



UNIVERSIDAD
NACIONAL
DE COLOMBIA

Libro de Sílabos

Programa Profesional de Administración
de Sistemas Informáticos

– 2022-II–

Sede Manizales: 15 de noviembre de 2022

Equipo de trabajo

Alex Cuadros-Vargas

Director del Departamento de Ciencia de la Computación, UCSP,
Arequipa-Perú

Miembro de la Sociedad Peruana de Computación

email: *alex@ucsp.edu.pe*

Kelly Vizconde La Motta

Profesora del Departamento de Ciencia de la Computación, UCSP,
Arequipa-Perú

email: *kvizconde@ucsp.edu.pe*

Ernesto Cuadros-Vargas (Editor)

Presidente de la Sociedad Peruana de Computación (SPC) 2001-2007, 2009
Miembro del *Steering Committee de ACM/IEEE-CS Computing Curricula for
Computer Science (CS2013)*

Miembro del *Steering Committee de ACM/IEEE-CS Computing Curricula
2020 (CS2020)*

email: *ecuadros@spc.org.pe*

http://socios.spc.org.pe/ecuadros

Además, han colaborado con este esfuerzo los siguientes profesionales: ?? a quienes dejamos público nuestro agradecimiento.

Índice general

Primer Semestre	5
1.1. CS111. Fundamentos de Computación	5
1.2. CS1D1. Estructuras Discretas I	12
1.3. MA100. Matemática I	17
1.4. FG101. Comunicación Integral	20
1.5. FG101. Comunicación Integral	22
1.6. FG102. Metodología del Estudio	24
Segundo Semestre	28
2.1. CS100. Introducción de Ciencia de la Computación	28
2.2. CS112. Ciencia de la Computación I	30
2.3. CS1D2. Estructuras Discretas II	38
2.4. MA101. Matemática II	42
Tercer Semestre	46
3.1. CS113. Ciencia de la Computación II	46
3.2. CS113. Ciencia de la Computación II	57
3.3. CS1D3. Álgebra Abstracta	68
3.4. CS221. Arquitectura de Computadores	72
3.5. CS2B1. Desarrollo Basado en Plataformas	80
3.6. MA102. Cálculo I	86
Cuarto Semestre	90
4.1. CS210. Algoritmos y Estructuras de Datos	90
4.2. CS211. Teoría de la Computación	93
4.3. CS271. Bases de Datos I	97
4.4. MA201. Cálculo II	103
4.5. MA203. Estadística y Probabilidades	106
Quinto Semestre	109
5.1. CS212. Análisis y Diseño de Algoritmos	109
5.2. CS272. Bases de Datos II	114
5.3. CS291. Ingeniería de Software I	119
5.4. CS342. Compiladores	125
5.5. CB111. Física Computacional	130
5.6. FG106. Teatro	134

Sexto Semestre	138
6.1. CS292. Ingeniería de Software II	138
6.2. CS2S1. Sistemas Operativos	144
6.3. CS311. Programación Competitiva	154
6.4. CS312. Estructuras de Datos Avanzadas	159
6.5. CS393. Sistemas de Infomación	163
6.6. MA307. Matemática aplicada a la computación	166
6.7. FG203. Oratoria	169
Séptimo Semestre	172
7.1. CS231. Redes y Comunicación	172
7.2. CS261. Inteligencia Artificial	177
7.3. CS2H1. Interacción Humano Computador	186
7.4. CS391. Ingeniería de Software III	193
7.5. CS401. Metodología de la Investigación en Computación	199
7.6. CS251. Computación Gráfica	201
7.7. FG350. Liderazgo	208
Octavo Semestre	211
8.1. CS281. Computación en la Sociedad	211
8.2. CS3I1. Seguridad en Computación	221
8.3. CS3P1. Computación Paralela y Distribuída	231
8.4. CS402. Proyecto de Final de Carrera I	238
8.5. ET201. Formación de Empresas de Base Tecnológica I	241
Noveno Semestre	246
9.1. CS370. Big Data	246
9.2. CS403. Proyecto de Final de Carrera II	249
9.3. CS351. Tópicos en Computación Gráfica	252
9.4. CS361. Tópicos en Inteligencia Artificial	254
9.5. CS392. Tópicos Avanzados en Ingeniería de Software	256
9.6. CB309. Bioinformática	261
9.7. ET301. Formación de Empresas de Base Tecnológica II	265
Décimo Semestre	269
10.1. CS3P2. Cloud Computing	269
10.2. CS3P3. Internet de las Cosas	275
10.3. CS404. Proyecto de Final de Carrera III	281
10.4. CS362. Robótica	284
10.5. FG211. Ética Profesional	288



Universidad Nacional de Colombia (UNAL) Sede
Manizales
Programa Profesional de
Administración de Sistemas Informáticos
SILABO

CS111. Fundamentos de Computación (Obligatorio)

2022-II

1. Información general	
1.1 Escuela	: Sistemas de Información
1.2 Curso	: CS111. Fundamentos de Computación
1.3 Semestre	: 1 ^{er} Semestre.
1.4 Prerrequisitos	: Ninguno
1.5 Condición	: Obligatorio
1.6 Modalidad de aprendizaje	: Presencial
1.7 horas	: 2 HT; 2 HP; 2 HL;
1.8 Créditos	: 4

2. Profesores

3. Fundamentación del curso
Este es el primer curso en la secuencia de los cursos introductorios a la Ciencia de la Computación. En este curso se pretende cubrir los conceptos señalados por la Computing Curricula IEEE-CS/ACM 2013. La programación es uno de los pilares de la Ciencia de la Computación; cualquier profesional del Área, necesitará programar para concretizar sus modelos y propuestas. Este curso introducción a los participantes en los conceptos fundamentales de este arte. Los tópicos incluyen tipos de datos, estructuras de control, funciones, listas, recursividad y la mecánica de la ejecución, prueba y depuración.

4. Resumen
1. Historia 2. Sistemas de tipos básicos 3. Conceptos Fundamentales de Programación 4. Análisis Básico 5. Algoritmos y Estructuras de Datos fundamentales 6. Algoritmos y Diseño 7. Métodos de Desarrollo

5. Objetivos Generales
<ul style="list-style-type: none">• Introducir los conceptos fundamentales de programación.• Desarrollar su capacidad de abstracción utilizar un lenguaje de programación.

6. Contribución a los resultados (<i>Outcomes</i>)
Esta disciplina contribuye al logro de los siguientes resultados de la carrera: <ol style="list-style-type: none">1) Analizar un problema computacional complejo y aplicar los principios computacionales y otras disciplinas relevantes para identificar soluciones. (Usar)2) Diseñar, implementar y evaluar una solución basada en computación para cumplir con un conjunto determinado de requisitos computacionales en el contexto de las disciplinas del programa. (Usar)6) Aplicar fundamentos de teoría de ciencias de la computación y desarrollo de software para producir soluciones basados en computación. (Usar)

7. Contenido

UNIDAD 1: Historia (5)

Competencias:	
Contenido	Objetivos Generales
<ul style="list-style-type: none">• Pre-historia – El mundo antes de 1946.• Historia del hardware, software, redes.• Pioneros de la Computación.• Historia de Internet.	<ul style="list-style-type: none">• Identificar importantes tendencias en la historia del campo de la computación [Familiarizarse]• Identificar las contribuciones de varios pioneros en el campo de la computación [Familiarizarse]• Discutir el contexto histórico de los paradigmas de diversos lenguajes de programación [Familiarizarse]• Comparar la vida diaria antes y después de la llegada de los ordenadores personales y el Internet [Evaluar]
Lecturas: Brookshear and Brylow (2019), Guttag (2013), Zelle (2010)	

UNIDAD 2: Sistemas de tipos básicos (2)

Competencias:	
Contenido	Objetivos Generales
<ul style="list-style-type: none">• Tipos como conjunto de valores junto con un conjunto de operaciones.<ul style="list-style-type: none">– Tipos primitivos (p.e. números, booleanos)– Composición de tipos contruídos de otros tipos (p.e., registros, uniones, arreglos, listas, funciones, referencias)• Asociación de tipos de variables, argumentos, resultados y campos.• Tipo de seguridad y los errores causados por el uso de valores de manera incompatible dadas sus tipos previstos.	<ul style="list-style-type: none">• Tanto para tipo primitivo y un tipo compuesto, describir de manera informal los valores que tiene dicho tipo [Familiarizarse]• Para un lenguaje con sistema de tipos estático, describir las operaciones que están prohibidas de forma estática, como pasar el tipo incorrecto de valor a una función o método [Familiarizarse]• Describir ejemplos de errores de programa detectadas por un sistema de tipos [Familiarizarse]• Para múltiples lenguajes de programación, identificar propiedades de un programa con verificación estática y propiedades de un programa con verificación dinámica [Usar]• Usar tipos y mensajes de error de tipos para escribir y depurar programas [Usar]• Definir y usar piezas de programas (tales como, funciones, clases, métodos) que usan tipos genéricos, incluyendo para colecciones [Usar]
Lecturas: Guttag (2013), Zelle (2010)	

UNIDAD 3: Conceptos Fundamentales de Programación (9)**Competencias:****Contenido**

- Sintaxis y semántica básica de un lenguaje de alto nivel.
- Variables y tipos de datos primitivos (ej., números, caracteres, booleanos)
- Expresiones y asignaciones.
- Operaciones básicas I/O incluyendo archivos I/O.
- Estructuras de control condicional e iterativas.
- Paso de funciones y parámetros.
- Concepto de recursividad.

Objetivos Generales

- Analiza y explica el comportamiento de programas simples que involucran estructuras fundamentales de programación variables, expresiones, asignaciones, E/S, estructuras de control, funciones, paso de parámetros, y recursividad [Evaluar]
- Identifica y describe el uso de tipos de datos primitivos [Familiarizarse]
- Escribe programas que usan tipos de datos primitivos [Usar]
- Modifica y expande programas cortos que usen estructuras de control condicionales e iterativas así como funciones [Usar]
- Diseña, implementa, prueba, y depura un programa que usa cada una de las siguientes estructuras de datos fundamentales: cálculos básicos, E/S simple, condicional estándar y estructuras iterativas, definición de funciones, y paso de parámetros [Usar]
- Escribe un programa que usa E/S de archivos para brindar persistencia a través de ejecuciones múltiples [Usar]
- Escoje estructuras de condición y repetición adecuadas para una tarea de programación dada [Familiarizarse]
- Describe el concepto de recursividad y da ejemplos de su uso [Evaluar]
- Identifica el caso base y el caso general de un problema basado en recursividad [Familiarizarse]

Lecturas: Guttag (2013), Zelle (2010)

UNIDAD 4: Análisis Básico (2)**Competencias:****Contenido**

- Diferencias entre el mejor, el esperado y el peor caso de un algoritmo.
- Definición formal de la Notación Big O.
- Clases de complejidad como constante, logarítmica, lineal, cuadrática y exponencial.
- Uso de la notación Big O.
- Análisis de algoritmos iterativos y recursivos.

Objetivos Generales

- Explique a que se refiere con “mejor”, “esperado” y “peor” caso de comportamiento de un algoritmo [Familiarizarse]
- En el contexto de a algoritmos específicos, identifique las características de data y/o otras condiciones o suposiciones que lleven a diferentes comportamientos [Familiarizarse]
- Indique la definición formal de Big O [Familiarizarse]
- Use la notación formal de la Big O para dar límites superiores asintóticos en la complejidad de tiempo y espacio de los algoritmos [Usar]
- Usar la notación formal Big O para dar límites de casos esperados en el tiempo de complejidad de los algoritmos [Usar]

Lecturas: Guttag (2013), Zelle (2010)

UNIDAD 5: Algoritmos y Estructuras de Datos fundamentales (8)**Competencias:****Contenido****Objetivos Generales**

- Algoritmos numéricos simples, tales como el cálculo de la media de una lista de números, encontrar el mínimo y máximo.
- Algoritmos de búsqueda secuencial y binaria.
- Algoritmos de ordenamiento de peor caso cuadrático (selección, inserción)
- Algoritmos de ordenamiento con peor caso o caso promedio en $O(N \lg N)$ (Quicksort, Heapsort, Mergesort)
- Tablas Hash, incluyendo estrategias para evitar y resolver colisiones.
- Árboles de búsqueda binaria:
 - Operaciones comunes en árboles de búsqueda binaria como seleccionar el mínimo, máximo, insertar, eliminar, recorrido en árboles.
- Grafos y algoritmos en grafos:
 - Representación de grafos (ej., lista de adyacencia, matriz de adyacencia)
 - Recorrido en profundidad y amplitud
- Montículos (Heaps)
- Grafos y algoritmos en grafos:
 - Problema de corte máximo y mínimo
 - Búsqueda local
- Búsqueda de patrones y algoritmos de cadenas/texto (ej. búsqueda de subcadena, búsqueda de expresiones regulares, algoritmos de subsecuencia común más larga)

- Implementar algoritmos numéricos básicos [Usar]
- Implementar algoritmos de búsqueda simple y explicar las diferencias en sus tiempos de complejidad [Evaluar]
- Ser capaz de implementar algoritmos de ordenamiento comunes cuadráticos y $O(N \log N)$ [Usar]
- Describir la implementación de tablas hash, incluyendo resolución y el evitamiento de colisiones [Familiarizarse]
- Discutir el tiempo de ejecución y eficiencia de memoria de los principales algoritmos de ordenamiento, búsqueda y hashing [Familiarizarse]
- Discutir factores otros que no sean eficiencia computacional que influyan en la elección de algoritmos, tales como tiempo de programación, mantenibilidad, y el uso de patrones específicos de la aplicación en los datos de entrada [Familiarizarse]
- Explicar como el balanceamiento del arbol afecta la eficiencia de varias operaciones de un arbol de búsqueda binaria [Familiarizarse]
- Resolver problemas usando algoritmos básicos de grafos, incluyendo búsqueda por profundidad y búsqueda por amplitud [Usar]
- Demostrar habilidad para evaluar algoritmos, para seleccionar de un rango de posibles opciones, para proveer una justificación por esa selección, y para implementar el algoritmo en un contexto en específico [Evaluar]
- Describir la propiedad del heap y el uso de heaps como una implementación de colas de prioridad [Familiarizarse]
- Resolver problemas usando algoritmos de grafos, incluyendo camino más corto de una sola fuente y camino más corto de todos los pares, y como mínimo un algoritmo de arbol de expansion minima [Usar]
- Trazar y/o implementar un algoritmo de comparación de string [Usar]

Lecturas: Gutttag (2013), Zelle (2010)

UNIDAD 6: Algoritmos y Diseño (9)	
Competencias:	
Contenido	Objetivos Generales
<ul style="list-style-type: none"> • Conceptos y propiedades de los algoritmos <ul style="list-style-type: none"> – Comparación informal de la eficiencia de los algoritmos (ej., conteo de operaciones) • Rol de los algoritmos en el proceso de solución de problemas • Estrategias de solución de problemas <ul style="list-style-type: none"> – Funciones matemáticas iterativas y recursivas – Recorrido iterativo y recursivo en estructura de datos – Estrategias Divide y Conquistar • Conceptos y principios fundamentales de diseño <ul style="list-style-type: none"> – Abstracción – Descomposición de Program – Encapsulamiento y camuflaje de información – Separación de comportamiento y aplicación 	<ul style="list-style-type: none"> • Discute la importancia de los algoritmos en el proceso de solución de un problema [Familiarizarse] • Discute como un problema puede ser resuelto por múltiples algoritmos, cada uno con propiedades diferentes [Familiarizarse] • Crea algoritmos para resolver problemas simples [Usar] • Usa un lenguaje de programación para implementar, probar, y depurar algoritmos para resolver problemas simples [Usar] • Implementa, prueba, y depura funciones recursivas simples y sus procedimientos [Usar] • Determina si una solución iterativa o recursiva es la más apropiada para un problema [Evaluar] • Implementa un algoritmo de divide y vencerás para resolver un problema [Usar] • Aplica técnicas de descomposición para dividir un programa en partes más pequeñas [Usar] • Identifica los componentes de datos y el comportamiento de múltiples tipos de datos abstractos [Usar] • Implementa un tipo de dato abstracto coherente, con la menor pérdida de acoplamiento entre componentes y comportamientos [Usar] • Identifica las fortalezas y las debilidades relativas entre múltiples diseños e implementaciones de un problema [Evaluar]
Lecturas: Guttag (2013), Zelle (2010)	

UNIDAD 7: Métodos de Desarrollo (1)	
Competencias:	
Contenido	Objetivos Generales
<ul style="list-style-type: none"> • Entornos modernos de programación: <ul style="list-style-type: none"> – Búsqueda de código. – Programación usando librería de componentes y sus APIs. 	<ul style="list-style-type: none"> • Construir y depurar programas que utilizan las bibliotecas estándar disponibles con un lenguaje de programación elegido [Familiarizarse]
Lecturas: Guttag (2013), Zelle (2010)	

8. Metodología

El profesor del curso presentará clases teóricas de los temas señalados en el programa propiciando la intervención de los alumnos.

El profesor del curso presentará demostraciones para fundamentar clases teóricas.

El profesor y los alumnos realizarán prácticas

Los alumnos deberán asistir a clase habiendo leído lo que el profesor va a presentar. De esta manera se facilitará la comprensión y los estudiantes estarán en mejores condiciones de hacer consultas en clase.

9. Evaluar

Evaluación Continua 1 : 20 %

Examen parcial : 30 %

Evaluación Continua 2 : 20 %

Examen final : 30 %

References

Brookshear, J. Glenn and Dennis Brylow (2019). *Computer Science: An Overview*. Ed. by PEARSON. Global Edition. Pearson. ISBN: 1292263423.

Gutttag, John V (2013). . *Introduction To Computation And Programming Using Python*. MIT Press.

Zelle, John (2010). *Python Programming: An Introduction to Computer Science*. Franklin, Beedle & Associates Inc.



Universidad Nacional de Colombia (UNAL) Sede
Manizales
Programa Profesional de
Administración de Sistemas Informáticos
SILABO

CS1D1. Estructuras Discretas I (Obligatorio)

2022-II

1. Información general	
1.1 Escuela	: Sistemas de Información
1.2 Curso	: CS1D1. Estructuras Discretas I
1.3 Semestre	: 1 ^{er} Semestre.
1.4 Prerrequisitos	: Ninguno
1.5 Condición	: Obligatorio
1.6 Modalidad de aprendizaje	: Presencial
1.7 horas	: 2 HT; 4 HP;
1.8 Créditos	: 4
2. Profesores	
3. Fundamentación del curso	
Las estructuras discretas proporcionan los fundamentos teóricos necesarios para la computación. Estos fundamentos no sólo son útiles para desarrollar la computación desde un punto de vista teórico como sucede En el curso de la teoría computacional, pero también es útil para la práctica de la informática; En particular en aplicaciones tales como verificación, Criptografía, métodos formales, etc.	
4. Resumen	
1. Funciones, relaciones y conjuntos 2. Lógica básica 3. Técnicas de demostración 4. Representación de Datos	
5. Objetivos Generales	
<ul style="list-style-type: none">• Aplicar Correctamente conceptos de matemáticas finitas (conjuntos, relaciones, funciones) para representar datos de problemas reales.• Modelar situaciones reales descritas en lenguaje natural, usando lógica proposicional y lógica predicada.• Determinar las propiedades abstractas de las relaciones binarias.• Elegir el método de demostración más apropiado para determinar la veracidad de una propuesta y construir argumentos matemáticos correctos.• Interpretar soluciones matemáticas a un problema y determinar su fiabilidad, ventajas y desventajas.• Expresar el funcionamiento de un circuito electrónico simple usando álgebra booleana.	

6. Contribución a los resultados (*Outcomes*)

Esta disciplina contribuye al logro de los siguientes resultados de la carrera:

- 1) Analizar un problema computacional complejo y aplicar los principios computacionales y otras disciplinas relevantes para identificar soluciones. (**Evaluar**)
- 6) Aplicar fundamentos de teoría de ciencias de la computación y desarrollo de software para producir soluciones basados en computación. (**Evaluar**)

7. Contenido

UNIDAD 1: Funciones, relaciones y conjuntos (22)

Competencias:

Contenido

Objetivos Generales

- Conjuntos:
 - Diagramas de Venn
 - Unión, intersección, complemento
 - Producto Cartesiano
 - Potencia de conjuntos
 - Cardinalidad de Conjuntos finitos
- Relaciones:
 - Reflexividad, simetría, transitividad
 - Relaciones de equivalencia
 - Relación de orden parcial y conjuntos parcialmente ordenados
 - Elementos extremos de un conjunto parcialmente ordenado
- Funciones:
 - Suryecciones, inyecciones, biyecciones
 - Inversas
 - Composición

- Explicar con ejemplos la terminología básica de funciones, relaciones y conjuntos [Evaluar]
- Realizar las operaciones asociadas con conjuntos, funciones y relaciones [Evaluar]
- Relacionar ejemplos prácticos para conjuntos funciones o modelos de relación apropiados e interpretar la asociación de operaciones y terminología en contexto [Evaluar]

Lecturas: Grimaldi (2003), Rosen (2007), Velleman (2006)

UNIDAD 2: Lógica básica (14)	
Competencias:	
Contenido	Objetivos Generales
<ul style="list-style-type: none"> • Lógica proposicional. • Conectores lógicos. • Tablas de verdad. • Forma normal (conjuntiva y disyuntiva) • Validación de fórmula bien formada. • Reglas de inferencia proposicional (conceptos de modus ponens y modus tollens) • Logica de predicados: <ul style="list-style-type: none"> – Cuantificación universal y existencial • Limitaciones de la lógica proposicional y de predicados (ej. problemas de expresividad) 	<ul style="list-style-type: none"> • Convertir declaraciones lógicas desde el lenguaje informal a expresiones de lógica proposicional y de predicados [Usar] • Aplicar métodos formales de simbolismo proposicional y lógica de predicados, como el cálculo de la validez de formulas y cálculo de formas normales [Usar] • Usar reglas de inferencia para construir demostraciones en lógica proposicional y de predicados [Usar] • Describir como la lógica simbólica puede ser usada para modelar situaciones o aplicaciones de la vida real, incluidos aquellos planteados en el contexto computacional como análisis de software (ejm. programas correctores), consulta de base de datos y algoritmos [Familiarizarse] • Aplicar demostraciones de lógica formal y/o informal, pero rigurosa, razonamiento lógico para problemas reales, como la predicción del comportamiento de software o solución de problemas tales como rompecabezas [Usar] • Describir las fortalezas y limitaciones de la lógica proposicional y de predicados [Usar]
Lecturas: Rosen (2007), Grimaldi (2003), Velleman (2006)	

UNIDAD 3: Técnicas de demostración (14)	
Competencias:	
Contenido	Objetivos Generales
<ul style="list-style-type: none"> • Nociones de implicancia, equivalencia, conversión, inversa, contrapositivo, negación, y contradicción • Estructura de pruebas matemáticas. • Demostración directa. • Refutar por contraejemplo. • Demostración por contradicción. • Inducción sobre números naturales. • Inducción estructural. • Inducción leve y fuerte (Ej. Primer y Segundo principio de la inducción) • Definiciones matemáticas recursivas. • Conjuntos bien ordenados. 	<ul style="list-style-type: none"> • Identificar la técnica de demostración utilizada en una demostración dada [Evaluar] • Describir la estructura básica de cada técnica de demostración (demostración directa, demostración por contradicción e inducción) descritas en esta unidad [Usar] • Aplicar las técnicas de demostración (demostración directa, demostración por contradicción e inducción) correctamente en la construcción de un argumento solido [Usar] • Determine que tipo de demostración es la mejor para un problema dado [Evaluar] • Explicar el paralelismo entre ideas matemáticas y/o inducción estructural para la recursión y definir estructuras recursivamente [Familiarizarse] • Explicar la relación entre inducción fuerte y débil y dar ejemplos del apropiado uso de cada uno [Evaluar] • Enunciar el principio del buen-orden y su relación con la inducción matemática [Familiarizarse]
Lecturas: Rosen (2007), Vel06, Scheinerman (2012), Velleman (2006)	

UNIDAD 4: Representación de Datos (10)	
Competencias:	
Contenido	Objetivos Generales
<ul style="list-style-type: none"> • Representaciones numéricas: signo magnitud, punto flotante. • Representaciones de otros objetos: conjuntos, relaciones, funciones 	<ul style="list-style-type: none"> • Conocer las formas de representación numérica como signo magnitud y punto flotante. [Evaluar]. • Llevar a cabo operaciones aritméticas utilizando las distintas formas de representación. [Evaluar]. • Conocer el estándar de punto flotante IEEE-754 [Familiarizarse].
Lecturas: Rosen (2007), Grimaldi (2003), Velleman (2006)	

8. Metodología
<p>El profesor del curso presentará clases teóricas de los temas señalados en el programa propiciando la intervención de los alumnos.</p> <p>El profesor del curso presentará demostraciones para fundamentar clases teóricas.</p> <p>El profesor y los alumnos realizarán prácticas</p> <p>Los alumnos deberán asistir a clase habiendo leído lo que el profesor va a presentar. De esta manera se facilitará la comprensión y los estudiantes estarán en mejores condiciones de hacer consultas en clase.</p>

9. Evaluar

Evaluación Continua 1 : 20 %

Examen parcial : 30 %

Evaluación Continua 2 : 20 %

Examen final : 30 %

References

Grimaldi, R. (2003). *Discrete and Combinatorial Mathematics: An Applied Introduction*. 5 ed. Pearson.

Rosen, Kenneth H. (2007). *Discrete Mathematics and Its Applications*. 7 ed. Mc Graw Hill.

Scheinerman, Edward R. (2012). *Mathematics: A Discrete Introduction*. 3 ed. Brooks Cole.

Velleman, Daniel J. (2006). *How to Prove It: A Structured Approach*. Ed. by Cambridge University Pres. 2nd. ISBN: 978-0521675994.



Universidad Nacional de Colombia (UNAL) Sede
Manizales
Programa Profesional de
Administración de Sistemas Informáticos
SILABO

MA100. Matemática I (Obligatorio)

2022-II

1. Información general	
1.1 Escuela	: Sistemas de Información
1.2 Curso	: MA100. Matemática I
1.3 Semestre	: 1 ^{er} Semestre.
1.4 Prerrequisitos	: Ninguno
1.5 Condición	: Obligatorio
1.6 Modalidad de aprendizaje	: Presencial
1.7 horas	: 2 HT; 6 HP;
1.8 Créditos	: 5
2. Profesores	
3. Fundamentación del curso	
The course aims to develop in students the skills to deal with models in science and engineering related to single variable differential calculus skills. In the course it is studied and applied concepts related to calculation limits, derivatives and integrals of real and vector functions of single real variables to be used as base and support for the study of new contents and subjects. Also seeks to achieve reasoning capabilities and applicability to interact with real-world problems by providing a mathematical basis for further professional development activities.	
4. Resumen	
1. Números complejos 2. Functions of a single variable 3. Límites y derivadas 4. Integrales	
5. Objetivos Generales	
<ul style="list-style-type: none">• .• .• .	
6. Contribución a los resultados (<i>Outcomes</i>)	
Esta disciplina contribuye al logro de los siguientes resultados de la carrera: 1) Analizar un problema computacional complejo y aplicar los principios computacionales y otras disciplinas relevantes para identificar soluciones. (Evaluar) 6) Aplicar fundamentos de teoría de ciencias de la computación y desarrollo de software para producir soluciones basados en computación. (Evaluar)	
7. Contenido	

UNIDAD 1: Números complejos (20)	
Competencias:	
Contenido	Objetivos Generales
<ul style="list-style-type: none"> • . • . 	<ul style="list-style-type: none"> • . • .
Lecturas: Stewart (2012), ión (2014)	

UNIDAD 2: Functions of a single variable (10)	
Competencias:	
Contenido	Objetivos Generales
<ul style="list-style-type: none"> • . • . • . • . • . • . 	<ul style="list-style-type: none"> • . • . • . • . • . • .
Lecturas: Stewart (2012), ión (2014)	

UNIDAD 3: Límites y derivadas (20)	
Competencias:	
Contenido	Objetivos Generales
<ul style="list-style-type: none"> • . • . • . • . • . 	<ul style="list-style-type: none"> • . • . • . • . • . • . • . • . • . • . • . • .
Lecturas: Stewart (2012), ión (2014)	

UNIDAD 4: Integrales (22)	
Competencias:	
Contenido	Objetivos Generales
<ul style="list-style-type: none"> • . • . • . • . 	<ul style="list-style-type: none"> • . • . • . • . • . • . • . • . • . • . • .
Lecturas: Stewart (2012), ión (2014)	

8. Metodología
<p>El profesor del curso presentará clases teóricas de los temas señalados en el programa propiciando la intervención de los alumnos.</p> <p>El profesor del curso presentará demostraciones para fundamentar clases teóricas.</p> <p>El profesor y los alumnos realizarán prácticas</p> <p>Los alumnos deberán asistir a clase habiendo leído lo que el profesor va a presentar. De esta manera se facilitará la comprensión y los estudiantes estarán en mejores condiciones de hacer consultas en clase.</p>

9. Evaluar
<p>Evaluación Continua 1 : 20 %</p> <p>Examen parcial : 30 %</p> <p>Evaluación Continua 2 : 20 %</p> <p>Examen final : 30 %</p>

References

ión, ROn Larson (2014). *Calculus*. 10th.
 Stewart, James (2012). *Calculus*. 7th.



Universidad Nacional de Colombia (UNAL) Sede
Manizales
Programa Profesional de
Administración de Sistemas Informáticos
SILABO

FG101. Comunicación Integral (Obligatorio)

2022-II

1. Información general	
1.1 Escuela	: Sistemas de Información
1.2 Curso	: FG101. Comunicación Integral
1.3 Semestre	: 1 ^{er} Semestre.
1.4 Prerrequisitos	: Ninguno
1.5 Condición	: Obligatorio
1.6 Modalidad de aprendizaje	: Presencial
1.7 horas	: 2 HT; 2 HP;
1.8 Créditos	: 3
2. Profesores	
3. Fundamentación del curso	
Para lograr una eficaz comunicación en el ámbito personal y profesional, es prioritario el manejo adecuado de la Lengua en forma oral y escrita. Se justifica, por lo tanto, que los alumnos de la Universidad Católica San Pablo conozcan, comprendan y apliquen los aspectos conceptuales y operativos de su idioma, para el desarrollo de sus habilidades comunicativas fundamentales: Escuchar, hablar, leer y escribir. En consecuencia el ejercicio permanente y el aporte de los fundamentos contribuyen grandemente en la formación académica y, en el futuro, en el desempeño de su profesión	
4. Resumen	
1. Primera Unidad	
5. Objetivos Generales	
<ul style="list-style-type: none">• Desarrollar capacidades comunicativas a través de la teoría y práctica del lenguaje que ayuden al estudiante a superar las exigencias académicas del pregrado y contribuyan a su formación humanística y como persona humana.	
6. Contribución a los resultados (Outcomes)	
Esta disciplina contribuye al logro de los siguientes resultados de la carrera:	
3) Comunicarse efectivamente en diversos contextos profesionales. (Usar)	
5) Funcionar efectivamente como miembro o líder de un equipo involucrado en actividades apropiadas a la disciplina del programa. (Usar)	
6) Aplicar fundamentos de teoría de ciencias de la computación y desarrollo de software para producir soluciones basados en computación. (Usar)	
7) Desarrollar tecnología computacional buscando el bien común, aportando con formación humana, capacidades científicas, tecnológicas y profesionales para solucionar problemas sociales de nuestro entorno. (Usar)	
7. Contenido	

UNIDAD 1: Primera Unidad (16)	
Competencias:	
Contenido	Objetivos Generales
<ul style="list-style-type: none"> • La comunicación, definición, relevancia. Elementos. Proceso. Funciones. Clasificación. Comunicación oral y escrita. • El lenguaje: definición. Características y funciones. Lengua: niveles. Sistema. Norma. Habla. El signo lingüístico: definición, características. • Multilingüismo en el Perú. Variaciones dialectales en el Perú. • La palabra: definición, clases y estructura. Los monemas: lexema y morfema. El morfema: clases. La etimología. • El Artículo académico: Definición, estructura, elección del tema, delimitación del tema. 	<ul style="list-style-type: none"> • Reconocer y valorar la comunicación como un proceso de comprensión e intercambio de mensajes, diferenciando sus elementos, funciones y clasificación [Usar]. • Analizar las características, funciones y elementos del lenguaje y de la lengua [Usar]. • Identificar las características del multilingüismo en el Perú, valorando su riqueza idiomática [Usar]. • Identificar las cualidades de la palabra y sus clases [Usar].
Lecturas: Lengua Española (2010)	

8. Metodología
<p>El profesor del curso presentará clases teóricas de los temas señalados en el programa propiciando la intervención de los alumnos.</p> <p>El profesor del curso presentará demostraciones para fundamentar clases teóricas.</p> <p>El profesor y los alumnos realizarán prácticas</p> <p>Los alumnos deberán asistir a clase habiendo leído lo que el profesor va a presentar. De esta manera se facilitará la comprensión y los estudiantes estarán en mejores condiciones de hacer consultas en clase.</p>

9. Evaluar
<p>Evaluación Continua 1 : 20 %</p> <p>Examen parcial : 30 %</p> <p>Evaluación Continua 2 : 20 %</p> <p>Examen final : 30 %</p>

References

Lengua Española, Real Academia de la (2010). *Nueva gramática de la lengua española, morfología y sintaxis*. Madrid, España: Ed. Espasa.



Universidad Nacional de Colombia (UNAL) Sede
Manizales
Programa Profesional de
Administración de Sistemas Informáticos
SILABO

FG101. Comunicación Integral (Obligatorio)

2022-II

1. Información general	
1.1 Escuela	: Sistemas de Información
1.2 Curso	: FG101. Comunicación Integral
1.3 Semestre	: 1 ^{er} Semestre.
1.4 Prerrequisitos	: Ninguno
1.5 Condición	: Obligatorio
1.6 Modalidad de aprendizaje	: Presencial
1.7 horas	: 2 HT; 2 HP;
1.8 Créditos	: 3
2. Profesores	
3. Fundamentación del curso	
Para lograr una eficaz comunicación en el ámbito personal y profesional, es prioritario el manejo adecuado de la Lengua en forma oral y escrita. Se justifica, por lo tanto, que los alumnos de la Universidad Católica San Pablo conozcan, comprendan y apliquen los aspectos conceptuales y operativos de su idioma, para el desarrollo de sus habilidades comunicativas fundamentales: Escuchar, hablar, leer y escribir. En consecuencia el ejercicio permanente y el aporte de los fundamentos contribuyen grandemente en la formación académica y, en el futuro, en el desempeño de su profesión	
4. Resumen	
1. Primera Unidad	
5. Objetivos Generales	
<ul style="list-style-type: none">• Desarrollar capacidades comunicativas a través de la teoría y práctica del lenguaje que ayuden al estudiante a superar las exigencias académicas del pregrado y contribuyan a su formación humanística y como persona humana.	
6. Contribución a los resultados (Outcomes)	
Esta disciplina contribuye al logro de los siguientes resultados de la carrera:	
3) Comunicarse efectivamente en diversos contextos profesionales. (Usar)	
5) Funcionar efectivamente como miembro o líder de un equipo involucrado en actividades apropiadas a la disciplina del programa. (Usar)	
6) Aplicar fundamentos de teoría de ciencias de la computación y desarrollo de software para producir soluciones basados en computación. (Usar)	
7) Desarrollar tecnología computacional buscando el bien común, aportando con formación humana, capacidades científicas, tecnológicas y profesionales para solucionar problemas sociales de nuestro entorno. (Usar)	
7. Contenido	

UNIDAD 1: Primera Unidad (16)	
Competencias:	
Contenido	Objetivos Generales
<ul style="list-style-type: none"> • La comunicación, definición, relevancia. Elementos. Proceso. Funciones. Clasificación. Comunicación oral y escrita. • El lenguaje: definición. Características y funciones. Lengua: niveles. Sistema. Norma. Habla. El signo lingüístico: definición, características. • Multilingüismo en el Perú. Variaciones dialectales en el Perú. • La palabra: definición, clases y estructura. Los monemas: lexema y morfema. El morfema: clases. La etimología. • El Artículo académico: Definición, estructura, elección del tema, delimitación del tema. 	<ul style="list-style-type: none"> • Reconocer y valorar la comunicación como un proceso de comprensión e intercambio de mensajes, diferenciando sus elementos, funciones y clasificación [Usar]. • Analizar las características, funciones y elementos del lenguaje y de la lengua [Usar]. • Identificar las características del multilingüismo en el Perú, valorando su riqueza idiomática [Usar]. • Identificar las cualidades de la palabra y sus clases [Usar].
Lecturas: Lengua Española (2010)	

8. Metodología
<p>El profesor del curso presentará clases teóricas de los temas señalados en el programa propiciando la intervención de los alumnos.</p> <p>El profesor del curso presentará demostraciones para fundamentar clases teóricas.</p> <p>El profesor y los alumnos realizarán prácticas</p> <p>Los alumnos deberán asistir a clase habiendo leído lo que el profesor va a presentar. De esta manera se facilitará la comprensión y los estudiantes estarán en mejores condiciones de hacer consultas en clase.</p>

9. Evaluar
<p>Evaluación Continua 1 : 20 %</p> <p>Examen parcial : 30 %</p> <p>Evaluación Continua 2 : 20 %</p> <p>Examen final : 30 %</p>

References

Lengua Española, Real Academia de la (2010). *Nueva gramática de la lengua española, morfología y sintaxis*. Madrid, España: Ed. Espasa.



Universidad Nacional de Colombia (UNAL) Sede
Manizales
Programa Profesional de
Administración de Sistemas Informáticos
SILABO

FG102. Metodología del Estudio (Obligatorio)

2022-II

1. Información general	
1.1 Escuela	: Sistemas de Información
1.2 Curso	: FG102. Metodología del Estudio
1.3 Semestre	: 1 ^{er} Semestre.
1.4 Prerrequisitos	: Ninguno
1.5 Condición	: Obligatorio
1.6 Modalidad de aprendizaje	: Presencial
1.7 horas	: 2 HT; 2 HP;
1.8 Créditos	: 3
2. Profesores	
3. Fundamentación del curso	
Los alumnos en formación profesional necesitan mejorar su actitud frente al trabajo y exigencia académicos. Además conviene que entiendan el proceso mental que se da en el ejercicio del estudio para lograr el aprendizaje; así sabrán dónde y cómo hacer los ajustes más convenientes a sus necesidades. Asimismo, requieren dominar variadas formas de estudiar, para que puedan seleccionar las estrategias más convenientes a su personal estilo de aprender y a la naturaleza de cada asignatura. De igual modo conocer y usar maneras de buscar información académica y realizar trabajos creativos de tipo académico formal, así podrán aplicarlos a su trabajo universitario, haciendo exitoso su esfuerzo.	
4. Resumen	
1. 2. 3. 4.	
5. Objetivos Generales	
<ul style="list-style-type: none">• Desarrollar en el estudiante actitudes y habilidades que promuevan la autonomía en el aprendizaje, el buen desempeño académico y su formación como persona y profesional.	

6. Contribución a los resultados (*Outcomes*)

Esta disciplina contribuye al logro de los siguientes resultados de la carrera:

- 1) Analizar un problema computacional complejo y aplicar los principios computacionales y otras disciplinas relevantes para identificar soluciones. (**Familiarizarse**)
- 2) Diseñar, implementar y evaluar una solución basada en computación para cumplir con un conjunto determinado de requisitos computacionales en el contexto de las disciplinas del programa. (**Familiarizarse**)
- 3) Comunicarse efectivamente en diversos contextos profesionales. (**Usar**)
- 5) Funcionar efectivamente como miembro o líder de un equipo involucrado en actividades apropiadas a la disciplina del programa. (**Usar**)
- 6) Aplicar fundamentos de teoría de ciencias de la computación y desarrollo de software para producir soluciones basados en computación. (**Usar**)
- 7) Desarrollar tecnología computacional buscando el bien común, aportando con formación humana, capacidades científicas, tecnológicas y profesionales para solucionar problemas sociales de nuestro entorno. (**Familiarizarse**)

7. Contenido

UNIDAD 1: (12)

Competencias:

Contenido	Objetivos Generales
<ul style="list-style-type: none">• El subrayado.• Toma de puntas.• La vocación, hábitos de la vida universitaria.• Interacción humana.• La voluntad como requisito para el aprendizaje.• La planificación y el tiempo	<ul style="list-style-type: none">• Analizar la documentación normativa de la Universidad valorando su importancia para la convivencia y desempeño académico. [Usar]• Comprender y valorar la exigencia de la vida universitaria como parte de la formación personal y profesional.[Usar]• Planificar adecuadamente el tiempo en función de sus metas personales y académicas.[Usar]• Elaborar un plan de mejora personal a partir del conocimiento de sí mismo.[Usar]

Lecturas: Bibliografía

UNIDAD 2: (12)	
Competencias:	
Contenido	Objetivos Generales
<ul style="list-style-type: none"> • Resumen. Notas al margen. Nemotecnias. • Procesos mentales: Simples, complejos. Fundamentos del aprendizaje significativo. • Los pasos o factores para el aprendizaje. Leyes del aprendizaje. Cuestionario de estilos de aprendizaje Identificación del estilo de aprendizaje personal • La lectura académica. Niveles de análisis de un texto: idea central, idea principal e ideas secundarias. El modelo de Meza de Vernet. • Exámenes: Preparación. Pautas y estrategias para antes, durante y después de un examen. Inteligencia emocional y exámenes. • Las fuentes de información. Aparato crítico: concepto y finalidad. Normas Vancouver. Referencias y citas. 	<ul style="list-style-type: none"> • Identificar los procesos mentales relacionándolos con el aprendizaje [Usar]. • Comprender el proceso del aprendizaje para determinar el estilo propio e incorporarlo en su actividad académica [Usar]. • Desarrollar estrategias para el análisis de textos potenciando la comprensión lectora [Usar]. • Diseñar un programa estratégico para afrontar con éxito los exámenes[Usar].
Lecturas: Rodríguez (2007), Perez (2010), Quintana (2007)	

UNIDAD 3: (12)	
Competencias:	
Contenido	Objetivos Generales
<ul style="list-style-type: none"> • Los mapas conceptuales. Características y elementos. • Los derechos de autor y el plagio. Derechos personales o morales. Derechos patrimoniales. “Copyright”. • Autoestima, Inteligencia Emocional, Asertividad y Resiliencia. Conceptos, desarrollo y fortalecimiento. • Aparato crítico: Normas Vancouver. Aplicación práctica. • Generación de ideas. Estrategias para organizar las ideas, redacción y revisión. 	<ul style="list-style-type: none"> • Aplicar las técnicas de estudio atendiendo a sus particularidades y adecuándolas a las distintas situaciones que demanda el aprendizaje [Usar]. • Reconocer la importancia del respeto a la propiedad Intelectual [Usar]. • Reconocer la importancia de la Inteligencia Emocional, la conducta asertiva, la autoestima y la resiliencia valorándolas como fortalezas para el desempeño universitario [Usar].
Lecturas: Chávez (2011), Velazco (1999)	

UNIDAD 4: (12)	
Competencias:	
Contenido	Objetivos Generales
<ul style="list-style-type: none"> • Cuadro Sinóptico. Los mapas mentales. Practicas con la temática del curso. • El método personal de estudio. • El aprendizaje cooperativo: definición, los grupos de estudio, organización, roles de los miembros. • Pautas para conformar grupos eficientes y armónicos. • El método personal de estudio.Reforzamiento de técnicas de estudio. • Presentación y exposición de trabajos de producción intelectual. • El debate y la argumentación. 	<ul style="list-style-type: none"> • Aplicar las técnicas de estudio atendiendo a sus particularidades y adecuándolas a las distintas situaciones que demanda el aprendizaje [Usar]. • Asumir el manejo de conductas y actitudes para el aprendizaje cooperativo y el desempeño en los equipos de trabajo [Usar]. • Formular un proyecto de método personal de estudio, de acuerdo a su estilo y necesidades, que incluya técnicas y estrategias [Usar].
Lecturas: Rodríguez (2007), Chávez (2011)	

8. Metodología
<p>El profesor del curso presentará clases teóricas de los temas señalados en el programa propiciando la intervención de los alumnos.</p> <p>El profesor del curso presentará demostraciones para fundamentar clases teóricas.</p> <p>El profesor y los alumnos realizarán prácticas</p> <p>Los alumnos deberán asistir a clase habiendo leído lo que el profesor va a presentar. De esta manera se facilitará la comprensión y los estudiantes estarán en mejores condiciones de hacer consultas en clase.</p>

9. Evaluar
<p>Evaluación Continua 1 : 20 %</p> <p>Examen parcial : 30 %</p> <p>Evaluación Continua 2 : 20 %</p> <p>Examen final : 30 %</p>

References

- Chávez, A. (2011). *Se necesita un tutor*. UCSP.
- Perez, A.E. (2010). *Teoría del Derecho*. Editorial Madrid.
- Quintana, V. (2007). *El estudio Universitario y elementos de investigación científica*. Editorial universitaria.
- Rodríguez, J. (2007). *Guía para el método de estudio universitario*. Educa.
- Velazco, Marco Flores (1999). *Mapas conceptuales en el aula*. Ed. San Marcos.



Universidad Nacional de Colombia (UNAL) Sede
Manizales
Programa Profesional de
Administración de Sistemas Informáticos
SILABO

CS100. Introducción de Ciencia de la Computación
(Obligatorio)

2022-II

1. Información general

1.1 Escuela	:	Sistemas de Información
1.2 Curso	:	CS100. Introducción de Ciencia de la Computación
1.3 Semestre	:	2 ^{do} Semestre.
1.4 Prerrequisitos	:	Ninguno
1.5 Condición	:	Obligatorio
1.6 Modalidad de aprendizaje	:	Presencial
1.7 horas	:	2 HT; 2 HP;
1.8 Créditos	:	3

2. Profesores

3. Fundamentación del curso

La Ciencia de la Computación es un campo de estudio enorme con muchas especialidades y aplicaciones. Este curso brindará a sus participantes, una visión panorámica de la informática y mostrará sus campos más representativos, como son: Algoritmos, Estructuras de Datos, Sistemas Operativos, Bases de Datos, etc.

4. Resumen

1. Introducción 2. Lógica básica

5. Objetivos Generales

- Brindar un panorama del área del conocimiento que es cubierta en la ciencia de la computación.

6. Contribución a los resultados (Outcomes)

Esta disciplina contribuye al logro de los siguientes resultados de la carrera:

- 1) Analizar un problema computacional complejo y aplicar los principios computacionales y otras disciplinas relevantes para identificar soluciones. (**Familiarizarse**)
- 4) Reconocer las responsabilidades profesionales y hacer juicios informados en el campo profesional de computación con principios éticos. (**Familiarizarse**)
- 5) Funcionar efectivamente como miembro o líder de un equipo involucrado en actividades apropiadas a la disciplina del programa. (**Familiarizarse**)
- 6) Aplicar fundamentos de teoría de ciencias de la computación y desarrollo de software para producir soluciones basados en computación. (**Familiarizarse**)
- 7) Desarrollar tecnología computacional buscando el bien común, aportando con formación humana, capacidades científicas, tecnológicas y profesionales para solucionar problemas sociales de nuestro entorno. (**Familiarizarse**)

7. Contenido

UNIDAD 1: Introducción (2)	
Competencias:	
Contenido	Objetivos Generales
<ul style="list-style-type: none">• Introducción a la computación.• Historia de la computación.	<ul style="list-style-type: none">• Incentivar a los alumnos el estudio de Computacion como una ciencia. [Familiarizarse]
Lecturas: Brookshear (2015)	

UNIDAD 2: Lógica básica (2)	
Competencias:	
Contenido	Objetivos Generales
<ul style="list-style-type: none">• Lógica proposicional.• Conectores lógicos.• Tablas de verdad.• Forma normal (conjuntiva y disyuntiva)	<ul style="list-style-type: none">• Convertir declaraciones lógicas desde el lenguaje informal a expresiones de lógica proposicional y de predicados [Familiarizarse]• Aplicar métodos formales de simbolismo proposicional y lógica de predicados, como el cálculo de la validez de formulas y cálculo de formas normales [Familiarizarse]
Lecturas: Brookshear (2015)	

8. Metodología
<p>El profesor del curso presentará clases teóricas de los temas señalados en el programa propiciando la intervención de los alumnos.</p> <p>El profesor del curso presentará demostraciones para fundamentar clases teóricas.</p> <p>El profesor y los alumnos realizarán prácticas</p> <p>Los alumnos deberán asistir a clase habiendo leído lo que el profesor va a presentar. De esta manera se facilitará la comprensión y los estudiantes estarán en mejores condiciones de hacer consultas en clase.</p>

9. Evaluar
<p>Evaluación Continua 1 : 20 %</p> <p>Examen parcial : 30 %</p> <p>Evaluación Continua 2 : 20 %</p> <p>Examen final : 30 %</p>

References

Brookshear, J. G. (2015). *Computer Science: An Overview*. 12th. Addison-Wesley.



Universidad Nacional de Colombia (UNAL) Sede
Manizales
Programa Profesional de
Administración de Sistemas Informáticos
SILABO

CS112. Ciencia de la Computación I (Obligatorio)

2022-II

1. Información general	
1.1 Escuela	: Sistemas de Información
1.2 Curso	: CS112. Ciencia de la Computación I
1.3 Semestre	: 2 ^{do} Semestre.
1.4 Prerrequisitos	: CS111. Fundamentos de Computación. (1 ^{er} Sem)
1.5 Condición	: Obligatorio
1.6 Modalidad de aprendizaje	: Presencial
1.7 horas	: 2 HT; 2 HP; 4 HL;
1.8 Créditos	: 5
2. Profesores	
3. Fundamentación del curso	
Este es el segundo curso en la secuencia de los cursos introductorios a la Ciencia de la Computación. El curso introducirá a los participantes en los diversos temas del área de computación como: algoritmos, estructuras de datos, ingeniería del software, etc.	
4. Resumen	
1. Visión General de los Lenguajes de Programación 2. Sistemas de tipos básicos 3. Conceptos Fundamentales de Programación 4. Programación orientada a objetos 5. Algoritmos y Diseño 6. Estrategias Algorítmicas 7. Análisis Básico 8. Algoritmos y Estructuras de Datos fundamentales	
5. Objetivos Generales	
<ul style="list-style-type: none">• Introducir al alumno a los fundamentos del paradigma de orientación a objetos, permitiendo asimilar los conceptos necesarios para desarrollar sistemas de información.	
6. Contribución a los resultados (Outcomes)	
Esta disciplina contribuye al logro de los siguientes resultados de la carrera:	
1) Analizar un problema computacional complejo y aplicar los principios computacionales y otras disciplinas relevantes para identificar soluciones. (Evaluar)	
2) Diseñar, implementar y evaluar una solución basada en computación para cumplir con un conjunto determinado de requisitos computacionales en el contexto de las disciplinas del programa. (Evaluar)	
5) Funcionar efectivamente como miembro o líder de un equipo involucrado en actividades apropiadas a la disciplina del programa. (Familiarizarse)	
6) Aplicar fundamentos de teoría de ciencias de la computación y desarrollo de software para producir soluciones basados en computación. (Usar)	

7. Contenido

UNIDAD 1: Visión General de los Lenguajes de Programación (1)

Competencias:

Contenido

- Breve revisión de los paradigmas de programación.
- Comparación entre programación funcional y programación imperativa.
- Historia de los lenguajes de programación.

Objetivos Generales

- Discutir el contexto histórico de los paradigmas de diversos lenguajes de programación [Familiarizarse]

Lecturas: Stroustrup2013, Deitel17

UNIDAD 2: Sistemas de tipos básicos (2)**Competencias:****Contenido****Objetivos Generales**

- Tipos como conjunto de valores junto con un conjunto de operaciones.
 - Tipos primitivos (p.e. números, booleanos)
 - Composición de tipos construídos de otros tipos (p.e., registros, uniones, arreglos, listas, funciones, referencias)
- Declaración de modelos (enlace, visibilidad, alcance y tiempo de vida).
- Vista general del chequeo de tipos.

- Tanto para tipo primitivo y un tipo compuesto, describir de manera informal los valores que tiene dicho tipo [Familiarizarse]
- Para un lenguaje con sistema de tipos estático, describir las operaciones que están prohibidas de forma estática, como pasar el tipo incorrecto de valor a una función o método [Familiarizarse]
- Describir ejemplos de errores de programa detectadas por un sistema de tipos [Familiarizarse]
- Para múltiples lenguajes de programación, identificar propiedades de un programa con verificación estática y propiedades de un programa con verificación dinámica [Usar]
- Dar un ejemplo de un programa que no verifique tipos en un lenguaje particular y sin embargo no tenga error cuando es ejecutado [Familiarizarse]
- Usar tipos y mensajes de error de tipos para escribir y depurar programas [Usar]
- Explicar como las reglas de tipificación definen el conjunto de operaciones que legales para un tipo [Familiarizarse]
- Escribir las reglas de tipo que rigen el uso de un particular tipo compuesto [Usar]
- Explicar por qué indecidibilidad requiere sistemas de tipo para conservadoramente aproximar el comportamiento de un programa [Familiarizarse]
- Definir y usar piezas de programas (tales como, funciones, clases, métodos) que usan tipos genéricos, incluyendo para colecciones [Usar]
- Discutir las diferencias entre, genéricos (*generics*), subtipo y sobrecarga [Familiarizarse]
- Explicar múltiples beneficios y limitaciones de tipificación estática en escritura, mantenimiento y depuración de un software [Familiarizarse]

Lecturas: Stroustrup2013, Deitel17

UNIDAD 3: Conceptos Fundamentales de Programación (6)**Competencias:****Contenido**

- Sintaxis y semántica básica de un lenguaje de alto nivel.
- Variables y tipos de datos primitivos (ej., números, caracteres, booleanos)
- Expresiones y asignaciones.
- Operaciones básicas I/O incluyendo archivos I/O.
- Estructuras de control condicional e iterativas.
- Paso de funciones y parámetros.

Objetivos Generales

- Analiza y explica el comportamiento de programas simples que involucran estructuras fundamentales de programación variables, expresiones, asignaciones, E/S, estructuras de control, funciones, paso de parámetros, y recursividad [Evaluar]
- Identifica y describe el uso de tipos de datos primitivos [Familiarizarse]
- Escribe programas que usan tipos de datos primitivos [Usar]
- Modifica y expande programas cortos que usen estructuras de control condicionales e iterativas así como funciones [Usar]
- Diseña, implementa, prueba, y depura un programa que usa cada una de las siguientes estructuras de datos fundamentales: cálculos básicos, E/S simple, condicional estándar y estructuras iterativas, definición de funciones, y paso de parámetros [Usar]
- Escribe un programa que usa E/S de archivos para brindar persistencia a través de ejecuciones múltiples [Usar]
- Escoje estructuras de condición y repetición adecuadas para una tarea de programación dada [Evaluar]
- Describe el concepto de recursividad y da ejemplos de su uso [Familiarizarse]
- Identifica el caso base y el caso general de un problema basado en recursividad [Evaluar]

Lecturas: Stroustrup2013, Deitel17

UNIDAD 4: Programación orientada a objetos (10)**Competencias:****Contenido****Objetivos Generales**

- Diseño orientado a objetos:
 - Descomposición en objetos que almacenan estados y poseen comportamiento
 - Diseño basado en jerarquía de clases para modelamiento
- Lenguajes orientados a objetos para la encapsulación:
 - privacidad y la visibilidad de miembros de la clase
 - Interfaces revelan único método de firmas
 - clases base abstractas
- Definición de las categorías, campos, métodos y constructores.
- Las subclases, herencia y método de alteración temporal.
- Subtipificación:
 - Polimorfismo artículo Subtipo; upcasts implícitos en lenguajes con tipos.
 - Noción de reemplazo de comportamiento: los subtipos de actuar como supertipos.
 - Relación entre subtipos y la herencia.
- Uso de colección de clases, iteradores, y otros componentes de la librería estándar.
- Asignación dinámica: definición de método de llamada.

- Diseñar e implementar una clase [Usar]
- Usar subclase para diseñar una jerarquía simple de clases que permita al código ser reusable por diferentes subclases [Usar]
- Razonar correctamente sobre el flujo de control en un programa mediante el envío dinámico [Usar]
- Comparar y contrastar (1) el enfoque procedurales/funcional- definiendo una función por cada operación con el uso de la función proporcionando un caso por cada variación de dato - y (2) el enfoque orientado a objetos - definiendo una clase por cada variación de dato con la definición de la clase proporcionando un método por cada operación. Entender ambos enfoques como una definición de variaciones y operaciones de una matriz [Evaluar]
- Explicar la relación entre la herencia orientada a objetos (código compartido y *overriding*) y subtipificación (la idea de un subtipo es ser utilizable en un contexto en el que espera al supertipo) [Familiarizarse]
- Usar mecanismos de encapsulación orientada a objetos, tal como interfaces y miembros privados [Usar]
- Definir y usar iteradores y otras operaciones sobre agregaciones, incluyendo operaciones que tienen funciones como argumentos, en múltiples lenguajes de programación, seleccionar la forma más natural por cada lenguaje [Usar]

Lecturas: Stroustrup2013, Deitel17

UNIDAD 5: Algoritmos y Diseño (3)**Competencias:****Contenido**

- Estrategias de solución de problemas
 - Funciones matemáticas iterativas y recursivas
 - Recorrido iterativo y recursivo en estructura de datos
 - Estrategias Divide y Conquistar
- Rol de los algoritmos en el proceso de solución de problemas
- Estrategias de solución de problemas
 - Funciones matemáticas iterativas y recursivas
 - Recorrido iterativo y recursivo en estructura de datos
 - Estrategias Divide y Conquistar
- Conceptos y principios fundamentales de diseño
 - Abstracción
 - Descomposición de Program
 - Encapsulamiento y camuflaje de información
 - Separación de comportamiento y aplicación

Objetivos Generales

- Discute la importancia de los algoritmos en el proceso de solución de un problema [Familiarizarse]
- Discute como un problema puede ser resuelto por múltiples algoritmos, cada uno con propiedades diferentes [Familiarizarse]
- Crea algoritmos para resolver problemas simples [Usar]
- Usa un lenguaje de programación para implementar, probar, y depurar algoritmos para resolver problemas simples [Usar]
- Implementa, prueba, y depura funciones recursivas simples y sus procedimientos [Usar]
- Determina si una solución iterativa o recursiva es la más apropiada para un problema [Evaluar]
- Implementa un algoritmo de divide y vencerás para resolver un problema [Usar]
- Aplica técnicas de descomposición para dividir un programa en partes más pequeñas [Usar]
- Identifica los componentes de datos y el comportamiento de múltiples tipos de datos abstractos [Usar]
- Implementa un tipo de dato abstracto coherente, con la menor pérdida de acoplamiento entre componentes y comportamientos [Usar]
- Identifica las fortalezas y las debilidades relativas entre múltiples diseños e implementaciones de un problema [Evaluar]

Lecturas: Stroustrup2013, Deitel17

UNIDAD 6: Estrategias Algorítmicas (3)	
Competencias:	
Contenido	Objetivos Generales
<ul style="list-style-type: none"> • Algoritmos de fuerza bruta. • Algoritmos voraces. • Divide y vencerás. • Backtracking recursivo. • Programación Dinámica. 	<ul style="list-style-type: none"> • Para cada una de las estrategias (fuerza bruta, algoritmo goloso, divide y vencerás, recursividad en reversa y programación dinámica), identifica un ejemplo práctico en el cual se pueda aplicar [Familiarizarse] • Utiliza un enfoque voraz para resolver un problema específico y determina si la regla escogida lo guía a una solución óptima [Evaluar] • Usa un algoritmo de divide-y-vencerás para resolver un determinado problema [Usar] • Usa recursividad en reversa a fin de resolver un problema como en el caso de recorrer un laberinto [Usar] • Usa programación dinámica para resolver un problema determinado [Usar] • Determina el enfoque algorítmico adecuado para un problema [Evaluar] • Describe varios métodos basados en heurísticas para resolver problemas [Familiarizarse]
Lecturas: Stroustrup2013, Deitel17	

UNIDAD 7: Análisis Básico (2)	
Competencias:	
Contenido	Objetivos Generales
<ul style="list-style-type: none"> • Diferencias entre el mejor, el esperado y el peor caso de un algoritmo. 	<ul style="list-style-type: none"> • Explique a que se refiere con “mejor”, “esperado” y “peor” caso de comportamiento de un algoritmo [Familiarizarse]
Lecturas: Stroustrup2013, Deitel17	

UNIDAD 8: Algoritmos y Estructuras de Datos fundamentales (6)	
Competencias:	
Contenido	Objetivos Generales
<ul style="list-style-type: none"> • Algoritmos numéricos simples, tales como el cálculo de la media de una lista de números, encontrar el mínimo y máximo. • Algoritmos de búsqueda secuencial y binaria. • Algoritmos de ordenamiento de peor caso cuadrático (selección, inserción) • Algoritmos de ordenamiento con peor caso o caso promedio en $O(N \lg N)$ (Quicksort, Heapsort, Mergesort) 	<ul style="list-style-type: none"> • Implementar algoritmos numéricos básicos [Usar] • Implementar algoritmos de búsqueda simple y explicar las diferencias en sus tiempos de complejidad [Evaluar] • Ser capaz de implementar algoritmos de ordenamiento comunes cuadráticos y $O(N \log N)$ [Usar] • Discutir el tiempo de ejecución y eficiencia de memoria de los principales algoritmos de ordenamiento, búsqueda y hashing [Familiarizarse] • Discutir factores otros que no sean eficiencia computacional que influyan en la elección de algoritmos, tales como tiempo de programación, mantenibilidad, y el uso de patrones específicos de la aplicación en los datos de entrada [Familiarizarse] • Explicar como el balanceamiento del arbol afecta la eficiencia de varias operaciones de un arbol de búsqueda binaria [Familiarizarse] • Demostrar habilidad para evaluar algoritmos, para seleccionar de un rango de posibles opciones, para proveer una justificación por esa selección, y para implementar el algoritmo en un contexto en específico [Evaluar] • Trazar y/o implementar un algoritmo de comparación de string [Usar]
Lecturas: Stroustrup2013, Deitel17	

8. Metodología
<p>El profesor del curso presentará clases teóricas de los temas señalados en el programa propiciando la intervención de los alumnos.</p> <p>El profesor del curso presentará demostraciones para fundamentar clases teóricas.</p> <p>El profesor y los alumnos realizarán prácticas</p> <p>Los alumnos deberán asistir a clase habiendo leído lo que el profesor va a presentar. De esta manera se facilitará la comprensión y los estudiantes estarán en mejores condiciones de hacer consultas en clase.</p>

9. Evaluar
<p>Evaluación Continua 1 : 20 %</p> <p>Examen parcial : 30 %</p> <p>Evaluación Continua 2 : 20 %</p> <p>Examen final : 30 %</p>



Universidad Nacional de Colombia (UNAL) Sede
Manizales
Programa Profesional de
Administración de Sistemas Informáticos
SILABO

CS1D2. Estructuras Discretas II (Obligatorio)

2022-II

1. Información general	
1.1 Escuela	: Sistemas de Información
1.2 Curso	: CS1D2. Estructuras Discretas II
1.3 Semestre	: 2 ^{do} Semestre.
1.4 Prerrequisitos	: CS1D1. Estructuras Discretas I. (1 ^{er} Sem)
1.5 Condición	: Obligatorio
1.6 Modalidad de aprendizaje	: Presencial
1.7 horas	: 2 HT; 2 HP; 2 HL;
1.8 Créditos	: 4
2. Profesores	
3. Fundamentación del curso	
Para entender las técnicas computacionales avanzadas, los estudiantes deberán tener un fuerte conocimiento de las diversas estructuras discretas, estructuras que serán implementadas y usadas en laboratorio en el lenguaje de programación.	
4. Resumen	
1. Lógica Digital y Representación de Datos 2. Fundamentos de conteo 3. Árboles y Grafos	
5. Objetivos Generales	
<ul style="list-style-type: none">• Que el alumno sea capaz de modelar problemas de ciencia de la computación usando grafos y árboles relacionados con estructuras de datos.• Que el alumno aplique eficientemente estrategias de recorrido para poder buscar datos de una manera óptima.• Que el alumno utilice las diversas técnicas de conteo para resolver problemas computacionales.	
6. Contribución a los resultados (Outcomes)	
Esta disciplina contribuye al logro de los siguientes resultados de la carrera:	
1) Analizar un problema computacional complejo y aplicar los principios computacionales y otras disciplinas relevantes para identificar soluciones. (Familiarizarse)	
6) Aplicar fundamentos de teoría de ciencias de la computación y desarrollo de software para producir soluciones basados en computación. (Familiarizarse)	
7. Contenido	

UNIDAD 1: Lógica Digital y Representación de Datos (10)**Competencias:****Contenido**

- Retículo: Tipos y propiedades.
- Álgebras booleanas.
- Funciones y expresiones booleanas.
- Representación de las funciones booleanas: Disjuntiva normal y conjuntiva normal.
- Puertas Lógicas.
- Minimización del Circuito.

Objetivos Generales

- Explicar la importancia del álgebra booleana como una unificación de la teoría de conjuntos y la lógica proposicional [Evaluar].
- Explicar las estructuras algebraicas del retículo y sus tipos [Evaluar].
- Explicar la relación entre el retículo y el conjunto de ordenadas y el uso prudente para demostrar que un conjunto es un retículo [Evaluar].
- Explicar las propiedades que satisfacen un álgebra booleana [Evaluar].
- Demostrar si una terna formada por un conjunto y dos operaciones internas es o no Álgebra booleana [Evaluar].
- Encuentra las formas canónicas de una función booleana [Evaluar].
- Representar una función booleana como un circuito booleano usando puertas lógicas [Evaluar].
- Minimizar una función booleana [Evaluar].

Lecturas: Rosen (2007), Grimaldi (2003)

UNIDAD 2: Fundamentos de conteo (40)	
Competencias:	
Contenido	Objetivos Generales
<ul style="list-style-type: none"> • Técnicas de Conteo: <ul style="list-style-type: none"> – Conteo y cardinalidad de un conjunto – Regla de la suma y producto – Principio de inclusión-exclusión – Progresión geométrica y aritmética • Principio de las casillas. • Permutaciones y combinaciones: <ul style="list-style-type: none"> – Definiciones básicas – Identidad de Pascal – Teorema del binomio • Resolviendo relaciones de recurrencia: <ul style="list-style-type: none"> – Un ejemplo de una relación de recurrencia simple, como los números de Fibonacci – Otras ejemplos, mostrando una variedad de soluciones • Aritmetica modular basica 	<ul style="list-style-type: none"> • Aplicar argumentos de conteo, incluyendo las reglas del producto y de la suma, principio de inclusión-exclusión y progresiones aritméticas/geométricas [Familiarizarse] • Aplicar el principio de las casillas en el contexto de una demostración formal [Familiarizarse] • Calcular permutaciones y combinaciones en un conjunto, e interpreta su significado en el contexto de una aplicación en particular [Familiarizarse] • Mapear aplicaciones del mundo real a formalismos de conteo adecuados, como el determinar el número de formas de acomodar a un conjunto de personas alrededor de una mesa, sujeto a restricciones en la disposición de los asientos, o en el número de maneras de determinar ciertas manos en juegos de cartas (ejm. una casa llena) [Familiarizarse] • Resolver una variedad de relaciones de recurrencia básicas [Familiarizarse] • Analizar un problema para determinar las relaciones de recurrencia implícitas [Familiarizarse] • Realizar cálculos que involucran aritmética modular [Familiarizarse]
Lecturas: Grimaldi (2003)	

UNIDAD 3: Árboles y Grafos (40)	
Competencias:	
Contenido	Objetivos Generales
<ul style="list-style-type: none"> • Árboles. <ul style="list-style-type: none"> – Propiedades – Estrategias de recorrido • Grafos no dirigidos • Grafos dirigidos • Grafos ponderados • Árboles de expansión/bosques. • Isomorfismo en grafos. 	<ul style="list-style-type: none"> • Ilustrar mediante ejemplos la terminología básica de teoría de grafos, y de alguna de las propiedades y casos especiales de cada tipo de grafos/árboles [Familiarizarse] • Demostrar diversos métodos de recorrer árboles y grafos, incluyendo recorridos pre, post e inorden de árboles [Familiarizarse] • Modelar una variedad de problemas del mundo real en ciencia de la computación usando formas adecuadas de grafos y árboles, como son la representación de una topología de red o la organización jerárquica de un sistema de archivos [Familiarizarse] • Demostrar como los conceptos de grafos y árboles aparecen en estructuras de datos, algoritmos, técnicas de prueba (inducción estructurada), y conteos [Familiarizarse] • Explicar como construir un árbol de expansión de un grafo [Familiarizarse] • Determinar si dos grafos son isomorfos [Familiarizarse]
Lecturas: Johnsonbaugh (1999)	

8. Metodología
<p>El profesor del curso presentará clases teóricas de los temas señalados en el programa propiciando la intervención de los alumnos.</p> <p>El profesor del curso presentará demostraciones para fundamentar clases teóricas.</p> <p>El profesor y los alumnos realizarán prácticas</p> <p>Los alumnos deberán asistir a clase habiendo leído lo que el profesor va a presentar. De esta manera se facilitará la comprensión y los estudiantes estarán en mejores condiciones de hacer consultas en clase.</p>

9. Evaluar
<p>Evaluación Continua 1 : 20 %</p> <p>Examen parcial : 30 %</p> <p>Evaluación Continua 2 : 20 %</p> <p>Examen final : 30 %</p>

References

- Grimaldi, R. (2003). *Discrete and Combinatorial Mathematics: An Applied Introduction*. 5 ed. Pearson.
- Johnsonbaugh, Richard (1999). *Matemáticas Discretas*. Prentice Hall, México.
- Rosen, Kenneth H. (2007). *Discrete Mathematics and Its Applications*. 7 ed. Mc Graw Hill.



Universidad Nacional de Colombia (UNAL) Sede
Manizales
Programa Profesional de
Administración de Sistemas Informáticos
SILABO

MA101. Matemática II (Obligatorio)

2022-II

1. Información general	
1.1 Escuela	: Sistemas de Información
1.2 Curso	: MA101. Matemática II
1.3 Semestre	: 2 ^{do} Semestre.
1.4 Prerrequisitos	: MA100. Matemática I. (1 ^{er} Sem)
1.5 Condición	: Obligatorio
1.6 Modalidad de aprendizaje	: Presencial
1.7 horas	: 2 HT; 4 HP;
1.8 Créditos	: 4
2. Profesores	
3. Fundamentación del curso	
El curso desarrolla en los estudiantes las habilidades para manejar modelos de habilidades de ingeniería y ciencia. En la primera parte Del curso un estudio de las funciones de varias variables, derivadas parciales, integrales múltiples y una Introducción a campos vectoriales. Luego el estudiante utilizará los conceptos básicos de cálculo para modelar y resolver ecuaciones diferenciales ordinarias utilizando técnicas como las transformadas de Laplace y las series de Fourier.	
4. Resumen	
1. Multi-Variable Function Differential 2. Multi-Variable function Integral 3. Series 4. Ordinary Differential Equations	
5. Objetivos Generales	
<ul style="list-style-type: none">• Aplicar reglas de derivación y diferenciación parcial en funciones de varias variables.• Aplicar técnicas para el cálculo de integrales múltiples.• Comprender y utilizar los conceptos de cálculo vectorial.• Comprender la importancia de las series.• Identificar y resolver ecuaciones diferenciales de primer orden y sus aplicaciones en problemas químicos y físicos.	
6. Contribución a los resultados (Outcomes)	
Esta disciplina contribuye al logro de los siguientes resultados de la carrera:	
1) Analizar un problema computacional complejo y aplicar los principios computacionales y otras disciplinas relevantes para identificar soluciones. (Evaluar)	
6) Aplicar fundamentos de teoría de ciencias de la computación y desarrollo de software para producir soluciones basados en computación. (Evaluar)	

7. Contenido

UNIDAD 1: Multi-Variable Function Differential (24)

Competencias:

Contenido

- Concepto de funciones multi-variables.
- Derivados Direccionales
- Línea tangente, plano normal a línea de curva y plano tangente, línea normal a un plano de curva. Conocer para calcular sus ecuaciones.
- Concepto de valor extremo y valor extremo condicional de funciones multi-variables.
- Problemas de aplicación tales como modelización de la producción total de un sistema económico, velocidad del sonido a través del océano, optimización del espesante, etc.

Objetivos Generales

- Comprender el concepto de funciones multi-variables.
- Dominar el concepto y método de cálculo de la derivada direccional y gradiente de la guía.
- Dominar el método de cálculo de la derivada parcial de primer orden y de segundo orden de las funciones compuestas.
- Dominar línea tangente, plano normal a línea de curva y plano tangente, línea normal a un plano de curva. Saber calcular sus ecuaciones.
- Dominar el método de las derivadas parciales para funciones implícitas.
- Entender línea tangente, plano normal a línea de curva y plano tangente, línea normal a un plano de curva. Saber calcular sus ecuaciones.
- Aprender el concepto de valor extremo y valor extremo condicional de funciones multi-variables; Saber para averiguar el valor extremo de la función binaria.
- Ser capaz de resolver problemas de aplicaciones simples.

Lecturas: Stewart (2012), Zill (2013)

UNIDAD 2: Multi-Variable function Integral (12)

Competencias:

Contenido

- Integral doble, integral triple y naturaleza de la integral múltiple.
- Método de doble integral
- Línea integral
- La Divergencia, Rotación y Laplaciano

Objetivos Generales

- Entender la integral doble, integral triple, y entender la naturaleza de la integral múltiple.
- Dominar el método de cálculo de la integral doble (coordenadas cartesianas, coordenadas polares), la integral triple (coordenadas cartesianas, coordenadas cilíndricas, coordenadas esféricas).
- Entender el concepto de línea Integral, sus propiedades y relaciones.
- Saber calcular la integral de línea.
- Dominar el cálculo de la rotación, la divergencia y Laplaciano.

Lecturas: Stewart (2012), Zill (2013)

UNIDAD 3: Series (24)	
Competencias:	
Contenido	Objetivos Generales
<ul style="list-style-type: none"> • Serie convergente. • Serie Taylor y MacLaurin. • Funciones ortogonales. 	<ul style="list-style-type: none"> • Dominio del cálculo si la serie es convergente, y si es convergente, encontrar la suma de la serie tratando de encontrar el radio de convergencia y el intervalo de convergencia de una serie de potencia. • Representa una función como una serie de potencias y encuentra la serie de Taylor y MacLaurin para estimar los valores de las funciones con la precisión deseada. • Entender los conceptos de funciones ortogonales y la expansión de una función dada f para encontrar su serie de Fourier.
Lecturas: Stewart (2012), Zill (2013)	

UNIDAD 4: Ordinary Differential Equations (30)	
Competencias:	
Contenido	Objetivos Generales
<ul style="list-style-type: none"> • Concepto de ecuaciones diferenciales • Métodos para resolver ecuaciones diferenciales • Métodos para resolver las ecuaciones diferenciales lineales de segundo orden • Ecuaciones diferenciales ordinarias lineales de orden superior • Problemas de aplicaciones con las transformaciones de Laplace 	<ul style="list-style-type: none"> • Comprender ecuaciones diferenciales, soluciones, orden, solución general, condiciones iniciales y soluciones especiales, etc. • Dominar el método de cálculo para las variables ecuación separable y ecuaciones lineales de primer orden. Conocido para resolver la ecuación homogénea y las ecuaciones de Bernoulli (Bernoulli); Entender la sustitución de la variable para resolver la ecuación. • Diminuo para resolver ecuaciones diferenciales totales. • Ser capaz de utilizar el método de orden reducido para resolver ecuaciones. • Comprender la estructura de la ecuación diferencial lineal de segundo orden. • Dominio del cálculo para las ecuaciones diferenciales lineales homogéneas de coeficiente constante; Y comprender el método de cálculo para las ecuaciones diferenciales lineales homogéneas de orden superior. • Saber aplicar el método de cálculo de ecuaciones diferenciales para resolver problemas simples de aplicación geométrica y física. • Resolver correctamente ciertos tipos de ecuaciones diferenciales utilizando transformadas de Laplace.
Lecturas: Stewart (2012), Zill (2013)	

8. Metodología

El profesor del curso presentará clases teóricas de los temas señalados en el programa propiciando la intervención de los alumnos.

El profesor del curso presentará demostraciones para fundamentar clases teóricas.

El profesor y los alumnos realizarán prácticas

Los alumnos deberán asistir a clase habiendo leído lo que el profesor va a presentar. De esta manera se facilitará la comprensión y los estudiantes estarán en mejores condiciones de hacer consultas en clase.

9. Evaluar

Evaluación Continua 1 : 20 %

Examen parcial : 30 %

Evaluación Continua 2 : 20 %

Examen final : 30 %

References

Stewart, James (2012). *Calculus*. 7th. CENGAGE Learning.

Zill, Dennis G. (2013). *Differential equations with Boundary value problems*. 8th. CENGAGE Learning.



Universidad Nacional de Colombia (UNAL) Sede
Manizales
Programa Profesional de
Administración de Sistemas Informáticos
SILABO

CS113. Ciencia de la Computación II (Obligatorio)

2022-II

1. Información general	
1.1 Escuela	: Sistemas de Información
1.2 Curso	: CS113. Ciencia de la Computación II
1.3 Semestre	: 3 ^{er} Semestre.
1.4 Prerrequisitos	: <ul style="list-style-type: none">• CS112. Ciencia de la Computación I. (2^{do} Sem)• CS112. Ciencia de la Computación I. (2^{do} Sem)
1.5 Condición	: Obligatorio
1.6 Modalidad de aprendizaje	: Presencial
1.7 horas	: 2 HT; 2 HP; 4 HL;
1.8 Créditos	: 5
2. Profesores	
3. Fundamentación del curso	
Este es el tercer curso en la secuencia de los cursos introductorios a la informática. En este curso se pretende cubrir los conceptos señalados por la Computing Curricula IEEE(c)-ACM 2001, bajo el enfoque funcional-first. El paradigma orientado a objetos nos permite combatir la complejidad haciendo modelos a partir de abstracciones de los elementos del problema y utilizando técnicas como encapsulamiento, modularidad, polimorfismo y herencia. El dominio de estos temas permitirá que los participantes puedan dar soluciones computacionales a problemas de diseño sencillos del mundo real.	
4. Resumen	
1. Conceptos Fundamentales de Programación 2. Programación orientada a objetos 3. Algoritmos y Diseño 4. Análisis Básico 5. Sistemas de tipos básicos 6. Algoritmos y Estructuras de Datos fundamentales 7. Programación reactiva y dirigida por eventos 8. Árboles y Grafos 9. Diseño de Software 10. Ingeniería de Requisitos	
5. Objetivos Generales	
<ul style="list-style-type: none">• Introducir al alumno a los fundamentos del paradigma de orientación a objetos, permitiendo asimilar los conceptos necesarios para desarrollar un sistema de información	

6. Contribución a los resultados (*Outcomes*)

Esta disciplina contribuye al logro de los siguientes resultados de la carrera:

- 1) Analizar un problema computacional complejo y aplicar los principios computacionales y otras disciplinas relevantes para identificar soluciones. (**Usar**)
- 3) Comunicarse efectivamente en diversos contextos profesionales. (**Usar**)
- 5) Funcionar efectivamente como miembro o líder de un equipo involucrado en actividades apropiadas a la disciplina del programa. (**Usar**)

7. Contenido

UNIDAD 1: Conceptos Fundamentales de Programación (5)

Competencias:

Contenido	Objetivos Generales
<ul style="list-style-type: none">• Sintaxis y semántica básica de un lenguaje de alto nivel.• Variables y tipos de datos primitivos (ej., números, caracteres, booleanos)• Expresiones y asignaciones.• Operaciones básicas I/O incluyendo archivos I/O.• Estructuras de control condicional e iterativas.• Paso de funciones y parámetros.• Concepto de recursividad.	<ul style="list-style-type: none">• Analiza y explica el comportamiento de programas simples que involucran estructuras fundamentales de programación variables, expresiones, asignaciones, E/S, estructuras de control, funciones, paso de parámetros, y recursividad [Usar]• Identifica y describe el uso de tipos de datos primitivos [Usar]• Escribe programas que usan tipos de datos primitivos [Usar]• Modifica y expande programas cortos que usen estructuras de control condicionales e iterativas así como funciones [Usar]• Diseña, implementa, prueba, y depura un programa que usa cada una de las siguientes estructuras de datos fundamentales: cálculos básicos, E/S simple, condicional estándar y estructuras iterativas, definición de funciones, y paso de parámetros [Usar]• Escribe un programa que usa E/S de archivos para brindar persistencia a través de ejecuciones múltiples [Usar]• Escoje estructuras de condición y repetición adecuadas para una tarea de programación dada [Usar]• Describe el concepto de recursividad y da ejemplos de su uso [Usar]• Identifica el caso base y el caso general de un problema basado en recursividad [Usar]

Lecturas: stroustrup2013, Vandervoorde (2002), Lippman and E.Moo (2013)

UNIDAD 2: Programación orientada a objetos (7)	
Competencias:	
Contenido	Objetivos Generales
<ul style="list-style-type: none"> • Diseño orientado a objetos: <ul style="list-style-type: none"> – Descomposición en objetos que almacenan estados y poseen comportamiento – Diseño basado en jerarquía de clases para modelamiento • Definición de las categorías, campos, métodos y constructores. • Las subclases, herencia y método de alteración temporal. • Asignación dinámica: definición de método de llamada. • Subtipificación: <ul style="list-style-type: none"> – Polimorfismo artículo Subtipo; upcasts implícitos en lenguajes con tipos. – Noción de reemplazo de comportamiento: los subtipos de actuar como supertipos. – Relación entre subtipos y la herencia. • Lenguajes orientados a objetos para la encapsulación: <ul style="list-style-type: none"> – privacidad y la visibilidad de miembros de la clase – Interfaces revelan único método de firmas – clases base abstractas • Uso de colección de clases, iteradores, y otros componentes de la librería estándar. 	<ul style="list-style-type: none"> • Diseñar e implementar una clase [Usar] • Usar subclase para diseñar una jerarquía simple de clases que permita al código ser reusable por diferentes subclases [Usar] • Razonar correctamente sobre el flujo de control en un programa mediante el envío dinámico [Usar] • Comparar y contrastar (1) el enfoque procedurador/funcional- definiendo una función por cada operación con el uso de la función proporcionando un caso por cada variación de dato - y (2) el enfoque orientado a objetos - definiendo una clase por cada variación de dato con la definición de la clase proporcionando un método por cada operación. Entender ambos enfoques como una definición de variaciones y operaciones de una matriz [Usar] • Explicar la relación entre la herencia orientada a objetos (código compartido y <i>overriding</i>) y subtipificación (la idea de un subtipo es ser utilizable en un contexto en el que espera al supertipo) [Usar] • Usar mecanismos de encapsulación orientada a objetos, tal como interfaces y miembros privados [Usar] • Definir y usar iteradores y otras operaciones sobre agregaciones, incluyendo operaciones que tienen funciones como argumentos, en múltiples lenguajes de programación, seleccionar la forma más natural por cada lenguaje [Usar]
Lecturas: Stroustrup (2013)	

UNIDAD 3: Algoritmos y Diseño (5)**Competencias:****Contenido****Objetivos Generales**

- Conceptos y propiedades de los algoritmos
 - Comparación informal de la eficiencia de los algoritmos (ej., conteo de operaciones)
- Rol de los algoritmos en el proceso de solución de problemas
- Estrategias de solución de problemas
 - Funciones matemáticas iterativas y recursivas
 - Recorrido iterativo y recursivo en estructura de datos
 - Estrategias Divide y Conquistar
- Conceptos y principios fundamentales de diseño
 - Abstracción
 - Descomposición de Program
 - Encapsulamiento y camuflaje de información
 - Separación de comportamiento y aplicación

- Discute la importancia de los algoritmos en el proceso de solución de un problema [Usar]
- Discute como un problema puede ser resuelto por múltiples algoritmos, cada uno con propiedades diferentes [Usar]
- Crea algoritmos para resolver problemas simples [Usar]
- Usa un lenguaje de programación para implementar, probar, y depurar algoritmos para resolver problemas simples [Usar]
- Implementa, prueba, y depura funciones recursivas simples y sus procedimientos [Usar]
- Determina si una solución iterativa o recursiva es la más apropiada para un problema [Usar]
- Implementa un algoritmo de divide y vencerás para resolver un problema [Usar]
- Aplica técnicas de descomposición para dividir un programa en partes más pequeñas [Usar]
- Identifica los componentes de datos y el comportamiento de múltiples tipos de datos abstractos [Usar]
- Implementa un tipo de dato abstracto coherente, con la menor pérdida de acoplamiento entre componentes y comportamientos [Usar]
- Identifica las fortalezas y las debilidades relativas entre múltiples diseños e implementaciones de un problema [Usar]

Lecturas: stroustrup2013, Weert16, Lippman and E.Moo (2013)

UNIDAD 4: Análisis Básico (3)**Competencias:****Contenido****Objetivos Generales**

- Diferencias entre el mejor, el esperado y el peor caso de un algoritmo.
- Análisis asintótico de complejidad de cotas superior y esperada.
- Definición formal de la Notación Big O.
- Clases de complejidad como constante, logarítmica, lineal, cuadrática y exponencial.
- Medidas empíricas de desempeño.
- Compensación entre espacio y tiempo en los algoritmos.
- Uso de la notación Big O.
- Notación Little o, Big omega y Big theta.
- Relaciones recurrentes.
- Análisis de algoritmos iterativos y recursivos.
- Teorema Maestro y Árboles Recursivos.

- Explique a que se refiere con “mejor”, “esperado” y “peor” caso de comportamiento de un algoritmo [Usar]
- En el contexto de a algoritmos específicos, identifique las características de data y/o otras condiciones o suposiciones que lleven a diferentes comportamientos [Usar]
- Determine informalmente el tiempo y el espacio de complejidad de diferentes algoritmos [Usar]
- Indique la definición formal de Big O [Usar]
- Lista y contraste de clases estándares de complejidad [Usar]
- Realizar estudios empíricos para validar una hipótesis sobre runtime stemming desde un análisis matemático Ejecute algoritmos con entrada de varios tamaños y compare el desempeño [Usar]
- Da ejemplos que ilustran las compensaciones entre espacio y tiempo que se dan en los algoritmos [Usar]
- Use la notación formal de la Big O para dar límites superiores asintóticos en la complejidad de tiempo y espacio de los algoritmos [Usar]
- Usar la notación formal Big O para dar límites de casos esperados en el tiempo de complejidad de los algoritmos [Usar]
- Explicar el uso de la notación theta grande, omega grande y o pequeña para describir la cantidad de trabajo hecho por un algoritmo [Usar]
- Usar relaciones recurrentes para determinar el tiempo de complejidad de algoritmos recursivamente definidos [Usar]
- Resuelve relaciones de recurrencia básicas, por ejemplo. usando alguna forma del Teorema Maestro [Usar]

Lecturas: Stroustrup (2013)

UNIDAD 5: Sistemas de tipos básicos (5)	
Competencias:	
Contenido	Objetivos Generales
<ul style="list-style-type: none"> • Tipos como conjunto de valores junto con un conjunto de operaciones. <ul style="list-style-type: none"> – Tipos primitivos (p.e. números, booleanos) – Composición de tipos contruídos de otros tipos (p.e., registros, uniones, arreglos, listas, funciones, referencias) • Asociación de tipos de variables, argumentos, resultados y campos. • Tipo de seguridad y los errores causados por el uso de valores de manera incompatible dadas sus tipos previstos. • Metas y limitaciones de tipos estáticos <ul style="list-style-type: none"> – Eliminación de algunas clases de errores sin ejecutar el programa – Indecisión significa que un análisis estatico puede aproximar el comportamiento de un programa • Tipos genéricos (polimorfismo paramétrico) <ul style="list-style-type: none"> – Definición – Uso de librerías genéricas tales como colecciones. – Comparación con polimorfismo ad-hoc y polimorfismo de subtipos • Beneficios complementarios de tipos estáticos y dinámicos: <ul style="list-style-type: none"> – Errores tempranos vs. errores tardíos/evitados. – Refuerzo invariante durante el desarrollo y mantenimiento del código vs. decisiones pospuestas de tipos durante la la creación de prototipos y permitir convenientemente la codificación flexible de patrones tales como colecciones heterogéneas. – Evitar el mal uso del código vs. permitir más reuso de código. – Detectar programas incompletos vs. permitir que programas incompletos se ejecuten 	<ul style="list-style-type: none"> • Tanto para tipo primitivo y un tipo compuesto, describir de manera informal los valores que tiene dicho tipo [Usar] • Para un lenguaje con sistema de tipos estático, describir las operaciones que están prohibidas de forma estática, como pasar el tipo incorrecto de valor a una función o método [Usar] • Describir ejemplos de errores de programa detectadas por un sistema de tipos [Usar] • Para múltiples lenguajes de programación, identificar propiedades de un programa con verificación estática y propiedades de un programa con verificación dinámica [Usar] • Dar un ejemplo de un programa que no verifique tipos en un lenguaje particular y sin embargo no tenga error cuando es ejecutado [Usar] • Usar tipos y mensajes de error de tipos para escribir y depurar programas [Usar] • Explicar como las reglas de tipificación definen el conjunto de operaciones que legales para un tipo [Usar] • Escribir las reglas de tipo que rigen el uso de un particular tipo compuesto [Usar] • Explicar por qué indecidibilidad requiere sistemas de tipo para conservadoramente aproximar el comportamiento de un programa [Usar] • Definir y usar piezas de programas (tales como, funciones, clases, métodos) que usan tipos genéricos, incluyendo para colecciones [Usar] • Discutir las diferencias entre, genéricos (<i>generics</i>), subtipo y sobrecarga [Usar] • Explicar múltiples beneficios y limitaciones de tipificación estática en escritura, mantenimiento y depuración de un software [Usar]
Lecturas: Stroustrup (2013)	

UNIDAD 6: Algoritmos y Estructuras de Datos fundamentales (3)**Competencias:****Contenido****Objetivos Generales**

- Algoritmos numéricos simples, tales como el cálculo de la media de una lista de números, encontrar el mínimo y máximo.
- Algoritmos de búsqueda secuencial y binaria.
- Algoritmos de ordenamiento de peor caso cuadrático (selección, inserción)
- Algoritmos de ordenamiento con peor caso o caso promedio en $O(N \lg N)$ (Quicksort, Heapsort, Mergesort)
- Tablas Hash, incluyendo estrategias para evitar y resolver colisiones.
- Árboles de búsqueda binaria:
 - Operaciones comunes en árboles de búsqueda binaria como seleccionar el mínimo, máximo, insertar, eliminar, recorrido en árboles.
- Grafos y algoritmos en grafos:
 - Representación de grafos (ej., lista de adyacencia, matriz de adyacencia)
 - Recorrido en profundidad y amplitud
- Montículos (Heaps)
- Grafos y algoritmos en grafos:
 - Problema de corte máximo y mínimo
 - Búsqueda local
- Búsqueda de patrones y algoritmos de cadenas/texto (ej. búsqueda de subcadena, búsqueda de expresiones regulares, algoritmos de subsecuencia común más larga)

- Implementar algoritmos numéricos básicos [Usar]
- Implementar algoritmos de búsqueda simple y explicar las diferencias en sus tiempos de complejidad [Usar]
- Ser capaz de implementar algoritmos de ordenamiento comunes cuadráticos y $O(N \log N)$ [Usar]
- Describir la implementación de tablas hash, incluyendo resolución y el evitamiento de colisiones [Usar]
- Discutir el tiempo de ejecución y eficiencia de memoria de los principales algoritmos de ordenamiento, búsqueda y hashing [Usar]
- Discutir factores otros que no sean eficiencia computacional que influyan en la elección de algoritmos, tales como tiempo de programación, mantenibilidad, y el uso de patrones específicos de la aplicación en los datos de entrada [Usar]
- Explicar como el balanceamiento del arbol afecta la eficiencia de varias operaciones de un arbol de búsqueda binaria [Usar]
- Resolver problemas usando algoritmos básicos de grafos, incluyendo búsqueda por profundidad y búsqueda por amplitud [Usar]
- Demostrar habilidad para evaluar algoritmos, para seleccionar de un rango de posibles opciones, para proveer una justificación por esa selección, y para implementar el algoritmo en un contexto en específico [Usar]
- Describir la propiedad del heap y el uso de heaps como una implementación de colas de prioridad [Usar]
- Resolver problemas usando algoritmos de grafos, incluyendo camino más corto de una sola fuente y camino más corto de todos los pares, y como mínimo un algoritmo de arbol de expansion minima [Usar]
- Trazar y/o implementar un algoritmo de comparación de string [Usar]

Lecturas: stroustrup2013, Pai and Abraham (2018)

UNIDAD 7: Programación reactiva y dirigida por eventos (2)	
Competencias:	
Contenido	Objetivos Generales
<ul style="list-style-type: none"> • Eventos y controladores de eventos. • Usos canónicos como interfaces gráficas de usuario, dispositivos móviles, robots, servidores. • Uso de frameworks reactivos. <ul style="list-style-type: none"> – Definición de controladores/oyentes (handles/listeners) de eventos. – Bucle principal de eventos no controlado por el escritor controlador de eventos (event-handler-writer) • Eventos y eventos del programa generados externamente generada. • La separación de modelo, vista y controlador. 	<ul style="list-style-type: none"> • Escribir manejadores de eventos para su uso en sistemas reactivos tales como GUIs [Usar] • Explicar porque el estilo de programación manejada por eventos es natural en dominios donde el programa reacciona a eventos externos [Usar] • Describir un sistema interactivo en términos de un modelo, una vista y un controlador [Usar]
Lecturas: stroustrup2013, Williams (2011)	

UNIDAD 8: Árboles y Grafos (7)	
Competencias:	
Contenido	Objetivos Generales
<ul style="list-style-type: none"> • Árboles. <ul style="list-style-type: none"> – Propiedades – Estrategias de recorrido • Grafos no dirigidos • Grafos dirigidos • Grafos ponderados • Árboles de expansion/bosques. • Isomorfismo en grafos. 	<ul style="list-style-type: none"> • Ilustrar mediante ejemplos la terminología básica de teoría de grafos, y de alguna de las propiedades y casos especiales de cada tipo de grafos/árboles [Usar] • Demostrar diversos métodos de recorrer árboles y grafos, incluyendo recorridos pre, post e inorden de árboles [Usar] • Modelar una variedad de problemas del mundo real en ciencia de la computación usando formas adecuadas de grafos y árboles, como son la representación de una topología de red o la organización jerárquica de un sistema de archivos [Usar] • Demostrar como los conceptos de grafos y árboles aparecen en estructuras de datos, algoritmos, técnicas de prueba (inducción estructurada), y conteos [Usar] • Explicar como construir un árbol de expansión de un grafo [Usar] • Determinar si dos grafos son isomorfos [Usar]
Lecturas: Nakariakov (2013)	

UNIDAD 9: Diseño de Software (6)**Competencias:****Contenido****Objetivos Generales**

- Principios de diseño del sistema: niveles de abstracción (diseño arquitectónico y el diseño detallado), separación de intereses, ocultamiento de información, de acoplamiento y de cohesión, de reutilización de estructuras estándar.
- Diseño de paradigmas tales como diseño estructurado (descomposición funcional de arriba hacia abajo), el análisis orientado a objetos y diseño, orientado a eventos de diseño, diseño de nivel de componente, centrado datos estructurada, orientada a aspectos, orientado a la función, orientado al servicio.
- Modelos estructurales y de comportamiento de los diseños de software.
- Diseño de patrones.
- Relaciones entre los requisitos y diseños: La transformación de modelos, el diseño de los contratos, invariantes.
- Conceptos de arquitectura de software y arquitecturas estándar (por ejemplo, cliente-servidor, n-capas, transforman centrados, tubos y filtros).
- El uso de componentes de diseño: selección de componentes, diseño, adaptación y componentes de ensamblaje, componentes y patrones, componentes y objetos (por ejemplo, construir una GUI usando un standard widget set)
- Diseños de refactorización utilizando patrones de diseño
- Calidad del diseño interno, y modelos para: eficiencia y desempeño, redundancia y tolerancia a fallos, trazabilidad de los requerimientos.
- Medición y análisis de la calidad de un diseño.
- Compensaciones entre diferentes aspectos de la calidad.
- Aplicaciones en frameworks.
- Middleware: El paradigma de la orientación a objetos con middleware, requerimientos para correr y clasificar objetos, monitores de procesamiento de transacciones y el sistema de flujo de trabajo.
- Principales diseños de seguridad y codificación (cross-reference IAS/Principles of secure design).
 - Principio de privilegios mínimos
 - Principio de falla segura por defecto
 - Principio de aceptabilidad psicológica

- Formular los principios de diseño, incluyendo la separación de problemas, ocultación de información, acoplamiento y cohesión, y la encapsulación [Usar]
- Usar un paradigma de diseño para diseñar un sistema de software básico y explicar cómo los principios de diseño del sistema se han aplicado en este diseño [Usar]
- Construir modelos del diseño de un sistema de software simple los cuales son apropiado para el paradigma utilizado para diseñarlo [Usar]
- En el contexto de un paradigma de diseño simple, describir uno o más patrones de diseño que podrían ser aplicables al diseño de un sistema de software simple [Usar]
- Para un sistema simple adecuado para una situación dada, discutir y seleccionar un paradigma de diseño apropiado [Usar]
- Crear modelos apropiados para la estructura y el comportamiento de los productos de software desde la especificaciones de requisitos [Usar]
- Explicar las relaciones entre los requisitos para un producto de software y su diseño, utilizando los modelos apropiados [Usar]
- Para el diseño de un sistema de software simple dentro del contexto de un único paradigma de diseño, describir la arquitectura de software de ese sistema [Usar]
- Dado un diseño de alto nivel, identificar la arquitectura de software mediante la diferenciación entre las arquitecturas comunes de software, tales como 3 capas (*3-tier*), *pipe-and-filter*, y cliente-servidor [Usar]
- Investigar el impacto de la selección arquitecturas de software en el diseño de un sistema simple [Usar]
- Aplicar ejemplos simples de patrones en un diseño de software [Usar]
- Describir una manera de refactorar y discutir cuando esto debe ser aplicado [Usar]
- Seleccionar componentes adecuados para el uso en un diseño de un producto de software [Usar]
- Explicar cómo los componentes deben ser adaptados para ser usados en el diseño de un producto de software [Usar]
- Diseñar un contrato para un típico componente de software pequeño para el uso de un dado sistema [Usar]
- Discutir y seleccionar la arquitectura de software adecuada para un sistema de software simple para un dado escenario [Usar]

UNIDAD 10: Ingeniería de Requisitos (1)	
Competencias:	
Contenido	Objetivos Generales
<ul style="list-style-type: none"> • Al describir los requisitos funcionales utilizando, por ejemplo, los casos de uso o historias de los usuarios. • Propiedades de requisitos, incluyendo la consistencia, validez, integridad y viabilidad. • Requisitos de software elicitation. • Descripción de datos del sistema utilizando, por ejemplo, los diagramas de clases o diagramas entidad-relación. • Requisitos no funcionales y su relación con la calidad del software. • Evaluación y uso de especificaciones de requisitos. • Requisitos de las técnicas de modelado de análisis. • La aceptabilidad de las consideraciones de certeza/incertidumbre sobre el comportamiento del software/sistema. • Prototipos. • Conceptos básicos de la especificación formal de requisitos. • Especificación de requisitos. • Validación de requisitos. • Rastreo de requisitos. 	<ul style="list-style-type: none"> • Enumerar los componentes clave de un caso de uso o una descripción similar de algún comportamiento que es requerido para un sistema [Usar] • Describir cómo el proceso de ingeniería de requisitos apoya la obtención y validación de los requisitos de comportamiento [Usar] • Interpretar un modelo de requisitos dada por un sistema de software simple [Usar] • Describir los retos fundamentales y técnicas comunes que se utilizan para la obtención de requisitos [Usar] • Enumerar los componentes clave de un modelo de datos (por ejemplo, diagramas de clases o diagramas ER) [Usar] • Identificar los requisitos funcionales y no funcionales en una especificación de requisitos dada por un sistema de software [Usar] • Realizar una revisión de un conjunto de requisitos de software para determinar la calidad de los requisitos con respecto a las características de los buenos requisitos [Usar] • Aplicar elementos clave y métodos comunes para la obtención y el análisis para producir un conjunto de requisitos de software para un sistema de software de tamaño medio [Usar] • Comparar los métodos ágiles y el dirigido por planes para la especificación y validación de requisitos y describir los beneficios y riesgos asociados con cada uno [Usar] • Usar un método común, no formal para modelar y especificar los requisitos para un sistema de software de tamaño medio [Usar] • Traducir al lenguaje natural una especificación de requisitos de software (por ejemplo, un contrato de componentes de software) escrito en un lenguaje de especificación formal [Usar] • Crear un prototipo de un sistema de software para reducir el riesgo en los requisitos [Usar] • Diferenciar entre el rastreo (<i>tracing</i>) hacia adelante y hacia atrás y explicar su papel en el proceso de validación de requisitos [Usar]
Lecturas: Stroustrup (2013)	

8. Metodología

El profesor del curso presentará clases teóricas de los temas señalados en el programa propiciando la intervención de los alumnos.

El profesor del curso presentará demostraciones para fundamentar clases teóricas.

El profesor y los alumnos realizarán prácticas

Los alumnos deberán asistir a clase habiendo leído lo que el profesor va a presentar. De esta manera se facilitará la comprensión y los estudiantes estarán en mejores condiciones de hacer consultas en clase.

9. Evaluar

Evaluación Continua 1 : 20 %

Examen parcial : 30 %

Evaluación Continua 2 : 20 %

Examen final : 30 %

References

- Lippman, Stanley B. and Barbara E.Moo (2013). *C++ Primer*. 5th. O'Reilly. ISBN: 9780133053043.
- Nakariakov, S. (2013). *The Boost C++ Libraries: Generic Programming*. CreateSpace Independent Publishing Platform.
- Pai, Praseed and Peter Abraham (2018). *C++ Reactive Programming*. 1st. Packt.
- Stroustrup, B (2013). *The C++ Programming Language, 4th edition*. Addison-Wesley.
- Vandervoorde, David (2002). *C++ Templates: The Complete Guide*. 1st. Addison-Wesley. ISBN: 978-0134448237.
- Williams, Anthony (2011). *C++ Concurrency in Action*. 1st. Manning.



Universidad Nacional de Colombia (UNAL) Sede
Manizales
Programa Profesional de
Administración de Sistemas Informáticos
SILABO

CS113. Ciencia de la Computación II (Obligatorio)

2022-II

1. Información general	
1.1 Escuela	: Sistemas de Información
1.2 Curso	: CS113. Ciencia de la Computación II
1.3 Semestre	: 3 ^{er} Semestre.
1.4 Prerrequisitos	: <ul style="list-style-type: none">• CS112. Ciencia de la Computación I. (2^{do} Sem)• CS112. Ciencia de la Computación I. (2^{do} Sem)
1.5 Condición	: Obligatorio
1.6 Modalidad de aprendizaje	: Presencial
1.7 horas	: 2 HT; 2 HP; 4 HL;
1.8 Créditos	: 5
2. Profesores	
3. Fundamentación del curso	
Este es el tercer curso en la secuencia de los cursos introductorios a la informática. En este curso se pretende cubrir los conceptos señalados por la Computing Curricula IEEE(c)-ACM 2001, bajo el enfoque funcional-first. El paradigma orientado a objetos nos permite combatir la complejidad haciendo modelos a partir de abstracciones de los elementos del problema y utilizando técnicas como encapsulamiento, modularidad, polimorfismo y herencia. El dominio de estos temas permitirá que los participantes puedan dar soluciones computacionales a problemas de diseño sencillos del mundo real.	
4. Resumen	
1. Conceptos Fundamentales de Programación 2. Programación orientada a objetos 3. Algoritmos y Diseño 4. Análisis Básico 5. Sistemas de tipos básicos 6. Algoritmos y Estructuras de Datos fundamentales 7. Programación reactiva y dirigida por eventos 8. Árboles y Grafos 9. Diseño de Software 10. Ingeniería de Requisitos	
5. Objetivos Generales	
<ul style="list-style-type: none">• Introducir al alumno a los fundamentos del paradigma de orientación a objetos, permitiendo asimilar los conceptos necesarios para desarrollar un sistema de información	

6. Contribución a los resultados (*Outcomes*)

Esta disciplina contribuye al logro de los siguientes resultados de la carrera:

- 1) Analizar un problema computacional complejo y aplicar los principios computacionales y otras disciplinas relevantes para identificar soluciones. (**Usar**)
- 3) Comunicarse efectivamente en diversos contextos profesionales. (**Usar**)
- 5) Funcionar efectivamente como miembro o líder de un equipo involucrado en actividades apropiadas a la disciplina del programa. (**Usar**)

7. Contenido

UNIDAD 1: Conceptos Fundamentales de Programación (5)

Competencias:

Contenido	Objetivos Generales
<ul style="list-style-type: none">• Sintaxis y semántica básica de un lenguaje de alto nivel.• Variables y tipos de datos primitivos (ej., números, caracteres, booleanos)• Expresiones y asignaciones.• Operaciones básicas I/O incluyendo archivos I/O.• Estructuras de control condicional e iterativas.• Paso de funciones y parámetros.• Concepto de recursividad.	<ul style="list-style-type: none">• Analiza y explica el comportamiento de programas simples que involucran estructuras fundamentales de programación variables, expresiones, asignaciones, E/S, estructuras de control, funciones, paso de parámetros, y recursividad [Usar]• Identifica y describe el uso de tipos de datos primitivos [Usar]• Escribe programas que usan tipos de datos primitivos [Usar]• Modifica y expande programas cortos que usen estructuras de control condicionales e iterativas así como funciones [Usar]• Diseña, implementa, prueba, y depura un programa que usa cada una de las siguientes estructuras de datos fundamentales: cálculos básicos, E/S simple, condicional estándar y estructuras iterativas, definición de funciones, y paso de parámetros [Usar]• Escribe un programa que usa E/S de archivos para brindar persistencia a través de ejecuciones múltiples [Usar]• Escoje estructuras de condición y repetición adecuadas para una tarea de programación dada [Usar]• Describe el concepto de recursividad y da ejemplos de su uso [Usar]• Identifica el caso base y el caso general de un problema basado en recursividad [Usar]

Lecturas: stroustrup2013, Vandervoorde (2002), Lippman and E.Moo (2013)

UNIDAD 2: Programación orientada a objetos (7)	
Competencias:	
Contenido	Objetivos Generales
<ul style="list-style-type: none"> ● Diseño orientado a objetos: <ul style="list-style-type: none"> – Descomposición en objetos que almacenan estados y poseen comportamiento – Diseño basado en jerarquía de clases para modelamiento ● Definición de las categorías, campos, métodos y constructores. ● Las subclasses, herencia y método de alteración temporal. ● Asignación dinámica: definición de método de llamada. ● Subtipificación: <ul style="list-style-type: none"> – Polimorfismo artículo Subtipo; upcasts implícitos en lenguajes con tipos. – Noción de reemplazo de comportamiento: los subtipos de actuar como supertipos. – Relación entre subtipos y la herencia. ● Lenguajes orientados a objetos para la encapsulación: <ul style="list-style-type: none"> – privacidad y la visibilidad de miembros de la clase – Interfaces revelan único método de firmas – clases base abstractas ● Uso de colección de clases, iteradores, y otros componentes de la librería estándar. 	<ul style="list-style-type: none"> ● Diseñar e implementar una clase [Usar] ● Usar subclase para diseñar una jerarquía simple de clases que permita al código ser reusable por diferentes subclasses [Usar] ● Razonar correctamente sobre el flujo de control en un programa mediante el envío dinámico [Usar] ● Comparar y contrastar (1) el enfoque procedurales/funcional- definiendo una función por cada operación con el uso de la función proporcionando un caso por cada variación de dato - y (2) el enfoque orientado a objetos - definiendo una clase por cada variación de dato con la definición de la clase proporcionando un método por cada operación. Entender ambos enfoques como una definición de variaciones y operaciones de una matriz [Usar] ● Explicar la relación entre la herencia orientada a objetos (código compartido y <i>overriding</i>) y subtipificación (la idea de un subtipo es ser utilizable en un contexto en el que espera al supertipo) [Usar] ● Usar mecanismos de encapsulación orientada a objetos, tal como interfaces y miembros privados [Usar] ● Definir y usar iteradores y otras operaciones sobre agregaciones, incluyendo operaciones que tienen funciones como argumentos, en múltiples lenguajes de programación, seleccionar la forma más natural por cada lenguaje [Usar]
Lecturas: Stroustrup (2013)	

UNIDAD 3: Algoritmos y Diseño (5)**Competencias:****Contenido****Objetivos Generales**

- Conceptos y propiedades de los algoritmos
 - Comparación informal de la eficiencia de los algoritmos (ej., conteo de operaciones)
- Rol de los algoritmos en el proceso de solución de problemas
- Estrategias de solución de problemas
 - Funciones matemáticas iterativas y recursivas
 - Recorrido iterativo y recursivo en estructura de datos
 - Estrategias Divide y Conquistar
- Conceptos y principios fundamentales de diseño
 - Abstracción
 - Descomposición de Program
 - Encapsulamiento y camuflaje de información
 - Separación de comportamiento y aplicación

- Discute la importancia de los algoritmos en el proceso de solución de un problema [Usar]
- Discute como un problema puede ser resuelto por múltiples algoritmos, cada uno con propiedades diferentes [Usar]
- Crea algoritmos para resolver problemas simples [Usar]
- Usa un lenguaje de programación para implementar, probar, y depurar algoritmos para resolver problemas simples [Usar]
- Implementa, prueba, y depura funciones recursivas simples y sus procedimientos [Usar]
- Determina si una solución iterativa o recursiva es la más apropiada para un problema [Usar]
- Implementa un algoritmo de divide y vencerás para resolver un problema [Usar]
- Aplica técnicas de descomposición para dividir un programa en partes más pequeñas [Usar]
- Identifica los componentes de datos y el comportamiento de múltiples tipos de datos abstractos [Usar]
- Implementa un tipo de dato abstracto coherente, con la menor pérdida de acoplamiento entre componentes y comportamientos [Usar]
- Identifica las fortalezas y las debilidades relativas entre múltiples diseños e implementaciones de un problema [Usar]

Lecturas: stroustrup2013, Weert16, Lippman and E.Moo (2013)

UNIDAD 4: Análisis Básico (3)**Competencias:**

Contenido	Objetivos Generales
<ul style="list-style-type: none"> • Diferencias entre el mejor, el esperado y el peor caso de un algoritmo. • Análisis asintótico de complejidad de cotas superior y esperada. • Definición formal de la Notación Big O. • Clases de complejidad como constante, logarítmica, lineal, cuadrática y exponencial. • Medidas empíricas de desempeño. • Compensación entre espacio y tiempo en los algoritmos. • Uso de la notación Big O. • Notación Little o, Big omega y Big theta. • Relaciones recurrentes. • Análisis de algoritmos iterativos y recursivos. • Teorema Maestro y Árboles Recursivos. 	<ul style="list-style-type: none"> • Explique a que se refiere con “mejor”, “esperado” y “peor” caso de comportamiento de un algoritmo [Usar] • En el contexto de a algoritmos específicos, identifique las características de data y/o otras condiciones o suposiciones que lleven a diferentes comportamientos [Usar] • Determine informalmente el tiempo y el espacio de complejidad de diferentes algoritmos [Usar] • Indique la definición formal de Big O [Usar] • Lista y contraste de clases estándares de complejidad [Usar] • Realizar estudios empíricos para validar una hipótesis sobre runtime stemming desde un análisis matemático Ejecute algoritmos con entrada de varios tamaños y compare el desempeño [Usar] • Da ejemplos que ilustran las compensaciones entre espacio y tiempo que se dan en los algoritmos [Usar] • Use la notación formal de la Big O para dar límites superiores asintóticos en la complejidad de tiempo y espacio de los algoritmos [Usar] • Usar la notación formal Big O para dar límites de casos esperados en el tiempo de complejidad de los algoritmos [Usar] • Explicar el uso de la notación theta grande, omega grande y o pequeña para describir la cantidad de trabajo hecho por un algoritmo [Usar] • Usar relaciones recurrentes para determinar el tiempo de complejidad de algoritmos recursivamente definidos [Usar] • Resuelve relaciones de recurrencia básicas, por ejemplo. usando alguna forma del Teorema Maestro [Usar]
Lecturas: Stroustrup (2013)	

UNIDAD 5: Sistemas de tipos básicos (5)	
Competencias:	
Contenido	Objetivos Generales
<ul style="list-style-type: none"> • Tipos como conjunto de valores junto con un conjunto de operaciones. <ul style="list-style-type: none"> – Tipos primitivos (p.e. números, booleanos) – Composición de tipos contruídos de otros tipos (p.e., registros, uniones, arreglos, listas, funciones, referencias) • Asociación de tipos de variables, argumentos, resultados y campos. • Tipo de seguridad y los errores causados por el uso de valores de manera incompatible dadas sus tipos previstos. • Metas y limitaciones de tipos estáticos <ul style="list-style-type: none"> – Eliminación de algunas clases de errores sin ejecutar el programa – Indecisión significa que un análisis estatico puede aproximar el comportamiento de un programa • Tipos genéricos (polimorfismo paramétrico) <ul style="list-style-type: none"> – Definición – Uso de librerías genéricas tales como colecciones. – Comparación con polimorfismo ad-hoc y polimorfismo de subtipos • Beneficios complementarios de tipos estáticos y dinámicos: <ul style="list-style-type: none"> – Errores tempranos vs. errores tardíos/evitados. – Refuerzo invariante durante el desarrollo y mantenimiento del código vs. decisiones pospuestas de tipos durante la la creación de prototipos y permitir convenientemente la codificación flexible de patrones tales como colecciones heterogéneas. – Evitar el mal uso del código vs. permitir más reuso de código. – Detectar programas incompletos vs. permitir que programas incompletos se ejecuten 	<ul style="list-style-type: none"> • Tanto para tipo primitivo y un tipo compuesto, describir de manera informal los valores que tiene dicho tipo [Usar] • Para un lenguaje con sistema de tipos estático, describir las operaciones que están prohibidas de forma estática, como pasar el tipo incorrecto de valor a una función o método [Usar] • Describir ejemplos de errores de programa detectadas por un sistema de tipos [Usar] • Para múltiples lenguajes de programación, identificar propiedades de un programa con verificación estática y propiedades de un programa con verificación dinámica [Usar] • Dar un ejemplo de un programa que no verifique tipos en un lenguaje particular y sin embargo no tenga error cuando es ejecutado [Usar] • Usar tipos y mensajes de error de tipos para escribir y depurar programas [Usar] • Explicar como las reglas de tipificación definen el conjunto de operaciones que legales para un tipo [Usar] • Escribir las reglas de tipo que rigen el uso de un particular tipo compuesto [Usar] • Explicar por qué indecidibilidad requiere sistemas de tipo para conservadoramente aproximar el comportamiento de un programa [Usar] • Definir y usar piezas de programas (tales como, funciones, clases, métodos) que usan tipos genéricos, incluyendo para colecciones [Usar] • Discutir las diferencias entre, genéricos (<i>generics</i>), subtipo y sobrecarga [Usar] • Explicar múltiples beneficios y limitaciones de tipificación estática en escritura, mantenimiento y depuración de un software [Usar]
Lecturas: Stroustrup (2013)	

UNIDAD 6: Algoritmos y Estructuras de Datos fundamentales (3)**Competencias:****Contenido****Objetivos Generales**

- Algoritmos numéricos simples, tales como el cálculo de la media de una lista de números, encontrar el mínimo y máximo.
- Algoritmos de búsqueda secuencial y binaria.
- Algoritmos de ordenamiento de peor caso cuadrático (selección, inserción)
- Algoritmos de ordenamiento con peor caso o caso promedio en $O(N \lg N)$ (Quicksort, Heapsort, Mergesort)
- Tablas Hash, incluyendo estrategias para evitar y resolver colisiones.
- Árboles de búsqueda binaria:
 - Operaciones comunes en árboles de búsqueda binaria como seleccionar el mínimo, máximo, insertar, eliminar, recorrido en árboles.
- Grafos y algoritmos en grafos:
 - Representación de grafos (ej., lista de adyacencia, matriz de adyacencia)
 - Recorrido en profundidad y amplitud
- Montículos (Heaps)
- Grafos y algoritmos en grafos:
 - Problema de corte máximo y mínimo
 - Búsqueda local
- Búsqueda de patrones y algoritmos de cadenas/texto (ej. búsqueda de subcadena, búsqueda de expresiones regulares, algoritmos de subsecuencia común más larga)

- Implementar algoritmos numéricos básicos [Usar]
- Implementar algoritmos de búsqueda simple y explicar las diferencias en sus tiempos de complejidad [Usar]
- Ser capaz de implementar algoritmos de ordenamiento comunes cuadráticos y $O(N \log N)$ [Usar]
- Describir la implementación de tablas hash, incluyendo resolución y el evitamiento de colisiones [Usar]
- Discutir el tiempo de ejecución y eficiencia de memoria de los principales algoritmos de ordenamiento, búsqueda y hashing [Usar]
- Discutir factores otros que no sean eficiencia computacional que influyan en la elección de algoritmos, tales como tiempo de programación, mantenibilidad, y el uso de patrones específicos de la aplicación en los datos de entrada [Usar]
- Explicar como el balanceamiento del arbol afecta la eficiencia de varias operaciones de un arbol de búsqueda binaria [Usar]
- Resolver problemas usando algoritmos básicos de grafos, incluyendo búsqueda por profundidad y búsqueda por amplitud [Usar]
- Demostrar habilidad para evaluar algoritmos, para seleccionar de un rango de posibles opciones, para proveer una justificación por esa selección, y para implementar el algoritmo en un contexto en específico [Usar]
- Describir la propiedad del heap y el uso de heaps como una implementación de colas de prioridad [Usar]
- Resolver problemas usando algoritmos de grafos, incluyendo camino más corto de una sola fuente y camino más corto de todos los pares, y como mínimo un algoritmo de arbol de expansion minima [Usar]
- Trazar y/o implementar un algoritmo de comparación de string [Usar]

Lecturas: stroustrup2013, Pai and Abraham (2018)

UNIDAD 7: Programación reactiva y dirigida por eventos (2)	
Competencias:	
Contenido	Objetivos Generales
<ul style="list-style-type: none"> • Eventos y controladores de eventos. • Usos canónicos como interfaces gráficas de usuario, dispositivos móviles, robots, servidores. • Uso de frameworks reactivos. <ul style="list-style-type: none"> – Definición de controladores/oyentes (handles/listeners) de eventos. – Bucle principal de eventos no controlado por el escritor controlador de eventos (event-handler-writer) • Eventos y eventos del programa generados externamente generada. • La separación de modelo, vista y controlador. 	<ul style="list-style-type: none"> • Escribir manejadores de eventos para su uso en sistemas reactivos tales como GUIs [Usar] • Explicar porque el estilo de programación manejada por eventos es natural en dominios donde el programa reacciona a eventos externos [Usar] • Describir un sistema interactivo en términos de un modelo, una vista y un controlador [Usar]
Lecturas: stroustrup2013, Williams (2011)	

UNIDAD 8: Árboles y Grafos (7)	
Competencias:	
Contenido	Objetivos Generales
<ul style="list-style-type: none"> • Árboles. <ul style="list-style-type: none"> – Propiedades – Estrategias de recorrido • Grafos no dirigidos • Grafos dirigidos • Grafos ponderados • Árboles de expansion/bosques. • Isomorfismo en grafos. 	<ul style="list-style-type: none"> • Ilustrar mediante ejemplos la terminología básica de teoría de grafos, y de alguna de las propiedades y casos especiales de cada tipo de grafos/árboles [Usar] • Demostrar diversos métodos de recorrer árboles y grafos, incluyendo recorridos pre, post e inorden de árboles [Usar] • Modelar una variedad de problemas del mundo real en ciencia de la computación usando formas adecuadas de grafos y árboles, como son la representación de una topología de red o la organización jerárquica de un sistema de archivos [Usar] • Demostrar como los conceptos de grafos y árboles aparecen en estructuras de datos, algoritmos, técnicas de prueba (inducción estructurada), y conteos [Usar] • Explicar como construir un árbol de expansión de un grafo [Usar] • Determinar si dos grafos son isomorfos [Usar]
Lecturas: Nakariakov (2013)	

UNIDAD 9: Diseño de Software (6)**Competencias:****Contenido****Objetivos Generales**

- Principios de diseño del sistema: niveles de abstracción (diseño arquitectónico y el diseño detallado), separación de intereses, ocultamiento de información, de acoplamiento y de cohesión, de reutilización de estructuras estándar.
- Diseño de paradigmas tales como diseño estructurado (descomposición funcional de arriba hacia abajo), el análisis orientado a objetos y diseño, orientado a eventos de diseño, diseño de nivel de componente, centrado datos estructurada, orientada a aspectos, orientado a la función, orientado al servicio.
- Modelos estructurales y de comportamiento de los diseños de software.
- Diseño de patrones.
- Relaciones entre los requisitos y diseños: La transformación de modelos, el diseño de los contratos, invariantes.
- Conceptos de arquitectura de software y arquitecturas estándar (por ejemplo, cliente-servidor, n-capas, transforman centrados, tubos y filtros).
- El uso de componentes de diseño: selección de componentes, diseño, adaptación y componentes de ensamblaje, componentes y patrones, componentes y objetos (por ejemplo, construir una GUI usando un standard widget set)
- Diseños de refactorización utilizando patrones de diseño
- Calidad del diseño interno, y modelos para: eficiencia y desempeño, redundancia y tolerancia a fallos, trazabilidad de los requerimientos.
- Medición y análisis de la calidad de un diseño.
- Compensaciones entre diferentes aspectos de la calidad.
- Aplicaciones en frameworks.
- Middleware: El paradigma de la orientación a objetos con middleware, requerimientos para correr y clasificar objetos, monitores de procesamiento de transacciones y el sistema de flujo de trabajo.
- Principales diseños de seguridad y codificación (cross-reference IAS/Principles of secure design).
 - Principio de privilegios mínimos
 - Principio de falla segura por defecto
 - Principio de aceptabilidad psicológica

- Formular los principios de diseño, incluyendo la separación de problemas, ocultación de información, acoplamiento y cohesión, y la encapsulación [Usar]
- Usar un paradigma de diseño para diseñar un sistema de software básico y explicar cómo los principios de diseño del sistema se han aplicado en este diseño [Usar]
- Construir modelos del diseño de un sistema de software simple los cuales son apropiado para el paradigma utilizado para diseñarlo [Usar]
- En el contexto de un paradigma de diseño simple, describir uno o más patrones de diseño que podrían ser aplicables al diseño de un sistema de software simple [Usar]
- Para un sistema simple adecuado para una situación dada, discutir y seleccionar un paradigma de diseño apropiado [Usar]
- Crear modelos apropiados para la estructura y el comportamiento de los productos de software desde la especificaciones de requisitos [Usar]
- Explicar las relaciones entre los requisitos para un producto de software y su diseño, utilizando los modelos apropiados [Usar]
- Para el diseño de un sistema de software simple dentro del contexto de un único paradigma de diseño, describir la arquitectura de software de ese sistema [Usar]
- Dado un diseño de alto nivel, identificar la arquitectura de software mediante la diferenciación entre las arquitecturas comunes de software, tales como 3 capas (*3-tier*), *pipe-and-filter*, y cliente-servidor [Usar]
- Investigar el impacto de la selección arquitecturas de software en el diseño de un sistema simple [Usar]
- Aplicar ejemplos simples de patrones en un diseño de software [Usar]
- Describir una manera de refactorar y discutir cuando esto debe ser aplicado [Usar]
- Seleccionar componentes adecuados para el uso en un diseño de un producto de software [Usar]
- Explicar cómo los componentes deben ser adaptados para ser usados en el diseño de un producto de software [Usar]
- Diseñar un contrato para un típico componente de software pequeño para el uso de un dado sistema [Usar]
- Discutir y seleccionar la arquitectura de software adecuada para un sistema de software simple para un dado escenario [Usar]

UNIDAD 10: Ingeniería de Requisitos (1)	
Competencias:	
Contenido	Objetivos Generales
<ul style="list-style-type: none"> • Al describir los requisitos funcionales utilizando, por ejemplo, los casos de uso o historias de los usuarios. • Propiedades de requisitos, incluyendo la consistencia, validez, integridad y viabilidad. • Requisitos de software elicitation. • Descripción de datos del sistema utilizando, por ejemplo, los diagramas de clases o diagramas entidad-relación. • Requisitos no funcionales y su relación con la calidad del software. • Evaluación y uso de especificaciones de requisitos. • Requisitos de las técnicas de modelado de análisis. • La aceptabilidad de las consideraciones de certeza/incertidumbre sobre el comportamiento del software/sistema. • Prototipos. • Conceptos básicos de la especificación formal de requisitos. • Especificación de requisitos. • Validación de requisitos. • Rastreo de requisitos. 	<ul style="list-style-type: none"> • Enumerar los componentes clave de un caso de uso o una descripción similar de algún comportamiento que es requerido para un sistema [Usar] • Describir cómo el proceso de ingeniería de requisitos apoya la obtención y validación de los requisitos de comportamiento [Usar] • Interpretar un modelo de requisitos dada por un sistema de software simple [Usar] • Describir los retos fundamentales y técnicas comunes que se utilizan para la obtención de requisitos [Usar] • Enumerar los componentes clave de un modelo de datos (por ejemplo, diagramas de clases o diagramas ER) [Usar] • Identificar los requisitos funcionales y no funcionales en una especificación de requisitos dada por un sistema de software [Usar] • Realizar una revisión de un conjunto de requisitos de software para determinar la calidad de los requisitos con respecto a las características de los buenos requisitos [Usar] • Aplicar elementos clave y métodos comunes para la obtención y el análisis para producir un conjunto de requisitos de software para un sistema de software de tamaño medio [Usar] • Comparar los métodos ágiles y el dirigido por planes para la especificación y validación de requisitos y describir los beneficios y riesgos asociados con cada uno [Usar] • Usar un método común, no formal para modelar y especificar los requisitos para un sistema de software de tamaño medio [Usar] • Traducir al lenguaje natural una especificación de requisitos de software (por ejemplo, un contrato de componentes de software) escrito en un lenguaje de especificación formal [Usar] • Crear un prototipo de un sistema de software para reducir el riesgo en los requisitos [Usar] • Diferenciar entre el rastreo (<i>tracing</i>) hacia adelante y hacia atrás y explicar su papel en el proceso de validación de requisitos [Usar]
Lecturas: Stroustrup (2013)	

8. Metodología

El profesor del curso presentará clases teóricas de los temas señalados en el programa propiciando la intervención de los alumnos.

El profesor del curso presentará demostraciones para fundamentar clases teóricas.

El profesor y los alumnos realizarán prácticas

Los alumnos deberán asistir a clase habiendo leído lo que el profesor va a presentar. De esta manera se facilitará la comprensión y los estudiantes estarán en mejores condiciones de hacer consultas en clase.

9. Evaluar

Evaluación Continua 1 : 20 %

Examen parcial : 30 %

Evaluación Continua 2 : 20 %

Examen final : 30 %

References

- Lippman, Stanley B. and Barbara E.Moo (2013). *C++ Primer*. 5th. O'Reilly. ISBN: 9780133053043.
- Nakariakov, S. (2013). *The Boost C++ Libraries: Generic Programming*. CreateSpace Independent Publishing Platform.
- Pai, Praseed and Peter Abraham (2018). *C++ Reactive Programming*. 1st. Packt.
- Stroustrup, B (2013). *The C++ Programming Language, 4th edition*. Addison-Wesley.
- Vandervoorde, David (2002). *C++ Templates: The Complete Guide*. 1st. Addison-Wesley. ISBN: 978-0134448237.
- Williams, Anthony (2011). *C++ Concurrency in Action*. 1st. Manning.



Universidad Nacional de Colombia (UNAL) Sede
Manizales
Programa Profesional de
Administración de Sistemas Informáticos
SILABO

CS1D3. Álgebra Abstracta (Obligatorio)

2022-II

1. Información general	
1.1 Escuela	: Sistemas de Información
1.2 Curso	: CS1D3. Álgebra Abstracta
1.3 Semestre	: 3 ^{er} Semestre.
1.4 Prerrequisitos	: <ul style="list-style-type: none">• CS1D1. Estructuras Discretas I. (1^{er} Sem)• CS112. Ciencia de la Computación I. (2^{do} Sem)
1.5 Condición	: Obligatorio
1.6 Modalidad de aprendizaje	: Presencial
1.7 horas	: 2 HT; 2 HL;
1.8 Créditos	: 3
2. Profesores	
3. Fundamentación del curso	
En algebra abstracta se explotará las nociones de teoria de números, grupos, anillos y campos para comprender en profundidad temas de computación como criptografía y teoría de la codificación.	
4. Resumen	
1. 2. 3. Criptografía 4.	
5. Objetivos Generales	
<ul style="list-style-type: none">• Entender los conceptos de estructuras algebraicas como anillos, dominios, cuerpos y grupos.• Utilizar las propiedades de las estructuras algebraicas para resolver problemas• Conocer las técnicas y métodos de sistemas criptográficos y como los teoremas permiten la realización de cálculos rápidos y eficientes.	
6. Contribución a los resultados (Outcomes)	
Esta disciplina contribuye al logro de los siguientes resultados de la carrera: <ol style="list-style-type: none">1) Analizar un problema computacional complejo y aplicar los principios computacionales y otras disciplinas relevantes para identificar soluciones. (Evaluar)6) Aplicar fundamentos de teoria de ciencias de la computación y desarrollo de software para producir soluciones basados en computación. (Evaluar)	
7. Contenido	

UNIDAD 1: (16)	
Competencias:	
Contenido	Objetivos Generales
<ul style="list-style-type: none"> • Número enteros, algoritmos de la división, máximo común divisor, algoritmo de Euclides y algoritmo extendido de Euclides. Ecuaciones diofánticas • Aritmética Modular y Operaciones en \mathbb{Z}_n: suma, resta, multiplicación, inversa y exponenciación. • Congruencia, conjunto de residuos, congruencia lineal, teorema chino del resto. • Generadores de números primos y pseudo-aleatorios, función phi de Euler, teorema pequeño de Fermat, teorema de Euler, teorema fundamental de la aritmética y factorización. 	<ul style="list-style-type: none"> • Realizar cálculos que involucren aritmética modular [Usar] • Describir algoritmos numérico teóricos básicos eficientes, incluyendo el algoritmo de Euclides y el algoritmo extendido de Euclides. [Evaluar] • Establecer la importancia del estudio de la teoría de números. [Familiarizarse] • Discutir la importancia de los números primos en criptografía y explicar su uso en algoritmos criptográficos [Familiarizarse]
Lecturas: Rosen (2011), Grimaldi (2003), Koshy (2007)	

UNIDAD 2: (14)	
Competencias:	
Contenido	Objetivos Generales
<ul style="list-style-type: none"> • Grupos: propiedades, operaciones, homomorfismos e isomorfismo, orden de un grupo, grupos cíclicos, teorema de Lagrange y raíces primitivas. • Anillos y cuerpos: propiedades, sub-anillos, dominios de integridad. 	<ul style="list-style-type: none"> • Adquirir habilidad en la resolución de problemas abstractos y en la formulación de conjeturas . [Familiarizarse] • Argumentar como los principales teoremas y algoritmos permiten resolver problemas criptográficos. [Evaluar]
Lecturas: Grimaldi (2003), Gallian (2012), Koshy (2007)	

UNIDAD 3: Criptografía (20)

Competencias:

Contenido

Objetivos Generales

- Terminología básica de criptografía cubriendo las nociones relacionadas con los diferentes socios (comunicación), canal seguro / inseguro, los atacantes y sus capacidades, cifrado, descifrado, llaves y sus características, firmas.
- Tipos de cifrado (por ejemplo, cifrado César, cifrado affine), junto con los métodos de ataque típicas como el análisis de frecuencia.
- Apoyo a la infraestructura de clave pública para la firma digital y el cifrado y sus desafíos.
- Preliminares matemáticos esenciales para la criptografía, incluyendo temas de álgebra lineal, teoría de números, teoría de la probabilidad y la estadística.
- Primitivas criptográficas:
 - generadores pseudo-aleatorios y cifrados de flujo
 - cifrados de bloque (permutaciones pseudo-aleatorios), por ejemplo, AES
 - funciones de pseudo-aleatorios
 - funciones de hash, por ejemplo, SHA2, resistencia colisión
 - códigos de autenticación de mensaje
 - funciones derivaciones clave
- Criptografía de clave simétrica:
 - El secreto perfecto y el cojín de una sola vez
 - Modos de funcionamiento para la seguridad semántica y encriptación autenticada (por ejemplo, cifrar-entonces-MAC, OCB, GCM)
 - Integridad de los mensajes (por ejemplo, CMAC, HMAC)
- La criptografía de clave pública:
 - Permutación de trampa, por ejemplo, RSA
 - Cifrado de clave pública, por ejemplo, el cifrado RSA, cifrado El Gamal
 - Las firmas digitales
 - Infraestructura de clave pública (PKI) y certificados
 - Supuestos de dureza, por ejemplo, Diffie-Hellman, factoring entero
- Protocolos de intercambio de claves autenticadas, por ejemplo, TLS .
- Los protocolos criptográficos: autenticación desafío-respuesta, protocolos de conocimiento cero, el compromiso, la transferencia inconsciente, seguro 2-partido o multipartidista computación, compartición de secretos y aplicaciones .
- Motivar a los conceptos que utilizan las aplicaciones

- Describir el propósito de la Criptografía y listar formas en las cuales es usada en comunicación de datos [Familiarizarse]
- Definir los siguientes términos: Cifrado, Criptoanálisis, Algoritmo Criptográfico, y Criptología y describe dos métodos básicos (cifrados) para transformar texto plano en un texto cifrado [Familiarizarse]
- Discutir la importancia de los números primos en criptografía y explicar su uso en algoritmos criptográficos [Familiarizarse]
- Explicar como una infraestructura de Clave Pública soporta firmas digitales y encriptación y discutir sus limitaciones/vulnerabilidades [Familiarizarse]
- Usar primitivas criptográficas y sus propiedades básicas [Familiarizarse]
- Ilustrar como medir la entropía y como generar aleatoriedad criptográfica [Familiarizarse]
- Usa primitivas de clave pública y sus aplicaciones [Familiarizarse]
- Explicar como los protocolos de intercambio de claves trabajan y como es que pueden fallar [Familiarizarse]
- Discutir protocolos criptográficos y sus propiedades [Familiarizarse]
- Describir aplicaciones del mundo real de primitivas criptográficas y sus protocolos [Familiarizarse]
- Resumir definiciones precisas de seguridad, capacidades de ataque y sus metas [Familiarizarse]
- Aplicar técnicas conocidas y apropiadas de criptografía para un escenario determinado [Familiarizarse]
- Apreciar los peligros de inventarse cada uno sus propios métodos criptográficos [Familiarizarse]
- Describir la criptografía cuántica y el impacto de la computación cuántica en algoritmos criptográficos [Familiarizarse]

UNIDAD 4: (10)	
Competencias:	
Contenido	Objetivos Generales
<ul style="list-style-type: none"> • Elementos, proceso de transmitir una palabra • Esquemas de codificación: paridad, triple repetición, verificación de paridad y generación de códigos de grupo. 	<ul style="list-style-type: none"> • Utilizar las propiedades de las estructuras algebraicas en el estudio de la teoría algebraica de los códigos. [Familiarizarse] • Aplicar técnicas que permitan la detección de errores, y si es necesario, proveer de métodos para reconstruir palabras originales. [Usar]
Lecturas: Grimaldi (2003), W.Trappe and Washington (2005)	

8. Metodología
<p>El profesor del curso presentará clases teóricas de los temas señalados en el programa propiciando la intervención de los alumnos.</p> <p>El profesor del curso presentará demostraciones para fundamentar clases teóricas.</p> <p>El profesor y los alumnos realizarán prácticas</p> <p>Los alumnos deberán asistir a clase habiendo leído lo que el profesor va a presentar. De esta manera se facilitará la comprensión y los estudiantes estarán en mejores condiciones de hacer consultas en clase.</p>

9. Evaluar
<p>Evaluación Continua 1 : 20 %</p> <p>Examen parcial : 30 %</p> <p>Evaluación Continua 2 : 20 %</p> <p>Examen final : 30 %</p>

References

- A.Menezes (1996). *Handbook of Applied Cryptography (Discrete Mathematics and Its Applications)*. CRC Press.
- Forouzan, B. (2008). *Introduction to Cryptography and Network Security*. McGraw-Hill.
- Gallian, J. (2012). *Contemporary Abstract Algebra*. 8 ed. Brooks/Cole.
- Grimaldi, R. (2003). *Discrete and Combinatorial Mathematics: An Applied Introduction*. 5 ed. Pearson.
- Koshy, T. (2007). *Elementary Number Theory with Applications*. 2 ed. Academic Press.
- Paar, C. and J. Pelzl (2011). *Understanding Cryptography: A Textbook for Students and Practitioners*. Springer.
- Rosen, Kenneth H. (2011). *Matemática Discreta y sus Aplicaciones*. 7 ed. McGraw Hill.
- W.Trappe and C. Washington (2005). *Introduction to Cryptography with Coding Theory*. Pearson Prentice Hall.



Universidad Nacional de Colombia (UNAL) Sede
Manizales
Programa Profesional de
Administración de Sistemas Informáticos
SILABO

CS221. Arquitectura de Computadores (Obligatorio)

2022-II

1. Información general	
1.1 Escuela	: Sistemas de Información
1.2 Curso	: CS221. Arquitectura de Computadores
1.3 Semestre	: 3 ^{er} Semestre.
1.4 Prerrequisitos	: CS1D2. Estructuras Discretas II. (2 ^{do} Sem)
1.5 Condición	: Obligatorio
1.6 Modalidad de aprendizaje	: Presencial
1.7 horas	: 2 HT; 2 HL;
1.8 Créditos	: 3
2. Profesores	
3. Fundamentación del curso	
Un profesional en Ciencia de la Computación debe tener un conocimiento sólido de la organización y los principios de diseño de diversos sistemas de computación, al comprender las limitaciones de los sistemas modernos serán capaces de proponer nuevos paradigmas en la próxima generación. Este curso enseña los fundamentos y principios de la arquitectura de computadoras. Esta clase incluye diseño de lógica digital, conceptos básicos de arquitectura de computadora y diseño de procesador (<i>Instruction Set Architecture</i> , microarquitectura, ejecución fuera de orden, predicción de <i>branches</i>), paradigmas de ejecución (superescalar, flujo de datos, VLIW, SIMD, GPU, sistólica, multiproceso) y organización del sistema de memoria.	
4. Resumen	
1. Lógica digital y sistemas digitales 2. Representación de datos a nivel máquina 3. Organización de la Máquina a Nivel Ensamblador 4. Organización funcional 5. Organización y Arquitectura del Sistema de Memoria 6. Interfaz y comunicación 7. Multiprocesamiento y arquitecturas alternativas 8. Mejoras de rendimiento	
5. Objetivos Generales	
<ul style="list-style-type: none">• Proporcionar un primer enfoque en Arquitectura de Computadoras.• Estudiar el diseño y la evolución de las arquitecturas de computador, que llevaron a las implementaciones de los sistemas modernos.• Proporcionar un estudio profundo del hardware y su relación con la ejecución del software.• Implementar un microprocesador simple usando el lenguaje Verilog.	

6. Contribución a los resultados (*Outcomes*)

Esta disciplina contribuye al logro de los siguientes resultados de la carrera:

- 1) Analizar un problema computacional complejo y aplicar los principios computacionales y otras disciplinas relevantes para identificar soluciones. (**Usar**)
- 6) Aplicar fundamentos de teoría de ciencias de la computación y desarrollo de software para producir soluciones basados en computación. (**Evaluar**)

7. Contenido

UNIDAD 1: Lógica digital y sistemas digitales (18)

Competencias:

Contenido

Objetivos Generales

- Revisión e historia de la Arquitectura de Computadores.
- Lógica combinacional y secuencial/*field programmable gate arrays* como bloque fundamental de construcción lógico combinacional secuencial.
- Modelos de representación (abstracción)
- Herramientas de diseño asistidas por computadora que procesan hardware y representaciones arquitecturales.
- Registrar transferencia notación / Hardware language descriptivo (Verilog/VHDL)
- Restricción física (Retrasos de Entrada, fan-in, fan-out, energía/potencia)

- Describir el avance de la tecnología de dispositivos, desde los tubos de vacío hasta VLSI, desde las arquitecturas mainframe a las arquitecturas en escala warehouse [Familiarizarse]
- Comprender que la tendencia de las arquitecturas modernas de computadores es hacia núcleos múltiples y que el paralelismo es inherente en todos los sistemas de hardware [Usar]
- Explicar las implicancias de los límites de potencia para mejoras adicionales en el rendimiento de los procesadores y también en el aprovechamiento del paralelismo [Usar]
- Relacionar las varias representaciones equivalentes de la funcionalidad de un computador, incluyendo expresiones y puertas lógicas, y ser capaces de utilizar expresiones matemáticas para describir las funciones de circuitos combinacionales y secuenciales sencillos [Familiarizarse]
- Diseñar los componentes básicos de construcción de un computador: unidad aritmético lógica (a nivel de puertas lógicas), unidad central de procesamiento (a nivel de registros de transferencia), memoria (a nivel de registros de transferencia) [Usar]
- Usar herramientas CAD para capturar, sintetizar, y simular bloques de construcción (como ALUs, registros, movimiento entre registros) de un computador simple [Familiarizarse]
- Evaluar el comportamiento de un diagrama de tiempos y funcional de un procesador simple implementado a nivel de circuitos lógicos [Evaluar]

Lecturas: D. Harris and S. Harris (2012), Patt and Patel (2005), Patterson and Hennessy (2004), J. Ashenden (2007), Hennessy and Patterson (2006), Parhami (2005), Stalings (2010), P. Chu (2006)

UNIDAD 2: Representación de datos a nivel máquina (8)	
Competencias:	
Contenido	Objetivos Generales
<ul style="list-style-type: none"> • Bits, Bytes y Words. • Representación de datos numérica y bases numéricas. • Sistemas de punto flotante y punto fijo. • Representaciones con signo y complemento a 2. • Representación de información no numérica (códigos de caracteres, información gráfica) • Representación de registros y arreglos. 	<ul style="list-style-type: none"> • Explicar porqué en computación todo es datos, inclusive las instrucciones [Evaluar] • Explicar las razones de usar formatos alternativos para representar datos numéricos [Familiarizarse] • Describir cómo los enteros negativos se almacenan con representaciones de bit de signo y complemento a 2 [Usar] • Explicar cómo las representaciones de tamaño fijo afectan en la exactitud y la precisión [Usar] • Describir la representación interna de datos no numéricos como caracteres, cadenas, registros y arreglos [Usar] • Convertir datos numéricos de un formato a otro [Usar]
Lecturas: D. Harris and S. Harris (2012), Patt and Patel (2005), Patterson and Hennessy (2004), J.Ashenden (2007), Hennessy and Patterson (2006), Parhami (2005), Stalings (2010), P.Chu (2006)	

UNIDAD 3: Organización de la Máquina a Nivel Ensamblador (8)	
Competencias:	
Contenido	Objetivos Generales
<ul style="list-style-type: none"> • Organización Básica de la Máquina de Von Neumann. • Unidad de Control. • <i>Instruction sets</i> y tipos (manipulación de información, control, I/O) • Assembler y Programación en Lenguaje de Máquina. • Formato de instrucciones. • Modos de direccionamiento. • Llamada a subrutinas y mecanismos de retorno. • I/O e Interrupciones. • Montículo (Heap) vs. Estático vs. Pila vs. Segmentos de código. 	<ul style="list-style-type: none"> • Explicar la organización de la maquina clásica de von Neumann y sus principales unidades funcionales [Familiarizarse] • Describir cómo se ejecuta una instrucción en una máquina de von Neumann con extensión para hebras, sincronización multiproceso y ejecución SIMD (máquina vectorial) [Familiarizarse] • Describir el paralelismo a nivel de instrucciones y sus peligros, y cómo es esto tratado en pipelines de proceso típicos [Familiarizarse] • Resumir cómo se representan las instrucciones, tanto a nivel de máquina bajo el contexto de un ensamblador simbólico [Familiarizarse] • Demostrar cómo se mapean los patrones de lenguajes de alto nivel en notaciones en lenguaje ensamblador o en código máquina [Usar] • Explicar los diferentes formatos de instrucciones, así como el direccionamiento por instrucción, y comparar formatos de tamaño fijo y variable [Usar] • Explicar como las llamadas a subrutinas son manejadas a nivel de ensamblador [Usar] • Explicar los conceptos básicos de interrupciones y operaciones de entrada y salida (I/O) [Familiarizarse] • Escribir segmentos de programa simples en lenguaje ensamblador [Usar] • Ilustrar cómo los bloques constructores fundamentales en lenguajes de alto nivel son implementados a nivel de lenguaje máquina [Usar]
Lecturas: D. Harris and S. Harris (2012), Patt and Patel (2005), Patterson and Hennessy (2004), J.Ashenden (2007), Hennessy and Patterson (2006), Parhami (2005), Stalings (2010), P.Chu (2006)	

UNIDAD 4: Organización funcional (8)	
Competencias:	
Contenido	Objetivos Generales
<ul style="list-style-type: none"> • Implementación de <i>datapath</i>, incluyendo un <i>pipeline</i> de instrucciones, detección de <i>hazards</i> y la resolución. • Control de unidades: Microprogramada. • Instrucción (Pipelining) • Introducción al paralelismo al nivel de instrucción (PNI) 	<ul style="list-style-type: none"> • Comparar implementaciones alternativas de ruta de datos [Evaluar] • Discutir el concepto de puntos de control y la generación de señales de control usando implementaciones a nivel de circuito o microprogramadas [Familiarizarse] • Explicar el paralelismo a nivel de instrucciones básicas usando pipelining y los mayores riesgos que pueden ocurrir [Usar] • Diseñar e implementar un procesador completo, incluyendo ruta de datos y control [Usar] • Calcular la cantidad promedio de ciclos por instrucción de una implementación con procesador y sistema de memoria determinados [Evaluar]
Lecturas: D. Harris and S. Harris (2012), Patt and Patel (2005), Patterson and Hennessy (2004), J.Ashenden (2007), Hennessy and Patterson (2006), Parhami (2005), Stalings (2010), P.Chu (2006)	

UNIDAD 5: Organización y Arquitectura del Sistema de Memoria (8)	
Competencias:	
Contenido	Objetivos Generales
<ul style="list-style-type: none"> • Sistemas de Almacenamiento y su Tecnología. • Jerarquía de Memoria: importancia de la localización temporal y espacial. • Organización y Operaciones de la Memoria Principal. • Latencia, ciclos de tiempo, ancho de banda e <i>inter-leaving</i>. • Memorias caché (Mapeo de direcciones, Tamaño de bloques, Reemplazo y Políticas de almacenamiento) • Multiprocesador coherencia cache / Usando el sistema de memoria para las operaciones de sincronización de memoria / atómica inter-core. • Memoria virtual (tabla de página, TLB) • Manejo de Errores y confiabilidad. • Error de codificación, compresión de datos y la integridad de datos. 	<ul style="list-style-type: none"> • Identificar las principales tecnologías de memoria (Por ejemplo: SRAM, DRAM, Flash, Disco Magnético) y su relación costo beneficio [Familiarizarse] • Explique el efecto de latencia de memoria en tiempo de ejecución [Familiarizarse] • Describir como el uso de jerarquía de memoria (caché, memoria virtual) es aplicado para reducir el atraso efectivo en la memoria [Usar] • Describir los principios de la administración de memoria [Usar] • Explique el funcionamiento de un sistema con gestión de memoria virtual [Usar] • Calcule el tiempo de acceso promedio a memoria bajo varias configuraciones de caché y memoria y para diversas combinaciones de instrucciones y referencias a datos [Evaluar]
Lecturas: D. Harris and S. Harris (2012), Patt and Patel (2005), Patterson and Hennessy (2004), J.Ashenden (2007), Hennessy and Patterson (2006), Parhami (2005), Stalings (2010), P.Chu (2006)	

UNIDAD 6: Interfaz y comunicación (8)	
Competencias:	
Contenido	Objetivos Generales
<ul style="list-style-type: none"> • Fundamentos de I/O: Handshaking, Bbuffering, I/O programadas, interrupciones dirigidas de I/O. • Interrumpir estructuras: interrumpir reconocimiento, vectorizado y priorizado. • Almacenamiento externo, organización física y discos. • Buses: Protocolos de bus, arbitraje, acceso directo a memoria (DMA). • Introducción a Redes: comunicación de redes como otra capa de acceso remoto. • Soporte Multimedia. • Arquitecturas RAID. 	<ul style="list-style-type: none"> • Explicar como las interrupciones son aplicadas para implementar control de entrada-salida y transferencia de datos [Familiarizarse] • Identificar diversos tipos de buses en un sistema computacional [Familiarizarse] • Describir el acceso a datos desde una unidad de disco magnético [Usar] • Comparar organizaciones de red conocidas como organizaciones en bus/Ethernet, en anillo y organizaciones conmutadas versus ruteadas [Evaluar] • Identificar las interfaces entre capas necesarios para el acceso y presentación multimedia, desde la captura de la imagen en almacenamiento remoto, a través del transporte por una red de comunicaciones, hasta la puesta en la memoria local y la presentación final en una pantalla gráfica [Familiarizarse] • Describir las ventajas y limitaciones de las arquitecturas RAID [Familiarizarse]
Lecturas: D. Harris and S. Harris (2012), Patt and Patel (2005), Patterson and Hennessy (2004), J.Ashenden (2007), Hennessy and Patterson (2006), Parhami (2005), Stalings (2010), P.Chu (2006)	

UNIDAD 7: Multiprocesamiento y arquitecturas alternativas (8)	
Competencias:	
Contenido	Objetivos Generales
<ul style="list-style-type: none"> • <i>Power Law</i>. • Ejemplos de <i>sets</i> de instrucciones y arquitecturas SIMD y MIMD. • Redes de interconexión (Hypercube, Shuffle-exchange, Mesh, Crossbar) • Sistemas de memoria de multiprocesador compartido y consistencia de memoria. • Coherencia de cache multiprocesador. 	<ul style="list-style-type: none"> • Discutir el concepto de procesamiento paralelo mas allá del clásico modelo de von Neumann [Evaluar] • Describir diferentes arquitecturas paralelas como SIMD y MIMD [Familiarizarse] • Explicar el concepto de redes de interconexión y mostrar diferentes enfoques [Usar] • Discutir los principales cuidados en los sistemas de multiprocesamiento presentes con respecto a la gestión de memoria y describir como son tratados [Familiarizarse] • Describir las diferencias entre conectores eléctricos en paralelo backplane, interconexión memoria procesador y memoria remota via red, sus implicaciones para la latencia de acceso y el impacto en el rendimiento de un programa [Evaluar]
Lecturas: D. Harris and S. Harris (2012), Patt and Patel (2005), Patterson and Hennessy (2004), J.Ashenden (2007), Hennessy and Patterson (2006), Parhami (2005), Stalings (2010), P.Chu (2006)	

UNIDAD 8: Mejoras de rendimiento (8)	
Competencias:	
Contenido	Objetivos Generales
<ul style="list-style-type: none"> • Arquitectura superescalar. • Predicción de ramificación, Ejecución especulativa, Ejecución fuera de orden. • Prefetching. • Procesadores vectoriales y GPU's • Soporte de hardware para multiprocesamiento. • Escalabilidad. • Arquitecturas alternativas, como VLIW / EPIC y aceleradores y otros tipos de procesadores de propósito especial. 	<ul style="list-style-type: none"> • Describir las arquitecturas superescalares y sus ventajas [Familiarizarse] • Explicar el concepto de predicción de bifurcaciones y su utilidad [Usar] • Caracterizar los costos y beneficios de la precarga prefetching [Evaluar] • Explicar la ejecución especulativa e identifique las condiciones que la justifican [Evaluar] • Discutir las ventajas de rendimiento ofrecida en una arquitectura de multihebras junto con los factores que hacen difícil dar el máximo beneficio de estas [Evaluar] • Describir la importancia de la escalabilidad en el rendimiento [Evaluar]
Lecturas: D. Harris and S. Harris (2012), Patt and Patel (2005), Patterson and Hennessy (2004), J.Ashenden (2007), Hennessy and Patterson (2006), Parhami (2005), Stalings (2010), P.Chu (2006)	

8. Metodología

El profesor del curso presentará clases teóricas de los temas señalados en el programa propiciando la intervención de los alumnos.

El profesor del curso presentará demostraciones para fundamentar clases teóricas.

El profesor y los alumnos realizarán prácticas

Los alumnos deberán asistir a clase habiendo leído lo que el profesor va a presentar. De esta manera se facilitará la comprensión y los estudiantes estarán en mejores condiciones de hacer consultas en clase.

9. Evaluar

Evaluación Continua 1 : 20 %

Examen parcial : 30 %

Evaluación Continua 2 : 20 %

Examen final : 30 %

References

- Harris, David and Sarah Harris (2012). *Digital Design and Computer Architecture*. 2nd. Morgan Kaufmann. ISBN: 978-0123944245.
- Hennessy, J. L. and D. A. Patterson (2006). *Computer Architecture: A Quantitative Approach*. 4th. Morgan Kaufman: San Mateo, CA.
- J.Ashenden, Peter (2007). *Digital Design (Verilog): An Embedded Systems Approach Using Verilog*. Morgan Kaufmann. ISBN: 978-0123695277.
- P.Chu, Pong (2006). *RTL Hardware Design Using VHDL*. 1st. Wiley-Interscience.
- Parhami, Behrooz (2005). *Computer Architecture: From Microprocessors to Supercomputers*. Oxford Univ. Press: New York. ISBN: ISBN 0-19-515455-X.
- Patt, Yale N and Sanjay J Patel (2005). *Introduction to Computing Systems*. 2nd. McGraw Hill.

Patterson, D. A. and J. L. Hennessy (2004). *Computer Organization and Design: The Hardware/Software Interface*. 3rd ed. Morgan Kaufman: San Mateo, CA.

Stalings, William (2010). *Computer Organization and Architecture: Designing for Performance*. 8th. Prentice Hall: Upper Saddle River, NJ.



Universidad Nacional de Colombia (UNAL) Sede
Manizales
Programa Profesional de
Administración de Sistemas Informáticos
SILABO

CS2B1. Desarrollo Basado en Plataformas (Obligatorio)

2022-II

1. Información general	
1.1 Escuela	: Sistemas de Información
1.2 Curso	: CS2B1. Desarrollo Basado en Plataformas
1.3 Semestre	: 3 ^{er} Semestre.
1.4 Prerrequisitos	: CS112. Ciencia de la Computación I. (2 ^{do} Sem)
1.5 Condición	: Obligatorio
1.6 Modalidad de aprendizaje	: Presencial
1.7 horas	: 1 HT; 2 HP; 2 HL;
1.8 Créditos	: 3
2. Profesores	
3. Fundamentación del curso	
<p>El mundo ha cambiado debido al uso de la web y tecnologías relacionadas, el acceso rápido, oportuno y personalizado de la información, a través de la tecnología web, ubicuo y pervasiva; han cambiado la forma de ¿cómo hacemos las cosas?, ¿cómo pensamos? y ¿cómo la industria se desarrolla?.</p> <p>Las tecnologías web, ubicuo y pervasivo se basan en el desarrollo de servicios web, aplicaciones web y aplicaciones móviles, las cuales son necesarias entender la arquitectura, el diseño, y la implementación de servicios web, aplicaciones web y aplicaciones móviles.</p>	
4. Resumen	
1. Introducción 2. Plataformas web 3. Desarrollo de servicios y aplicaciones web 4. Plataformas móviles 5. Aplicaciones Móviles para dispositivos Android	
5. Objetivos Generales	
<ul style="list-style-type: none">• Que el alumno sea capaz de diseño e implementación de servicios, aplicaciones web utilizando herramientas y lenguajes como HTML, CSS, JavaScript (incluyendo AJAX) , back-end scripting y una base de datos, a un nivel intermedio.• Que el alumno sea capaz de desarrollar aplicaciones móviles, administrar servidores web en sistemas basados en UNIX y aplicar técnicas de seguridad en la web a un nivel intermedio.	

6. Contribución a los resultados (*Outcomes*)

Esta disciplina contribuye al logro de los siguientes resultados de la carrera:

- 2) Diseñar, implementar y evaluar una solución basada en computación para cumplir con un conjunto determinado de requisitos computacionales en el contexto de las disciplinas del programa. (**Usar**)
- 3) Comunicarse efectivamente en diversos contextos profesionales. (**Usar**)
- 5) Funcionar efectivamente como miembro o líder de un equipo involucrado en actividades apropiadas a la disciplina del programa. (**Usar**)
- 6) Aplicar fundamentos de teoría de ciencias de la computación y desarrollo de software para producir soluciones basados en computación. (**Usar**)
- 7) Desarrollar tecnología computacional buscando el bien común, aportando con formación humana, capacidades científicas, tecnológicas y profesionales para solucionar problemas sociales de nuestro entorno. (**Usar**)

7. Contenido

UNIDAD 1: Introducción (5)

Competencias:

Contenido	Objetivos Generales
<ul style="list-style-type: none">• Visión general de plataformas (ejemplo, Web, Mobil, Juegos, Industrial)• Programación a través de APIs específicos.• Visión general de lenguajes de plataforma (ejemplo, Objective C, HTML5)• Programación bajo restricciones de plataforma.	<ul style="list-style-type: none">• Describir cómo el desarrollo basado en plataforma difiere de la programación de propósito general [Familiarizarse]• Listar las características de lenguajes de plataforma [Familiarizarse]• Escribir y ejecutar un programa simple basado en plataforma [Familiarizarse]• Listar las ventajas y desventajas de la programación con restricciones de plataforma [Familiarizarse]

Lecturas: grove2009web, annuzzi2013introduction, Cornez2015

UNIDAD 2: Plataformas web (5)	
Competencias:	
Contenido	Objetivos Generales
<ul style="list-style-type: none"> • Lenguajes de programación web (e.g., HTML5, Javascript, PHP, CSS) • Restricciones de las plataformas web: Client-Server, Stateless-Stateful, Caché, Uniform Interface, Layered System, Code on Demand, ReST. • Restricción de plataformas web. • Software como servicio. • Estándares web. 	<ul style="list-style-type: none"> • Diseñar e implementar una aplicación web sencilla [Familiarizarse] • Describir las limitaciones que la web pone a los desarrolladores [Familiarizarse] • Comparar y contrastar la programación web con la programación de proposito general [Familiarizarse] • Describir las diferencias entre software como un servicio y productos de software tradicionales [Familiarizarse] • Discutir cómo los estándares de web impactan el desarrollo de software [Familiarizarse] • Revisar una aplicación web existente con un estándar web actual [Familiarizarse]
Lecturas: fielding2000fielding	

UNIDAD 3: Desarrollo de servicios y aplicaciones web (25)**Competencias:**

Contenido	Objetivos Generales
<ul style="list-style-type: none"> • Describir, identificar y depurar problemas relacionados con el desarrollo de aplicaciones web. • Diseño y desarrollo de aplicaciones web interactivas usando HTML5 y Python. • Utilice MySQL para la gestión de datos y manipular MySQL con Python. • Diseño y desarrollo de aplicaciones web asíncronos utilizando técnicas Ajax. • Uso del lado del cliente dinámico lenguaje de script Javascript y del lado del servidor lenguaje de scripting python con Ajax. • Aplicar las tecnologías XML / JSON para la gestión de datos. • Utilizar los servicios, APIs Web, Ajax y aplicar los patrones de diseño para el desarrollo de aplicaciones web. 	<ul style="list-style-type: none"> • Del lado del servidor lenguaje de scripting python: variables, tipos de datos, operaciones, cadenas, funciones, sentencias de control, matrices, archivos y el acceso a directorios, mantener el estado. [Usar] • Enfoque de programación web usando python incrustado. [Usar] • El acceso y la manipulación de MySQL. [Usar] • El enfoque de desarrollo de aplicaciones web Ajax. [Usar] • DOM y CSS utilizan en JavaScript. [Usar] • Tecnologías de actualización de contenido asíncrono. [Usar] • Objetos XMLHttpRequest utilizar para comunicarse entre clientes y servidores. [Usar] • XML y JSON. [Usar] • XSLT y XPath como mecanismos para transformar documentos XML. [Usar] • Servicios web y APIs (especialmente Google Maps). [Usar] • Marcos Ajax para el desarrollo de aplicaciones web contemporánea. [Usar] • Los patrones de diseño utilizados en aplicaciones web. [Usar]

Lecturas: freeman2011head**UNIDAD 4: Plataformas móviles (5)****Competencias:**

Contenido	Objetivos Generales
<ul style="list-style-type: none"> • Lenguajes de Programación para Móviles. • Principios de diseño: Segregación de Interfaces, Responsabilidad Única, Separación de Responsabilidades, Inversión de Dependencias. • Desafíos con movilidad y comunicación inalámbrica. • Aplicaciones Location-aware. • Rendimiento / Compensación de Potencia. • Restricciones de las Plataformas Móviles. • Tecnologías Emergentes. 	<ul style="list-style-type: none"> • Diseñar e implementar una aplicación móvil para una plataforma móvil dada [Familiarizarse] • Discutir las limitaciones que las plataformas móviles ponen a los desarrolladores [Familiarizarse] • Discutir los principios de diseño que guían la construcción de aplicaciones móviles [Familiarizarse] • Discutir el rendimiento vs pérdida de potencia [Familiarizarse] • Compare y contraste la programación móvil con la programación de propósito general [Familiarizarse]

Lecturas: martin2017clean

UNIDAD 5: Aplicaciones Móviles para dispositivos Android (25)	
Competencias:	
Contenido	Objetivos Generales
<ul style="list-style-type: none"> • The Android Platform • The Android Development Environment • Application Fundamentals • The Activity Class • The Intent Class • Permissions • The Fragment Class • User Interface Classes • User Notifications • The BroadcastReceiver Class • Threads, AsyncTask & Handlers • Alarms • Networking (http class) • Multi-touch & Gestures • Sensors • Location & Maps 	<ul style="list-style-type: none"> • Los estudiantes identifican software necesario y lo instalan en sus ordenadores personales. Los estudiantes realizan varias tareas para familiarizarse con la plataforma Android y Ambiente para el Desarrollo. [Usar] • Los estudiantes construyen aplicaciones que trazan los métodos de devolución de llamada de ciclo de vida emitidas por la plataforma Android y que demuestran el comportamiento de Android cuando los cambios de configuración de dispositivos (por ejemplo, cuando el dispositivo se mueve de vertical a horizontal y viceversa). [Usar] • Los estudiantes construyen aplicaciones que requieren iniciar múltiples actividades a través de ambos métodos estándar y personalizados. [Usar] • Los estudiantes construyen aplicaciones que requieren permisos estándar y personalizados. [Usar] • Los estudiantes construyen una aplicación que utiliza una única base de código, sino que crea diferentes interfaces de usuario dependiendo del tamaño de la pantalla de un dispositivo. [Usar] • Los estudiantes construyen un gestor de listas de tareas pendientes utilizando los elementos de la interfaz de usuario discutidos en clase. La aplicación permite a los usuarios crear nuevos elementos y para mostrarlos en un ListView. [Usar] • Los estudiantes construyen una aplicación que utiliza la información de ubicación para recoger latitud, longitud de los lugares que visitan. [Usar]
Lecturas: annuzzi2013introduction, Cornez2015	

<p>8. Metodología</p> <p>El profesor del curso presentará clases teóricas de los temas señalados en el programa propiciando la intervención de los alumnos.</p> <p>El profesor del curso presentará demostraciones para fundamentar clases teóricas.</p> <p>El profesor y los alumnos realizarán prácticas</p> <p>Los alumnos deberán asistir a clase habiendo leído lo que el profesor va a presentar. De esta manera se facilitará la comprensión y los estudiantes estarán en mejores condiciones de hacer consultas en clase.</p>

9. Evaluar

Evaluación Continua 1 : 20 %

Examen parcial : 30 %

Evaluación Continua 2 : 20 %

Examen final : 30 %



Universidad Nacional de Colombia (UNAL) Sede
Manizales
Programa Profesional de
Administración de Sistemas Informáticos
SILABO

MA102. Cálculo I (Obligatorio)

2022-II

1. Información general	
1.1 Escuela	: Sistemas de Información
1.2 Curso	: MA102. Cálculo I
1.3 Semestre	: 3 ^{er} Semestre.
1.4 Prerrequisitos	: MA100. Matemática I. (1 ^{er} Sem)
1.5 Condición	: Obligatorio
1.6 Modalidad de aprendizaje	: Presencial
1.7 horas	: 2 HT; 4 HP;
1.8 Créditos	: 4
2. Profesores	
3. Fundamentación del curso	
Este curso introduce los primeros conceptos del álgebra lineal, así como los métodos numéricos con un énfasis en la resolución de problemas con el paquete de software libre de código abierto Scilab. La teoría matemática se limita a los fundamentos, mientras que la aplicación efectiva para la resolución de problemas es privilegiada. En cada tópico, se enseña unos cuantos métodos de relevancia para la ingeniería. Los conocimientos sobre estos métodos prepara a los estudiantes para la búsqueda de alternativas más avanzadas, si se lo requiere.	
4. Resumen	
1. Introducción 2. Álgebra lineal 3. Métodos Numéricos	
5. Objetivos Generales	
<ul style="list-style-type: none">• Capacidad para aplicar los conocimientos sobre Matemáticas.• Capacidad para aplicar los conocimientos sobre Ingeniería .• Capacidad para aplicar los conocimientos, técnicas, habilidades y herramientas modernas de la ingeniería moderna para la práctica de la ingeniería.	
6. Contribución a los resultados (Outcomes)	
Esta disciplina contribuye al logro de los siguientes resultados de la carrera:	
1) Analizar un problema computacional complejo y aplicar los principios computacionales y otras disciplinas relevantes para identificar soluciones. (Evaluar)	
6) Aplicar fundamentos de teoría de ciencias de la computación y desarrollo de software para producir soluciones basados en computación. (Evaluar)	
7. Contenido	

UNIDAD 1: Introducción (18)	
Competencias:	
Contenido	Objetivos Generales
<ul style="list-style-type: none"> • Importancia del álgebra lineal y métodos numéricos. Ejemplos. 	<ul style="list-style-type: none"> • Ser capaz de entender los conceptos básicos y la importancia de Álgebra Lineal y Métodos Numéricos.
Lecturas: Anton and Rorres (2014), Chapra and Canale (12015)	

UNIDAD 2: Álgebra lineal (14)	
Competencias:	
Contenido	Objetivos Generales
<ul style="list-style-type: none"> • Álgebra matricial elemental y determinantes. • Espacio nulo y soluciones exactas de sistemas de ecuaciones lineales $Ax = b$: <ul style="list-style-type: none"> – Sistemas tridiagonal y triangular y eliminación gaussiana con y sin giro. – Factorización LU y algoritmo Crout. • Conceptos básicos sobre valores propios y vectores propios <ul style="list-style-type: none"> – Polinomios característicos. – Multiplicaciones algebraicas y geométricas. • Estimación de mínimos cuadrados. • Transformaciones lineales. 	<ul style="list-style-type: none"> • Comprender los conceptos básicos del Álgebra Lineal. • Resolver problemas de transformaciones lineales.
Lecturas: Anton and Rorres (2014), Chapra and Canale (12015)	

UNIDAD 3: Métodos Numéricos (22)	
Competencias:	
Contenido	Objetivos Generales
<ul style="list-style-type: none"> • Fundamentos de soluciones de sistemas de ecuaciones lineales $Ax = b$: métodos de Jacobi y Gauss Seidel • Aplicación de factorizaciones de matriz a la solución de sistemas lineales (descomposición de valores singulares, QR, Cholesky) Cálculo numérico del espacio nulo, rango y número de condición • Conclusión de la raíz: <ul style="list-style-type: none"> – Bisección. – Iteración de punto fijo. – Métodos de Newton-Raphson. • Fundamentos de la interpolación: <ul style="list-style-type: none"> – Interpolaciones polinomiales de Newton y Lagrange. – Interpolación de spline. • Fundamentos de la diferenciación numérica y la aproximación de Taylor. • Aspectos básicos de la integración numérica: <ul style="list-style-type: none"> – Trapecio, punto medio y regla de Simpson – Cuadratura gaussiana • Conceptos básicos sobre las soluciones numéricas a las EDOs: <ul style="list-style-type: none"> – Diferencias finitas; Métodos de Euler y Runge-Kutta – Convertir ODEs de orden superior en un sistema de ODEs de bajo orden. – Métodos de Runge-Kutta para sistemas de ecuaciones – Método simple.XYZ • Breve introducción a las técnicas de optimización: visión general sobre la programación lineal, sistemas lineales acotados, programación cuadrática, descenso gradiente. 	<ul style="list-style-type: none"> • Comprender los conceptos básicos de los métodos numéricos. • Aplicar los métodos más frecuentes para la resolución de problemas matemáticos. • Implementación y aplicación de algoritmos numéricos para la solución de problemas matemáticos utilizando el paquete computacional Scilab open-source. • Aplicación de Scilab para la solución de problemas matemáticos y para trazar graficas.
Lecturas: Anton and Rorres (2014), Chapra and Canale (2015)	

8. Metodología

El profesor del curso presentará clases teóricas de los temas señalados en el programa propiciando la intervención de los alumnos.

El profesor del curso presentará demostraciones para fundamentar clases teóricas.

El profesor y los alumnos realizarán prácticas

Los alumnos deberán asistir a clase habiendo leído lo que el profesor va a presentar. De esta manera se facilitará la comprensión y los estudiantes estarán en mejores condiciones de hacer consultas en clase.

9. Evaluar

Evaluación Continua 1 : 20 %

Examen parcial : 30 %

Evaluación Continua 2 : 20 %

Examen final : 30 %

References

Anton, H. and C. Rorres (2014). *Elementary Linear Algebra, Applications Version*. 11th. Wiley.

Chapra, S.C. and R.P. Canale (2015). *Numerical Methods for Engineers*, 7th. Vol. 1. McGraw-Hill.



Universidad Nacional de Colombia (UNAL) Sede
Manizales
Programa Profesional de
Administración de Sistemas Informáticos
SILABO

CS210. Algoritmos y Estructuras de Datos
(Obligatorio)

2022-II

1. Información general

1.1 Escuela	:	Sistemas de Información
1.2 Curso	:	CS210. Algoritmos y Estructuras de Datos
1.3 Semestre	:	4 ^{to} Semestre.
1.4 Prerrequisitos	:	CS113. Ciencia de la Computación II. (3 ^{er} Sem)
1.5 Condición	:	Obligatorio
1.6 Modalidad de aprendizaje	:	Presencial
1.7 horas	:	2 HT; 2 HP; 2 HL;
1.8 Créditos	:	4

2. Profesores

3. Fundamentación del curso

El fundamento teórico de todas las ramas de la informática descansa sobre los algoritmos y estructuras de datos, este curso brindará a los participantes una introducción a estos temas, formando así una base que servirá para los siguientes cursos en la carrera.

4. Resumen

1. Grafos 2. Matrices Esparzas 3. Arboles Equilibrados

5. Objetivos Generales

- Hacer que el alumno entienda la importancia de los algoritmos para la solución de problemas.
- Introducir al alumno hacia el campo de la aplicación de las estructuras de datos.

6. Contribución a los resultados (Outcomes)

Esta disciplina contribuye al logro de los siguientes resultados de la carrera:

- 1) Analizar un problema computacional complejo y aplicar los principios computacionales y otras disciplinas relevantes para identificar soluciones. (**Usar**)
- 2) Diseñar, implementar y evaluar una solución basada en computación para cumplir con un conjunto determinado de requisitos computacionales en el contexto de las disciplinas del programa. (**Usar**)
- 6) Aplicar fundamentos de teoría de ciencias de la computación y desarrollo de software para producir soluciones basados en computación. (**Usar**)

7. Contenido

UNIDAD 1: Grafos (12)	
Competencias:	
Contenido	Objetivos Generales
<ul style="list-style-type: none"> • Concepto de Grafos. • Grafos Dirigidos y Grafos no Dirigidos. • Utilización de los Grafos. • Medida de la Eficiencia. En tiempo y espacio. • Matrices de Adyacencia. • Matrices de Adyacencia etiquetada. • Listas de Adyacencia. • Implementación de Grafos usando Matrices de Adyacencia. • Implementación de Grafos usando Listas de Adyacencia. • Inserción, Búsqueda y Eliminación de nodos y aristas. • Algoritmos de búsqueda en grafos. 	<ul style="list-style-type: none"> • Adquirir destreza para realizar una implementación correcta. [Usar] • Desarrollar los conocimientos para decidir cuando es mejor usar una técnica de implementación que otra. [Usar]
Lecturas: Cormen et al. (2009), Fager et al. (2014), Knuth (1997), Knuth (1998)	

UNIDAD 2: Matrices Esparzas (8)	
Competencias:	
Contenido	Objetivos Generales
<ul style="list-style-type: none"> • Conceptos Iniciales. • Matrices poco densas • Medida de la Eficiencia en Tiempo y en Espacio • Creación de la matriz esparza estática vs Dinámicas. • Métodos de inserción, búsqueda y eliminación 	<ul style="list-style-type: none"> • Comprender el uso y implementación de matrices esparzas. [Evaluar]
Lecturas: Cormen et al. (2009), Fager et al. (2014), Knuth (1997), Knuth (1998)	

UNIDAD 3: Árboles Equilibrados (16)	
Competencias:	
Contenido	Objetivos Generales
<ul style="list-style-type: none"> • Árboles AVL. • Medida de la Eficiencia. • Rotaciones Simples y Compuestas • Inserción, Eliminación y Búsqueda. • Árboles B , B+ B* y Patricia. 	<ul style="list-style-type: none"> • Comprender las funciones básicas de estas estructuras complejas con el fin de adquirir la capacidad para su implementación. [Evaluar]
Lecturas: Cormen et al. (2009), Fager et al. (2014), Knuth (1997), Knuth (1998)	

8. Metodología

El profesor del curso presentará clases teóricas de los temas señalados en el programa propiciando la intervención de los alumnos.

El profesor del curso presentará demostraciones para fundamentar clases teóricas.

El profesor y los alumnos realizarán prácticas

Los alumnos deberán asistir a clase habiendo leído lo que el profesor va a presentar. De esta manera se facilitará la comprensión y los estudiantes estarán en mejores condiciones de hacer consultas en clase.

9. Evaluar

Evaluación Continua 1 : 20 %

Examen parcial : 30 %

Evaluación Continua 2 : 20 %

Examen final : 30 %

References

- Cormen, Thomas H. et al. (2009). *Introduction to Algorithms*. Third Edition. ISBN: 978-0-262-53305-8. MIT Press.
- Fager, José et al. (2014). *Estructura de datos*. First Edition. Iniciativa Latinoamericana de Libros de Texto Abiertos (LATIN).
- Knuth, Donald E. (1997). *The Art of Computer Programming, Vol. 1: Fundamental Algorithms*. 3rd. Addison-Wesley Professional.
- Knuth, Donald E. (1998). *The art of computer programming, volume 3:Sorting and searching*. 2nd. Addison-Wesley Professional.



Universidad Nacional de Colombia (UNAL) Sede
Manizales
Programa Profesional de
Administración de Sistemas Informáticos
SILABO

CS211. Teoría de la Computación (Obligatorio)

2022-II

1. Información general	
1.1 Escuela	: Sistemas de Información
1.2 Curso	: CS211. Teoría de la Computación
1.3 Semestre	: 4 ^{to} Semestre.
1.4 Prerrequisitos	: CS1D2. Estructuras Discretas II. (2 ^{do} Sem)
1.5 Condición	: Obligatorio
1.6 Modalidad de aprendizaje	: Presencial
1.7 horas	: 2 HT; 2 HP; 2 HL;
1.8 Créditos	: 4
2. Profesores	
3. Fundamentación del curso	
Este curso hace énfasis en los lenguajes formales, modelos de computación y computabilidad, además de incluir fundamentos de la complejidad computacional y de los problemas NP completos.	
4. Resumen	
1. Computabilidad y complejidad básica de autómatas 2. Complejidad Computacional Avanzada 3. Teoría y Computabilidad Avanzada de Autómatas	
5. Objetivos Generales	
<ul style="list-style-type: none">• Que el alumno aprenda los conceptos fundamentales de la teoría de lenguajes formales.	
6. Contribución a los resultados (Outcomes)	
Esta disciplina contribuye al logro de los siguientes resultados de la carrera: 1) Analizar un problema computacional complejo y aplicar los principios computacionales y otras disciplinas relevantes para identificar soluciones. (Evaluar) 6) Aplicar fundamentos de teoría de ciencias de la computación y desarrollo de software para producir soluciones basados en computación. (Evaluar)	
7. Contenido	

UNIDAD 1: Computabilidad y complejidad básica de autómatas (20)	
Competencias:	
Contenido	Objetivos Generales
<ul style="list-style-type: none"> • Máquinas de estado finito. • Expresiones regulares. • Problema de la parada. • Gramáticas libres de contexto. • Introducción a las clases P y NP y al problema P vs. NP. • Introducción y ejemplos de problemas NP- Completos y a clases NP-Completos. • Máquinas de Turing, o un modelo formal equivalente de computación universal. • Máquinas de Turing no determinísticas. • Jerarquía de Chomsky. • La tesis de Church-Turing. • Computabilidad. • Teorema de Rice. • Ejemplos de funciones no computables. • Implicaciones de la no-computabilidad. 	<ul style="list-style-type: none"> • Discute el concepto de máquina de estado finito [Evaluar] • Diseña una máquina de estado finito determinista para aceptar un determinado lenguaje [Evaluar] • Genere una expresión regular para representar un lenguaje específico [Evaluar] • Explique porque el problema de la parada no tiene solución algorítmica [Evaluar] • Diseña una gramática libre de contexto para representar un lenguaje especificado [Evaluar] • Defina las clases P y NP [Evaluar] • Explique el significado de NP-Complejidad [Evaluar] • Explica la tesis de Church-Turing y su importancia [Familiarizarse] • Explica el teorema de Rice y su importancia [Familiarizarse] • Da ejemplos de funciones no computables [Familiarizarse] • Demuestra que un problema es no computable al reducir un problema clásico no computable en base a él [Familiarizarse]
Lecturas: Martin (2010), Linz (2011), Sipser (2012)	

UNIDAD 2: Complejidad Computacional Avanzada (20)	
Competencias:	
Contenido	Objetivos Generales
<ul style="list-style-type: none"> • Revisión de las clases P y NP; introducir espacio P y EXP. • Jerarquía polinomial. • NP completitud (Teorema de Cook). • Problemas NP completos clásicos. • Técnicas de reducción. 	<ul style="list-style-type: none"> • Defina las clases P y NP (También aparece en AL / Automata Básico, Computabilidad y Complejidad) [Evaluar] • Defina la clase P-Space y su relación con la clase EXP [Evaluar] • Explique el significado de NP-Completo (También aparece en AL / Automata Básico, Computabilidad y Complejidad) [Evaluar] • Muestre ejemplos de problemas clásicos en NP - Completo [Evaluar] • Pruebe que un problema es NP- Completo reduciendo un problema conocido como NP-Completo [Evaluar]
Lecturas: Martin (2010), Linz (2011), Sipser (2012), Hopcroft and Ullman (2013)	

UNIDAD 3: Teoría y Computabilidad Avanzada de Autómatas (20)	
Competencias:	
Contenido	Objetivos Generales
<ul style="list-style-type: none"> • Conjuntos y Lenguajes: <ul style="list-style-type: none"> – Lenguajes Regulares. – Revisión de autómatas finitos determinísticos (Deterministic Finite Automata DFAs) – Autómata finito no determinístico (Nondeterministic Finite Automata NFAs) – Equivalencia de DFAs y NFAs. – Revisión de expresiones regulares; su equivalencia con autómatas finitos. – Propiedades de cierre. – Probando no-regularidad de lenguajes, a través del lema de bombeo (Pumping Lemma) o medios alternativos. • Lenguajes libres de contexto: <ul style="list-style-type: none"> – Autómatas de pila (Push-down automata (PDAs) – Relación entre PDA y gramáticas libres de contexto. – Propiedades de los lenguajes libres de contexto. 	<ul style="list-style-type: none"> • Determina la ubicación de un lenguaje en la jerarquía de Chomsky (regular, libre de contexto, enumerable recursivamente) [Evaluar] • Convierte entre notaciones igualmente poderosas para un lenguaje, incluyendo entre estas AFDs, AFNDs, expresiones regulares, y entre AP y GLCs [Evaluar]
Lecturas: Hopcroft and Ullman (2013), Brookshear (1993)	

8. Metodología
<p>El profesor del curso presentará clases teóricas de los temas señalados en el programa propiciando la intervención de los alumnos.</p> <p>El profesor del curso presentará demostraciones para fundamentar clases teóricas.</p> <p>El profesor y los alumnos realizarán prácticas</p> <p>Los alumnos deberán asistir a clase habiendo leído lo que el profesor va a presentar. De esta manera se facilitará la comprensión y los estudiantes estarán en mejores condiciones de hacer consultas en clase.</p>

9. Evaluar
<p>Evaluación Continua 1 : 20 %</p> <p>Examen parcial : 30 %</p> <p>Evaluación Continua 2 : 20 %</p> <p>Examen final : 30 %</p>

References

- Brookshear, J. Glenn (1993). *Teoría de la Computación*. Addison Wesley Iberoamericana.
- Hopcroft, John E. and Jeffrey D. Ullman (2013). *Introducción a la Teoría de Autómatas, Lenguajes y Computación*. Pearson Education.
- Linz, Peter (2011). *An Introduction to Formal Languages and Automata*. 5th. Jones and Bartlett Learning.

Martin, John (2010). *Introduction to Languages and the Theory of Computation*. 4th. McGraw-Hill.
Sipser, Michael (2012). *Introduction to the Theory of Computation*. 3rd. Cengage Learning.



Universidad Nacional de Colombia (UNAL) Sede
Manizales
Programa Profesional de
Administración de Sistemas Informáticos
SILABO

CS271. Bases de Datos I (Obligatorio)

2022-II

1. Información general	
1.1 Escuela	: Sistemas de Información
1.2 Curso	: CS271. Bases de Datos I
1.3 Semestre	: 4 ^{to} Semestre.
1.4 Prerrequisitos	: CS1D3. Álgebra Abstracta. (3 ^{er} Sem)
1.5 Condición	: Obligatorio
1.6 Modalidad de aprendizaje	: Presencial
1.7 horas	: 2 HT; 4 HL;
1.8 Créditos	: 4
2. Profesores	
3. Fundamentación del curso	
La gestión de la información (IM) juega un rol principal en casi todas las áreas donde los computadores son usados. Esta área incluye la captura, digitalización, representación, organización, transformación y presentación de información; algoritmos para mejorar la eficiencia y efectividad del acceso y actualización de información almacenada, modelamiento de datos y abstracción, y técnicas de almacenamiento de archivos físicos. Este también abarca la seguridad de la información, privacidad, integridad y protección en un ambiente compartido. Los estudiantes necesitan ser capaces de desarrollar modelos de datos conceptuales y físicos, determinar que métodos de (IM) y técnicas son apropiados para un problema dado, y ser capaces de seleccionar e implementar una apropiada solución de IM que refleje todas las restricciones aplicables, incluyendo escalabilidad y usabilidad.	
4. Resumen	
1. Sistemas de Bases de Datos 2. Modelado de datos 3. Indexación 4. Bases de Datos Relacionales 5. Lenguajes de Consulta	
5. Objetivos Generales	
<ul style="list-style-type: none">• Que el alumno aprenda a representar información en una base de datos priorizando la eficiencia en la recuperación de la misma• Que el alumno aprenda los conceptos fundamentales de gestión de bases de datos. Esto incluye aspectos de diseño de bases de datos, lenguajes de bases de datos y realización de bases de datos• Discutir el modelo de bases de datos con base en el álgebra relacional, cálculo relacional y en el estudio de sentencias SQL.	

6. Contribución a los resultados (*Outcomes*)

Esta disciplina contribuye al logro de los siguientes resultados de la carrera:

- 1) Analizar un problema computacional complejo y aplicar los principios computacionales y otras disciplinas relevantes para identificar soluciones. (**Evaluar**)
- 3) Comunicarse efectivamente en diversos contextos profesionales. (**Usar**)
- 4) Reconocer las responsabilidades profesionales y hacer juicios informados en el campo profesional de computación con principios éticos. (**Usar**)
- 5) Funcionar efectivamente como miembro o líder de un equipo involucrado en actividades apropiadas a la disciplina del programa. (**Usar**)
- 6) Aplicar fundamentos de teoría de ciencias de la computación y desarrollo de software para producir soluciones basados en computación. (**Evaluar**)
- 7) Desarrollar tecnología computacional buscando el bien común, aportando con formación humana, capacidades científicas, tecnológicas y profesionales para solucionar problemas sociales de nuestro entorno. (**Usar**)

7. Contenido

UNIDAD 1: Sistemas de Bases de Datos (14)

Competencias:

Contenido

Objetivos Generales

- Enfoque y Evolución de Sistemas de Bases de Datos.
- Componentes del Sistema de Bases de Datos.
- Diseño de las funciones principales de un DBMS.
- Arquitectura de base de datos e independencia de datos.
- Uso de un lenguaje de consulta declarativa.
- Sistemas de apoyo a contenido estructurado y / o corriente.
- Enfoques para la gestión de grandes volúmenes de datos (por ejemplo, sistemas de bases de datos NoSQL, uso de MapReduce).

- Explica las características que distinguen un esquema de base de datos de aquellos basados en la programación de archivos de datos [Usar]
- Describe los diseños más comunes para los componentes base de sistemas de bases de datos incluyendo el optimizador de consultas, ejecutor de consultas, administrador de almacenamiento, métodos de acceso y procesador de transacciones [Usar]
- Cita las metas básicas, funciones y modelos de un sistema de bases de datos [Usar]
- Describe los componentes de un sistema de bases de datos y da ejemplos de su uso [Usar]
- Identifica las funciones principales de un SGBD y describe sus roles en un sistema de bases de datos [Usar]
- Explica los conceptos de independencia de datos y su importancia en un sistema de bases de datos [Usar]
- Usa un lenguaje de consulta declarativo para recoger información de una base de datos [Usar]
- Describe las capacidades que las bases de datos brindan al apoyar estructuras y/o la secuencia de flujo de datos, ejm. texto [Usar]
- Describe los enfoques principales para almacenar y procesar largos volúmenes de datos [Usar]

Lecturas: Rob and Coronel (2004), Elmasri and Navathe (2004), Ramakrishnan and Gehrke (2003), Emil Eifrem and Robinson (2015), C.J (2011), Korth and Silberschatz (2002)

UNIDAD 2: Modelado de datos (14)	
Competencias:	
Contenido	Objetivos Generales
<ul style="list-style-type: none"> • Modelado de datos • Modelos conceptuales (e.g., entidad-relación, diagramas UML) • Modelos de hoja de cálculo • Modelos Relacionales. • Modelos orientados a objetos. • Modelos de datos semi-estructurados (expresados usando DTD o XML Schema, por ejemplo) 	<ul style="list-style-type: none"> • Compare y contrasta modelos apropiados de datos, incluyendo estructuras sus estructuras internas, para diversos tipos de datos [Usar] • Describe los conceptos en notación de modelos (ejm. Diagramas Entidad-Relación o UML) y cómo deben de ser usados [Usar] • Define la terminología fundamental a ser usada en un modelo relacional de datos [Usar] • Describe los principios básicos del modelo relacional de datos [Usar] • Aplica los conceptos de modelado y la notación de un modelo relacional de datos [Usar] • Describe los conceptos principales del modelado OO como son identidad de objetos, constructores de tipos, encapsulación, herencia, polimorfismo, y versiones [Usar] • Describe las diferencias entre modelos de datos relacionales y semi-estructurados [Usar] • Da una semi estructura equivalente (ejm. en DTD o Esquema XML) para un esquema relacional dado [Usar]
Lecturas: Simsion and Witt (2004), Elmasri and Navathe (2004), Korth and Silberschatz (2002)	

UNIDAD 3: Indexación (4)	
Competencias:	
Contenido	Objetivos Generales
<ul style="list-style-type: none"> • El impacto de índices en el rendimiento de consultas. • La estructura basica de un indice. • Mantener un buffer de datos en memoria. • Creando indices con SQL. • Indexando texto. • Indexando la web (e.g., web crawling) 	<ul style="list-style-type: none"> • Generar un archivo índice para una colección de recursos [Usar] • Explicar la función de un índice invertido en la localización de un documento en una colección [Usar] • Explicar cómo rechazar y detener palabras que afectan a la indexación [Usar] • Identificar los índices adecuados para determinado el esquema relacional y el conjunto de consultas [Usar] • Estimar el tiempo para recuperar información, cuando son usados los índices comparado con cuando no son usados [Usar] • Describir los desafíos claves en el rastreo web, por ejemplo, la detección de documentos duplicados, la determinación de la frontera de rastreo [Usar]
Lecturas: Whitehorn and Marklyn (2001), Ramakrishnan and Gehrke (2003), Emil Eifrem and Robinson (2015), C.J (2011), Korth and Silberschatz (2002)	

UNIDAD 4: Bases de Datos Relacionales (14)	
Competencias:	
Contenido	Objetivos Generales
<ul style="list-style-type: none"> • Mapeo de esquemas conceptuales a esquemas relacionales. • Entidad y integridad referencial. • Algebra relacional y calculo relacional. • Diseño de bases de datos relacionales. • Dependencia funcional. • Descomposición de un esquema. • Llaves candidatas, SuperLlaves y cierre de un conjunto de atributos. • Formas Normales (BCNF) • Dependencias multi-valoradas (4NF) • Uniendo dependencias (PJNF, 5NF) • Teoría de la representación. 	<ul style="list-style-type: none"> • Prepara un esquema relacional de un modelo conceptual desarrollado usando el modelo entidad-relación [Usar] • Explica y demuestra los conceptos de restricciones de integridad de la entidad e integridad referencial (incluyendo la definición del concepto de clave foránea) [Usar] • Demuestra el uso de las operaciones de álgebra relacional de la teoría matemática de conjuntos (unión, intersección, diferencia, y producto Cartesiano) y de las operaciones de álgebra relacional desarrolladas específicamente para las bases de datos relacionales (selección (restringida), proyección, unión y división) [Usar] • Escribe consultas en álgebra relacional [Usar] • Escribe consultas en cálculo relacional de tuplas [Usar] • Determina la dependencia funcional entre dos o más atributos que son subconjunto de una relación [Usar] • Conecta restricciones expresadas como clave primaria y foránea, con dependencias funcionales [Usar] • Calcula la cerradura de un conjunto de atributos dado dependencias funcionales [Usar] • Determina si un conjunto de atributos forma una superclave y/o una clave candidata de una relación dada dependencias funcionales [Usar] • Evalua una descomposición propuesta, a fin de determinar si tiene una unión sin pérdidas o preservación de dependencias [Usar] • Describe las propiedades de la FNBC, FNUP (forma normal unión de proyecto), 5FN [Usar] • Explica el impacto de la normalización en la eficacia de las operaciones de una base de datos especialmente en la optimización de consultas [Usar] • Describe que es una dependencia de multi valor y cual es el tipo de restricciones que especifica [Usar]
Lecturas: Whitehorn and Marklyn (2001), Ramakrishnan and Gehrke (2003), Emil Eifrem and Robinson (2015), C.J (2011), Korth and Silberschatz (2002)	

UNIDAD 5: Lenguajes de Consulta (12)	
Competencias:	
Contenido	Objetivos Generales
<ul style="list-style-type: none"> • Visión general de lenguajes de base de datos. • SQL (definición de datos, formulacion de consultas, sublenguaje update, restricciones, integridad) • Selecciones • Proyecciones • Select-project-join • Agregaciones y agrupaciones. • Subconsultas. • Entornos QBE de cuarta generación. • Diferentes maneras de invocar las consultas no procedimentales en lenguajes convencionales. • Introducción a otros lenguajes importantes de consulta (por ejemplo, XPATH, SPARQL) • Procedimientos almacenados. 	<ul style="list-style-type: none"> • Crear un esquema relacional de bases de datos en SQL que incorpora restricciones clave y restricciones de integridad de entidad e integridad referencial [Usar] • Usar SQL para crear tablas y devuelve (SELECT) la información de una base de datos [Usar] • Evaluar un conjunto de estrategias de procesamiento de consultas y selecciona la estrategia óptima [Usar] • Crear una consulta no-procedimental al llenar plantillas de relaciones para construir un ejemplo del resultado de una consulta requerida [Usar] • Adicionar consultas orientadas a objetos en un lenguaje stand-alone como C++ o Java (ejm. SELECT ColMethod() FROM Objeto) [Usar] • Escribe un procedimiento almacenado que trata con parámetros y con algo de flujo de control de tal forma que tenga funcionalidad [Usar]
Lecturas: Dietrich (2001), Elmasri and Navathe (2004), Celko (2005), Korth and Silberschatz (2002)	

8. Metodología
<p>El profesor del curso presentará clases teóricas de los temas señalados en el programa propiciando la intervención de los alumnos.</p> <p>El profesor del curso presentará demostraciones para fundamentar clases teóricas.</p> <p>El profesor y los alumnos realizarán prácticas</p> <p>Los alumnos deberán asistir a clase habiendo leído lo que el profesor va a presentar. De esta manera se facilitará la comprensión y los estudiantes estarán en mejores condiciones de hacer consultas en clase.</p>

9. Evaluar
<p>Evaluación Continua 1 : 20 %</p> <p>Examen parcial : 30 %</p> <p>Evaluación Continua 2 : 20 %</p> <p>Examen final : 30 %</p>

References

- C.J, Date (2011). *SQL and Relational Theory: How to Write Accurate SQL Code*. O'Reilly Media.
- Celko, Joe (2005). *Joe Celko's SQL Programming Style*. Elsevier.
- Dietrich, Suzanne W (2001). *Understanding Relational Database Query Languages, First Edition*. Prentice Hall.
- Elmasri, Ramez and Shamkant B. Navathe (2004). *Fundamentals of Database Systems, Fourth Edition*. Addison Wesley.
- Emil Eifrem, Jim Webber and Ian Robinson (2015). *Graph Databases*. 2nd. O'Reilly Media.
- Korth, Henry F. and Abraham Silberschatz (2002). *Fundamentos de Base de Datos*. McGraw-Hill.
- Ramakrishnan, Raghu and Johannes Gehrke (2003). *Database Management Systems*. 3rd. McGraw-Hill.

Rob, Peter and Carlos Coronel (2004). *Database Systems: Design, Implementation and Management, Sixth Edition*. Morgan Kaufmann.

Simsion, Graeme and Graham Witt (2004). *Data Modeling Essentials, Third Edition*. Morgan Kaufmann.

Whitehorn, Mark and Bill Marklyn (2001). *Inside Relational Databases, Second Edition*. Springer.



Universidad Nacional de Colombia (UNAL) Sede
Manizales
Programa Profesional de
Administración de Sistemas Informáticos
SILABO

MA201. Cálculo II (Obligatorio)

2022-II

1. Información general	
1.1 Escuela	: Sistemas de Información
1.2 Curso	: MA201. Cálculo II
1.3 Semestre	: 4 ^{to} Semestre.
1.4 Prerrequisitos	: <ul style="list-style-type: none">• MA101. Matemática II. (2^{do} Sem)• MA102. Cálculo I. (3^{er} Sem)
1.5 Condición	: Obligatorio
1.6 Modalidad de aprendizaje	: Presencial
1.7 horas	: 2 HT; 4 HP;
1.8 Créditos	: 4

2. Profesores

3. Fundamentación del curso
Es una extensión de los cursos de Análisis Matemático I y Análisis Matemático II, tomando en cuenta dos o más variables, indispensables para aquellas materias que requieren trabajar con geometría en curvas y superficies, así como en procesos de búsqueda de puntos extremos.

4. Resumen
1. 2. 3. 4. 5. 6.

5. Objetivos Generales
<ul style="list-style-type: none">• Diferenciar e integrar funciones vectoriales de variable real, entender y manejar el concepto de parametrización. Describir una curva en forma paramétrica.• Describir, analizar, diseñar y formular modelos continuos que dependen de más de una variable.• Establecer relaciones entre diferenciación e integración y aplicar el cálculo diferencial e integral ala resolución de problemas geométricos y de optimización.

6. Contribución a los resultados (<i>Outcomes</i>)
Esta disciplina contribuye al logro de los siguientes resultados de la carrera: <ol style="list-style-type: none">1) Analizar un problema computacional complejo y aplicar los principios computacionales y otras disciplinas relevantes para identificar soluciones. (Evaluar)6) Aplicar fundamentos de teoria de ciencias de la computación y desarrollo de software para producir soluciones basados en computación. (Evaluar)

7. Contenido

UNIDAD 1: (8)	
Competencias:	
Contenido	Objetivos Generales
<ul style="list-style-type: none">• R^3 como espacio euclídeo y álgebra .• Superficies básicas en el espacio.	<ul style="list-style-type: none">• Manejar el álgebra vectorial en R^3 [Usar].• Identificar tipos de superficies en el espacio [Usar].• Graficar superficies básicas [Usar].
Lecturas: Apóstol (1973), Simmons (1995)	

UNIDAD 2: (20)	
Competencias:	
Contenido	Objetivos Generales
<ul style="list-style-type: none">• Funciones vectoriales de variable real. Reparametrizaciones• Diferenciación e integración• Velocidad, aceleración , curvatura, torsión	<ul style="list-style-type: none">• Describir las diferentes características de una curva [Usar].
Lecturas: Apóstol (1973), Simmons (1995)	

UNIDAD 3: (20)	
Competencias:	
Contenido	Objetivos Generales
<ul style="list-style-type: none">• Curvas de nivel• Límites y continuidad• Diferenciación	<ul style="list-style-type: none">• Graficar campos escalares• Discutir la existencia de un límite y la continuidad de un campo escalar [Usar].• Calcular derivadas parciales y totales [Usar].
Lecturas: Apóstol (1973), Bartle (1976), Simmons (1995)	

UNIDAD 4: (12)	
Competencias:	
Contenido	Objetivos Generales
<ul style="list-style-type: none">• Máximos y mínimos• Multiplicadores de Lagrange	<ul style="list-style-type: none">• Interpretar la noción de gradiente en curvas de nivel y en superficies de nivel [Usar].• Usar técnicas para hallar extremos [Usar].
Lecturas: Apóstol (1973), Simmons (1995), Bartle (1976)	

UNIDAD 5: (12)	
Competencias:	
Contenido	Objetivos Generales
<ul style="list-style-type: none"> • Integración de Riemann • Integración sobre regiones • Cambio de coordenadas • Aplicaciones 	<ul style="list-style-type: none"> • Reconocer regiones de integración adecuadas [Usar]. • Realizar cambios de coordenadas adecuados [Usar]. • Aplicar la integración múltiple a problemas [Usar].
Lecturas: Apóstol (1973)	

UNIDAD 6: (18)	
Competencias:	
Contenido	Objetivos Generales
<ul style="list-style-type: none"> • Integrales de línea • Campos conservativos • Integrales de superficie 	<ul style="list-style-type: none"> • Calcular la integral de línea de campos vectoriales [Usar]. • Reconocer campos conservativos [Usar]. • Hallar funciones potenciales de campos conservativos [Usar]. • Hallar integrales de superficies y aplicarlas [Usar].
Lecturas: Apóstol (1973)	

8. Metodología
<p>El profesor del curso presentará clases teóricas de los temas señalados en el programa propiciando la intervención de los alumnos.</p> <p>El profesor del curso presentará demostraciones para fundamentar clases teóricas.</p> <p>El profesor y los alumnos realizarán prácticas</p> <p>Los alumnos deberán asistir a clase habiendo leído lo que el profesor va a presentar. De esta manera se facilitará la comprensión y los estudiantes estarán en mejores condiciones de hacer consultas en clase.</p>

9. Evaluar
<p>Evaluación Continua 1 : 20 %</p> <p>Examen parcial : 30 %</p> <p>Evaluación Continua 2 : 20 %</p> <p>Examen final : 30 %</p>

References

- Apóstol, Tom M (1973). *Calculus*. Vol. II. Editorial Reverté.
- Bartle, Robert G. (1976). *The Elements of Real Analysis*. Wiley; 2 edition. ISBN: 047105464X.
- Simmons, George F (1995). *Calculus With Analytic Geometry*. McGraw-Hill Science/Engineering. ISBN: 0070576424.



Universidad Nacional de Colombia (UNAL) Sede
Manizales
Programa Profesional de
Administración de Sistemas Informáticos
SILABO

MA203. Estadística y Probabilidades (Obligatorio)

2022-II

1. Información general	
1.1 Escuela	: Sistemas de Información
1.2 Curso	: MA203. Estadística y Probabilidades
1.3 Semestre	: 4 ^{to} Semestre.
1.4 Prerrequisitos	: MA102. Cálculo I. (3 ^{er} Sem)
1.5 Condición	: Obligatorio
1.6 Modalidad de aprendizaje	: Presencial
1.7 horas	: 2 HT; 2 HP; 2 HL;
1.8 Créditos	: 4
2. Profesores	
3. Fundamentación del curso	
Provee de una introducción a la teoría de las probabilidades e inferencia estadística con aplicaciones, necesarias en el análisis de datos, diseño de modelos aleatorios y toma de decisiones.	
4. Resumen	
1. Tipo de variable 2. Estadísticas descriptiva 3. Estadística inferencial	
5. Objetivos Generales	
<ul style="list-style-type: none">• Capacidad para diseñar y conducir experimentos, así como usar tecnología como para analizar e interpretar datos.• Capacidad para identificar, formular y resolver problemas reales.	
6. Contribución a los resultados (Outcomes)	
Esta disciplina contribuye al logro de los siguientes resultados de la carrera:	
1) Analizar un problema computacional complejo y aplicar los principios computacionales y otras disciplinas relevantes para identificar soluciones. (Evaluar)	
6) Aplicar fundamentos de teoría de ciencias de la computación y desarrollo de software para producir soluciones basados en computación. (Evaluar)	
7. Contenido	

UNIDAD 1: Tipo de variable (6)	
Competencias:	
Contenido	Objetivos Generales
<ul style="list-style-type: none"> • Tipo de variable: Continua, discreta. 	<ul style="list-style-type: none"> • Clasificar las variables relevantes identificadas según su tipo: continuo (intervalo y razón), categórico (nominal, ordinario, dicotómico). • Identificar las variables relevantes de un sistema utilizando un enfoque de proceso.
Lecturas: M.Ross (2014), Mendenhall (2014)	

UNIDAD 2: Estadísticas descriptiva (6)	
Competencias:	
Contenido	Objetivos Generales
<ul style="list-style-type: none"> • Tendencia Central (Media, mediana, modo) • Dispersión (Rango, desviación estándar, cuartil) • Gráficos: histograma, boxplot, etc. : Capacidad de comunicación. 	<ul style="list-style-type: none"> • Utilizar medidas de tendencia central y medidas de dispersión para describir los datos recopilados. • Utilizar gráficos para comunicar las características de los datos recopilados.
Lecturas: M.Ross (2014), Mendenhall (2014)	

UNIDAD 3: Estadística inferencial (6)	
Competencias:	
Contenido	Objetivos Generales
<ul style="list-style-type: none"> • Determinación del tamaño de la muestra • Intervalo de confianza • Tipo I y error del tipo II • Tipo de distribución • Prueba de hipótesis (t-student, medias, proporciones y ANOVA) • Relaciones entre variables: correlación, regresión. 	<ul style="list-style-type: none"> • Proponer preguntas e hipótesis de interés. • Analizar los datos recopilados utilizando diferentes herramientas estadísticas para responder preguntas de interés. • Dibujar conclusiones basadas en el análisis realizado.
Lecturas: M.Ross (2014), Mendenhall (2014)	

8. Metodología
<p>El profesor del curso presentará clases teóricas de los temas señalados en el programa propiciando la intervención de los alumnos.</p> <p>El profesor del curso presentará demostraciones para fundamentar clases teóricas.</p> <p>El profesor y los alumnos realizarán prácticas</p> <p>Los alumnos deberán asistir a clase habiendo leído lo que el profesor va a presentar. De esta manera se facilitará la comprensión y los estudiantes estarán en mejores condiciones de hacer consultas en clase.</p>

9. Evaluar

Evaluación Continua 1 : 20 %

Examen parcial : 30 %

Evaluación Continua 2 : 20 %

Examen final : 30 %

References

M.Ross, Sheldon (2014). *Introduction to Probability and Statistics for Engineers and Scientists*. 5th. Academic Press.
Mendenhall, Beaver (2014). *Introducción a la probabilidad y estadística*. 13th. Cengage Learning.



**Universidad Nacional de Colombia (UNAL) Sede
Manizales
Programa Profesional de
Administración de Sistemas Informáticos
SILABO**

CS212. Análisis y Diseño de Algoritmos (Obligatorio)

2022-II

1. Información general	
1.1 Escuela	: Sistemas de Información
1.2 Curso	: CS212. Análisis y Diseño de Algoritmos
1.3 Semestre	: 5 ^{to} Semestre.
1.4 Prerrequisitos	: CS210. Algoritmos y Estructuras de Datos. (4 ^{to} Sem)
1.5 Condición	: Obligatorio
1.6 Modalidad de aprendizaje	: Presencial
1.7 horas	: 2 HT; 2 HP; 2 HL;
1.8 Créditos	: 4

2. Profesores

3. Fundamentación del curso
<p>Un algoritmo es, esencialmente, un conjunto bien definido de reglas o instrucciones que permitan resolver un problema computacional. El estudio teórico del desempeño de los algoritmos y los recursos utilizados por estos, generalmente tiempo y espacio, nos permite evaluar si un algoritmo es adecuado para un resolver un problema específico, compararlo con otros algoritmos para el mismo problema o incluso delimitar la frontera entre lo viable y lo imposible. Esta materia es tan importante que incluso Donald E. Knuth definió a Ciencia de la Computación como el estudio de algoritmos.</p> <p>En este curso serán presentadas las técnicas más comunes utilizadas en el análisis y diseño de algoritmos eficientes, con el propósito de aprender los principios fundamentales del diseño, implementación y análisis de algoritmos para la solución de problemas computacionales.</p>

4. Resumen
1. Análisis Básico 2. Estrategias Algorítmicas 3. Algoritmos y Estructuras de Datos fundamentales 4. Computabilidad y complejidad básica de autómatas 5. Estructuras de Datos Avanzadas y Análisis de Algoritmos

5. Objetivos Generales
<ul style="list-style-type: none">• Desarrollar la capacidad para evaluar la complejidad y calidad de algoritmos propuestos para un determinado problema.• Estudiar los algoritmos más representativos, introductorios de las clases más importantes de problemas tratados en computación.• Desarrollar la capacidad de resolución de problemas algorítmicos utilizando los principios fundamentales de diseño de algoritmos aprendidos.• Ser capaz de responder a las siguientes preguntas cuando le sea presentado un nuevo algoritmo: ¿Cuán buen desempeño tiene?, ¿Existe una mejor forma de resolver el problema?

6. Contribución a los resultados (*Outcomes*)

Esta disciplina contribuye al logro de los siguientes resultados de la carrera:

- 1) Analizar un problema computacional complejo y aplicar los principios computacionales y otras disciplinas relevantes para identificar soluciones. (**E**valuar)
- 5) Funcionar efectivamente como miembro o líder de un equipo involucrado en actividades apropiadas a la disciplina del programa. (**U**sar)
- 6) Aplicar fundamentos de teoría de ciencias de la computación y desarrollo de software para producir soluciones basados en computación. (**U**sar)

7. Contenido

UNIDAD 1: Análisis Básico (10)

Competencias:

Contenido

Objetivos Generales

- Diferencias entre el mejor, el esperado y el peor caso de un algoritmo.
- Análisis asintótico de complejidad de cotas superior y esperada.
- Definición formal de la Notación Big O.
- Clases de complejidad como constante, logarítmica, lineal, cuadrática y exponencial.
- Uso de la notación Big O.
- Relaciones recurrentes.
- Análisis de algoritmos iterativos y recursivos.
- Teorema Maestro y Árboles Recursivos.

- Explique a que se refiere con “mejor”, “esperado” y “peor” caso de comportamiento de un algoritmo [Evaluar]
- En el contexto de algoritmos específicos, identifique las características de datos y/o otras condiciones o suposiciones que lleven a diferentes comportamientos [Evaluar]
- Determine informalmente el tiempo y el espacio de complejidad de diferentes algoritmos [Evaluar]
- Indique la definición formal de Big O [Evaluar]
- Lista y contraste de clases estándares de complejidad [Evaluar]
- Use la notación formal de la Big O para dar límites superiores asintóticos en la complejidad de tiempo y espacio de los algoritmos [Evaluar]
- Usar la notación formal Big O para dar límites de casos esperados en el tiempo de complejidad de los algoritmos [Evaluar]
- Explicar el uso de la notación theta grande, omega grande y o pequeña para describir la cantidad de trabajo hecho por un algoritmo [Evaluar]
- Usar relaciones recurrentes para determinar el tiempo de complejidad de algoritmos recursivamente definidos [Evaluar]
- Resuelve relaciones de recurrencia básicas, por ejemplo, usando alguna forma del Teorema Maestro [Evaluar]

Lecturas: Kleinberg and Tardos (2005), Dasgupta, Papadimitriou, and Vazirani (2006), Rivest and Stein (2009), Sedgewick and Flajolet (2013), Knuth (1997)

UNIDAD 2: Estrategias Algorítmicas (30)**Competencias:****Contenido**

- Algoritmos de fuerza bruta.
- Algoritmos voraces.
- Divide y vencerás.
- Programación Dinámica.

Objetivos Generales

- Para cada una de las estrategias (fuerza bruta, algoritmo goloso, divide y vencerás, recursividad en reversa y programación dinámica), identifica un ejemplo práctico en el cual se pueda aplicar [Evaluar]
- Utiliza un enfoque voraz para resolver un problema específico y determina si la regla escogida lo guía a una solución óptima [Evaluar]
- Usa un algoritmo de divide-y-vencerás para resolver un determinado problema [Evaluar]
- Usa programación dinámica para resolver un problema determinado [Evaluar]
- Determina el enfoque algorítmico adecuado para un problema [Evaluar]

Lecturas: Kleinberg and Tardos (2005), Dasgupta, Papadimitriou, and Vazirani (2006), Rivest and Stein (2009), Alsuwaiyel (1999)

UNIDAD 3: Algoritmos y Estructuras de Datos fundamentales (10)	
Competencias:	
Contenido	Objetivos Generales
<ul style="list-style-type: none"> • Algoritmos numéricos simples, tales como el cálculo de la media de una lista de números, encontrar el mínimo y máximo. • Algoritmos de búsqueda secuencial y binaria. • Algoritmos de ordenamiento de peor caso cuadrático (selección, inserción) • Algoritmos de ordenamiento con peor caso o caso promedio en $O(N \lg N)$ (Quicksort, Heapsort, Mergesort) • Grafos y algoritmos en grafos: <ul style="list-style-type: none"> – Representación de grafos (ej., lista de adyacencia, matriz de adyacencia) – Recorrido en profundidad y amplitud • Montículos (Heaps) • Grafos y algoritmos en grafos: <ul style="list-style-type: none"> – Problema de corte máximo y mínimo – Búsqueda local 	<ul style="list-style-type: none"> • Implementar algoritmos numéricos básicos [Evaluar] • Implementar algoritmos de búsqueda simple y explicar las diferencias en sus tiempos de complejidad [Evaluar] • Ser capaz de implementar algoritmos de ordenamiento comunes cuadráticos y $O(N \log N)$ [Evaluar] • Discutir el tiempo de ejecución y eficiencia de memoria de los principales algoritmos de ordenamiento, búsqueda y hashing [Usar] • Discutir factores otros que no sean eficiencia computacional que influyan en la elección de algoritmos, tales como tiempo de programación, mantenibilidad, y el uso de patrones específicos de la aplicación en los datos de entrada [Familiarizarse] • Resolver problemas usando algoritmos básicos de grafos, incluyendo búsqueda por profundidad y búsqueda por amplitud [Evaluar] • Demostrar habilidad para evaluar algoritmos, para seleccionar de un rango de posibles opciones, para proveer una justificación por esa selección, y para implementar el algoritmo en un contexto en específico [Evaluar] • Describir la propiedad del heap y el uso de heaps como una implementación de colas de prioridad [Evaluar] • Resolver problemas usando algoritmos de grafos, incluyendo camino más corto de una sola fuente y camino más corto de todos los pares, y como mínimo un algoritmo de arbol de expansion minima [Evaluar]
Lecturas: Kleinberg and Tardos (2005), Dasgupta, Papadimitriou, and Vazirani (2006), Rivest and Stein (2009), Sedgewick and Wayne (2011), Goodrich and Tamassia (2009)	

UNIDAD 4: Computabilidad y complejidad básica de autómatas (2)	
Competencias:	
Contenido	Objetivos Generales
<ul style="list-style-type: none"> • Introducción a las clases P y NP y al problema P vs. NP. • Introducción y ejemplos de problemas NP- Completos y a clases NP-Completos. 	<ul style="list-style-type: none"> • Define las clases P y NP [Familiarizarse] • Explique el significado de NP-Complejidad [Familiarizarse]
Lecturas: Kleinberg and Tardos (2005), Dasgupta, Papadimitriou, and Vazirani (2006), Rivest and Stein (2009)	

UNIDAD 5: Estructuras de Datos Avanzadas y Análisis de Algoritmos (8)	
Competencias:	
Contenido	Objetivos Generales
<ul style="list-style-type: none"> • Grafos (ej. Ordenamiento Topológico, encontrando componentes fuertemente conectados) • Algoritmos Teórico-Numéricos (Aritmética Modular, Prueba del Número Primo, Factorización Entera) • Algoritmos aleatorios. • Análisis amortizado. • Análisis Probabilístico. 	<ul style="list-style-type: none"> • Entender el mapeamiento de problemas del mundo real a soluciones algorítmicas (ejemplo, problemas de grafos, programas lineales, etc) [Familiarizarse] • Seleccionar y aplicar técnicas de algoritmos avanzadas (ejemplo, randomización, aproximación) para resolver problemas reales [Usar] • Seleccionar y aplicar técnicas avanzadas de análisis (ejemplo, amortizado, probabilístico, etc) para algoritmos [Usar]
Lecturas: Kleinberg and Tardos (2005), Dasgupta, Papadimitriou, and Vazirani (2006), Rivest and Stein (2009), Tarjan (1983), Rawlins (1992)	

8. Metodología
<p>El profesor del curso presentará clases teóricas de los temas señalados en el programa propiciando la intervención de los alumnos.</p> <p>El profesor del curso presentará demostraciones para fundamentar clases teóricas.</p> <p>El profesor y los alumnos realizarán prácticas</p> <p>Los alumnos deberán asistir a clase habiendo leído lo que el profesor va a presentar. De esta manera se facilitará la comprensión y los estudiantes estarán en mejores condiciones de hacer consultas en clase.</p>

9. Evaluar
<p>Evaluación Continua 1 : 20 %</p> <p>Examen parcial : 30 %</p> <p>Evaluación Continua 2 : 20 %</p> <p>Examen final : 30 %</p>

References

- Alsuwaiyel, H. (1999). *Algorithms: Design Techniques and Analysis*. World Scientific. ISBN: 9789810237400.
- Dasgupta, S., C. Papadimitriou, and U. Vazirani (2006). *Algorithms*. McGraw-Hill Education. ISBN: 9780073523408.
- Goodrich, Michael T. and Roberto Tamassia (2009). *Algorithm Design: Foundations, Analysis and Internet Examples*. 2nd. John Wiley & Sons, Inc. ISBN: 0470088540, 9780470088548.
- Kleinberg, Jon and Eva Tardos (2005). *Algorithm Design*. Addison-Wesley Longman Publishing Co., Inc. ISBN: 0321295358.
- Knuth, D.E. (1997). *The Art of Computer Programming: Fundamental algorithms Vol 1*. Third Edition. Addison-Wesley. ISBN: 9780201896831.
- Rawlins, G.J.E. (1992). *Compared to What?: An Introduction to the Analysis of Algorithms*. Computer Science Press. ISBN: 9780716782438.
- Rivest, Thomas H. Cormen; Charles E. Leiserson ; Ronald L. and Clifford Stein (2009). *Introduction to Algorithms, Third Edition*. 3rd. The MIT Press. ISBN: 0262033844.
- Sedgewick, R. and P. Flajolet (2013). *An Introduction to the Analysis of Algorithms*. Pearson Education. ISBN: 9780133373486.
- Sedgewick, R. and K. Wayne (2011). *Algorithms*. Pearson Education. ISBN: 9780132762564.
- Tarjan, Robert Endre (1983). *Data Structures and Network Algorithms*. Society for Industrial and Applied Mathematics. ISBN: 0-89871-187-8.



Universidad Nacional de Colombia (UNAL) Sede
Manizales
Programa Profesional de
Administración de Sistemas Informáticos
SILABO

CS272. Bases de Datos II (Obligatorio)

2022-II

1. Información general	
1.1 Escuela	: Sistemas de Información
1.2 Curso	: CS272. Bases de Datos II
1.3 Semestre	: 5 ^{to} Semestre.
1.4 Prerrequisitos	: CS271. Bases de Datos I. (4 ^{to} Sem)
1.5 Condición	: Obligatorio
1.6 Modalidad de aprendizaje	: Presencial
1.7 horas	: 1 HT; 2 HP; 2 HL;
1.8 Créditos	: 3
2. Profesores	
3. Fundamentación del curso	
La Gestión de la Información (<i>IM-Information Management</i>) juega un rol principal en casi todas las áreas donde los computadores son usados. Esta área incluye la captura, digitalización, representación, organización, transformación y presentación de información; algoritmos para mejorar la eficiencia y efectividad del acceso y actualización de información almacenada, modelamiento de datos y abstracción, y técnicas de almacenamiento de archivos físicos. Este también abarca la seguridad de la información, privacidad, integridad y protección en un ambiente compartido. Los estudiantes necesitan ser capaces de desarrollar modelos de datos conceptuales y físicos, determinar que métodos de IM y técnicas son apropiados para un problema dado, y ser capaces de seleccionar e implementar una apropiada solución de IM que refleje todas las restricciones aplicables, incluyendo escalabilidad y usabilidad.	
4. Resumen	
1. Diseño Físico de Bases de Datos 2. Procesamiento de Transacciones 3. Almacenamiento y Recuperación de Información 4. Bases de Datos Distribuidas	
5. Objetivos Generales	
<ul style="list-style-type: none">• Hacer que el alumno entienda las diferentes aplicaciones que tienen las bases de datos, en las diversas áreas de conocimiento.• Mostrar las formas adecuadas de almacenamiento de información basada en sus diversos enfoques y su posterior recuperación de información.	

6. Contribución a los resultados (*Outcomes*)

Esta disciplina contribuye al logro de los siguientes resultados de la carrera:

- 1) Analizar un problema computacional complejo y aplicar los principios computacionales y otras disciplinas relevantes para identificar soluciones. (**Evaluar**)
- 4) Reconocer las responsabilidades profesionales y hacer juicios informados en el campo profesional de computación con principios éticos. (**Evaluar**)
- 6) Aplicar fundamentos de teoría de ciencias de la computación y desarrollo de software para producir soluciones basados en computación. (**Evaluar**)
- 7) Desarrollar tecnología computacional buscando el bien común, aportando con formación humana, capacidades científicas, tecnológicas y profesionales para solucionar problemas sociales de nuestro entorno. (**Evaluar**)

7. Contenido

UNIDAD 1: Diseño Físico de Bases de Datos (10)

Competencias:

Contenido	Objetivos Generales
<ul style="list-style-type: none">• Almacenamiento y estructura de archivos.• Archivos indexados.• Archivos Hash.• Archivos de Firma.• Árboles B.• Archivos con índice denso.• Archivos con registros de tamaño variable.• Eficiencia y Afinación de Bases de Datos.	<ul style="list-style-type: none">• Explica los conceptos de registro, tipos de registro, y archivos, así como las diversas técnicas para colocar registros de archivos en un disco [Usar]• Da ejemplos de la aplicación de índices primario, secundario y de agrupamiento [Usar]• Distingue entre un índice no denso y uno denso [Usar]• Implementa índices de multinivel dinámicos usando árboles-B [Usar]• Explica la teoría y la aplicación de técnicas de hash internas y externas [Usar]• Usa técnicas de hasp para facilitar la expansión de archivos dinámicos [Usar]• Describe las relaciones entre hashing, compresión, y búsquedas eficientes en bases de datos [Usar]• Evalúa el costo y beneficio de diversos esquemas de hashing [Usar]• Explica como el diseño físico de una base de datos afecta la eficiencia de las transacciones en ésta [Usar]
Lecturas: Burleson (2004), Celko (2005)	

UNIDAD 2: Procesamiento de Transacciones (12)	
Competencias:	
Contenido	Objetivos Generales
<ul style="list-style-type: none"> • Transacciones. • Fallo y recuperación. • Control concurrente. • Interacción de gestión de transacciones con el almacenamiento, especialmente en almacenamiento. 	<ul style="list-style-type: none"> • Crear una transacción mediante la incorporación de SQL en un programa de aplicación [Usar] • Explicar el concepto de confirmaciones implícitas [Usar] • Describir los problemas específicos para la ejecución de una transacción eficiente [Usar] • Explicar cuando y porqué se necesita un <i>rollback</i>, y cómo registrar todo asegura un <i>rollback</i> adecuado [Usar] • Explicar el efecto de diferentes niveles de aislamiento sobre los mecanismos de control de concurrencia [Usar] • Elegir el nivel de aislamiento adecuado para la aplicación de un protocolo de transacción especificado [Usar] • Identificar los límites apropiados de la transacción en programas de aplicación [Usar]
Lecturas: Philip A. Bernstein (1997), Ramez Elmasri (2004)	

UNIDAD 3: Almacenamiento y Recuperación de Información (10)	
Competencias:	
Contenido	Objetivos Generales
<ul style="list-style-type: none"> • Documentos, publicación electrónica, markup, y lenguajes markup. • Tries, archivos invertidos, Árboles PAT, archivos de firma, indexación. • Análisis Morfológico, stemming, frases, stop lists. • Distribuciones de frecuencia de términos, incertidumbre, fuzificación (fuzzyness), ponderación. • Espacio vectorial, probabilidad, lógica, y modelos avanzados. • Necesidad de Información , Relevancia, evaluación, efectividad. • Thesauri, ontologías, clasificación y categorización, metadata. • Información bibliográfica, bibliometría, citasiones. • Enrutamiento y filtrado. • Búsqueda multimedia. • Información de resumen y visualización. • Búsqueda por facetas (por ejemplo, el uso de citas, palabras clave, esquemas de clasificación). • Librerías digitales. • Digitalización, almacenamiento, intercambio, objetos digitales, composición y paquetes. • Metadata y catalogación. • Nombramiento, repositorios, archivos • Archivamiento y preservación, integrdad • Espacios (Conceptual, geográfico, 2/3D, Realidad virtual) • Arquitecturas (agentes, autobuses, envolturas / mediadores), de interoperabilidad. • Servicios (búsqueda, de unión, de navegación, y así sucesivamente). • Gestión de derechos de propiedad intelectual, la privacidad y la protección (marcas de agua). 	<ul style="list-style-type: none"> • Explica los conceptos básicos de almacenamiento y recuperación de la información [Usar] • Describe que temas son específicos para una recuperación de la información eficiente [Usar] • Da aplicaciones de estrategias alternativas de búsqueda y explica porqué una estrategia en particular es apropiada para una aplicación [Usar] • Diseña e implementa un sistema de almacenamiento y recuperación de la información o librería digital de tamaño pequeño a mediano [Usar] • Describe algunas de las soluciones técnicas a los problemas relacionados al archivamiento y preservación de la información en una librería digital [Usar]
Lecturas: Peter Brusilovsky (1998), Ramez Elmasri (2004)	

UNIDAD 4: Bases de Datos Distribuidas (36)	
Competencias:	
Contenido	Objetivos Generales
<ul style="list-style-type: none"> • DBMS Distribuidas <ul style="list-style-type: none"> – Almacenamiento de datos distribuido – Procesamiento de consultas distribuido – Modelo de transacciones distribuidas – Soluciones homogéneas y heterogéneas – Bases de datos distribuidas cliente-servidor • Parallel DBMS <ul style="list-style-type: none"> – Arquitecturas paralelas DBMS: memoria compartida, disco compartido, nada compartido; – Aceleración y ampliación, por ejemplo, el uso del modelo de procesamiento MapReduce – Replicación de información y modelos de consistencia débil 	<ul style="list-style-type: none"> • Explicar las técnicas usadas para la fragmentación de datos, replicación, y la asignación durante el proceso de diseño de base de datos distribuida [Usar] • Evaluar estrategias simples para la ejecución de una consulta distribuida para seleccionar una estrategia que minimice la cantidad de transferencia de datos [Usar] • Explicar como el protocolo de dos fases de <i>commit</i> es usado para resolver problemas de transacciones que acceden a bases de datos almacenadas en múltiples nodos [Usar] • Describir el control concurrente distribuido basados en técnicas de copia distinguidos y el método de votación. [Usar] • Describir los tres niveles del software en el modelo cliente servidor [Usar]
Lecturas: M. Tamer Ozsu (1999)	

8. Metodología
<p>El profesor del curso presentará clases teóricas de los temas señalados en el programa propiciando la intervención de los alumnos.</p> <p>El profesor del curso presentará demostraciones para fundamentar clases teóricas.</p> <p>El profesor y los alumnos realizarán prácticas</p> <p>Los alumnos deberán asistir a clase habiendo leído lo que el profesor va a presentar. De esta manera se facilitará la comprensión y los estudiantes estarán en mejores condiciones de hacer consultas en clase.</p>

9. Evaluar
<p>Evaluación Continua 1 : 20 %</p> <p>Examen parcial : 30 %</p> <p>Evaluación Continua 2 : 20 %</p> <p>Examen final : 30 %</p>

References

- Burleson, Donald K. (2004). *Physical Database Design Using Oracle*. CRC Press.
- Celko, Joe (2005). *Joe Celko's SQL Programming Style*. Elsevier.
- M. Tamer Ozsu, Patrick Valduriez (1999). *Principles of Distributed Database Systems, Second Edition*. Prentice Hall.
- Peter Brusilovsky Alfred Kobsa, Julita Vassileva (1998). *Adaptive Hypertext and Hypermedia, First Edition*. Springer.
- Philip A. Bernstein, Eric Newcomer (1997). *Principles of Transaction Processing, First Edition*. Morgan Kaufmann.
- Ramez Elmasri, Shamkant B. Navathe (2004). *Fundamentals of Database Systems, Fourth Edition*. Addison Wesley.



Universidad Nacional de Colombia (UNAL) Sede
Manizales
Programa Profesional de
Administración de Sistemas Informáticos
SILABO

CS291. Ingeniería de Software I (Obligatorio)

2022-II

1. Información general	
1.1 Escuela	: Sistemas de Información
1.2 Curso	: CS291. Ingeniería de Software I
1.3 Semestre	: 5 ^{to} Semestre.
1.4 Prerrequisitos	: <ul style="list-style-type: none">• CS113. Ciencia de la Computación II. (3^{er} Sem)• CS271. Bases de Datos I. (4^{to} Sem)
1.5 Condición	: Obligatorio
1.6 Modalidad de aprendizaje	: Presencial
1.7 horas	: 2 HT; 2 HP; 2 HL;
1.8 Créditos	: 4

2. Profesores

3. Fundamentación del curso
La tarea de desarrollar software, excepto para aplicaciones sumamente simples, exige la ejecución de un proceso de desarrollo bien definido. Los profesionales de esta área requieren un alto grado de conocimiento de los diferentes modelos e proceso de desarrollo, para que sean capaces de elegir el más idóneo para cada proyecto de desarrollo. Por otro lado, el desarrollo de sistemas de mediana y gran escala requiere del uso de bibliotecas de patrones y componentes y del dominio de técnicas relacionadas al diseño basado en componentes.

4. Resumen
1. Ingeniería de Requisitos 2. Diseño de Software 3. Construcción de Software

5. Objetivos Generales
<ul style="list-style-type: none">• Brindar al alumno un marco teórico y práctico para el desarrollo de software bajo estándares de calidad.• Familiarizar al alumno con los procesos de modelamiento y construcción de software a través del uso de herramientas CASE.• Los alumnos debe ser capaces de seleccionar Arquitecturas y Plataformas tecnológicas ad-hoc a los escenarios de implementación.• Aplicar el modelamiento basado en componentes y fin de asegurar variables como calidad, costo y <i>time-to-market</i> en los procesos de desarrollo.• Brindar a los alumnos mejores prácticas para la verificación y validación del software.

6. Contribución a los resultados (*Outcomes*)

Esta disciplina contribuye al logro de los siguientes resultados de la carrera:

- 1) Analizar un problema computacional complejo y aplicar los principios computacionales y otras disciplinas relevantes para identificar soluciones. (**Usar**)
- 2) Diseñar, implementar y evaluar una solución basada en computación para cumplir con un conjunto determinado de requisitos computacionales en el contexto de las disciplinas del programa. (**Usar**)
- 3) Comunicarse efectivamente en diversos contextos profesionales. (**Usar**)
- 6) Aplicar fundamentos de teoría de ciencias de la computación y desarrollo de software para producir soluciones basados en computación. (**Evaluar**)

7. Contenido

UNIDAD 1: Ingeniería de Requisitos (18)	
Competencias:	
Contenido	Objetivos Generales
<ul style="list-style-type: none"> • Al describir los requisitos funcionales utilizando, por ejemplo, los casos de uso o historias de los usuarios. • Propiedades de requisitos, incluyendo la consistencia, validez, integridad y viabilidad. • Requisitos de software elicitation. • Descripción de datos del sistema utilizando, por ejemplo, los diagramas de clases o diagramas entidad-relación. • Requisitos no funcionales y su relación con la calidad del software. • Evaluación y uso de especificaciones de requisitos. • Requisitos de las técnicas de modelado de análisis. • La aceptabilidad de las consideraciones de certeza/incertidumbre sobre el comportamiento del software/sistema. • Prototipos. • Conceptos básicos de la especificación formal de requisitos. • Especificación de requisitos. • Validación de requisitos. • Rastreo de requisitos. 	<ul style="list-style-type: none"> • Enumerar los componentes clave de un caso de uso o una descripción similar de algún comportamiento que es requerido para un sistema [Evaluar] • Describir cómo el proceso de ingeniería de requisitos apoya la obtención y validación de los requisitos de comportamiento [Evaluar] • Interpretar un modelo de requisitos dada por un sistema de software simple [Evaluar] • Describir los retos fundamentales y técnicas comunes que se utilizan para la obtención de requisitos [Evaluar] • Enumerar los componentes clave de un modelo de datos (por ejemplo, diagramas de clases o diagramas ER) [Evaluar] • Identificar los requisitos funcionales y no funcionales en una especificación de requisitos dada por un sistema de software [Evaluar] • Realizar una revisión de un conjunto de requisitos de software para determinar la calidad de los requisitos con respecto a las características de los buenos requisitos [Evaluar] • Aplicar elementos clave y métodos comunes para la obtención y el análisis para producir un conjunto de requisitos de software para un sistema de software de tamaño medio [Evaluar] • Comparar los métodos ágiles y el dirigido por planes para la especificación y validación de requisitos y describir los beneficios y riesgos asociados con cada uno [Evaluar] • Usar un método común, no formal para modelar y especificar los requisitos para un sistema de software de tamaño medio [Evaluar] • Traducir al lenguaje natural una especificación de requisitos de software (por ejemplo, un contrato de componentes de software) escrito en un lenguaje de especificación formal [Evaluar] • Crear un prototipo de un sistema de software para reducir el riesgo en los requisitos [Evaluar] • Diferenciar entre el rastreo (<i>tracing</i>) hacia adelante y hacia atrás y explicar su papel en el proceso de validación de requisitos [Evaluar]
Lecturas: Eric Freeman and Sierra (2014), Hans-Erik Eriksson and Fado (2003)	

UNIDAD 2: Diseño de Software (18)

Competencias:

Contenido

Objetivos Generales

- Principios de diseño del sistema: niveles de abstracción (diseño arquitectónico y el diseño detallado), separación de intereses, ocultamiento de información, de acoplamiento y de cohesión, de reutilización de estructuras estándar.
- Diseño de paradigmas tales como diseño estructurado (descomposición funcional de arriba hacia abajo), el análisis orientado a objetos y diseño, orientado a eventos de diseño, diseño de nivel de componente, centrado datos estructurada, orientada a aspectos, orientado a la función, orientado al servicio.
- Modelos estructurales y de comportamiento de los diseños de software.
- Diseño de patrones.
- Relaciones entre los requisitos y diseños: La transformación de modelos, el diseño de los contratos, invariantes.
- Conceptos de arquitectura de software y arquitecturas estándar (por ejemplo, cliente-servidor, n-capas, transforman centrados, tubos y filtros).
- El uso de componentes de diseño: selección de componentes, diseño, adaptación y componentes de ensamblaje, componentes y patrones, componentes y objetos (por ejemplo, construir una GUI usando un standard widget set)
- Diseños de refactorización utilizando patrones de diseño
- Calidad del diseño interno, y modelos para: eficiencia y desempeño, redundancia y tolerancia a fallos, trazabilidad de los requerimientos.
- Medición y análisis de la calidad de un diseño.
- Compensaciones entre diferentes aspectos de la calidad.
- Aplicaciones en frameworks.
- Middleware: El paradigma de la orientación a objetos con middleware, requerimientos para correr y clasificar objetos, monitores de procesamiento de transacciones y el sistema de flujo de trabajo.
- Principales diseños de seguridad y codificación (cross-reference IAS/Principles of secure design).
 - Principio de privilegios mínimos
 - Principio de falla segura por defecto
 - Principio de aceptabilidad psicológica

- Formular los principios de diseño, incluyendo la separación de problemas, ocultación de información, acoplamiento y cohesión, y la encapsulación [Familiarizarse]
- Usar un paradigma de diseño para diseñar un sistema de software básico y explicar cómo los principios de diseño del sistema se han aplicado en este diseño [Usar]
- Construir modelos del diseño de un sistema de software simple los cuales son apropiado para el paradigma utilizado para diseñarlo [Usar]
- En el contexto de un paradigma de diseño simple, describir uno o más patrones de diseño que podrían ser aplicables al diseño de un sistema de software simple [Familiarizarse]
- Para un sistema simple adecuado para una situación dada, discutir y seleccionar un paradigma de diseño apropiado [Usar]
- Crear modelos apropiados para la estructura y el comportamiento de los productos de software desde la especificaciones de requisitos [Usar]
- Explicar las relaciones entre los requisitos para un producto de software y su diseño, utilizando los modelos apropiados [Evaluar]
- Para el diseño de un sistema de software simple dentro del contexto de un único paradigma de diseño, describir la arquitectura de software de ese sistema [Familiarizarse]
- Dado un diseño de alto nivel, identificar la arquitectura de software mediante la diferenciación entre las arquitecturas comunes de software, tales como 3 capas (*3-tier*), *pipe-and-filter*, y cliente-servidor [Familiarizarse]
- Investigar el impacto de la selección arquitecturas de software en el diseño de un sistema simple [Evaluar]
- Aplicar ejemplos simples de patrones en un diseño de software [Usar]
- Describir una manera de refactorar y discutir cuando esto debe ser aplicado [Familiarizarse]
- Seleccionar componentes adecuados para el uso en un diseño de un producto de software [Usar]
- Explicar cómo los componentes deben ser adaptados para ser usados en el diseño de un producto de software [Familiarizarse]
- Diseñar un contrato para un típico componente de software pequeño para el uso de un dado sistema [Usar]
- Discutir y seleccionar la arquitectura de software

UNIDAD 3: Construcción de Software (24)	
Competencias:	
Contenido	Objetivos Generales
<ul style="list-style-type: none"> • Prácticas de codificación: técnicas, idiomas/patrones, mecanismos para construcción de programas de calidad: <ul style="list-style-type: none"> – Prácticas de codificación defensiva – Prácticas de codificación segura – Utilizando mecanismos de manejo de excepciones para hacer el programa más robusto, tolerante a fallas • Normas de codificación. • Estrategias de integración. • Desarrollando contexto: "campo verde" frente a la base de código existente : <ul style="list-style-type: none"> – Análisis de cambio impacto – Cambio de actualización • Los problemas de seguridad potenciales en los programas : <ul style="list-style-type: none"> – Buffer y otros tipos de desbordamientos – Condiciones elemento Race – Inicialización incorrecta, incluyendo la elección de los privilegios – Entrada Comprobación – Suponiendo éxito y corrección – La validación de las hipótesis 	<ul style="list-style-type: none"> • Describir técnicas, lenguajes de codificación y mecanismos de implementación para conseguir las propiedades deseadas, tales como la confiabilidad, la eficiencia y la robustez [Evaluar] • Construir código robusto utilizando los mecanismos de manejo de excepciones [Evaluar] • Describir la codificación segura y prácticas de codificación de defensa [Evaluar] • Seleccionar y utilizar un estándar de codificación definido en un pequeño proyecto de software [Evaluar] • Comparar y contrastar las estrategias de integración incluyendo: de arriba hacia abajo (<i>top-down</i>), de abajo hacia arriba (<i>bottom-up</i>), y la integración Sándwich [Evaluar] • Describir el proceso de analizar e implementar los cambios a la base de código desarrollado para un proyecto específico [Evaluar] • Describir el proceso de analizar e implementar los cambios a una gran base de código existente [Evaluar] • Reescribir un programa sencillo para eliminar vulnerabilidades comunes, tales como desbordamientos de búffer, desbordamientos de enteros y condiciones de carrera [Evaluar] • Escribir un componente de software que realiza alguna tarea no trivial y es resistente a errores en la entrada y en tiempo de ejecución [Evaluar]
Lecturas: Eric Freeman and Sierra (2014), Hans-Erik Eriksson and Fado (2003)	

8. Metodología
<p>El profesor del curso presentará clases teóricas de los temas señalados en el programa propiciando la intervención de los alumnos.</p> <p>El profesor del curso presentará demostraciones para fundamentar clases teóricas.</p> <p>El profesor y los alumnos realizarán prácticas</p> <p>Los alumnos deberán asistir a clase habiendo leído lo que el profesor va a presentar. De esta manera se facilitará la comprensión y los estudiantes estarán en mejores condiciones de hacer consultas en clase.</p>

9. Evaluar

Evaluación Continua 1 : 20 %

Examen parcial : 30 %

Evaluación Continua 2 : 20 %

Examen final : 30 %

References

Eric Freeman Elisabeth Robson, Bert Bates and Kathy Sierra (July 2014). *Head First Design Patterns*. 2nd. O'Reilly Media, Inc.

Hans-Erik Eriksson Magnus Penker, Brian Lyons and Davis Fado (Oct. 2003). *UML 2 Toolkit*. 2nd. Wiley.



Universidad Nacional de Colombia (UNAL) Sede
Manizales
Programa Profesional de
Administración de Sistemas Informáticos
SILABO

CS342. Compiladores (Obligatorio)

2022-II

1. Información general	
1.1 Escuela	: Sistemas de Información
1.2 Curso	: CS342. Compiladores
1.3 Semestre	: 5 ^{to} Semestre.
1.4 Prerrequisitos	: CS211. Teoría de la Computación. (4 ^{to} Sem)
1.5 Condición	: Obligatorio
1.6 Modalidad de aprendizaje	: Presencial
1.7 horas	: 2 HT; 2 HP; 2 HL;
1.8 Créditos	: 4
2. Profesores	
3. Fundamentación del curso	
Que el alumno conozca y comprenda los conceptos y principios fundamentales de la teoría de compilación para realizar la construcción de un compilador	
4. Resumen	
1. Representación de programas 2. Traducción y ejecución de lenguajes 3. Análisis de sintaxis 4. Análisis semántico de compiladores 5. Generación de código	
5. Objetivos Generales	
<ul style="list-style-type: none">• Conocer las técnicas básicas empleadas durante el proceso de generación intermedio, optimización y generación de código.• Aprender a implementar pequeños compiladores.	
6. Contribución a los resultados (Outcomes)	
Esta disciplina contribuye al logro de los siguientes resultados de la carrera:	
1) Analizar un problema computacional complejo y aplicar los principios computacionales y otras disciplinas relevantes para identificar soluciones. (Evaluar)	
6) Aplicar fundamentos de teoría de ciencias de la computación y desarrollo de software para producir soluciones basados en computación. (Evaluar)	
7. Contenido	

UNIDAD 1: Representación de programas (5)	
Competencias:	
Contenido	Objetivos Generales
<ul style="list-style-type: none"> • Programas que tienen otros programas como entrada tales como interpretes, compiladores, revisores de tipos y generadores de documentación. • Árboles de sintaxis abstracta, para contrastar la sintaxis correcta. • Estructuras de datos que representan código para ejecución, traducción o transmisión. • Compilación en tiempo just-in time y re-compilación dinámica. • Otras características comunes de las máquinas virtuales, tales como carga de clases, hilos y seguridad. 	<ul style="list-style-type: none"> • Explicar como programas que procesan otros programas tratan a los otros programas como su entrada de datos [Familiarizarse] • Describir un árbol de sintaxis abstracto para un lenguaje pequeño [Familiarizarse] • Describir los beneficios de tener representaciones de programas que no sean cadenas de código fuente [Familiarizarse] • Escribir un programa para procesar alguna representación de código para algún propósito, tales como un interprete, una expresión optimizada, o un generador de documentación [Familiarizarse] • Explicar el uso de metadatos en las representaciones de tiempo de ejecución de objetos y registros de activación, tales como los punteros de la clase, las longitudes de arreglos, direcciones de retorno, y punteros de <i>frame</i> [Familiarizarse] • Discutir las ventajas, desventajas y dificultades del término (<i>just-in-time</i>) y recompilación automática [Familiarizarse] • Identificar los servicios proporcionados por los sistemas de tiempo de ejecución en lenguajes modernos [Familiarizarse]
Lecturas: Louden (2004b)	

UNIDAD 2: Traducción y ejecución de lenguajes (10)	
Competencias:	
Contenido	Objetivos Generales
<ul style="list-style-type: none"> • Interpretación vs. compilación a código nativo vs. compilación de representación portable intermedia. • Pipeline de traducción de lenguajes: análisis, revisión opcional de tipos, traducción, enlazamiento, ejecución: <ul style="list-style-type: none"> – Ejecución como código nativo o con una máquina virtual – Alternativas como carga dinámica y codificación dinámica de código (o “just-in-time”) • Representación en tiempo de ejecución de construcción del lenguaje núcleo tales como objetos (tablas de métodos) y funciones de primera clase (cerradas) • Ejecución en tiempo real de asignación de memoria: pila de llamadas, montículo, datos estáticos: <ul style="list-style-type: none"> – Implementación de bucles, recursividad y llamadas de cola • Gestión de memoria: <ul style="list-style-type: none"> – Gestión manual de memoria: asignación, limpieza y reuso de la pila de memoria – Gestión automática de memoria: recolección de datos no utilizados (<i>garbage collection</i>) como una técnica automática usando la noción de accesibilidad 	<ul style="list-style-type: none"> • Distinguir una definición de un lenguaje de una implementación particular de un lenguaje (compilador vs interprete, tiempo de ejecución de la representación de los objetos de datos, etc) [Evaluar] • Distinguir sintaxis y parseo de la semántica y la evaluación [Evaluar] • Bosqueje una representación de bajo nivel de tiempo de ejecución de construcciones del lenguaje base, tales como objetos o cierres (<i>closures</i>) [Evaluar] • Explicar cómo las implementaciones de los lenguajes de programación típicamente organizan la memoria en datos globales, texto, <i>heap</i>, y secciones de pila y cómo las características tales como recursión y administración de memoria son mapeados a este modelo de memoria [Evaluar] • Identificar y corregir las pérdidas de memoria y punteros desreferenciados [Evaluar] • Discutir los beneficios y limitaciones de la recolección de basura (<i>garbage collection</i>), incluyendo la noción de accesibilidad [Evaluar]
Lecturas: Aho et al. (2011), Louden (2004a), Teufel and Schmidt (1998), Appel (2002)	

UNIDAD 3: Análisis de sintaxis (10)	
Competencias:	
Contenido	Objetivos Generales
<ul style="list-style-type: none"> • Exploración (análisis léxico) usando expresiones regulares. • Estrategias de análisis incluyendo técnicas de arriba a abajo (top-down) (p.e. descenso recursivo, análisis temprano o LL) y de abajo a arriba (bottom-up) (ej. ‘llamadas hacia atrás - bracktracking, o LR); rol de las gramáticas libres de contexto. • Generación de exploradores (scanners) y analizadores a partir de especificaciones declarativas. 	<ul style="list-style-type: none"> • Usar gramáticas formales para especificar la sintaxis de los lenguajes [Evaluar] • Usar herramientas declarativas para generar parseadores y escáneres [Evaluar] • Identificar las características clave en las definiciones de sintaxis: ambigüedad, asociatividad, precedencia [Evaluar]
Lecturas: Aho et al. (2011), Louden (2004a), Teufel and Schmidt (1998), Appel (2002)	

UNIDAD 4: Análisis semántico de compiladores (15)	
Competencias:	
Contenido	Objetivos Generales
<ul style="list-style-type: none"> • Representaciones de programas de alto nivel tales como árboles de sintaxis abstractas. • Alcance y resolución de vínculos. • Revisión de tipos. • Especificaciones declarativas tales como gramáticas atribuidas. 	<ul style="list-style-type: none"> • Implementar analizadores sensibles al contexto y estáticos a nivel de fuente, tales como, verificadores de tipos o resolvedores de identificadores para identificar las ocurrencias de vínculo [Evaluar] • Describir analizadores semánticos usando una gramática con atributos [Evaluar]
Lecturas: Aho et al. (2011), Louden (2004a), Teufel and Schmidt (1998), Appel (2002)	

UNIDAD 5: Generación de código (20)	
Competencias:	
Contenido	Objetivos Generales
<ul style="list-style-type: none"> • Llamadas a procedimientos y métodos en envío. • Compilación separada; vinculación. • Selección de instrucciones. • Calendarización de instrucciones. • Asignación de registros. • Optimización por rendija (peephole) 	<ul style="list-style-type: none"> • Identificar todos los pasos esenciales para convertir automáticamente código fuente en código ensamblador o otros lenguajes de bajo nivel [Evaluar] • Generar código de bajo nivel para llamadas a funciones en lenguajes modernos [Evaluar] • Discutir por qué la compilación separada requiere convenciones de llamadas uniformes [Evaluar] • Discutir por qué la compilación separada limita la optimización debido a efectos de llamadas desconocidas [Evaluar] • Discutir oportunidades para optimización introducida por la traducción y enfoques para alcanzar la optimización, tales como la selección de la instrucción, planificación de instrucción, asignación de registros y optimización de tipo mirilla (<i>peephole optimization</i>) [Evaluar]
Lecturas: Aho et al. (2011), Louden (2004a), Teufel and Schmidt (1998), Appel (2002)	

8. Metodología
<p>El profesor del curso presentará clases teóricas de los temas señalados en el programa propiciando la intervención de los alumnos.</p> <p>El profesor del curso presentará demostraciones para fundamentar clases teóricas.</p> <p>El profesor y los alumnos realizarán prácticas</p> <p>Los alumnos deberán asistir a clase habiendo leído lo que el profesor va a presentar. De esta manera se facilitará la comprensión y los estudiantes estarán en mejores condiciones de hacer consultas en clase.</p>

9. Evaluar

Evaluación Continua 1 : 20 %

Examen parcial : 30 %

Evaluación Continua 2 : 20 %

Examen final : 30 %

References

- Aho, Alfred et al. (2011). *Compilers Principles Techniques And Tools*. 2nd. ISBN:10-970-26-1133-4. Pearson.
- Appel, A. W. (2002). *Modern compiler implementation in Java*. 2.a edición. Cambridge University Press.
- Louden, Kenneth C. (2004a). *Compiler Construction: Principles and Practice*. Thomson.
- Louden, Kenneth C. (2004b). *Lenguajes de Programacion*. Thomson.
- Teufel, Bernard and Stephanie Schmidt (1998). *Fundamentos de Compiladores*. Addison Wesley Iberoamericana.



Universidad Nacional de Colombia (UNAL) Sede
Manizales
Programa Profesional de
Administración de Sistemas Informáticos
SILABO

CB111. Física Computacional (Obligatorio)

2022-II

1. Información general	
1.1 Escuela	: Sistemas de Información
1.2 Curso	: CB111. Física Computacional
1.3 Semestre	: 5 ^{to} Semestre.
1.4 Prerrequisitos	: MA102. Cálculo I. (3 ^{er} Sem)
1.5 Condición	: Obligatorio
1.6 Modalidad de aprendizaje	: Presencial
1.7 horas	: 2 HT; 2 HP; 2 HL;
1.8 Créditos	: 4
2. Profesores	
3. Fundamentación del curso	
Física I es un curso que le permitirá al estudiante entender las leyes de física de macropartículas y micropartículas considerado desde un punto material hasta un sistemas de part'../..../2020-I copy/Syllabi/BasicSciences'ículas; debiéndose tener en cuenta que los fenómenos aquí estudiados se relacionan a la física clásica: Cinemática, Dinámica, Trabajo y Energía; además se debe asociar que éstos problemas deben ser resueltos con algoritmos computacionales. Poseer capacidad y habilidad en la interpretación de problemas clásicos con condiciones de frontera reales que contribuyen en la elaboración de soluciones eficientes y factibles en diferentes áreas de la Ciencia de la Computación.	
4. Resumen	
1. Vectores 2. 3. 4. 5. 6.	
5. Objetivos Generales	
<ul style="list-style-type: none">• Conocer los principios básicos de los fenómenos que gobiernan la física clásica.• Aplicar los principios básicos a situaciones específicas y poder asociarlos con situaciones reales.• Analizar algunos de los fenómenos físicos así como su aplicación a situaciones reales.	
6. Contribución a los resultados (Outcomes)	
Esta disciplina contribuye al logro de los siguientes resultados de la carrera:	
1) Analizar un problema computacional complejo y aplicar los principios computacionales y otras disciplinas relevantes para identificar soluciones. (Usar)	
6) Aplicar fundamentos de teoria de ciencias de la computación y desarrollo de software para producir soluciones basados en computación. (Usar)	
7. Contenido	

UNIDAD 1: Vectores (6)	
Competencias:	
Contenido	Objetivos Generales
<ul style="list-style-type: none"> • Análisis dimensional. • Vectores. Propiedades. Operaciones. • Caso práctico: Estimación de fuerzas. 	<ul style="list-style-type: none"> • Entender y trabajar con las magnitudes físicas del SI.[Usar] • Abstractar de la naturaleza los conceptos físicos rigurosos y representarlos en modelos vectoriales.[Usar] • Entender y aplicar los conceptos vectoriales a problemas físicos reales.[Usar]
Lecturas: Burbano (2006), Resnik (2007), Serway (2009), Tipler (2009)	

UNIDAD 2: (6)	
Competencias:	
Contenido	Objetivos Generales
<ul style="list-style-type: none"> • Primera y tercera Ley de Newton. • Diagrama de cuerpo libre. • Primera condición de equilibrio. • Caso práctico: Estimación de la fuerza humana. • Segunda condición de equilibrio. • Torque. • Casos prácticos: Aplicaciones en dispositivos mecánicos. • Fricción. 	<ul style="list-style-type: none"> • Conocer los conceptos que rigen la primera Ley y tercera Ley de Newton. • Conocer y aplicar los conceptos de la primera y segunda condición de equilibrio. • Capacidad para resolver problemas de casos prácticos. • Entender el concepto de fricción y resolver problemas.
Lecturas: Burbano (2006), Resnik (2007), Serway (2009), Tipler (2009)	

UNIDAD 3: (6)	
Competencias:	
Contenido	Objetivos Generales
<ul style="list-style-type: none"> • Posición, Velocidad, Aceleración. • Gráficas de movimiento. • Casos prácticos: Representación gráfica de movimiento utilizando Excel. • Movimiento circular. • Velocidad angular y velocidad tangencial. • Mecanismos rotativos. • Caso práctico: Operación de la caja de cambios de un automóvil. 	<ul style="list-style-type: none"> • Poder determinar la posición, velocidad y aceleración de un cuerpo. • Conocer el concepto de composición de movimientos y saberlo aplicar, en la descripción de un movimiento circular. • Conocer el significado de las componentes tangencial y normal de la aceleración y saberlas calcular en un instante determinado. • Utilizar excel para el procesamiento de datos experimentales.
Lecturas: Burbano (2006), Resnik (2007), Serway (2009), Tipler (2009)	

UNIDAD 4: (6)	
Competencias:	
Contenido	Objetivos Generales
<ul style="list-style-type: none"> • Segunda Ley de Newton. • Fuerza y movimiento. • Momento de inercia. 	<ul style="list-style-type: none"> • Aplicar las leyes de Newton en la solución de problemas. • Describir las diversas interacciones por sus correspondientes fuerzas. • Determinar el momento de inercia de un cuerpo usando un método dinámico
Lecturas: Burbano (2006), Resnik (2007), Serway (2009), Tipler (2009)	

UNIDAD 5: (6)	
Competencias:	
Contenido	Objetivos Generales
<ul style="list-style-type: none"> • Trabajo. • Fuerzas constantes. • Fuerzas variables. • Potencia. • Caso práctico: Estimación de la potencia de una planta hidroeléctrica. 	<ul style="list-style-type: none"> • Comprender el concepto de Trabajo. • Comprender y aplicar el concepto de Potencia a la resolución de problemas. • Resolver problemas.
Lecturas: Burbano (2006), Resnik (2007), Serway (2009), Tipler (2009)	

UNIDAD 6: (6)	
Competencias:	
Contenido	Objetivos Generales
<ul style="list-style-type: none"> • Tipos de energía. • Conservación de la energía. • Dinámica de un sistema de part'../..../2020-I copy/Syllabi/BasicSciences'ículas. • Colisiones. 	<ul style="list-style-type: none"> • Conocer los tipos de energía que existen. • Aplicar el principio de conservación de la energía mecánica a distintas situaciones, diferenciando aquellas en las que la energía total no se mantiene constante. • Aplicar los principios de conservación del momento lineal y de la energía a un sistema aislado de dos o más part'../..../2020-I copy/Syllabi/BasicSciences'ículas interactuantes.
Lecturas: Burbano (2006), Resnik (2007), Serway (2009), Tipler (2009)	

8. Metodología
<p>El profesor del curso presentará clases teóricas de los temas señalados en el programa propiciando la intervención de los alumnos.</p> <p>El profesor del curso presentará demostraciones para fundamentar clases teóricas.</p> <p>El profesor y los alumnos realizarán prácticas</p> <p>Los alumnos deberán asistir a clase habiendo leído lo que el profesor va a presentar. De esta manera se facilitará la comprensión y los estudiantes estarán en mejores condiciones de hacer consultas en clase.</p>

9. Evaluar

Evaluación Continua 1 : 20 %

Examen parcial : 30 %

Evaluación Continua 2 : 20 %

Examen final : 30 %

References

Burbano, S. (2006). *Física General*. Alfaomega.

Resnik R. y Halliday, D. (2007). *Física*. 5th. Vol. 1. Patria.

Serway R. A. y Jewett, J.W. (2009). *Física para Ciencias e Ingeniería con Física Moderna*. 7th. Vol. 1. Cengage Learning.

Tipler P. y Mosca, G. (2009). *Física para la ciencia y la tecnología*. 7th. Vol. 1. Reverte.



Universidad Nacional de Colombia (UNAL) Sede
Manizales
Programa Profesional de
Administración de Sistemas Informáticos
SILABO

FG106. Teatro (Obligatorio)

2022-II

1. Información general	
1.1 Escuela	: Sistemas de Información
1.2 Curso	: FG106. Teatro
1.3 Semestre	: 5 ^{to} Semestre.
1.4 Prerrequisitos	: Ninguno
1.5 Condición	: Obligatorio
1.6 Modalidad de aprendizaje	: Presencial
1.7 horas	: 1 HT; 2 HP;
1.8 Créditos	: 2
2. Profesores	
3. Fundamentación del curso	
Favorece al estudiante a identificarse a la “Comunidad Académica” de la Universidad, en la medida en que le brinda canales naturales de integración a su grupo y a su Centro de Estudios y le permite, desde una visión alternativa, visualizar la valía interior de las personas a su alrededor, a la vez que puede conocer mejor la suya propia. Relaciona al universitario, a través de la experimentación, con un nuevo lenguaje, un medio de comunicación y expresión que va más allá de la expresión verbal conceptualizada. Coadyuva al estudiante en su formación integral, desarrollando en él capacidades corporales. Estimula en él, actitudes anímicas positivas, aptitudes cognitivas y afectivas. Enriquece su sensibilidad y despierta su solidaridad. Desinhibe y socializa, relaja y alegra, abriendo un camino de apertura de conocimiento del propio ser y el ser de los demás.	
4. Resumen	
1. 2. 3. 4. 5. 6.	
5. Objetivos Generales	
<ul style="list-style-type: none">• Contribuir a la formación personal y profesional del estudiante, reconociendo, valorando y desarrollando su lenguaje corporal, integrándolo a su grupo, afianzando su seguridad personal, enriqueciendo su intuición, su imaginación y creatividad, motivándolo a abrir caminos de búsqueda de conocimiento de sí mismo y de comunicación con los demás a través de su sensibilidad, de ejercicios de introspección y de nuevas vías de expresión.	
6. Contribución a los resultados (Outcomes)	
Esta disciplina contribuye al logro de los siguientes resultados de la carrera:	
3) Comunicarse efectivamente en diversos contextos profesionales. (Usar)	
4) Reconocer las responsabilidades profesionales y hacer juicios informados en el campo profesional de computación con principios éticos. (Usar)	
7) Desarrollar tecnología computacional buscando el bien común, aportando con formación humana, capacidades científicas, tecnológicas y profesionales para solucionar problemas sociales de nuestro entorno. (Usar)	

7. Contenido

UNIDAD 1: (6)	
Competencias:	
Contenido	Objetivos Generales
<ul style="list-style-type: none">• ¿Qué es el Arte? Una experiencia vivencial y personal.• La llave maestra: la creatividad.• La importancia del teatro en la formación personal y profesional.• Utilidad y enfoque del arte teatral.	<ul style="list-style-type: none">• Reconocer la vigencia del Arte y la creatividad en el desarrollo personal y social [Usar].• Relacionar al estudiante con su grupo valorando la importancia de la comunicación humana y del colectivo social [Usar].• Reconocer nociones básicas del teatro [Usar].
Lecturas: Majorana (1958), Pavis (1998)	

UNIDAD 2: (6)	
Competencias:	
Contenido	Objetivos Generales
<ul style="list-style-type: none">• Juego, luego existo.• El juego del niño y el juego dramático.• Juegos de integración grupal y juegos de creatividad.• La secuencia teatral.	<ul style="list-style-type: none">• Reconocer el juego como herramienta fundamental del teatro [Usar].• Interiorizar y revalorar el juego como aprendizaje creativo [Usar].• Acercar al estudiante de manera espontánea y natural, a la vivencia teatral [Usar].
Lecturas: Majorana (1958), Pavis (1998)	

UNIDAD 3: (9)	
Competencias:	
Contenido	Objetivos Generales
<ul style="list-style-type: none">• Toma de conciencia del cuerpo.• Toma de conciencia del espacio• Toma de conciencia del tiempo• Creación de secuencias individuales y colectivas: Cuerpo, espacio y tiempo.• El uso dramático del elemento: El juego teatral.• Presentaciones teatrales con el uso del elemento.	<ul style="list-style-type: none">• Experimentar con nuevas formas de expresión y comunicación [Usar].• Conocer algunos mecanismos de control y manejo corporal [Usar].• Brindar caminos para que el alumno pueda desarrollar creativamente su imaginación, su capacidad de relación y captación de estímulos auditivos, rítmicos y visuales [Usar].• Conocer y desarrollar el manejo de su espacio propio y de sus relaciones espaciales [Usar].• Experimentar estados emocionales diferentes y climas colectivos nuevos [Usar].
Lecturas: Majorana (1958), Pavis (1998)	

UNIDAD 4: (12)	
Competencias:	
Contenido	Objetivos Generales
<ul style="list-style-type: none"> • Relajación, concentración y respiración. • Desinhibición e interacción con el grupo. • La improvisación. • Equilibrio, peso, tiempo y ritmo. • Análisis del movimiento. Tipos de movimiento. • La presencia teatral. • La danza, la coreografía teatral. 	<ul style="list-style-type: none"> • Ejercitarse en el manejo de destrezas comunicativas no verbales [Usar]. • Practicar juegos y ejercicios de lenguaje corporal, individual y grupalmente [Usar]. • Expresar libre y creativamente sus emociones y sentimientos y su visión de la sociedad a través de representaciones originales con diversos lenguajes [Usar]. • Conocer los tipos de actuación [Usar].
Lecturas: Majorana (1958), Pavis (1998)	

UNIDAD 5: (3)	
Competencias:	
Contenido	Objetivos Generales
<ul style="list-style-type: none"> • El origen del teatro, el teatro griego y el teatro romano. • El teatro medieval , la comedia del arte. • De la pasión a la razón: Romanticismo e Ilustración. • El teatro realista, teatro épico. Brech y Stanislavski. • El teatro del absurdo, teatro contemporáneo y teatro total. • Teatro en el Perú: Yuyashkani, La Tarumba, pataclaun, otros. 	<ul style="list-style-type: none"> • Conocer la influencia que la sociedad ha ejercido en el teatro y la respuesta de este arte ante los diferentes momentos de la historia [Usar]. • Apreciar el valor y aporte de las obras de dramaturgos importantes [Usar]. • Analizar el contexto social del arte teatral [Usar]. • Reflexionar sobre el Teatro Peruano y arequipeño [Usar].
Lecturas: Majorana (1958), Pavis (1998)	

UNIDAD 6: (12)	
Competencias:	
Contenido	Objetivos Generales
<ul style="list-style-type: none"> • Apreciación teatral. Expectación de una o más obras teatrales. • El espacio escénico. • Construcción del personaje • Creación y montaje de una obra teatral . • Presentación en público de pequeñas obras haciendo uso de vestuario, maquillaje, escenografía, utilería y del empleo dramático del objeto. 	<ul style="list-style-type: none"> • Emplear la creación teatral, como manifestación de ideas y sentimientos propios ante la sociedad [Usar]. • Aplicar las técnicas practicadas y los conocimientos aprendidos en una apreciación y/o expresión teatral concreta que vincule el rol de la educación [Usar]. • Intercambiar experiencias y realizar presentaciones breves de ejercicios teatrales en grupo, frente a público [Usar].
Lecturas: Majorana (1958), Pavis (1998)	

8. Metodología

El profesor del curso presentará clases teóricas de los temas señalados en el programa propiciando la intervención de los alumnos.

El profesor del curso presentará demostraciones para fundamentar clases teóricas.

El profesor y los alumnos realizarán prácticas

Los alumnos deberán asistir a clase habiendo leído lo que el profesor va a presentar. De esta manera se facilitará la comprensión y los estudiantes estarán en mejores condiciones de hacer consultas en clase.

9. Evaluar

Evaluación Continua 1 : 20 %

Examen parcial : 30 %

Evaluación Continua 2 : 20 %

Examen final : 30 %

References

Majorana, Angel (1958). *El arte de hablar en publico*. La España Moderna.
Pavis, Patrice (1998). *Diccionario del Teatro*. Edit. Piados BA.



Universidad Nacional de Colombia (UNAL) Sede
Manizales
Programa Profesional de
Administración de Sistemas Informáticos
SILABO

CS292. Ingeniería de Software II (Obligatorio)

2022-II

1. Información general	
1.1 Escuela	: Sistemas de Información
1.2 Curso	: CS292. Ingeniería de Software II
1.3 Semestre	: 6 ^{to} Semestre.
1.4 Prerrequisitos	: CS291. Ingeniería de Software I. (5 ^{to} Sem)
1.5 Condición	: Obligatorio
1.6 Modalidad de aprendizaje	: Presencial
1.7 horas	: 2 HT; 2 HP; 2 HL;
1.8 Créditos	: 4

2. Profesores

3. Fundamentación del curso
Los tópicos de este curso extienden las ideas del diseño y desarrollo de software desde la secuencia de introducción a la programación para abarcar los problemas encontrados en proyectos de gran escala. Es una visión más amplia y completa de la Ingeniería de Software apreciada desde un punto de vista de Proyectos.

4. Resumen
1. Herramientas y Entornos 2. Verificación y Validación de Software 3. Evolución de Software 4. Gestión de Proyectos de Software

5. Objetivos Generales
<ul style="list-style-type: none">• Capacitar a los alumnos para formar parte y definir equipos de desarrollo de software que afronten problemas de envergadura real.• Familiarizar a los alumnos con el proceso de administración de un proyecto de software de tal manera que sea capaz de crear, mejorar y utilizar herramientas y métricas que le permitan realizar la estimación y seguimiento de un proyecto de software.• Crear, evaluar e implementar un plan de prueba para segmentos de código de tamaño medio , Distinguir entre los diferentes tipos de pruebas , sentar las bases para crear, mejorar los procedimientos de prueba y las herramientas utilizadas con ese propósito.• Seleccionar con justificación un apropiado conjunto de herramientas para soportar el desarrollo de un rango de productos de software.• Crear, mejorar y utilizar los patrones existentes para el mantenimiento de software . Dar a conocer las características y patrones de diseño para la reutilización de software.• Identificar y discutir diferentes sistemas especializados , crear , mejorar y utilizar los patrones especializados para el diseño , implementación , mantenimiento y prueba de sistemas especializados

6. Contribución a los resultados (*Outcomes*)

Esta disciplina contribuye al logro de los siguientes resultados de la carrera:

- 1) Analizar un problema computacional complejo y aplicar los principios computacionales y otras disciplinas relevantes para identificar soluciones. (**Usar**)
- 2) Diseñar, implementar y evaluar una solución basada en computación para cumplir con un conjunto determinado de requisitos computacionales en el contexto de las disciplinas del programa. (**Usar**)
- 3) Comunicarse efectivamente en diversos contextos profesionales. (**Usar**)
- 6) Aplicar fundamentos de teoría de ciencias de la computación y desarrollo de software para producir soluciones basados en computación. (**Evaluar**)

7. Contenido

UNIDAD 1: Herramientas y Entornos (12)

Competencias:

Contenido	Objetivos Generales
<ul style="list-style-type: none">• Administración de configuración de software y control de versiones.• Administración de despliegues.• Análisis de requerimientos y herramientas para modelado del diseño.• Herramientas de <i>testing</i> incluyendo herramientas de análisis estático y dinámico.• Entornos de programación que automatizan el proceso de construcción de partes de programa (ejem., construcciones automatizadas)<ul style="list-style-type: none">– Integración continua.• Mecanismos y conceptos de herramientas de integración.	<ul style="list-style-type: none">• Administración de configuración de software y control de versiones. [Usar]• Administración de despliegues. [Usar]• Análisis de requerimientos y herramientas para modelado del diseño. [Usar]• Herramientas de <i>testing</i> incluyendo herramientas de análisis estático y dinámico. [Usar]• Entornos de programación que automatizan el proceso de construcción de partes de programa (ejem., construcciones automatizadas)<ul style="list-style-type: none">– Integración continua.[Usar]• Mecanismos y conceptos de herramientas de integración. [Usar]
Lecturas: Pressman (2004), Blum (1992), Schach (2004), Wang and King (2000), Keyes (2004), Windle and Abreo (2002), Priest and Sanchez (2001), Schach (2004), Montangero (1996), Ambriola (2001), Conradi (2000), Oquendo (2003)	

UNIDAD 2: Verificación y Validación de Software (12)**Competencias:****Contenido****Objetivos Generales**

- Verificación y validación de conceptos.
- Inspecciones, revisiones, auditorias.
- Tipos de pruebas, incluyendo la interfaz humano computador, usabilidad, confiabilidad, seguridad, desempeño para la especificación.
- Fundamentos de testeo:
 - Pruebas de Unit, integración, validación y de Sistema
 - Creación de plan de pruebas y generación de casos de test
 - Técnicas de test de caja negra y caja blanca
 - Test de regresión y automatización de pruebas
- Seguimiento de defectos.
- Limitaciones de testeo en dominios particulares, tales como sistemas paralelos o críticos en cuanto a seguridad.
- Enfoques estáticos y enfoques dinámicos para la verificación.
- Desarrollo basado en pruebas.
- Plan de Validación, documentación para validación.
- Pruebas Orientadas a Objetos, Sistema de Pruebas.
- Verificación y validación de artefactos no codificados (documentación, archivos de ayuda, materiales de entrenamiento)
- Logeo fallido, error crítico y apoyo técnico para dichas actividades.
- Estimación fallida y terminación de las pruebas que incluye la envios por defecto.

- Distinguir entre la validación y verificación del programa [Usar]
- Describir el papel que las herramientas pueden desempeñar en la validación de software [Usar]
- Realizar, como parte de una actividad de equipo, una inspección de un segmento de código de tamaño medio [Usar]
- Describir y distinguir entre diferentes tipos y niveles de pruebas (unitaria, integración, sistemas y aceptación) [Usar]
- Describir técnicas para identificar casos de prueba representativos para integración, regresión y pruebas del sistema [Usar]
- Crear y documentar un conjunto de pruebas para un segmento de código de mediano tamaño [Usar]
- Describir cómo seleccionar buenas pruebas de regresión y automatizarlas [Usar]
- Utilizar una herramienta de seguimiento de defectos para manejar defectos de software en un pequeño proyecto de software [Usar]
- Discutir las limitaciones de las pruebas en un dominio particular [Usar]
- Evaluar un banco de pruebas (*a test suite*) para un segmento de código de tamaño medio [Usar]
- Comparar los enfoques estáticos y dinámicos para la verificación [Usar]
- Identificar los principios fundamentales de los métodos de desarrollo basado en pruebas y explicar el papel de las pruebas automatizadas en estos métodos [Usar]
- Discutir los temas relacionados con las pruebas de software orientado a objetos [Usar]
- Describir las técnicas para la verificación y validación de los artefactos de no código [Usar]
- Describir los enfoques para la estimación de fallos [Usar]
- Estimar el número de fallos en una pequeña aplicación de software basada en la densidad de defectos y siembra de errores [Usar]
- Realizar una inspección o revisión del de código fuente de un software para un proyecto de software de tamaño pequeño o mediano [Usar]

Lecturas: Pressman (2004), Blum (1992), Schach (2004), Wang and King (2000), Keyes (2004), Windle and Abreo (2002), Priest and Sanchez (2001), Schach (2004),³ Montangero (1996), Ambriola (2001), Conradi (2000), Oquendo (2003)

UNIDAD 3: Evolución de Software (12)**Competencias:****Contenido****Objetivos Generales**

- Desarrollo de Software en el contexto de código grande pre existente
 - Cambios de software
 - Preocupaciones y ubicación de preocupaciones
 - *Refactoring*
- Evolución de Software.
- Características de Software mantenible.
- Sistemas de Reingeniería.
- Reuso de Software.
 - Segmentos de código
 - Bibliotecas y *frameworks*
 - Componentes
 - Líneas de Producto

- Identificar los problemas principales asociados con la evolución del software y explicar su impacto en el ciclo de vida del software [Usar]
- Estimar el impacto del cambio de requerimientos en productos existentes de tamaño medio [Usar]
- Usar refactorización en el proceso de modificación de un componente de software [Usar]
- Estudiar los desafíos de mejorar sistemas en un entorno cambiante [Usar]
- Perfilar los procesos de pruebas de regresión y su rol en el manejo de versiones [Usar]
- Estudiar las ventajas y desventajas de diferentes tipos de niveles de confiabilidad [Usar]

Lecturas: Pressman (2004), Blum (1992), Schach (2004), Wang and King (2000), Keyes (2004), Windle and Abreo (2002), Priest and Sanchez (2001), Schach (2004), Montangero (1996), Ambriola (2001), Conradi (2000), Oquendo (2003)

UNIDAD 4: Gestión de Proyectos de Software (12)**Competencias:****Contenido****Objetivos Generales**

- La participación del equipo:
 - Procesos elemento del equipo, incluyendo responsabilidades de tarea, la estructura de reuniones y horario de trabajo
 - Roles y responsabilidades en un equipo de software
 - Equipo de resolución de conflictos
 - Los riesgos asociados con los equipos virtuales (comunicación, la percepción, la estructura)
- Estimación de esfuerzo (a nivel personal)
- Riesgo.
 - El papel del riesgo en el ciclo de vida
 - Categorías elemento de riesgo, incluyendo la seguridad, la seguridad, mercado, finanzas, tecnología, las personas, la calidad, la estructura y el proceso de
- Gestión de equipos:
 - Organización de equipo y la toma de decisiones
 - Roles de identificación y asignación
 - Individual y el desempeño del equipo de evaluación
- Gestión de proyectos:
 - Programación y seguimiento de elementos
 - Herramientas de gestión de proyectos
 - Análisis de Costo/Beneficio
- Software de medición y técnicas de estimación.
- Aseguramiento de la calidad del software y el rol de las mediciones.
- Riesgo.
 - Identificación de riesgos y gestión.
 - Análisis riesgo y evaluación.
 - La tolerancia al riesgo (por ejemplo, riesgo adverso, riesgo neutral, la búsqueda de riesgo)
 - Planificación de Riesgo
- En todo el sistema de aproximación al riesgo, incluyendo riesgos asociados con herramientas.

- Discutir los comportamientos comunes que contribuyen al buen funcionamiento de un equipo [Usar]
- Crear y seguir un programa para una reunión del equipo [Usar]
- Identificar y justificar las funciones necesarias en un equipo de desarrollo de software [Usar]
- Entender las fuentes, obstáculos y beneficios potenciales de un conflicto de equipo [Usar]
- Aplicar una estrategia de resolución de conflictos en un ambiente de equipo [Usar]
- Utilizar un método ad hoc para estimar el esfuerzo de desarrollo del software (ejemplo, tiempo) y comparar con el esfuerzo actual requerido [Usar]
- Listar varios ejemplos de los riesgos del software [Usar]
- Describir el impacto del riesgo en el ciclo de vida de desarrollo de software [Usar]
- Describir las diferentes categorías de riesgo en los sistemas de software [Usar]
- Demostrar a través de la colaboración de proyectos de equipo los elementos centrales de la construcción de equipos y gestión de equipos [Usar]

Lecturas: Pressman (2004), Blum (1992), Schach (2004), Wang and King (2000), Keyes (2004), Windle and Abreo (2002), Priest and Sanchez (2001), Schach (2004), Montangero (1996), Ambriola (2001), Conradi (2000), Oquendo (2003)

8. Metodología

El profesor del curso presentará clases teóricas de los temas señalados en el programa propiciando la intervención de los alumnos.

El profesor del curso presentará demostraciones para fundamentar clases teóricas.

El profesor y los alumnos realizarán prácticas

Los alumnos deberán asistir a clase habiendo leído lo que el profesor va a presentar. De esta manera se facilitará la comprensión y los estudiantes estarán en mejores condiciones de hacer consultas en clase.

9. Evaluar

Evaluación Continua 1 : 20 %

Examen parcial : 30 %

Evaluación Continua 2 : 20 %

Examen final : 30 %

References

- Ambriola, Vincenzo (July 2001). *Software Process Technology*. Springer.
- Blum, Bruce I. (May 1992). *Software Engineering: A Holistic View*. 7th. Oxford University Press US.
- Conradi, R (Mar. 2000). *Software Process Technology*. Springer.
- Keyes, Jessica (Feb. 2004). *Software Configuration Management*. CRC Press.
- Montangero, Carlo (Sept. 1996). *Software Process Technology*. Springer.
- Oquendo, Flavio (Sept. 2003). *Software Process Technology*. Springer.
- Pressman, Roger S. (Mar. 2004). *Software Engineering: A Practitioner's Approach*. 6th. McGraw-Hill.
- Priest, John W. and Jose M. Sanchez (Jan. 2001). *Product Development and Design for Manufacturing*. Marcel Dekker.
- Schach, Stephen R (Jan. 2004). *Object-Oriented and Classical Software Engineering*. McGraw-Hill.
- Wang, Yingxu and Graham King (Apr. 2000). *Software Engineering Processes: Principles and Applications*. CRC Press.
- Windle, Daniel R. and L. Rene Abreo (Aug. 2002). *Software Requirements Using the Unified Process*. Prentice Hall.



Universidad Nacional de Colombia (UNAL) Sede
Manizales
Programa Profesional de
Administración de Sistemas Informáticos
SILABO

CS2S1. Sistemas Operativos (Obligatorio)

2022-II

1. Información general	
1.1 Escuela	: Sistemas de Información
1.2 Curso	: CS2S1. Sistemas Operativos
1.3 Semestre	: 6 ^{to} Semestre.
1.4 Prerrequisitos	: CS221. Arquitectura de Computadores. (3 ^{er} Sem)
1.5 Condición	: Obligatorio
1.6 Modalidad de aprendizaje	: Presencial
1.7 horas	: 2 HT; 2 HP; 2 HL;
1.8 Créditos	: 4
2. Profesores	
3. Fundamentación del curso	
Un sistema operativo (SO) gestiona los recursos computacionales para completar la ejecución de múltiples aplicaciones y sus procesos asociados. Este curso enseña el diseño de sistemas operativos modernos; e introduce sus conceptos fundamentales que cubren la ejecución multi-programa, <i>scheduling</i> , gerencia de memoria, sistemas de archivos y seguridad. Además, el curso incluye actividades de programación en un <i>sistema operativo mínimo</i> para resolver problemas y ampliar su funcionalidad. Tenga en cuenta que estas actividades requieren mucho tiempo para completarse. Sin embargo, trabajar en ellos proporciona un valioso aprendizaje sobre los sistemas operativos.	
4. Resumen	
1. Visión general de Sistemas Operativos 2. Principios de Sistemas Operativos 3. Concurrencia 4. Planificación y despacho 5. Manejo de memoria 6. Seguridad y protección 7. Máquinas virtuales 8. Manejo de dispositivos 9. Sistema de archivos 10. Sistemas <i>embedded</i> y de tiempo real 11. Tolerancia a fallas 12. Evaluación del desempeño de sistemas	
5. Objetivos Generales	
<ul style="list-style-type: none">• Estudiar el diseño de sistemas operativos modernos.• Proveer una experiencia práctica al diseñar e implementar un sistema operativo mínimo.	

6. Contribución a los resultados (*Outcomes*)

Esta disciplina contribuye al logro de los siguientes resultados de la carrera:

- 1) Analizar un problema computacional complejo y aplicar los principios computacionales y otras disciplinas relevantes para identificar soluciones. (**Evaluar**)
- 4) Reconocer las responsabilidades profesionales y hacer juicios informados en el campo profesional de computación con principios éticos. (**Familiarizarse**)
- 5) Funcionar efectivamente como miembro o líder de un equipo involucrado en actividades apropiadas a la disciplina del programa. (**Usar**)
- 6) Aplicar fundamentos de teoría de ciencias de la computación y desarrollo de software para producir soluciones basados en computación. (**Usar**)
- 7) Desarrollar tecnología computacional buscando el bien común, aportando con formación humana, capacidades científicas, tecnológicas y profesionales para solucionar problemas sociales de nuestro entorno. (**Evaluar**)

7. Contenido

UNIDAD 1: Visión general de Sistemas Operativos (3)

Competencias:

Contenido	Objetivos Generales
<ul style="list-style-type: none">• Papel y el propósito del sistema operativo.• Funcionalidad de un sistema operativo típico.• Los mecanismos de apoyo modelos cliente-servidor.• Cuestiones de diseño (eficiencia, robustez, flexibilidad, portabilidad, seguridad, compatibilidad)• Influencias de seguridad, creación de redes, multimedia, sistemas de ventanas.	<ul style="list-style-type: none">• Explicar los objetivos y funciones de un sistema operativo moderno [Familiarizarse]• Analizar las ventajas y desventajas inherentes en el diseño de un sistema operativo [Evaluar]• Describir las funciones de un sistema operativo contemporáneo respecto a conveniencia, eficiencia, y su habilidad para evolucionar [Familiarizarse]• Discutir acerca de sistemas operativos cliente-servidor, en red, distribuidos y cómo se diferencian de los sistemas operativos de un solo usuario [Familiarizarse]• Identificar amenazas potenciales a sistemas operativos y las características del diseño de seguridad para protegerse de ellos [Familiarizarse]
Lecturas: Avi Silberschatz (2012), Stallings (2005), Tanenbaum (2006), Tanenbaum (2001), Anderson and Dahlin (2014)	

UNIDAD 2: Principios de Sistemas Operativos (6)	
Competencias:	
Contenido	Objetivos Generales
<ul style="list-style-type: none"> • Estructuración de Sistemas Operativos (monolítico, capas, modular, los modelos micro-kernel) • Abstracciones, procesos y recursos. • Los conceptos de interfaces de programa de aplicación (API) • La evolución de las técnicas de hardware / software y las necesidades de aplicación • Organización de dispositivos. • Interrupciones: métodos e implementaciones. • Concepto de estado de usuario / sistema y la protección, la transición al modo kernel. 	<ul style="list-style-type: none"> • Explicar el concepto de una capa lógica [Familiarizarse] • Explicar los beneficios de construir capas abstractas en forma jerárquica [Familiarizarse] • Describir el valor de la API y <i>middleware</i> [Familiarizarse] • Describir como los recursos computacionales son usados por aplicaciones de software y administradas por el software del sistema [Familiarizarse] • Contrastar el modo <i>kernel</i> y modo usuario en un sistema operativo [Evaluar] • Discutir las ventajas y desventajas del uso de procesamiento con interrupciones [Familiarizarse] • Explicar el uso de una lista de dispositivos y el controlador de colas de entrada y salida [Familiarizarse]
Lecturas: Avi Silberschatz (2012), Stallings (2005), Tanenbaum (2006), Tanenbaum (2001), Anderson and Dahlin (2014)	

UNIDAD 3: Concurrencia (9)	
Competencias:	
Contenido	Objetivos Generales
<ul style="list-style-type: none"> • Diagramas de estado. • Estructuras (lista preparada, bloques de control de procesos, y así sucesivamente) • <i>Dispatching</i> y cambio de contexto. • El papel de las interrupciones. • Gestionar el acceso a los objetos del sistema operativo de forma atómica. • La implementación de primitivas de sincronización. • Problemas de multiprocesador (spin-locks, reentrada) 	<ul style="list-style-type: none"> • Describir la necesidad de concurrencia en el marco de un sistema operativo [Familiarizarse] • Demostrar los potenciales problemas de tiempo de ejecución derivados de la operación simultánea de muchas tareas diferentes [Usar] • Resumir el rango de mecanismos que pueden ser usados a nivel del sistema operativo para realizar sistemas concurrentes y describir los beneficios de cada uno [Familiarizarse] • Explicar los diferentes estados por los que una tarea debe pasar y las estructuras de datos necesarias para el manejo de varias tareas [Familiarizarse] • Resumir las técnicas para lograr sincronización en un sistema operativo (por ejemplo, describir como implementar semáforos usando primitivas del sistema operativo.) [Familiarizarse] • Describir las razones para usar interrupciones, <i>dispatching</i>, y cambio de contexto para soportar concurrencia en un sistema operativo [Familiarizarse] • Crear diagramas de estado y transición para los problemas de dominios simples [Usar]
Lecturas: Avi Silberschatz (2012), Stallings (2005), Tanenbaum (2006), Tanenbaum (2001), Anderson and Dahlin (2014)	

UNIDAD 4: Planificación y despacho (6)	
Competencias:	
Contenido	Objetivos Generales
<ul style="list-style-type: none"> • <i>Scheduling preemptive y non-preemptive.</i> • <i>Scheduling</i> y políticas. • Procesos y subprocesos. • Plazos y cuestiones en tiempo real. 	<ul style="list-style-type: none"> • Comparar y contrastar los algoritmos comunes que se utilizan tanto para <i>scheduling preemptive</i> y <i>pre-emptive</i> de tareas en los sistemas operativos, como la comparación de prioridad, el rendimiento, y los esquemas de distribución equitativa [Evaluar] • Describir las relaciones entre los algoritmos de <i>scheduling</i> y dominios de aplicación [Familiarizarse] • Discutir los tipos de <i>scheduling</i> en procesadores en de corto, mediano, largo plazo y I/O [Familiarizarse] • Describir las diferencias entre procesos y <i>threads</i> [Familiarizarse] • Comparar y contrastar enfoques estáticos y dinámicos para <i>scheduling</i> en tiempo real [Evaluar] • Discutir sobre la necesidad de <i>preemption</i> y <i>deadline scheduling</i> [Familiarizarse] • Identificar formas en que la lógica expresada en algoritmos de planificación son de aplicación a otros ámbitos, tales como I/O del disco, la programación de disco de red, programación de proyectos y problemas más allá de la computación [Familiarizarse]
Lecturas: Avi Silberschatz (2012), Stallings (2005), Tanenbaum (2006), Tanenbaum (2001), Anderson and Dahlin (2014)	

UNIDAD 5: Manejo de memoria (6)	
Competencias:	
Contenido	Objetivos Generales
<ul style="list-style-type: none"> • Revisión de la memoria física y hardware de gestión de memoria. • Conjuntos de trabajo y thrashing. • El almacenamiento en caché 	<ul style="list-style-type: none"> • Explicar la jerarquía de la memoria y <i>tradeoffs</i> de costo-rendimiento [Familiarizarse] • Resumir los principios de memoria virtual tal como se aplica para el almacenamiento en cache y paginación [Familiarizarse] • Evaluar las ventajas y desventajas en términos del tamaño de memoria (memoria principal, memoria caché, memoria auxiliar) y la velocidad del procesador [Evaluar] • Describir las diferentes formas de asignar memoria a las tareas, citando las ventajas relativas de cada uno [Familiarizarse] • Describir el motivo y el uso de memoria caché (rendimiento y proximidad, dimensión diferente de como los caches complican el aislamiento y abstracción en VM) [Familiarizarse] • Estudiar los conceptos de <i>thrashing</i>, tanto en términos de las razones por las que se produce y las técnicas usadas para el reconocimiento y manejo del problema [Familiarizarse]
Lecturas: Avi Silberschatz (2012), Stallings (2005), Tanenbaum (2006), Tanenbaum (2001), Anderson and Dahlin (2014)	

UNIDAD 6: Seguridad y protección (6)	
Competencias:	
Contenido	Objetivos Generales
<ul style="list-style-type: none"> • Visión general de la seguridad del sistema . • Política / mecanismo de separación. • Métodos de seguridad y dispositivos. • Protección, control de acceso y autenticación. • Las copias de seguridad. 	<ul style="list-style-type: none"> • Explicar la necesidad para la protección y seguridad en un sistema operativo [Familiarizarse] • Resumir las características y limitaciones de un sistema operativo usado para proporcionar protección y seguridad [Familiarizarse] • Explicar el mecanismo disponible en un OS para controlar los accesos a los recursos [Familiarizarse] • Realizar tareas de administración de sistemas sencillas de acuerdo a una política de seguridad, por ejemplo la creación de cuentas, el establecimiento de permisos, aplicación de parches y organización de backups regulares [Familiarizarse]
Lecturas: Avi Silberschatz (2012), Stallings (2005), Tanenbaum (2006), Tanenbaum (2001), Anderson and Dahlin (2014)	

UNIDAD 7: Máquinas virtuales (6)	
Competencias:	
Contenido	Objetivos Generales
<ul style="list-style-type: none"> • Tipos de virtualización (incluyendo Hardware / Software, OS, Servidor, Servicio, Red) • Paginación y la memoria virtual. • Sistemas de archivos virtuales. • Los Hypervisores. • Virtualización portátil; emulación vs aislamiento. • Costo de la virtualización. 	<ul style="list-style-type: none"> • Explicar el concepto de memoria virtual y la forma cómo se realiza en hardware y software [Familiarizarse] • Diferenciar emulación y el aislamiento [Familiarizarse] • Evaluar virtualización de compensaciones [Evaluar] • Discutir sobre hipervisores y la necesidad para ellos en conjunto con diferentes tipos de hipervisores [Familiarizarse]
Lecturas: Avi Silberschatz (2012), Stallings (2005), Tanenbaum (2006), Tanenbaum (2001), Anderson and Dahlin (2014)	

UNIDAD 8: Manejo de dispositivos (6)	
Competencias:	
Contenido	Objetivos Generales
<ul style="list-style-type: none"> • Características de los dispositivos serie y paralelo. • Haciendo de abstracción de dispositivos. • Estrategias de buffering. • Acceso directo a memoria. • La recuperación de fallos. 	<ul style="list-style-type: none"> • Explique la diferencia clave entre dispositivos seriales y paralelos e identificar las condiciones en las cuales cada uno es apropiado [Familiarizarse] • Identificar la relación entre el hardware físico y los dispositivos virtuales mantenidos por el sistema operativo [Familiarizarse] • Explicar <i>buffering</i> y describir las estrategias para su aplicación [Familiarizarse] • Diferenciar los mecanismos utilizados en la interconexión de un rango de dispositivos (incluyendo dispositivos portátiles, redes, multimedia) a un ordenador y explicar las implicaciones de éstas para el diseño de un sistema operativo [Familiarizarse] • Describir las ventajas y desventajas de acceso directo a memoria y discutir las circunstancias en cuales se justifica su uso [Familiarizarse] • Identificar los requerimientos para recuperación de errores [Familiarizarse] • Implementar un controlador de dispositivo simple para una gama de posibles equipos [Usar]
Lecturas: Avi Silberschatz (2012), Stallings (2005), Tanenbaum (2006), Tanenbaum (2001), Anderson and Dahlin (2014)	

UNIDAD 9: Sistema de archivos (6)	
Competencias:	
Contenido	Objetivos Generales
<ul style="list-style-type: none"> • Archivos: los datos, metadatos, operaciones, organización, amortiguadores, secuenciales, no secuencial. • Directorios: contenido y estructura. • Los sistemas de archivos: partición, montar / desmontar sistemas de archivos virtuales. • Técnicas estándar de implementación . • Archivos asignados en memoria. • Sistemas de archivos de propósito especial. • Naming, búsqueda, acceso, copias de seguridad. • La bitacora y los sistemas de archivos estructurados (log) 	<ul style="list-style-type: none"> • Describir las decisiones que deben tomarse en el diseño de sistemas de archivos [Familiarizarse] • Comparar y contrastar los diferentes enfoques para la organización de archivos, el reconocimiento de las fortalezas y debilidades de cada uno. [Evaluar] • Resumir cómo el desarrollo de hardware ha dado lugar a cambios en las prioridades para el diseño y la gestión de sistemas de archivos [Familiarizarse] • Resumir el uso de diarios y como los sistemas de archivos de registro estructurado mejora la tolerancia a fallos [Familiarizarse]
Lecturas: Avi Silberschatz (2012), Stallings (2005), Tanenbaum (2006), Tanenbaum (2001), Anderson and Dahlin (2014)	

UNIDAD 10: Sistemas <i>embedded</i> y de tiempo real (6)	
Competencias:	
Contenido	Objetivos Generales
<ul style="list-style-type: none"> • Proceso y programación de tareas. • Los requisitos de gestión de memoria / disco en un entorno en tiempo real. • Los fracasos, los riesgos y la recuperación. • Preocupaciones especiales en sistemas de tiempo real. 	<ul style="list-style-type: none"> • Describir que hace a un sistema un sistema en tiempo real [Familiarizarse] • Explicar la presencia y describir las características de latencia en sistemas de tiempo real [Familiarizarse] • Resumir los problemas especiales que los sistemas en tiempo real presentan, incluyendo el riesgo, y cómo se tratan estos problemas [Familiarizarse]
Lecturas: Avi Silberschatz (2012), Stallings (2005), Tanenbaum (2006), Tanenbaum (2001), Anderson and Dahlin (2014)	

UNIDAD 11: Tolerancia a fallas (3)	
Competencias:	
Contenido	Objetivos Generales
<ul style="list-style-type: none"> • Conceptos fundamentales: sistemas fiables y disponibles. • Redundancia espacial y temporal. • Los métodos utilizados para implementar la tolerancia a fallos. • Los ejemplos de los mecanismos del sistema operativo para la detección, recuperación, reinicio para implementar la tolerancia a fallos, el uso de estas técnicas para los servicios propios del sistema operativo. 	<ul style="list-style-type: none"> • Explicar la importancia de los términos tolerancia a fallos, fiabilidad y disponibilidad [Familiarizarse] • Explicar en términos generales la gama de métodos para implementar la tolerancia a fallos en un sistema operativo [Familiarizarse] • Explicar cómo un sistema operativo puede continuar funcionando después de que ocurra una falla [Familiarizarse]
Lecturas: Avi Silberschatz (2012), Stallings (2005), Tanenbaum (2006), Tanenbaum (2001), Anderson and Dahlin (2014)	

UNIDAD 12: Evaluación del desempeño de sistemas (3)	
Competencias:	
Contenido	Objetivos Generales
<ul style="list-style-type: none"> • ¿Por qué el rendimiento del sistema debe ser evaluado? • ¿Qué se va a evaluar? • Sistemas de políticas de rendimiento, por ejemplo, el almacenamiento en caché, de paginación, la programación, la gestión de memoria, y la seguridad. • Modelos de evaluación: analítica, simulación, o de implementación específico determinista. • Cómo recoger los datos de evaluación (perfiles y mecanismos de localización) 	<ul style="list-style-type: none"> • Describir las medidas de rendimiento utilizados para determinar cómo el sistema funciona [Familiarizarse] • Explicar los principales modelos de evaluación utilizados para evaluar un sistema [Familiarizarse]
Lecturas: Avi Silberschatz (2012), Stallings (2005), Tanenbaum (2006), Tanenbaum (2001), Anderson and Dahlin (2014)	

8. Metodología
<p>El profesor del curso presentará clases teóricas de los temas señalados en el programa propiciando la intervención de los alumnos.</p> <p>El profesor del curso presentará demostraciones para fundamentar clases teóricas.</p> <p>El profesor y los alumnos realizarán prácticas</p> <p>Los alumnos deberán asistir a clase habiendo leído lo que el profesor va a presentar. De esta manera se facilitará la comprensión y los estudiantes estarán en mejores condiciones de hacer consultas en clase.</p>

9. Evaluar

Evaluación Continua 1 : 20 %

Examen parcial : 30 %

Evaluación Continua 2 : 20 %

Examen final : 30 %

References

Anderson, Thomas and Michael Dahlin (2014). *Operating Systems: Principles and Practice*. 2nd. Recursive Books. ISBN: 978-0985673529.

Avi Silberschatz Peter Baer Galvin, Greg Gagne (2012). *Operating System Concepts, 9/E*. John Wiley & Sons, Inc. ISBN: 978-1-118-06333-0.

Stallings, William (2005). *Operating Systems: Internals and Design Principles, 5/E*. Prentice Hall. ISBN: 0-13-147954-7.

Tanenbaum, Andrew S. (2001). *Modern Operating Systems, 4/E*. Prentice Hall. ISBN: 0-13-031358-0.

Tanenbaum, Andrew S. (2006). *Operating Systems Design and Implementation, 3/E*. Prentice Hall. ISBN: 0-13-142938-8.



Universidad Nacional de Colombia (UNAL) Sede
Manizales
Programa Profesional de
Administración de Sistemas Informáticos
SILABO

CS311. Programación Competitiva (Obligatorio)

2022-II

1. Información general	
1.1 Escuela	: Sistemas de Información
1.2 Curso	: CS311. Programación Competitiva
1.3 Semestre	: 6 ^{to} Semestre.
1.4 Prerrequisitos	: CS212. Análisis y Diseño de Algoritmos. (5 ^{to} Sem)
1.5 Condición	: Obligatorio
1.6 Modalidad de aprendizaje	: Presencial
1.7 horas	: 2 HT; 2 HP; 2 HL;
1.8 Créditos	: 4
2. Profesores	
3. Fundamentación del curso	
La Programación Competitiva combina retos de solucionar problemas con el añadido de poder competir con otras personas. Enseña a los participantes a pensar más rápido y desarrollar habilidades para resolver problemas, que son de gran demanda en la industria. Este curso enseñará la resolución de problemas algorítmicos de manera rápida combinando la teoría de algoritmos y estructuras de datos con la práctica la solución de los problemas.	
4. Resumen	
1. Introducción 2. Estructuras de datos 3. Paradigmas de diseño 4. Gráfos 5. Tópicos avanzados 6. Problemas de dominio específico	
5. Objetivos Generales	
<ul style="list-style-type: none">• Que el alumno utilice técnicas de estructuras de datos y algoritmos complejos.• Que el alumno aplique los conceptos aprendidos para la aplicación sobre un problema real.• Que el alumno investigue la posibilidad de crear un nuevo algoritmo y/o técnica nueva para resolver un problema real.	
6. Contribución a los resultados (Outcomes)	
Esta disciplina contribuye al logro de los siguientes resultados de la carrera:	
1) Analizar un problema computacional complejo y aplicar los principios computacionales y otras disciplinas relevantes para identificar soluciones. (Usar)	
6) Aplicar fundamentos de teoría de ciencias de la computación y desarrollo de software para producir soluciones basados en computación. (Usar)	
7. Contenido	

UNIDAD 1: Introducción (20)	
Competencias:	
Contenido	Objetivos Generales
<ul style="list-style-type: none"> • Introducción a la Programación competitiva • Modelo computacional • Complejidad algorítmica • Problemas sobre búsqueda y ordenamiento • Recursión y recurrencia • Estrategia divide y conquista 	<ul style="list-style-type: none"> • Reconocer y saber como usar los recursos del modelo de computación RAM (Random Access Machine). [Usar] • Determinar el tiempo y espacio de complejidad de algoritmos. [Usar] • Determinar relaciones de recurrencia para algoritmos recursivos.[Usar] • Resolver problemas de búsqueda y ordenamiento.[Usar] • Aprender a seleccionar los algoritmos adecuados para problemas de tipo divide y conquista.[Usar] • Diseñar nuevos algoritmos para la resolución de problemas.[Usar]
Lecturas: Cormen et al. (2009), Halim (2013), Kulikov (2019), Miguel A. Revilla (2003), Laaksonen (2017), Aziz, Lee, and Prakash (2012)	

UNIDAD 2: Estructuras de datos (20)	
Competencias:	
Contenido	Objetivos Generales
<ul style="list-style-type: none"> • Problemas sobre arrays y strings • Problemas sobre listas enlazadas • Problemas sobre pilas, colas • Problemas sobre arboles • Problemas sobre Hash tables • Problemas sobre Heaps 	<ul style="list-style-type: none"> • Reconocer las distintas estructuras de datos sus complejidades usos y restricciones. [Usar] • Identificar el tipo de estructura de datos adecuado a la resolución del problema. [Usar] • Reconocer tipos de problemas asociado a operaciones sobre estructuras de datos como búsqueda, inserción, eliminación y actualización.[Usar]
Lecturas: Cormen et al. (2009), Halim (2013), Kulikov (2019), Miguel A. Revilla (2003), Laaksonen (2017), Aziz, Lee, and Prakash (2012)	

UNIDAD 3: Paradigmas de diseño (20)	
Competencias:	
Contenido	Objetivos Generales
<ul style="list-style-type: none"> • Fuerza bruta • Divide y conquista • Backtracking • Greedy • Programación Dinamica 	<ul style="list-style-type: none"> • Aprender los distintos paradigmas de resolución de problemas.[Usar] • Aprender a seleccionar los algoritmos adecuados para distintos problemas según el tipo de paradigma.[Usar]
Lecturas: Cormen et al. (2009), Halim (2013), Kulikov (2019), Miguel A. Revilla (2003), Laaksonen (2017), Aziz, Lee, and Prakash (2012)	

UNIDAD 4: Gráfos (20)	
Competencias:	
Contenido	Objetivos Generales
<ul style="list-style-type: none"> • Recorrido de gráfos • Aplicaciones y problemas sobre gráfos • Camino mas corto • Redes y flujos 	<ul style="list-style-type: none"> • Identificar problemas clasificados como problemas de grafos. [Usar] • Aprender a seleccionar los algoritmos adecuados para problemas de grafos (recorrido, MST, camino mas costo, redes y flujos) y conocer sus soluciones eficientes. [Usar]
Lecturas: Cormen et al. (2009), Halim (2013), Kulikov (2019), Miguel A. Revilla (2003), Laaksonen (2017), Aziz, Lee, and Prakash (2012)	

UNIDAD 5: Tópicos avanzados (20)	
Competencias:	
Contenido	Objetivos Generales
<ul style="list-style-type: none"> • Teoria de números • Probabilidad y combinaciones • Algoritmos para manejos de strings (tries, string hashing, z-algorithm) • Geometria y sweep line algorithms, segment trees 	<ul style="list-style-type: none"> • Aprender a elegir los algoritmos adecuados para problemas sobre teoria de números y matemáticas ya que son importantes en programación competitiva. [Usar] • Aprender a seleccionar los algoritmos adecuados para problemas sobre probabilidades y combinaciones, manejos de strings y geometría computacional. [Usar]
Lecturas: Cormen et al. (2009), Halim (2013), Kulikov (2019), Miguel A. Revilla (2003), Laaksonen (2017), Aziz, Lee, and Prakash (2012)	

UNIDAD 6: Problemas de dominio específico (20)	
Competencias:	
Contenido	Objetivos Generales
<ul style="list-style-type: none"> • Latencia y rendimiento • Paralelismo • Redes • Almacenamiento • Alta disponibilidad • Caching • Proxies • Equilibradores de carga • Almacenamiento clave-valor • Replicar y compartir • Elección del líder • Limitación de la tasa • Registro y monitoreo 	<ul style="list-style-type: none"> • Aprender a diseñar sistemas para diferentes problemas de dominio específico aplicando conocimiento sobre redes, computación distribuida, alta disponibilidad, almacenamiento y arquitectura de sistemas. [Usar]
Lecturas: Cormen et al. (2009), Halim (2013), Kulikov (2019), Miguel A. Revilla (2003), Laaksonen (2017), Aziz, Lee, and Prakash (2012)	

8. Metodología
<p>El profesor del curso presentará clases teóricas de los temas señalados en el programa propiciando la intervención de los alumnos.</p> <p>El profesor del curso presentará demostraciones para fundamentar clases teóricas.</p> <p>El profesor y los alumnos realizarán prácticas</p> <p>Los alumnos deberán asistir a clase habiendo leído lo que el profesor va a presentar. De esta manera se facilitará la comprensión y los estudiantes estarán en mejores condiciones de hacer consultas en clase.</p>

9. Evaluar
<p>Evaluación Continua 1 : 20 %</p> <p>Examen parcial : 30 %</p> <p>Evaluación Continua 2 : 20 %</p> <p>Examen final : 30 %</p>

References

- Aziz, A., T.H. Lee, and A. Prakash (2012). *Elements of Programming Interviews: The Insiders' Guide*. ElementsOfProgrammingInterviews.com. ISBN: 9781479274833.
- Cormen, T. H. et al. (2009). *Introduction to Algorithms*. MIT Press.
- Halim, Steven (2013). *Competitive Programming*. 3 rd. Lulu.
- Kulikov, Alexander S. (2019). *Learning Algorithms Through Programming and Puzzle Solving*. Active Learning Technologies.

- Laaksonen, Antti (2017). *Guide to Competitive Programming: Learning and Improving Algorithms Through Contests*.
Stringer.
- Miguel A. Revilla, Steve Skiena (May 2003). *Programming Challenges: The Programming Contest Training Manual*.
Springer. ISBN: 978-0387001630.



Universidad Nacional de Colombia (UNAL) Sede
Manizales
Programa Profesional de
Administración de Sistemas Informáticos
SILABO

CS312. Estructuras de Datos Avanzadas (Obligatorio)

2022-II

1. Información general	
1.1 Escuela	: Sistemas de Información
1.2 Curso	: CS312. Estructuras de Datos Avanzadas
1.3 Semestre	: 6 ^{to} Semestre.
1.4 Prerrequisitos	: CS212. Análisis y Diseño de Algoritmos. (5 ^{to} Sem)
1.5 Condición	: Obligatorio
1.6 Modalidad de aprendizaje	: Presencial
1.7 horas	: 2 HT; 2 HP; 2 HL;
1.8 Créditos	: 4

2. Profesores

3. Fundamentación del curso
Los algoritmos y estructuras de datos son una parte fundamental de la ciencia de la computación que nos permiten organizar la información de una manera más eficiente, por lo que es importante para todo profesional del área tener una sólida formación en este aspecto. En el curso de estructuras de datos avanzadas nuestro objetivo es que el alumno conozca y analice estructuras complejas, como los Métodos de Acceso Multidimensional, Métodos de Acceso Espacio-Temporal y Métodos de Acceso Métrico, etc.

4. Resumen
1. Técnicas Básicas de Implementación de Estructuras de Datos 2. Métodos de Acceso Multidimensionales 3. Métodos de Acceso Métrico 4. Métodos de Acceso Aproximados 5. Seminarios

5. Objetivos Generales
<ul style="list-style-type: none">• Que el alumno entienda, diseñe, implemente, aplique y proponga estructuras de datos innovadoras para solucionar problemas relacionados al tratamiento de datos multidimensionales, recuperación de información por similitud, motores de búsqueda y otros problemas computacionales.

6. Contribución a los resultados (Outcomes)
Esta disciplina contribuye al logro de los siguientes resultados de la carrera: <ol style="list-style-type: none">1) Analizar un problema computacional complejo y aplicar los principios computacionales y otras disciplinas relevantes para identificar soluciones. (Usar)2) Diseñar, implementar y evaluar una solución basada en computación para cumplir con un conjunto determinado de requisitos computacionales en el contexto de las disciplinas del programa. (Familiarizarse)6) Aplicar fundamentos de teoría de ciencias de la computación y desarrollo de software para producir soluciones basados en computación. (Familiarizarse)

7. Contenido

UNIDAD 1: Técnicas Básicas de Implementación de Estructuras de Datos (16)

Competencias:

Contenido	Objetivos Generales
<ul style="list-style-type: none">• Programación estructurada• Programación Orientada a Objetos• Tipos Abstractos de Datos• Independencia del lenguaje de programación del usuario de la estructura• Independencia de Plataforma• Control de concurrencia• Protección de Datos• Niveles de encapsulamiento (struct, class, namespace, etc)	<ul style="list-style-type: none">• Que el alumno entienda las diferencias básicas que involucran las distintas técnicas de implementación de estructuras de datos[Usar]• Que el alumno analice las ventajas y desventajas de cada una de las técnicas existentes[Usar]

Lecturas: Cuadros-Vargas et al. (2004), Knuth (2007a), Knuth (2007b), Gamma et al. (1994), Björnander (2018), David Vandevoorde (2018)

UNIDAD 2: Métodos de Acceso Multidimensionales (16)

Competencias:

Contenido	Objetivos Generales
<ul style="list-style-type: none">• Métodos de Acceso para datos puntuales• Métodos de Acceso para datos no puntuales• Problemas relacionados con el aumento de dimensión	<ul style="list-style-type: none">• Que el alumno entienda conozca e implemente algunos Métodos de Acceso para datos multidimensionales y espacio temporales[Usar]• Que el alumno entienda el potencial de estos Métodos de Acceso en el futuro de las bases de datos comerciales[Usar]

Lecturas: Samet (2006), Gaede and ünther (1998)

UNIDAD 3: Métodos de Acceso Métrico (20)

Competencias:

Contenido	Objetivos Generales
<ul style="list-style-type: none">• Métodos de Acceso Métrico para distancias discretas• Métodos de Acceso Métrico para distancias continuas	<ul style="list-style-type: none">• Que el alumno entienda conozca e implemente algunos métodos de acceso métrico[Usar]• Que el alumno entienda la importancia de estos Métodos de Acceso para la Recuperación de Información por Similitud[Usar]

Lecturas: Samet (2006), Chávez et al. (2001), Traina Jr et al. (2000), Zezula et al. (2007)

UNIDAD 4: Métodos de Acceso Aproximados (20)	
Competencias:	
Contenido	Objetivos Generales
<ul style="list-style-type: none"> • Space Filling Curves • Locality Sensitive Hashing 	<ul style="list-style-type: none"> • Que el alumno entienda conozca e implemente algunos métodos de acceso aproximados[Usar] • Que el alumno entienda la importancia de estos Métodos de Acceso para la Recuperación de Información por Similitud en entornos donde la Escalabilidad sea una factor muy importante[Usar]
Lecturas: Samet (2006), PGregory Shakhnarovich and Indyk (2006), Zezula et al. (2007)	

UNIDAD 5: Seminarios (8)	
Competencias:	
Contenido	Objetivos Generales
<ul style="list-style-type: none"> • Métodos de Acceso Espacio Temporal • Estructuras de Datos con programación genérica 	<ul style="list-style-type: none"> • Que el alumno pueda discutir sobre los últimos avances en métodos de acceso para distintos dominios de conocimiento[Usar]
Lecturas: Samet (2006), Chávez et al. (2001)	

8. Metodología
<p>El profesor del curso presentará clases teóricas de los temas señalados en el programa propiciando la intervención de los alumnos.</p> <p>El profesor del curso presentará demostraciones para fundamentar clases teóricas.</p> <p>El profesor y los alumnos realizarán prácticas</p> <p>Los alumnos deberán asistir a clase habiendo leído lo que el profesor va a presentar. De esta manera se facilitará la comprensión y los estudiantes estarán en mejores condiciones de hacer consultas en clase.</p>

9. Evaluar
<p>Evaluación Continua 1 : 20 %</p> <p>Examen parcial : 30 %</p> <p>Evaluación Continua 2 : 20 %</p> <p>Examen final : 30 %</p>

References

- Björnander, Stefan (Feb. 2018). *C++17 By Example: Practical projects to get you up and running with C++17*. Packt Publishing.
- Chávez, E. et al. (Sept. 2001). "Proximity Searching in Metric Spaces". In: *ACM Computing Surveys* 33(3), pp. 273–321.
- Cuadros-Vargas, Ernesto et al. (2004). "Implementing data structures: An incremental approach". <http://socios.spc.org.pe/ecuadros/cursos/pdfs/>.
- David Vandevoorde Nicolai M. Josuttis, Doug Gregor (Sept. 2018). *C++ Templates: The Complete Guide*. Addison-Wesley Professional.
- Gaede, Volker and Oliver ünther (1998). "Multidimensional Access Methods". In: *ACM Computing Surveys* 30(2), pp. 170–231.
- Gamma, Erich et al. (Nov. 1994). *Design Patterns: Elements of Reusable Object-Oriented Software*. Computing Series. ISBN-10: 0201633612. Addison-Wesley Professional.

- Knuth, Donald Ervin (Feb. 2007a). *The Art of Computer Programming, Fundamental Algorithms*. 3rd. Vol. I. 0-201-89683-4. Addison-Wesley.
- Knuth, Donald Ervin (Feb. 2007b). *The Art of Computer Programming, Sorting and Searching*. 2nd. Vol. II. 0-201-89685-0. Addison-Wesley.
- PGregory Shakhnarovich, Trevor Darrell and Piotr Indyk (Mar. 2006). *Nearest-Neighbor Methods in Learning and Vision: Theory and Practice*. 1st. ISBN 0-262-19547-X. MIT Press.
- Samet, Hanan (Aug. 2006). *Foundations of Multidimensional and Metric Data Structures*. Illustrated. Elsevier/Morgan Kaufmann. ISBN: 9780123694461.
- Traina Jr, C. et al. (Mar. 2000). “Slim-Trees: High Performance Metric Trees Minimizing Overlap between Nodes”. In: *Advances in Database Technology - EDBT 2000, 6th International Conference on Extending Database Technology*. Vol. 1777. Lecture Notes in Computer Science. Springer: Konstanz, Germany, pp. 51–65.
- Zezula, Pavel et al. (Nov. 2007). *Similarity Search: The Metric Space Approach*. 1st. ISBN-10: 0387291466. Springer.



Universidad Nacional de Colombia (UNAL) Sede
Manizales
Programa Profesional de
Administración de Sistemas Informáticos
SILABO

CS393. Sistemas de Infomación (Obligatorio)

2022-II

1. Información general	
1.1 Escuela	: Sistemas de Información
1.2 Curso	: CS393. Sistemas de Infomación
1.3 Semestre	: 6 ^{to} Semestre.
1.4 Prerrequisitos	: CS291. Ingeniería de Software I. (5 ^{to} Sem)
1.5 Condición	: Obligatorio
1.6 Modalidad de aprendizaje	: Presencial
1.7 horas	: 2 HT; 2 HP; 2 HL;
1.8 Créditos	: 4
2. Profesores	
3. Fundamentación del curso	
Analizar técnicas para la correcta implementación de Sistemas de Información escalables, robustos, confiables y eficientes en las organizaciones.	
4. Resumen	
1. Introducción 2. Estrategia 3. Implementación	
5. Objetivos Generales	
<ul style="list-style-type: none">● Implementar de forma correcta (escalables, robustos, confiables y eficientes) Sistemas de Información en las organizaciones.	
6. Contribución a los resultados (Outcomes)	
Esta disciplina contribuye al logro de los siguientes resultados de la carrera:	
2) Diseñar, implementar y evaluar una solución basada en computación para cumplir con un conjunto determinado de requisitos computacionales en el contexto de las disciplinas del programa. (Usar)	
6) Aplicar fundamentos de teoría de ciencias de la computación y desarrollo de software para producir soluciones basados en computación. (Evaluar)	
7. Contenido	

UNIDAD 1: Introducción (15)	
Competencias:	
Contenido	Objetivos Generales
<ul style="list-style-type: none"> • Introducción a la gestión de la información • Software para gestión de información. • Tecnología para gestión de información. 	<ul style="list-style-type: none"> • Aplicar correctamente la tecnología para la gestión de la información [Evaluar]
Lecturas: Sommerville (2017), Pressman and Maxim (2015), K. C. Laudon and J. P. Laudon (2017)	

UNIDAD 2: Estrategia (15)	
Competencias:	
Contenido	Objetivos Generales
<ul style="list-style-type: none"> • Estrategia para gestión de información • Estrategia para gestión conocimiento • Estrategia para sistema de información. 	<ul style="list-style-type: none"> • Aplicar y evaluar correctamente estrategias de gestión [Evaluar]
Lecturas: Sommerville (2017), Pressman and Maxim (2015)	

UNIDAD 3: Implementación (15)	
Competencias:	
Contenido	Objetivos Generales
<ul style="list-style-type: none"> • Gestión de desarrollo de sistemas de información. • Gestión del cambio • Arquitectura de Información 	<ul style="list-style-type: none"> • Aplicar y evaluar correctamente estrategias de implementación [Evaluar]
Lecturas: Sommerville (2017), Pressman and Maxim (2015)	

8. Metodología
<p>El profesor del curso presentará clases teóricas de los temas señalados en el programa propiciando la intervención de los alumnos.</p> <p>El profesor del curso presentará demostraciones para fundamentar clases teóricas.</p> <p>El profesor y los alumnos realizarán prácticas</p> <p>Los alumnos deberán asistir a clase habiendo leído lo que el profesor va a presentar. De esta manera se facilitará la comprensión y los estudiantes estarán en mejores condiciones de hacer consultas en clase.</p>

9. Evaluar
<p>Evaluación Continua 1 : 20 %</p> <p>Examen parcial : 30 %</p> <p>Evaluación Continua 2 : 20 %</p> <p>Examen final : 30 %</p>

References

- Laudon, Kenneth C. and Jane P. Laudon (Mar. 2017). *Management Information Systems: Managing the Digital Firm*. 15th. Pearson.
- Pressman, Roger S. and Bruce Maxim (Jan. 2015). *Software Engineering: A Practitioner's Approach*. 8th. McGraw-Hill.
- Sommerville, Ian (Mar. 2017). *Software Engineering*. 10th. Pearson.



Universidad Nacional de Colombia (UNAL) Sede
Manizales
Programa Profesional de
Administración de Sistemas Informáticos
SILABO

MA307. Matemática aplicada a la computación
(Obligatorio)

2022-II

1. Información general

1.1 Escuela	:	Sistemas de Información
1.2 Curso	:	MA307. Matemática aplicada a la computación
1.3 Semestre	:	6 ^{to} Semestre.
1.4 Prerrequisitos	:	MA201. Cálculo II. (4 ^{to} Sem)
1.5 Condición	:	Obligatorio
1.6 Modalidad de aprendizaje	:	Presencial
1.7 horas	:	2 HT; 2 HP; 2 HL;
1.8 Créditos	:	4

2. Profesores

3. Fundamentación del curso

Este curso es importante porque desarrolla tópicos del Álgebra Lineal y de Ecuaciones Diferenciales Ordinarias útiles en todas aquellas áreas de la ciencia de la computación donde se trabaja con sistemas lineales y sistemas dinámicos.

4. Resumen

1. Espacios Lineales 2. Transformaciones lineales 3. Autovalores y autovectores 4. Sistemas de ecuaciones diferenciales 5. Teoría fundamental 6. Estabilidad de equilibrio

5. Objetivos Generales

- Que el alumno tenga la base matemática para el modelamiento de sistemas lineales y sistemas dinámicos necesarios en el Área de Computación Gráfica e Inteligencia Artificial.

6. Contribución a los resultados (Outcomes)

Esta disciplina contribuye al logro de los siguientes resultados de la carrera:

- 1) Analizar un problema computacional complejo y aplicar los principios computacionales y otras disciplinas relevantes para identificar soluciones. (Usar)
- 6) Aplicar fundamentos de teoría de ciencias de la computación y desarrollo de software para producir soluciones basados en computación. (Usar)

7. Contenido

UNIDAD 1: Espacios Lineales (0)	
Competencias:	
Contenido	Objetivos Generales
<ul style="list-style-type: none"> • Espacios vectoriales. • Independencia, base y dimensión. • Dimensiones y ortogonalidad de los cuatro subespacios. • Aproximaciones por mínimos cuadrados. • Proyecciones • Bases ortogonales y Gram-Schmidt 	<ul style="list-style-type: none"> • Identificar espacios generados por vectores linealmente independientes[Usar] • Construir conjuntos de vectores ortogonales[Usar] • Aproximar funciones por polinomios trigonométricos[Usar]
Lecturas: Strang (2003), Apóstol (1973)	
UNIDAD 2: Transformaciones lineales (0)	
Competencias:	
Contenido	Objetivos Generales
<ul style="list-style-type: none"> • Concepto de transformación lineal. • Matriz de una transformación lineal. • Cambio de base. • Diagonalización y pseudoinversa 	<ul style="list-style-type: none"> • Determinar el núcleo y la imagen de una transformación[Usar] • Construir la matriz de una transformación[Usar] • Determinar la matriz de cambio de base[Usar]
Lecturas: Strang (2003), Apóstol (1973)	
UNIDAD 3: Autovalores y autovectores (0)	
Competencias:	
Contenido	Objetivos Generales
<ul style="list-style-type: none"> • Diagonalización de una matriz • Matrices simétricas • Matrices definidas positivas • Matrices similares • La descomposición de valor singular 	<ul style="list-style-type: none"> • Encontrar la representación diagonal de una matriz[Usar] • Determinar la similaridad entre matrices[Usar] • Reducir una forma cuadrática real a diagonal[Usar]
Lecturas: Strang (2003), Apóstol (1973)	
UNIDAD 4: Sistemas de ecuaciones diferenciales (0)	
Competencias:	
Contenido	Objetivos Generales
<ul style="list-style-type: none"> • Exponencial de una matriz • Teoremas de existencia y unicidad para sistemas lineales homogéneos con coeficientes constantes • Sistemas lineales no homogéneos con coeficientes constantes. 	<ul style="list-style-type: none"> • Hallar la solución general de un sistema lineal no homogéneo[Usar] • Resolver problemas donde intervengan sistemas de ecuaciones diferenciales[Usar]
Lecturas: Zill (2002), Apóstol (1973)	

UNIDAD 5: Teoría fundamental (0)	
Competencias:	
Contenido	Objetivos Generales
<ul style="list-style-type: none"> • Sistemas dinámicos • El teorema fundamental • Existencia y unicidad • El flujo de una ecuación diferencial 	<ul style="list-style-type: none"> • Discutir la existencia y la unicidad de una ecuación diferencial[Usar] • Analizar la continuidad de las soluciones[Usar] • Estudiar la prolongación de una solución[Usar]
Lecturas: Hirsh and Smale (1974)	

UNIDAD 6: Estabilidad de equilibrio (0)	
Competencias:	
Contenido	Objetivos Generales
<ul style="list-style-type: none"> • Estabilidad • Funciones de Liapunov • Sistemas gradientes 	<ul style="list-style-type: none"> • Analizar la estabilidad de una solución[Usar] • Hallar la función de Liapunov para puntos de equilibrio[Usar] • Trazar el retrato de fase un flujo gradiente[Usar]
Lecturas: Zill (2002), Hirsh and Smale (1974)	

8. Metodología
<p>El profesor del curso presentará clases teóricas de los temas señalados en el programa propiciando la intervención de los alumnos.</p> <p>El profesor del curso presentará demostraciones para fundamentar clases teóricas.</p> <p>El profesor y los alumnos realizarán prácticas</p> <p>Los alumnos deberán asistir a clase habiendo leído lo que el profesor va a presentar. De esta manera se facilitará la comprensión y los estudiantes estarán en mejores condiciones de hacer consultas en clase.</p>

9. Evaluar
<p>Evaluación Continua 1 : 20 %</p> <p>Examen parcial : 30 %</p> <p>Evaluación Continua 2 : 20 %</p> <p>Examen final : 30 %</p>

References

- Apóstol, Tom M (1973). *Calculus Vol II*. Editorial Reverté.
- Hirsh, Morris W. and Stephen Smale (1974). *Differential Equations, Dynamical Systems, and Linear Algebra*. Academia Press.
- Strang, Gilbert (2003). *Introduction to Linear Algebra, 3ª edición*. Wellesley-Cambridge Press.
- Zill, Dennis G. (2002). *Ecuaciones Diferenciales con Problemas de Valores en la Frontera*. Thomson Learning. ISBN: 970-686-133-5.



Universidad Nacional de Colombia (UNAL) Sede
Manizales
Programa Profesional de
Administración de Sistemas Informáticos
SILABO

FG203. Oratoria (Obligatorio)

2022-II

1. Información general	
1.1 Escuela	: Sistemas de Información
1.2 Curso	: FG203. Oratoria
1.3 Semestre	: 6 ^{to} Semestre.
1.4 Prerrequisitos	: FG106. Teatro. (5 ^{to} Sem)
1.5 Condición	: Obligatorio
1.6 Modalidad de aprendizaje	: Presencial
1.7 horas	: 1 HT; 2 HP;
1.8 Créditos	: 2
2. Profesores	
3. Fundamentación del curso	
En la sociedad competitiva como la nuestra, se exige que la persona sea un comunicador eficaz y sepa utilizar sus potencialidades a fin de resolver problemas y enfrentar los desafíos del mundo moderno dentro de la actividad laboral, intelectual y social. Tener el conocimiento no basta, lo importante es saber comunicarlo y en la medida que la persona sepa emplear sus facultades comunicativas, derivará en éxito o fracaso aquello que tenga que realizar en su desenvolvimiento personal y profesional. Por ello es necesario para lograr un buen decir, recurrir a conocimientos, estrategias y recursos, que debe tener todo orador, para llegar con claridad, precisión y convicción al interlocutor	
4. Resumen	
1. 2.	
5. Objetivos Generales	
<ul style="list-style-type: none">• Al término del curso, el alumno será capaz de organizar y asumir la palabra desde la perspectiva del orador, en cualquier situación, en forma más correcta, coherente y adecuada, mediante el uso de conocimientos y habilidades lingüísticas, buscando en todo momento su realización personal y social a través de su expresión, teniendo como base la verdad y la preparación constante.	
6. Contribución a los resultados (Outcomes)	
Esta disciplina contribuye al logro de los siguientes resultados de la carrera:	
3) Comunicarse efectivamente en diversos contextos profesionales. (Usar)	
4) Reconocer las responsabilidades profesionales y hacer juicios informados en el campo profesional de computación con principios éticos. (Usar)	
7) Desarrollar tecnología computacional buscando el bien común, aportando con formación humana, capacidades científicas, tecnológicas y profesionales para solucionar problemas sociales de nuestro entorno. (Usar)	
7. Contenido	

UNIDAD 1: (3)	
Competencias:	
Contenido	Objetivos Generales
<ul style="list-style-type: none"> • La Oratoria • La función de la palabra. • El proceso de la comunicación. • Bases racionales y emocionales de la oratoria <ul style="list-style-type: none"> – La expresión oral en la participación. • Fuentes de conocimiento para la oratoria: niveles de cultura general. 	<ul style="list-style-type: none"> • Comprensión: interpretar, ejemplificar y generalizar las bases de la oratoria como fundamento teórico y práctico. [Usar].
Lecturas: Monroe and Ehninger (1976), Rodríguez (n.d.)	

UNIDAD 2: (4)	
Competencias:	
Contenido	Objetivos Generales
<ul style="list-style-type: none"> • Cualidades de un buen orador. • Normas para primeros discursos. • El cuerpo humano como instrumento de comunicación: <ul style="list-style-type: none"> – La expresión corporal en el discurso – La voz en el discurso. • Oradores con historia y su ejemplo. 	<ul style="list-style-type: none"> • Comprensión: Interpretar, ejemplificar y generalizar conocimientos y habilidades de la comunicación oral mediante la experiencia de grandes oradores y la suya propia. [Usar]. • Aplicación: Implementar, usar, elegir y desempeñar los conocimientos adquiridos para expresarse en público en forma eficiente, inteligente y agradable. [Usar].
Lecturas: Rodríguez (n.d.)	

8. Metodología
<p>El profesor del curso presentará clases teóricas de los temas señalados en el programa propiciando la intervención de los alumnos.</p> <p>El profesor del curso presentará demostraciones para fundamentar clases teóricas.</p> <p>El profesor y los alumnos realizarán prácticas</p> <p>Los alumnos deberán asistir a clase habiendo leído lo que el profesor va a presentar. De esta manera se facilitará la comprensión y los estudiantes estarán en mejores condiciones de hacer consultas en clase.</p>

9. Evaluar
<p>Evaluación Continua 1 : 20 %</p> <p>Examen parcial : 30 %</p> <p>Evaluación Continua 2 : 20 %</p> <p>Examen final : 30 %</p>

References

Monroe, A. and D. Ehninger (1976). *La comunicación oral*. Hispano Europea.

Rodríguez, María L. (n.d.). *Cómo manejar la información en una presentación.*



Universidad Nacional de Colombia (UNAL) Sede
Manizales
Programa Profesional de
Administración de Sistemas Informáticos
SILABO

CS231. Redes y Comunicación (Obligatorio)

2022-II

1. Información general	
1.1 Escuela	: Sistemas de Información
1.2 Curso	: CS231. Redes y Comunicación
1.3 Semestre	: 7 ^{mo} Semestre.
1.4 Prerrequisitos	: CS2S1. Sistemas Operativos. (6 ^{to} Sem)
1.5 Condición	: Obligatorio
1.6 Modalidad de aprendizaje	: Presencial
1.7 horas	: 1 HT; 2 HP; 2 HL;
1.8 Créditos	: 3
2. Profesores	
3. Fundamentación del curso	
El siempre creciente desarrollo de las tecnologías de comunicación y la información hace que exista una marcada tendencia a establecer más redes de computadores que permitan una mejor gestión de la información. En este segundo curso se brindará a los participantes una introducción a los problemas que conlleva la comunicación entre computadores, a través del estudio e implementación de protocolos de comunicación como TCP/IP y la implementación de software sobre estos protocolos.	
4. Resumen	
1. Introducción 2. Aplicaciones en red 3. Entrega confiable de datos 4. Ruteo y reenvío 5. Redes de área local 6. Asignación de recursos 7. Celulares 8. Redes sociales	
5. Objetivos Generales	
<ul style="list-style-type: none">• Que el alumno implemente y/o modifique un protocolo de comunicación de datos.• Que el alumno domine las técnicas de transmisión de datos utilizadas por los protocolos de red existentes.• Que el alumno conozca las últimas tendencias en redes que se están aplicando en el Internet.	

6. Contribución a los resultados (*Outcomes*)

Esta disciplina contribuye al logro de los siguientes resultados de la carrera:

- 1) Analizar un problema computacional complejo y aplicar los principios computacionales y otras disciplinas relevantes para identificar soluciones. (**Usar**)
- 2) Diseñar, implementar y evaluar una solución basada en computación para cumplir con un conjunto determinado de requisitos computacionales en el contexto de las disciplinas del programa. (**Usar**)
- 4) Reconocer las responsabilidades profesionales y hacer juicios informados en el campo profesional de computación con principios éticos. (**Familiarizarse**)
- 6) Aplicar fundamentos de teoría de ciencias de la computación y desarrollo de software para producir soluciones basados en computación. (**Usar**)
- 7) Desarrollar tecnología computacional buscando el bien común, aportando con formación humana, capacidades científicas, tecnológicas y profesionales para solucionar problemas sociales de nuestro entorno. (**Evaluar**)

7. Contenido

UNIDAD 1: Introducción (5)

Competencias:

Contenido	Objetivos Generales
<ul style="list-style-type: none">• Organización de la Internet (proveedores de servicios de Internet, proveedores de contenido, etc)• Técnicas de Switching (por ejemplo, de circuitos, de paquetes)• Piezas físicas de una red, incluidos hosts, routers, switches, ISPs, inalámbrico, LAN, punto de acceso y firewalls.• Principios de capas (encapsulación, multiplexación)• Roles de las diferentes capas (aplicación, transporte, red, enlace de datos, física)	<ul style="list-style-type: none">• Articular la organización de la Internet [Familiarizarse]• Listar y definir la terminología de red apropiada [Familiarizarse]• Describir la estructura en capas de una arquitectura típica en red [Familiarizarse]• Identificar los diferentes tipos de complejidad en una red (bordes, núcleo, etc.) [Familiarizarse]

Lecturas: Kurose and Ross (2013)

UNIDAD 2: Aplicaciones en red (5)

Competencias:

Contenido	Objetivos Generales
<ul style="list-style-type: none">• Esquemas de denominación y dirección (DNS, direcciones IP, identificadores de recursos uniformes, etc)• Las aplicaciones distribuidas (cliente / servidor, peer-to-peer, nube, etc)• HTTP como protocolo de capa de aplicación .• Multiplexación con TCP y UDP• API de Socket	<ul style="list-style-type: none">• Listar las diferencias y las relaciones entre los nombres y direcciones en una red [Familiarizarse]• Definir los principios detrás de esquemas de denominación y ubicación del recurso [Familiarizarse]• Implementar una aplicación simple cliente-servidor basada en <i>sockets</i> [Usar]

Lecturas: Kurose and Ross (2013)

UNIDAD 3: Entrega confiable de datos (10)	
Competencias:	
Contenido	Objetivos Generales
<ul style="list-style-type: none"> • Control de errores (técnicas de retransmisión, temporizadores) • El control de flujo (agradecimientos, ventana deslizante) • Problemas de rendimiento (pipelining) • TCP 	<ul style="list-style-type: none"> • Describir el funcionamiento de los protocolos de entrega fiables [Familiarizarse] • Listar los factores que afectan al rendimiento de los protocolos de entrega fiables [Familiarizarse] • Diseñar e implementar un protocolo confiable simple [Usar]
Lecturas: Kurose and Ross (2013)	

UNIDAD 4: Ruteo y reenvío (12)	
Competencias:	
Contenido	Objetivos Generales
<ul style="list-style-type: none"> • Enrutamiento vs reenvío . • Enrutamiento estático . • Protocolo de Internet (IP) • Problemas de escalabilidad (direccionamiento jerárquico) 	<ul style="list-style-type: none"> • Describir la organización de la capa de red [Familiarizarse] • Describir cómo los paquetes se envían en una red IP [Familiarizarse] • Listar las ventajas de escalabilidad de direccionamiento jerárquico [Familiarizarse]
Lecturas: Kurose and Ross (2013)	

UNIDAD 5: Redes de área local (10)	
Competencias:	
Contenido	Objetivos Generales
<ul style="list-style-type: none"> • Problemas de Acceso Múltiple. • Enfoques comunes a Acceso múltiple (exponencial backoff, multiplexación por división de tiempo, etc) • Redes de área local . • Ethernet . • Switching . 	<ul style="list-style-type: none"> • Describir como los paquetes son enviados en una red Ethernet [Familiarizarse] • Describir las relaciones entre IP y Ethernet [Familiarizarse] • Describir las etapas usadas en un enfoque común para el problema de múltiples accesos [Familiarizarse]
Lecturas: Kurose and Ross (2013)	

UNIDAD 6: Asignación de recursos (12)	
Competencias:	
Contenido	Objetivos Generales
<ul style="list-style-type: none"> • Necesidad de asignación de recursos . • Asignación fija (TDM, FDM, WDM) versus la asignación dinámica . • De extremo a extremo frente a las red de enfoque asistida . • Justicia. • Principios del control de congestión. • Enfoques para la congestión (por ejemplo, redes de distribución de contenidos) 	<ul style="list-style-type: none"> • Describir como los recursos pueden ser almacenados en la red [Familiarizarse] • Describir los problemas de congestión en una red grande [Familiarizarse] • Comparar y contrastar las técnicas de almacenamiento estático y dinámico [Familiarizarse] • Comparar y contrastar los enfoques actuales de la congestión [Familiarizarse]
Lecturas: Kurose and Ross (2013)	

UNIDAD 7: Celulares (5)	
Competencias:	
Contenido	Objetivos Generales
<ul style="list-style-type: none"> • Principios de redes celulares. • Redes 802.11 • Problemas en el apoyo a los nodos móviles (agente local) 	<ul style="list-style-type: none"> • Describir la organización de una red inalámbrica [Familiarizarse] • Describir como las redes inalámbricas soportan usuarios móviles [Familiarizarse]
Lecturas: Kurose and Ross (2013), Chayapathi (2016)	

UNIDAD 8: Redes sociales (5)	
Competencias:	
Contenido	Objetivos Generales
<ul style="list-style-type: none"> • Panorama de las redes sociales. • Ejemplo plataformas de redes sociales. • Estructura de los grafos de redes sociales. • Análisis de redes sociales. 	<ul style="list-style-type: none"> • Discutir los principios fundamentales (como pertenencia, confianza) de una red social [Familiarizarse] • Describir como redes sociales existentes operan [Familiarizarse] • Construir un grafo de una red social a partir de datos de la red [Usar] • Analizar una red social para determinar quienes son las personas importantes [Usar] • Evaluar una determinada interpretación de una pregunta de red social con los datos asociados [Familiarizarse]
Lecturas: Kurose and Ross (2013), Kadushin (2011)	

8. Metodología

El profesor del curso presentará clases teóricas de los temas señalados en el programa propiciando la intervención de los alumnos.

El profesor del curso presentará demostraciones para fundamentar clases teóricas.

El profesor y los alumnos realizarán prácticas

Los alumnos deberán asistir a clase habiendo leído lo que el profesor va a presentar. De esta manera se facilitará la comprensión y los estudiantes estarán en mejores condiciones de hacer consultas en clase.

9. Evaluar

Evaluación Continua 1 : 20 %

Examen parcial : 30 %

Evaluación Continua 2 : 20 %

Examen final : 30 %

References

- Chayapathi Rajendra; Syed F. Hassan; Shah, Paresh (2016). *Network Functions Virtualization (NFV) with a Touch of SDN*. Addison-Wesley Professional; 1 edition. ISBN: 978-0134463056.
- Kadushin, Charles (2011). *Understanding Social Networks: Theories, Concepts, And Findings*. Oxford University Press, Usa; 1 edition. ISBN: 978-0195379471.
- Kurose, J.F. and K.W. Ross (2013). *Computer Networking: A Top-down Approach*. 7th. Always learning. Pearson. ISBN: 978-0133594140.



Universidad Nacional de Colombia (UNAL) Sede
Manizales
Programa Profesional de
Administración de Sistemas Informáticos
SILABO

CS261. Inteligencia Artificial (Obligatorio)

2022-II

1. Información general	
1.1 Escuela	: Sistemas de Información
1.2 Curso	: CS261. Inteligencia Artificial
1.3 Semestre	: 7 ^{mo} Semestre.
1.4 Prerrequisitos	: MA203. Estadística y Probabilidades. (4 ^{to} Sem)
1.5 Condición	: Obligatorio
1.6 Modalidad de aprendizaje	: Presencial
1.7 horas	: 2 HT; 2 HP; 2 HL;
1.8 Créditos	: 4
2. Profesores	
3. Fundamentación del curso	
La investigación en Inteligencia Artificial ha conducido al desarrollo de numerosas técnicas relevantes, dirigidas a la automatización de la inteligencia humana, dando una visión panorámica de diferentes algoritmos que simulan los diferentes aspectos del comportamiento y la inteligencia del ser humano.	
4. Resumen	
1. Cuestiones fundamentales 2. Agentes 3. Estrategias de búsquedas básicas 4. Búsqueda Avanzada 5. Razonamiento Bajo Incertidumbre 6. Aprendizaje Automático Básico 7. Aprendizaje de máquina avanzado 8. Procesamiento del Lenguaje Natural 9. Visión y percepción por computador	
5. Objetivos Generales	
<ul style="list-style-type: none">• Evaluar las posibilidades de simulación de la inteligencia, para lo cual se estudiarán las técnicas de modelización del conocimiento.• Construir una noción de inteligencia que soporte después las tareas de su simulación.	
6. Contribución a los resultados (Outcomes)	
Esta disciplina contribuye al logro de los siguientes resultados de la carrera:	
1) Analizar un problema computacional complejo y aplicar los principios computacionales y otras disciplinas relevantes para identificar soluciones. (Usar)	
5) Funcionar efectivamente como miembro o líder de un equipo involucrado en actividades apropiadas a la disciplina del programa. (Familiarizarse)	
6) Aplicar fundamentos de teoría de ciencias de la computación y desarrollo de software para producir soluciones basados en computación. (Familiarizarse)	
7. Contenido	

UNIDAD 1: Cuestiones fundamentales (2)	
Competencias:	
Contenido	Objetivos Generales
<ul style="list-style-type: none"> • Descripción general de los problemas de Inteligencia Artificial, ejemplos recientes de aplicaciones de Inteligencia artificial. • ¿Qué es comportamiento inteligente? <ul style="list-style-type: none"> – El Test de Turing – Razonamiento Racional versus No Racional • Características del Problema: <ul style="list-style-type: none"> – Observable completamente versus observable parcialmente – Individual versus multi-agente – Determinístico versus estocástico – Estático versus dinámico – Discreto versus continuo • Naturaleza de agentes: <ul style="list-style-type: none"> – Autónomo versus semi-autónomo – Reflexivo, basado en objetivos, y basado en utilidad – La importancia en percepción e interacciones con el entorno • Cuestiones filosóficas y éticas. 	<ul style="list-style-type: none"> • Describir el test de Turing y el experimento pensado "cuarto chino" (<i>Chinese Room</i>) [Usar] • Determinando las características de un problema dado que sistemas inteligentes deberían resolver [Usar]
Lecturas: De Castro (2006), Ponce-Gallegos et al. (2014)	

UNIDAD 2: Agentes (2)	
Competencias:	
Contenido	Objetivos Generales
<ul style="list-style-type: none"> • Definición de Agentes • Arquitectura de agentes (Ej. reactivo, en capa, cognitivo) • Teoría de agentes • Racionalidad, teoría de juegos: <ul style="list-style-type: none"> – Agentes de decisión teórica – Procesos de decisión de Markov (MDP) • Agentes de Software, asistentes personales, y acceso a información: <ul style="list-style-type: none"> – Agentes colaborativos – Agentes de recolección de información – Agentes creíbles (carácter sintético, modelamiento de emociones en agentes) • Agentes de aprendizaje • Sistemas Multi-agente <ul style="list-style-type: none"> – Agentes Colaborativos – Equipos de Agentes – Agentes Competitivos (ej., subastas, votaciones) – Sistemas de enjambre y modelos biológicamente inspirados 	<ul style="list-style-type: none"> • Lista las características que definen un agente inteligente [Usar] • Describe y contrasta las arquitecturas de agente estándares [Usar] • Describe las aplicaciones de teoría de agentes para dominios como agentes de software, asistentes personales, y agentes creíbles [Usar] • Describe los paradigmas primarios usados por agentes de aprendizaje [Usar] • Demuestra mediante ejemplos adecuados como los sistemas multi-agente soportan interacción entre agentes [Usar]
Lecturas: Nilsson (2001), Russell and Norvig (2003), Ponce-Gallegos et al. (2014)	

UNIDAD 3: Estrategias de búsquedas básicas (2)	
Competencias:	
Contenido	Objetivos Generales
<ul style="list-style-type: none"> • Espacios de Problemas (estados, metas y operadores), solución de problemas mediante búsqueda. • Factored representation (factoring state hacia variables) • Uninformed search (breadth-first, depth-first, depth-first with iterative deepening) • Heurísticas y búsqueda informada (hill-climbing, generic best-first, A*) • El espacio y el tiempo de la eficiencia de búsqueda. • Dos jugadores juegos (introducción a la búsqueda minimax). • Satisfacción de restricciones (backtracking y métodos de búsqueda local). 	<ul style="list-style-type: none"> • Formula el espacio eficiente de un problema para un caso expresado en lenguaje natural (ejm. Inglés) en términos de estados de inicio y final, así como sus operadores [Usar] • Describe el rol de las heurísticas y describe los intercambios entre completitud, óptimo, complejidad de tiempo, y complejidad de espacio [Usar] • Describe el problema de la explosión combinatoria del espacio de búsqueda y sus consecuencias [Usar] • Compara y contrasta tópicos de búsqueda básica con temas jugabilidad de juegos [Usar]
Lecturas: Nilsson (2001), Ponce-Gallegos et al. (2014)	

UNIDAD 4: Búsqueda Avanzada (18)	
Competencias:	
Contenido	Objetivos Generales
<ul style="list-style-type: none"> • Búsqueda estocástica: <ul style="list-style-type: none"> – Simulated annealing – Algoritmos genéticos – Búsqueda de árbol Monte-Carlo • Construcción de árboles de búsqueda, espacio de búsqueda dinámico, explosión combinatoria del espacio de búsqueda. • Implementación de búsqueda A*, búsqueda en haz. • Búsqueda Minimax, poda alfa-beta. • Búsqueda Expectimax (MDP-Solving) y los nodos de azar. 	<ul style="list-style-type: none"> • Diseñar e implementar una solución a un problema con algoritmo genético [Usar] • Diseñar e implementar un esquema de recocido simulado (<i>simulated annealing</i>) para evitar mínimos locales en un problema [Usar] • Diseñar e implementar una búsqueda A* y búsqueda en haz (<i>beam search</i>) para solucionar un problema [Usar] • Aplicar búsqueda minimax con poda alfa-beta para simplificar el espacio de búsqueda en un juego con dos jugadores [Usar] • Comparar y contrastar los algoritmos genéticos con técnicas clásicas de búsqueda [Usar] • Comparar y contrastar la aplicabilidad de varias heurísticas de búsqueda, para un determinado problema [Usar]
Lecturas: Goldberg (1989), Nilsson (2001), Russell and Norvig (2003), Ponce-Gallegos et al. (2014)	

UNIDAD 5: Razonamiento Bajo Incertidumbre (18)	
Competencias:	
Contenido	Objetivos Generales
<ul style="list-style-type: none"> • Revisión de Probabilidad Básica • Variables aleatorias y distribuciones de probabilidad: <ul style="list-style-type: none"> – Axiomas de probabilidad – Inferencia probabilística – Regla de Bayes • Independencia Condicional • Representaciones del conocimiento: <ul style="list-style-type: none"> – Redes bayesianas <ul style="list-style-type: none"> * Inferencia exacta y su complejidad * Métodos de Muestreo aleatorio (Monte Carlo) (p.e. Muestreo de Gibbs) – Redes Markov – Modelos de probabilidad relacional – Modelos ocultos de Markov 	<ul style="list-style-type: none"> • Aplicar la regla de Bayes para determinar el cumplimiento de una hipótesis [Usar] • Explicar cómo al tener independencia condicional permite una gran eficiencia en sistemas probabilísticos [Usar] • Identificar ejemplos de representación de conocimiento para razonamiento bajo incertidumbre [Usar] • Indicar la complejidad de la inferencia exacta. Identificar métodos para inferencia aproximada [Usar]
Lecturas: Koller and Friedman (2009), Russell and Norvig (2003)	

UNIDAD 6: Aprendizaje Automático Básico (4)	
Competencias:	
Contenido	Objetivos Generales
<ul style="list-style-type: none"> • Definición y ejemplos de la extensa variedad de tareas de aprendizaje de máquina, incluida la clasificación. • Aprendizaje inductivo • Aprendizaje simple basado en estadísticas, como el clasificador ingenuo de Bayes, árboles de decisión. • El problema exceso de ajuste. • Medición clasificada con exactitud. 	<ul style="list-style-type: none"> • Listar las diferencias entre los tres principales tipos de aprendizaje: supervisado, no supervisado y por refuerzo [Usar] • Identificar ejemplos de tareas de clasificación, considerando las características de entrada disponibles y las salidas a ser predecidas [Usar] • Explicar la diferencia entre aprendizaje inductivo y deductivo [Usar] • Describir el sobre ajuste (<i>overfitting</i>) en el contexto de un problema [Usar] • Aplicar un algoritmo de aprendizaje estadístico simple como el Clasificador Naive Bayesiano e un problema de clasificación y medir la precisión del clasificador [Usar]
Lecturas: Mitchell (1998), Russell and Norvig (2003), Ponce-Gallegos et al. (2014)	

UNIDAD 7: Aprendizaje de máquina avanzado (20)	
Competencias:	
Contenido	Objetivos Generales
<ul style="list-style-type: none"> • Definición y ejemplos de una amplia variedad de tareas de aprendizaje de máquina • Aprendizaje general basado en estadística, estimación de parámetros (máxima probabilidad) • Programación lógica inductiva (<i>Inductive logic programming ILP</i>) • Aprendizaje supervisado <ul style="list-style-type: none"> – Aprendizaje basado en árboles de decisión – Aprendizaje basado en redes neuronales – Aprendizaje basado en máquinas de soporte vectorial (<i>Support vector machines SVMs</i>) • Aprendizaje y <i>clustering</i> no supervisado <ul style="list-style-type: none"> – EM – K-means – Mapas auto-organizados • Aprendizaje semi-supervisado. • Aprendizaje de modelos gráficos • Evaluación del desempeño (tal como cross-validation, area bajo la curva ROC) • Aplicación de algoritmos Machine Learning para Minería de datos. 	<ul style="list-style-type: none"> • Explica las diferencias entre los tres estilos de aprendizaje: supervisado, por refuerzo y no supervisado [Usar] • Implementa algoritmos simples para el aprendizaje supervisado, aprendizaje por refuerzo, y aprendizaje no supervisado [Usar] • Determina cuál de los tres estilos de aprendizaje es el apropiado para el dominio de un problema en particular [Usar] • Compara y contrasta cada una de las siguientes técnicas, dando ejemplo de cuando una estrategia es la mejor: árboles de decisión, redes neuronales, y redes bayesianas [Usar] • Evalúa el rendimiento de un sistema de aprendizaje simple en un conjunto de datos reales [Usar] • Describe el estado del arte en la teoría del aprendizaje, incluyendo sus logros y limitantes [Usar] • Explica el problema del sobreajuste, conjuntamente con técnicas para determinar y manejar el problema [Usar]
Lecturas: Russell and Norvig (2003), Koller and Friedman (2009), Murphy (2012)	

UNIDAD 8: Procesamiento del Lenguaje Natural (12)	
Competencias:	
Contenido	Objetivos Generales
<ul style="list-style-type: none"> • Gramaticas determinísticas y estocásticas • Algoritmos de parseo <ul style="list-style-type: none"> – Gramáticas libres de contexto (CFGs) y cuadros de parseo (e.g. Cocke-Younger-Kasami CYK) – CFGs probabilísticos y ponderados CYK • Representación del significado / Semántica <ul style="list-style-type: none"> – Representación de conocimiento basado en lógica – Roles semánticos – Representaciones temporales – Creencias, deseos e intenciones • Metodos basados en el corpus • N-gramas y Modelos ocultos de Markov (HMMs) • Suavizado y back-off • Ejemplos de uso: POS etiquetado y morfología • Recuperación de la información: <ul style="list-style-type: none"> – Modelo de espacio vectorial <ul style="list-style-type: none"> * TF & IDF – Precision y cobertura • Extracción de información • Traducción de lenguaje • Clasificación y categorización de texto: <ul style="list-style-type: none"> – Modelo de bolsa de palabras 	<ul style="list-style-type: none"> • Define y contrasta gramáticas de tipo estocásticas y determinísticas, dando ejemplos y demostrando como adecuar cada una de ellas [Usar] • Simula, aplica, o implementa algoritmos clásicos y estocásticos para el parseo de un lenguaje natural [Usar] • Identifica los retos de la representación del significado [Usar] • Lista las ventajas de usar corpus estándares. Identifica ejemplos de corpus actuales para una variedad de tareas de PLN [Usar] • Identifica técnicas para la recuperación de la información, traducción de lenguajes, y clasificación de textos [Usar]
Lecturas: Nilsson (2001), Russell and Norvig (2003), Ponce-Gallegos et al. (2014)	

UNIDAD 9: Visión y percepción por computador (12)	
Competencias:	
Contenido	Objetivos Generales
<ul style="list-style-type: none"> • Visión Computacional <ul style="list-style-type: none"> – Adquisición de imágenes, representación, procesamiento y propiedades – Representación de formas, reconocimiento y segmentación de objetos – Análisis de movimiento • Modularidad en reconocimiento. • Enfoques de reconocimiento de patrones <ul style="list-style-type: none"> – Algoritmos de clasificación y medidas de calidad de la clasificación. – Técnicas estadísticas. 	<ul style="list-style-type: none"> • Resumir la importancia del reconocimiento de imágenes y objetos en Inteligencia Artificial (AI) e indicar varias aplicaciones significativas de esta tecnología [Usar] • Listar al menos tres aproximaciones de segmentación de imágenes, tales como algoritmos de límites (thresholding), basado en el borde y basado en regiones, junto con sus características definitorias, fortalezas y debilidades [Usar] • Implementar reconocimiento de objetos en 2d basados en la representación del contorno y/o regiones basadas en formas [Usar] • Proporcionar al menos dos ejemplos de transformación de una fuente de datos de un dominio sensorial a otro, ejemplo, datos táctiles interpretados como imágenes en 2d de una sola banda [Usar] • Implementar un algoritmo para la extracción de características en información real, ejemplo, un detector de bordes o esquinas para imágenes o vectores de coeficientes de Fourier describiendo una pequeña porción de señal de audio [Usar] • Implementar un algoritmo de clasificación que segmenta percepciones de entrada en categorías de salida y evalúa cuantitativamente la clasificación resultante [Usar] • Evaluar el desempeño de la función de extracción subyacente, en relación con al menos una aproximación alternativa posible (ya sea implementado o no) en su contribución a la tarea de clasificación (8) anterior [Usar]
Lecturas: Nilsson (2001), Russell and Norvig (2003), Ponce-Gallegos et al. (2014)	

<p>8. Metodología</p> <p>El profesor del curso presentará clases teóricas de los temas señalados en el programa propiciando la intervención de los alumnos.</p> <p>El profesor del curso presentará demostraciones para fundamentar clases teóricas.</p> <p>El profesor y los alumnos realizarán prácticas</p> <p>Los alumnos deberán asistir a clase habiendo leído lo que el profesor va a presentar. De esta manera se facilitará la comprensión y los estudiantes estarán en mejores condiciones de hacer consultas en clase.</p>

9. Evaluar

Evaluación Continua 1 : 20 %

Examen parcial : 30 %

Evaluación Continua 2 : 20 %

Examen final : 30 %

References

- De Castro, L.N. (2006). *Fundamentals of natural computing: basic concepts, algorithms, and applications*. CRC Press.
- Goldberg, David (1989). *Genetic Algorithms in Search, Optimization and Machine Learning*. Addison Wesley.
- Koller, Daphne and Nir Friedman (2009). *Probabilistic Graphical Models: Principles and Techniques - Adaptive Computation and Machine Learning*. The MIT Press. ISBN: 0262013193.
- Mitchell, M. (1998). *An introduction to genetic algorithms*. The MIT press.
- Murphy, Kevin P. (2012). *Machine Learning: A Probabilistic Perspective*. The MIT Press. ISBN: 0262018020.
- Nilsson, Nils (2001). *Inteligencia Artificial: Una nueva visión*. McGraw-Hill.
- Ponce-Gallegos, Julio et al. (2014). *Inteligencia Artificial*. Iniciativa Latinoamericana de Libros de Texto Abiertos (LATIn).
- Russell, Stuart and Peter Norvig (2003). *Inteligencia Artificial: Un enfoque moderno*. Prentice Hall.



**Universidad Nacional de Colombia (UNAL) Sede
Manizales
Programa Profesional de
Administración de Sistemas Informáticos
SILABO**

CS2H1. Interacción Humano Computador (Obligatorio)

2022-II

1. Información general	
1.1 Escuela	: Sistemas de Información
1.2 Curso	: CS2H1. Interacción Humano Computador
1.3 Semestre	: 7 ^{mo} Semestre.
1.4 Prerrequisitos	: CS393. Sistemas de Infomación. (6 ^{to} Sem)
1.5 Condición	: Obligatorio
1.6 Modalidad de aprendizaje	: Presencial
1.7 horas	: 1 HT; 4 HL;
1.8 Créditos	: 3

2. Profesores

3. Fundamentación del curso
<p>El lenguaje ha sido una de las creaciones más significativas de la humanidad. Desde el lenguaje corporal y gestual, pasando por la comunicación verbal y escrita, hasta códigos simbólicos icónicos y otros, ha posibilitado interacciones complejas entre los seres humanos y facilitado considerablemente la comunicación de información. Con la invención de dispositivos automáticos y semiautomáticos, entre los que se cuentan las computadoras, la necesidad de lenguajes o interfaces para poder interactuar con ellos, ha cobrado gran importancia.</p> <p>La usabilidad del software, aunada a la satisfacción del usuario y su incremento de productividad, depende de la eficacia de la Interfaz Usuario-Computador. Tanto es así, que a menudo la interfaz es el factor más importante en el éxito o el fracaso de cualquier sistema computacional. El diseño e implementación de adecuadas Interfaces Humano-Computador, que además de cumplir los requisitos técnicos y la lógica transaccional de la aplicación, considere las sutiles implicaciones psicológicas, culturales y estéticas de los usuarios, consume buena parte del ciclo de vida de un proyecto software, y requiere habilidades especializadas, tanto para la construcción de las mismas, como para la realización de pruebas de usabilidad.</p>

4. Resumen
1. Fundamentos 2. Factores Humanos 3. Diseño y Testing centrados en el usuario 4. Diseño de Interacción 5. Nuevas Tecnologías Interactivas 6. Colaboración y Comunicación

5. Objetivos Generales
<ul style="list-style-type: none"> • Conocer y aplicar criterios de usabilidad y accesibilidad al diseño y construcción de interfaces humano-computador, buscando siempre que la tecnología se adapte a las personas y no las personas a la tecnología. • Que el alumno tenga una visión centrada en la experiencia de usuario al aplicar apropiados enfoques conceptuales y tecnológicos. • Entender como la tecnologica emergente hace posible nuevos estilos de interacción. • Determinar los requerimientos básicos a nivel de interfaces, hardware y software para la construcción de ambientes inmersivos.

6. Contribución a los resultados (*Outcomes*)

Esta disciplina contribuye al logro de los siguientes resultados de la carrera:

- 1) Analizar un problema computacional complejo y aplicar los principios computacionales y otras disciplinas relevantes para identificar soluciones. (**Familiarizarse**)
- 2) Diseñar, implementar y evaluar una solución basada en computación para cumplir con un conjunto determinado de requisitos computacionales en el contexto de las disciplinas del programa. (**Evaluar**)
- 3) Comunicarse efectivamente en diversos contextos profesionales. (**Usar**)
- 4) Reconocer las responsabilidades profesionales y hacer juicios informados en el campo profesional de computación con principios éticos. (**Familiarizarse**)
- 5) Funcionar efectivamente como miembro o líder de un equipo involucrado en actividades apropiadas a la disciplina del programa. (**Usar**)
- 7) Desarrollar tecnología computacional buscando el bien común, aportando con formación humana, capacidades científicas, tecnológicas y profesionales para solucionar problemas sociales de nuestro entorno. (**Familiarizarse**)

7. Contenido

UNIDAD 1: Fundamentos (8)

Competencias:

Contenido	Objetivos Generales
<ul style="list-style-type: none">• Contextos para IHC (cualquiera relacionado con una interfaz de usuario, p.e., página web, aplicaciones de negocios, aplicaciones móviles y juegos)• Heurística de usabilidad y los principios de pruebas de usabilidad.• Procesos para desarrollo centrado en usuarios, p.e., enfoque inicial en usuarios, pruebas empíricas, diseño iterativo.• Principios del buen diseño y buenos diseñadores; ventajas y desventajas de ingeniería.• Diferentes medidas para evaluación, p.e., utilidad, eficiencia, facilidad de aprendizaje, satisfacción de usuario.	<ul style="list-style-type: none">• Discutir por qué el desarrollo de software centrado en el hombre es importante [Familiarizarse]• Define un proceso de diseño centrado en el usuario que de forma explícita considere el hecho que un usuario no es como un desarrollador o como sus conocimientos [Familiarizarse]• Resumir los preceptos básicos de la interacción psicológica y social [Familiarizarse]• Desarrollar y usar un vocabulario conceptual para analizar la interacción humana con el software: disponibilidad, modelo conceptual, retroalimentación, y demás [Familiarizarse]
Lecturas: Dix et al. (2004), Stone et al. (2005), Rogers and Sharp (2011)	

UNIDAD 2: Factores Humanos (8)**Competencias:****Contenido**

- Modelos cognoscitivos que informan diseño de interacciones, p.e., atención, percepción y reconocimiento, movimiento, memoria, golfos de expectativa y ejecución.
- Capacidades físicas que informan diseño de interacción, p.e. percepción del color, ergonomía.
- Accesibilidad, p.e., interfaces para poblaciones con diferentes habilidades (p.e., invidentes, discapacitados)
- Interfaces para grupos de población de diferentes edades (p.e., niños, mayores de 80)

Objetivos Generales

- Crear y dirigir una simple prueba de usabilidad para una aplicación existente de software [Familiarizarse]

Lecturas: Dix et al. (2004), Stone et al. (2005), Rogers and Sharp (2011), Mathis (2011), Norman (2004)

UNIDAD 3: Diseño y Testing centrados en el usuario (16)**Competencias:****Contenido****Objetivos Generales**

- Enfoque y características del proceso de diseño.
- Requerimientos de funcionalidad y usabilidad.
- Técnicas de recolección de requerimientos, ej. entrevistas, encuestas, etnografía e investigación contextual.
- Técnicas y herramientas para el análisis y presentación de requerimientos ej. reportes, personas.
- Análisis de tareas, incluidos los aspectos cualitativos de la generación de modelos de análisis de tareas.
- Consideración de IHC como una disciplina de diseño:
 - Sketching
 - Diseño participativo
 - Sketching
 - Diseño participativo
- Técnicas de creación de prototipos y herramientas, ej. bosquejos, *storyboards*, prototipos de baja fidelidad, esquemas de página.
- Prototipos de baja fidelidad (papel)
- Técnicas de evaluación cuantitativa ej. evaluación Keystroke-level.
- Evaluación sin usuarios, usando ambas técnicas cualitativas y cuantitativas. Ej. Revisión estructurada, GOMS, análisis basado en expertos, heurísticas, lineamientos y estándar.
- Evaluación con usuarios. Ej. Observación, Método de pensamiento en voz alta, entrevistas, encuestas, experimentación.
- Desafíos para la evaluación efectiva, por ejemplo, toma de muestras, la generalización.
- Reportar los resultados de las evaluaciones.
- Internacionalización, diseño para usuarios de otras culturas, intercultural.

- Llevar a cabo una evaluación cuantitativa y discutir / informar sobre los resultados [Familiarizarse]
- Para un grupo de usuarios determinado, realizar y documentar un análisis de sus necesidades [Familiarizarse]
- Discutir al menos un standard nacional o internacional de diseño de interfaz de usuario [Familiarizarse]
- Explicar cómo el diseño centrado en el usuario complementa a otros modelos de proceso software [Familiarizarse]
- Utilizar *lo-fi* (baja fidelidad) técnicas de prototipado para recopilar y reportar, las respuestas del usuario [Usar]
- Elegir los métodos adecuados para apoyar el desarrollo de una específica interfaz de usuario [Evaluar]
- Utilizar una variedad de técnicas para evaluar una interfaz de usuario dada [Evaluar]
- Comparar las limitaciones y beneficios de los diferentes métodos de evaluación [Evaluar]

Lecturas: Dix et al. (2004), Stone et al. (2005), Rogers and Sharp (2011), Mathis (2011), Buxton (2007)

UNIDAD 4: Diseño de Interacción (8)**Competencias:****Contenido****Objetivos Generales**

- Principios de interfaces gráficas de usuario (GUIs)
- Elementos de diseño visual (disposición, color, fuentes, etiquetado)
- Manejo de fallas humanas/sistema.
- Estándares de interfaz de usuario.
- Presentación de información: navegación, representación, manipulación.
- Técnicas de animación de interfaz (ej. grafo de escena)
- Clases Widget y bibliotecas.
- Internacionalización, diseño para usuarios de otras culturas, intercultural.
- Elección de estilos de interacción y técnicas de interacción.

- Crear una aplicación simple, junto con la ayuda y la documentación, que soporta una interfaz gráfica de usuario [Usar]

Lecturas: Dix et al. (2004), Stone et al. (2005), Rogers and Sharp (2011), Johnson (2010), Mathis (2011), Leavitt and Shneiderman (2006)

UNIDAD 5: Nuevas Tecnologías Interactivas (8)	
Competencias:	
Contenido	Objetivos Generales
<ul style="list-style-type: none"> • Elección de estilos de interacción y técnicas de interacción. • Enfoques para el diseño, implementación y evaluación de la interacción sin mouse <ul style="list-style-type: none"> – Interfaces táctiles y multitáctiles. – Interfaces compartidas, incorporadas y grandes – Nuevas modalidades de entrada (tales como datos de sensores y localización) – Nuevas ventanas, por ejemplo, iPhone, Android – Reconocimiento de voz y procesamiento del lenguaje natural – Interfaces utilizables y tangibles – Interacción persuasiva y emoción – Tecnologías de interacción ubicuas y contextuales (UbiComp) – Inferencia bayesiana (por ejemplo, texto predictivo, orientación guiada) – Visualización e interacción de ambiente / periféricos • Salida: <ul style="list-style-type: none"> – Sonido – Visualización estereoscópica – Forzar la simulación de retroalimentación, dispositivos hápticos • Arquitectura de Sistemas: <ul style="list-style-type: none"> – Motores de Juego – Realidad Aumentada móvil – Simuladores de vuelo – CAVEs – Imágenes médicas 	<ul style="list-style-type: none"> • Describe cuando son adecuadas las interfaces sin uso de ratón [Familiarizarse] • Comprende las posibilidades de interacción que van más allá de las interfaces de ratón y puntero [Familiarizarse] • Discute las ventajas (y desventajas) de las interfaces no basadas en ratón [Usar] • Describir el modelo óptico realizado por un sistema de gráficos por computadora para sintetizar una visión estereoscópica [Familiarizarse] • Describir los principios de las diferentes tecnologías de seguimiento de espectador [Familiarizarse] • Determinar los requerimientos básicos en interfaz, software, hardware, y configuraciones de software de un sistema VR para una aplicación específica [Evaluar]
Lecturas: Dix et al. (2004), Stone et al. (2005), Rogers and Sharp (2011), Wigdor and Wixon (2011), Mathis (2011)	

UNIDAD 6: Colaboración y Comunicación (8)	
Competencias:	
Contenido	Objetivos Generales
<ul style="list-style-type: none"> • La comunicación asíncrona en grupo, por ejemplo, el correo electrónico, foros, redes sociales. • Medios de comunicación social, informática social, y el análisis de redes sociales. • Colaboración en línea, espacios "inteligentes" y aspectos de coordinación social de tecnologías de flujo de trabajo. • Comunidades en línea. • Personajes de Software y agentes inteligentes, mundos virtuales y avatares. • Psicología Social 	<ul style="list-style-type: none"> • Describir la diferencia entre la comunicación sincrónica y asíncrona [Familiarizarse] • Comparar los problemas de IHC en la interacción individual con la interacción del grupo [Familiarizarse] • Discuta varias problemas de interés social planteados por el software colaborativo [Usar] • Discutir los problemas de IHC en software que personifica la intención humana [Evaluar]
Lecturas: Dix et al. (2004), Stone et al. (2005), Rogers and Sharp (2011)	

8. Metodología
<p>El profesor del curso presentará clases teóricas de los temas señalados en el programa propiciando la intervención de los alumnos.</p> <p>El profesor del curso presentará demostraciones para fundamentar clases teóricas.</p> <p>El profesor y los alumnos realizarán prácticas</p> <p>Los alumnos deberán asistir a clase habiendo leído lo que el profesor va a presentar. De esta manera se facilitará la comprensión y los estudiantes estarán en mejores condiciones de hacer consultas en clase.</p>

9. Evaluar
<p>Evaluación Continua 1 : 20 %</p> <p>Examen parcial : 30 %</p> <p>Evaluación Continua 2 : 20 %</p> <p>Examen final : 30 %</p>

References

- Buxton, Bill (2007). *Sketching User Experiences: Getting the Design Right and the Right Design*. Morgan Kaufmann Publishers Inc.
- Dix, Alan et al. (2004). *Human-computer Interaction*. 3 ed. Prentice-Hall, Inc.
- Johnson, Jeff (2010). *Designing with the Mind in Mind: Simple Guide to Understanding User Interface Design Rules*. 3 ed. Morgan Kaufmann Publishers Inc.
- Leavitt, M. and B. Shneiderman (2006). *Research-Based Web Design & Usability Guidelines*. Health and Human Services Dept.
- Mathis, Lukas (2011). *Designed for Use: Create Usable Interfaces for Applications and the Web*. Pragmatic Bookshelf.
- Norman, Donald A. (2004). *Emotional Design: Why We Love (or Hate) Everyday Things*. Basic Book.
- Rogers, Y. and J Sharp H. & Preece (2011). *Interaction Design: Beyond Human-Computer Interaction*. 3 ed. John Wiley and Sons Ltd.
- Stone, D. et al. (2005). *User Interface Design and Evaluation*. Morgan Kaufmann Series in Interactive Technologies.
- Wigdor, D. and D. Wixon (2011). *Brave NUI World: Designing Natural User Interfaces for Touch and Gesture*. Morgan Kaufmann Publishers Inc.



Universidad Nacional de Colombia (UNAL) Sede
Manizales
Programa Profesional de
Administración de Sistemas Informáticos
SILABO

CS391. Ingeniería de Software III (Obligatorio)

2022-II

1. Información general	
1.1 Escuela	: Sistemas de Información
1.2 Curso	: CS391. Ingeniería de Software III
1.3 Semestre	: 7 ^{mo} Semestre.
1.4 Prerrequisitos	: CS292. Ingeniería de Software II. (6 ^{to} Sem)
1.5 Condición	: Obligatorio
1.6 Modalidad de aprendizaje	: Presencial
1.7 horas	: 2 HT; 2 HP;
1.8 Créditos	: 3
2. Profesores	
3. Fundamentación del curso	
El desarrollo de software requiere del uso de mejores prácticas de desarrollo, gestión de proyectos de TI, manejo de equipos y uso eficiente y racional de frameworks de aseguramiento de la calidad, estos elemento son pieza clave y transversal durante todo el proceso productivo. La construcción de software contempla la implementación y uso de procesos, métodos, modelos y herramientas que permitan lograr la realización de los atributos de calidad de un producto.	
4. Resumen	
1. Evolución de Software 2. Gestión de Proyectos de Software 3. Gestión de Proyectos de Software 4. Procesos de Software 5. Estándares ISO/IEC	
5. Objetivos Generales	
<ul style="list-style-type: none">• Comprender y poner en práctica los conceptos fundamentales sobre la gestión de proyectos y manejo de equipos de software.• Comprender los fundamentos de la gestión de proyectos, incluyendo su definición, alcance, y la necesidad de gestión de proyectos en la organización moderna.• Los alumnos deben comprender los conceptos fundamentales de CMMI, PSP, TSP para que sean adoptados en los proyectos de software.• Describir y comprender los modelos de aseguramiento de la calidad como marco clave para el éxitos de los proyectos de TI.	

6. Contribución a los resultados (*Outcomes*)

Esta disciplina contribuye al logro de los siguientes resultados de la carrera:

- 1) Analizar un problema computacional complejo y aplicar los principios computacionales y otras disciplinas relevantes para identificar soluciones. (**E**valuar)
- 2) Diseñar, implementar y evaluar una solución basada en computación para cumplir con un conjunto determinado de requisitos computacionales en el contexto de las disciplinas del programa. (**E**valuar)
- 3) Comunicarse efectivamente en diversos contextos profesionales. (**U**sar)
- 5) Funcionar efectivamente como miembro o líder de un equipo involucrado en actividades apropiadas a la disciplina del programa. (**U**sar)
- 6) Aplicar fundamentos de teoría de ciencias de la computación y desarrollo de software para producir soluciones basados en computación. (**E**valuar)
- 7) Desarrollar tecnología computacional buscando el bien común, aportando con formación humana, capacidades científicas, tecnológicas y profesionales para solucionar problemas sociales de nuestro entorno. (**E**valuar)

7. Contenido

UNIDAD 1: Evolución de Software (12)

Competencias:

Contenido

Objetivos Generales

- Desarrollo de Software en el contexto de código grande pre existente
 - Cambios de software
 - Preocupaciones y ubicación de preocupaciones
 - *Refactoring*
- Evolución de Software.
- Características de Software mantenible.
- Sistemas de Reingeniería.
- Reuso de Software.
 - Segmentos de código
 - Bibliotecas y *frameworks*
 - Componentes
 - Líneas de Producto

- Identificar los problemas principales asociados con la evolución del software y explicar su impacto en el ciclo de vida del software [Familiarizarse]
- Estimar el impacto del cambio de requerimientos en productos existentes de tamaño medio [Usar]
- Usar refactorización en el proceso de modificación de un componente de software [Usar]
- Estudiar los desafíos de mejorar sistemas en un entorno cambiante [Familiarizarse]
- Perfilar los procesos de pruebas de regresión y su rol en el manejo de versiones [Familiarizarse]
- Estudiar las ventajas y desventajas de diferentes tipos de niveles de confiabilidad [Familiarizarse]

Lecturas: Pressman and Maxim (2015), Sommerville (2017)

UNIDAD 2: Gestión de Proyectos de Software (10)	
Competencias:	
Contenido	Objetivos Generales
<ul style="list-style-type: none"> • La participación del equipo: <ul style="list-style-type: none"> – Procesos elemento del equipo, incluyendo responsabilidades de tarea, la estructura de reuniones y horario de trabajo – Roles y responsabilidades en un equipo de software – Equipo de resolución de conflictos – Los riesgos asociados con los equipos virtuales (comunicación, la percepción, la estructura) • Estimación de esfuerzo (a nivel personal) • Riesgo. <ul style="list-style-type: none"> – El papel del riesgo en el ciclo de vida – Categorías elemento de riesgo, incluyendo la seguridad, la seguridad, mercado, finanzas, tecnología, las personas, la calidad, la estructura y el proceso de • Gestión de equipos: <ul style="list-style-type: none"> – Organización de equipo y la toma de decisiones – Roles de identificación y asignación – Individual y el desempeño del equipo de evaluación • Gestión de proyectos: <ul style="list-style-type: none"> – Programación y seguimiento de elementos – Herramientas de gestión de proyectos – Análisis de Costo/Beneficio 	<ul style="list-style-type: none"> • Discutir los comportamientos comunes que contribuyen al buen funcionamiento de un equipo [Familiarizarse] • Crear y seguir un programa para una reunión del equipo [Usar] • Identificar y justificar las funciones necesarias en un equipo de desarrollo de software [Usar] • Entender las fuentes, obstáculos y beneficios potenciales de un conflicto de equipo [Usar] • Aplicar una estrategia de resolución de conflictos en un ambiente de equipo [Usar] • Utilizar un método ad hoc para estimar el esfuerzo de desarrollo del software (ejemplo, tiempo) y comparar con el esfuerzo actual requerido [Usar] • Listar varios ejemplos de los riesgos del software [Familiarizarse] • Describir el impacto del riesgo en el ciclo de vida de desarrollo de software [Familiarizarse] • Describir las diferentes categorías de riesgo en los sistemas de software [Familiarizarse] • Demostrar a través de la colaboración de proyectos de equipo los elementos centrales de la construcción de equipos y gestión de equipos [Usar] • Describir como la elección de modelos de procesos afectan la estructura organizacional de equipos y procesos de toma de decisiones [Familiarizarse] • Crear un equipo mediante la identificación de los roles apropiados y la asignación de funciones a los miembros del equipo [Usar] • Evaluar y retroalimentar a los equipos e individuos sobre su desempeño en un ambiente de equipo [Usar] • Usando un software particular procesar, describir los aspectos de un proyecto que necesita ser planeado y monitoreado, (ejemplo, estimar el tamaño y esfuerzo, un horario, reasignación de recursos, control de configuración, gestión de cambios, identificación de riesgos en un proyecto y gestión) [Familiarizarse]
Lecturas: Pressman and Maxim (2015), Sommerville (2017)	

UNIDAD 3: Gestión de Proyectos de Software (8)	
Competencias:	
Contenido	Objetivos Generales
<ul style="list-style-type: none"> • Software de medición y técnicas de estimación. • Aseguramiento de la calidad del software y el rol de las mediciones. • Riesgo. <ul style="list-style-type: none"> – Identificación de riesgos y gestión. – Análisis riesgo y evaluación. – La tolerancia al riesgo (por ejemplo, riesgo adverso, riesgo neutral, la búsqueda de riesgo) – Planificación de Riesgo • En todo el sistema de aproximación al riesgo, incluyendo riesgos asociados con herramientas. 	<ul style="list-style-type: none"> • Realizar el seguimiento del progreso de alguna etapa de un proyecto que utiliza métricas de proyectos apropiados [Usar] • Comparar las técnicas simples de tamaño de software y estimación de costos [Usar] • Usar una herramienta de gestión de proyectos para ayudar en la asignación y rastreo de tareas en un proyecto de desarrollo de software [Usar] • Describir el impacto de la tolerancia de riesgos en el proceso de desarrollo de software [Evaluar] • Identificar riesgos y describir enfoques para manejar riesgos (evitar, aceptar, transferir, mitigar) y caracterizar fortalezas y defectos para cada uno [Familiarizarse] • Explicar cómo el riesgo afecta las decisiones en el proceso de desarrollo de software [Usar] • Identificar los riesgos de seguridad para un sistema de software [Usar] • Demostrar un enfoque sistemático para la tarea de identificar los peligros y riesgos en una situación particular [Usar] • Aplicar los principios básicos del manejo de riesgos en una variedad de escenarios simples incluyendo una situación de seguridad [Usar] • Dirigir un análisis de costo/beneficio para el enfoque de mitigación de riesgos [Usar] • Identificar y analizar alguno de los riesgos para un sistema entero que surgen de aspectos distintos del software [Usar]
Lecturas: Pressman and Maxim (2015), Sommerville (2017)	

UNIDAD 4: Procesos de Software (12)	
Competencias:	
Contenido	Objetivos Generales
<ul style="list-style-type: none"> • Consideraciones a nivel de sistemas, ejem., la interacción del software con su entorno. • Introducción a modelos del proceso de software (e.g., cascada, incremental, ágil): <ul style="list-style-type: none"> – Actividades con ciclos de vida de software. • Programación a gran escala versus programación individual. • Evaluación de modelos de proceso de software. • Conceptos de calidad de software. • Mejoramiento de procesos. • Modelos de madurez de procesos de software. • Mediciones del proceso de software. 	<ul style="list-style-type: none"> • Describa cómo el software puede interactuar y participar en varios sistemas, incluyendo la gestión de información, integración, control de procesos y sistemas de comunicaciones [Usar] • Describir las ventajas y desventajas relativas entre varios modelos importantes de procesos (por ejemplo, la cascada, iterativo y ágil) [Usar] • Describir las diferentes prácticas que son componentes clave de los diversos modelos de procesos [Usar] • Diferenciar entre las fases de desarrollo de software [Usar] • Describir cómo la programación en grandes equipos difiere de esfuerzos individuales con respecto a la comprensión de una gran base de código, lectura de código, comprensión de las construcciones, y comprensión de contexto de cambios [Usar] • Explicar el concepto de ciclo de vida del software y proporcionar un ejemplo que ilustra sus fases incluyendo los entregables que se producen [Usar] • Comparar varios modelos comunes de procesos con respecto a su valor para el desarrollo de las clases particulares de sistemas de software, teniendo en cuenta diferentes aspectos tales como, estabilidad de los requisitos, tamaño y características no funcionales [Usar] • Definir la calidad del software y describir el papel de las actividades de aseguramiento de la calidad en el proceso de software [Usar] • Describir el objetivo y similitudes fundamentales entre los enfoques de mejora de procesos [Usar] • Comparar varios modelos de mejora de procesos, tales como CMM, CMMI, CQI, <i>Plan-Do-Check-Act</i>, o ISO9000 [Usar] • Evaluar un esfuerzo de desarrollo y recomendar cambios potenciales al participar en la mejora de procesos (usando un modelo como PSP) o involucración en una retrospectiva de un proyecto [Usar] • Explicar el papel de los modelos de madurez de procesos en la mejora de procesos [Usar] • Describir varias métricas de procesos para la evaluación y el control de un proyecto [Usar] • Usar las medidas en proyecto para describir el estado actual de un proyecto [Usar]
Lecturas: Pressman and Maxim (2015), Sommerville (2017)	

UNIDAD 5: Estándares ISO/IEC (6)	
Competencias:	
Contenido	Objetivos Generales
<ul style="list-style-type: none"> • ISO 9001:2001. • ISO 9000-3. • ISO/IEC 9126. • ISO/IEC 12207. • ISO/IEC 15939. • ISO/IEC 14598. • ISO/IEC 15504-SPICE. • IT Mark. • SCRUM. • SQuaRE. • CISQ. 	<ul style="list-style-type: none"> • Aprender y aplicar correctamente normas y estándares internacionales. [Usar]
Lecturas: Sommerville (2017), Pressman and Maxim (2015)	

8. Metodología
<p>El profesor del curso presentará clases teóricas de los temas señalados en el programa propiciando la intervención de los alumnos.</p> <p>El profesor del curso presentará demostraciones para fundamentar clases teóricas.</p> <p>El profesor y los alumnos realizarán prácticas</p> <p>Los alumnos deberán asistir a clase habiendo leído lo que el profesor va a presentar. De esta manera se facilitará la comprensión y los estudiantes estarán en mejores condiciones de hacer consultas en clase.</p>

9. Evaluar
<p>Evaluación Continua 1 : 20 %</p> <p>Examen parcial : 30 %</p> <p>Evaluación Continua 2 : 20 %</p> <p>Examen final : 30 %</p>

References

Pressman, Roger S. and Bruce Maxim (Jan. 2015). *Software Engineering: A Practitioner's Approach*. 8th. McGraw-Hill.
 Sommerville, Ian (Mar. 2017). *Software Engineering*. 10th. Pearson.



Universidad Nacional de Colombia (UNAL) Sede
Manizales
Programa Profesional de
Administración de Sistemas Informáticos
SILABO

CS401. Metodología de la Investigación en
Computación (Obligatorio)

2022-II

1. Información general	
1.1 Escuela	: Sistemas de Información
1.2 Curso	: CS401. Metodología de la Investigación en Computación
1.3 Semestre	: 7 ^{mo} Semestre.
1.4 Prerrequisitos	: CS212. Análisis y Diseño de Algoritmos. (5 ^{to} Sem)
1.5 Condición	: Obligatorio
1.6 Modalidad de aprendizaje	: Presencial
1.7 horas	: 2 HT; 2 HP;
1.8 Créditos	: 3
2. Profesores	
3. Fundamentación del curso	
Este curso tiene por objetivo que el alumno aprenda a realizar una investigación de carácter científico en el área de computación. Los docentes del curso determinarán un área de estudio para cada alumno, y se le hará entrega de bibliografía para analizar y a partir de la misma, y de fuentes bibliográficas adicionales (investigadas por el alumno), el alumno deberá ser capaz de construir un artículo del tipo survey del tema asignado.	
4. Resumen	
1. Iniciación científica en el área de computación	
5. Objetivos Generales	
<ul style="list-style-type: none">• Que el alumno aprenda como se inicia una investigación científica en el área de computación.• Que el alumno conozca las principales fuentes para obtener bibliografía relevante para trabajos de investigación en el área de computación: Researchindex, IEEE-CS¹, ACM².• Que el alumno sea capaz de analizar las propuestas existentes sobre un determinado tópico y relacionarlos de forma coherente en una revisión bibliográfica.• Que el alumno pueda redactar documentos técnicos en computación utilizando L^AT_EX.• Que el alumno sea capaz de <u>reproducir</u> los resultados ya existentes en un determinado tópico a través de la experimentación.• Los entregables de este curso son: Avance parcial: Dominio del tema del artículo y bibliografía preliminar en formato de artículo L^AT_EX. Final: Entendimiento del artículo del tipo survey, documento concluido donde se contenga, opcionalmente, los resultados experimentales de la(s) técnica(s) estudiada(s).	

¹<http://www.computer.org>

²<http://www.acm.org>

6. Contribución a los resultados (*Outcomes*)

Esta disciplina contribuye al logro de los siguientes resultados de la carrera:

Nooutcomes

7. Contenido

UNIDAD 1: Iniciación científica en el área de computación (60)

Competencias:

Contenido	Objetivos Generales
<ul style="list-style-type: none">• Búsqueda bibliográfica en computación.• Redacción de artículos técnicos en computación.	<ul style="list-style-type: none">• Aprender a hacer una investigación correcta en el área de computación[Usar]• Conocer las fuentes de bibliografía adecuada para esta área[Usar]• Saber redactar un documento de acorde con las características que las conferencias de esta área exigen[Usar]

Lecturas: IEEE-Computer Society (2008), Association for Computing Machinery (2008), CiteSeer.IST (2008)

8. Metodología

El profesor del curso presentará clases teóricas de los temas señalados en el programa propiciando la intervención de los alumnos.

El profesor del curso presentará demostraciones para fundamentar clases teóricas.

El profesor y los alumnos realizarán prácticas

Los alumnos deberán asistir a clase habiendo leído lo que el profesor va a presentar. De esta manera se facilitará la comprensión y los estudiantes estarán en mejores condiciones de hacer consultas en clase.

9. Evaluar

Evaluación Continua 1 : 20 %

Examen parcial : 30 %

Evaluación Continua 2 : 20 %

Examen final : 30 %

References

Association for Computing Machinery (2008). *Digital Libray*. <http://portal.acm.org/dl.cfm>. Association for Computing Machinery.

CiteSeer.IST (2008). *Scientific Literature Digital Libray*. <http://citeseer.ist.psu.edu>. College of Information Sciences and Technology, Penn State University.

IEEE-Computer Society (2008). *Digital Libray*. <http://www.computer.org/publications/dlib>. IEEE-Computer Society.



Universidad Nacional de Colombia (UNAL) Sede
Manizales
Programa Profesional de
Administración de Sistemas Informáticos
SILABO

CS251. Computación Gráfica (Electivo)

2022-II

1. Información general	
1.1 Escuela	: Sistemas de Información
1.2 Curso	: CS251. Computación Gráfica
1.3 Semestre	: 7 ^{mo} Semestre.
1.4 Prerrequisitos	: <ul style="list-style-type: none">• CS312. Estructuras de Datos Avanzadas. (6^{to} Sem)• MA307. Matemática aplicada a la computación. (6^{to} Sem)
1.5 Condición	: Electivo
1.6 Modalidad de aprendizaje	: Presencial
1.7 horas	: 2 HT; 2 HP; 2 HL;
1.8 Créditos	: 4
2. Profesores	
3. Fundamentación del curso	
Ofrece una introducción para el área de Computación Gráfica, la cual es una parte importante dentro de Ciencias de la Computación. El proposito de este curso es investigar los principios, técnicas y herramientas fundamentales para esta área.	
4. Resumen	
1. Conceptos Fundamentales 2. Rendering Básico 3. Programación de Sistemas Interactivos 4. Modelado Geométrico 5. Renderizado Avanzado 6. Animación por computadora	
5. Objetivos Generales	
<ul style="list-style-type: none">• Acercar al alumno a conceptos y técnicas usados en aplicaciones gráficas 3-D complejas.• Dar al alumno las herramientas necesarias para determinar que software gráfico y que plataforma son los más adecuados para desarrollar una aplicación específica.	
6. Contribución a los resultados (Outcomes)	
Esta disciplina contribuye al logro de los siguientes resultados de la carrera: <ol style="list-style-type: none">1) Analizar un problema computacional complejo y aplicar los principios computacionales y otras disciplinas relevantes para identificar soluciones. (Usar)6) Aplicar fundamentos de teoria de ciencias de la computación y desarrollo de software para producir soluciones basados en computación. (Usar)	
7. Contenido	

UNIDAD 1: Conceptos Fundamentales (6)**Competencias:****Contenido**

- Aplicaciones multimedia, incluyendo interfaces de usuario, edición de audio y vídeo, motores de juego, cad, visualización, realidad virtual.
- Soluciones de compensación entre el almacenamiento de datos y los datos re-computing es personalizado por vectores y raster en representaciones de imágenes.
- Modelos de color sustractivo Aditivo y (CMYK y RGB) y por qué estos proporcionan una gama de colores.
- Animación como una secuencia de imágenes fijas.

Objetivos Generales

- Explicar en términos generales cómo las señales analógicas pueden ser representadas por muestras discretas, por ejemplo, cómo las imágenes pueden ser representadas por píxeles [Familiarizarse]
- Describir modelos de color y su uso en los dispositivos de visualización de gráficos [Familiarizarse]
- Describir las ventajas y desventajas entre el almacenamiento de información vs almacenar suficiente información para reproducir la información, como en la diferencia entre el vector y la representación de la trama [Familiarizarse]
- Describir los procesos básico de la producción de movimiento continuo a partir de una secuencia de cuadros discretos (algunas veces llamado it flicker fusion) [Familiarizarse]

Lecturas: Hearn and Baker (1990)

UNIDAD 2: Rendering Básico (12)**Competencias:**

Contenido	Objetivos Generales
<ul style="list-style-type: none"> • Renderizado en la naturaleza, por ejemplo, la emisión y dispersión de la luz y su relación con la integración numérica. • Renderizado Forward and Backward (i.e., <i>ray-casting</i> y rasterización) • Radiometría básica, triángulos similares y modelos de proyecciones • Afinamiento y Transformaciones de Sistemas de coordenadas • <i>Ray tracing</i> • Visibilidad y oclusión, incluyendo soluciones a este problema, como el almacenamiento en búfer de profundidad, algoritmo del pintor, y el trazado de rayos. • Rasterización triangular simple. • Renderización con una API basada en shader. • Aplicación de la representación de estructuras de datos espaciales. • Muestreo y anti-aliasing. • Renderizado Forward and Backward (i.e., <i>ray-casting</i> y rasterización) 	<ul style="list-style-type: none"> • Discutir el problema de transporte de la luz y su relación con la integración numérica, es decir, se emite luz, dispersa alrededor de la escena, y es medida por el ojo [Familiarizarse] • Describir la tubería básica gráficos y cómo el factor de representación va hacia adelante y atrás en esta [Familiarizarse] • Crear un programa para visualizar modelos 3D de imágenes gráficas simples [Usar] • Obtener puntos en 2-dimensiones y 3-dimensiones por aplicación de transformaciones afín [Usar] • Aplicar sistema de coordenadas de 3-dimensiones y los cambios necesarios para extender las operaciones de transformación 2D para manejar las transformaciones en 3D [Usar] • Contrastar la renderización hacia adelante <i>forward</i> y hacia atrás <i>backward</i> [Evaluar] • Explicar el concepto y las aplicaciones de mapeo de texturas, muestreo y el <i>anti-aliasing</i> [Familiarizarse] • Explicar la dualidad de rastreo de rayos/rasterización para el problema de visibilidad [Familiarizarse] • Implementar un sencillo renderizador en tiempo real utilizando una API de rasterización (por ejemplo, OpenGL) utilizando buffers de vértices y <i>shaders</i> [Usar] • Calcular las necesidades de espacio en base a la resolución y codificación de color [Evaluar] • Calcular los requisitos de tiempo sobre la base de las frecuencias de actualización, técnicas de rasterización [Evaluar]
Lecturas: Marschner and Shirley (2016)	

UNIDAD 3: Programación de Sistemas Interactivos (2)**Competencias:****Contenido****Objetivos Generales**

- Manejo de eventos e interacción de usuario.
- Enfoques para el diseño, implementación y evaluación de la interacción sin mouse
 - Interfaces táctiles y multitáctiles.
 - Interfaces compartidas, incorporadas y grandes
 - Nuevas modalidades de entrada (tales como datos de sensores y localización)
 - Nuevas ventanas, por ejemplo, iPhone, Android
 - Reconocimiento de voz y procesamiento del lenguaje natural
 - Interfaces utilizables y tangibles
 - Interacción persuasiva y emoción
 - Tecnologías de interacción ubicuas y contextuales (UbiComp)
 - Inferencia bayesiana (por ejemplo, texto predictivo, orientación guiada)
 - Visualización e interacción de ambiente / periféricos

- Discute las ventajas (y desventajas) de las interfaces no basadas en ratón [Evaluar]

Lecturas: Marschner and Shirley (2016)

UNIDAD 4: Modelado Geométrico (15)**Competencias:****Contenido****Objetivos Generales**

- Operaciones geométricas básicas como cálculo de intersección y pruebas de proximidad.
- Volúmenes, voxels y representaciones basadas en puntos.
- Curvas polinomiales y Superficies paramétricas.
- Representación implícita de curvas y superficies.
- Técnicas de aproximación, tales como curvas polinómicas, curvas Bezier, curvas spline y superficies, y base racional no uniforme (NURB) espinas, y el método de ajuste de nivel.
- Técnicas de superficie de representación incluyendo teselación, la representación de malla, carenado malla, y las técnicas de generación de mallas, como la triangulación de Delaunay, marchando cubos.
- Técnicas de subdivisión espacial.
- Modelos procedimentales como fractales, modelamiento generativo y sistemas L.
- Modelos deformables de forma libre y elásticamente deformables.
- Subdivisión de superficies.
- Modelado multiresolución.
- Reconstrucción.
- Representación de Geometría Sólida Constructiva (GSC)

- Representar curvas y superficies utilizando formas tanto implícitas y paramétricas [Usar]
- Crear modelos poliédrico simples por teselación de superficies [Usar]
- Generar una representación de malla de una superficie implícita [Usar]
- Generar una malla de un conjunto de puntos adquiridos por un scanner laser [Usar]
- Construct modelos de geometría sólida constructiva a partir de simples primitivas, tales como cubos y superficies cuádricas [Usar]
- Contrastar métodos de modelización con respecto a espacio y tiempo de complejidad y calidad de imagen [Evaluar]

Lecturas: Marschner and Shirley (2016)

UNIDAD 5: Renderizado Avanzado (6)	
Competencias:	
Contenido	Objetivos Generales
<ul style="list-style-type: none"> • Tiempo (desenfoque de movimiento), la posición del objetivo (enfoque), y la frecuencia continua (color) y su impacto en la representación. • Mapeo de Sombras. • Selectiva de oclusión. • Dispersión de la Superficie. • Renderizado no fotorealístico. • Arquitectura del GPU. • Sistemas visuales humanos incluida la adaptación a la luz, la sensibilidad al ruido, y la fusión de parpadeo. 	<ul style="list-style-type: none"> • Demostrar como un algoritmo calcula una solución a la ecuación de renderización [Evaluar] • Demostrar las propiedades de un algoritmo de renderización, por ejemplo, completo, consistente, e imparcial [Evaluar] • Implementar un algoritmo no trivial de sombreado (por ejemplo, sombreado caricaturizado (<i>toon shading</i>), mapas de sombras en cascada (<i>cascaded shadow maps</i>)) bajo una API de rasterización [Usar] • Discutir como una técnica artística particular puede ser implementada en un renderizador [Familiarizarse] • Explicar como reconocer las técnicas gráficas usadas para crear una imagen en particular [Familiarizarse]
Lecturas: Marschner and Shirley (2016)	

UNIDAD 6: Animación por computadora (4)	
Competencias:	
Contenido	Objetivos Generales
<ul style="list-style-type: none"> • Cinemática directa e inversa. • Detección de colisiones y respuesta. • Animación procedimental empleando ruido, reglas (boids/crowds) y sistemas de partículas. • Algoritmos Skinning. • Movimientos basado en la física, incluyendo la dinámica del cuerpo rígido, sistemas de partículas físicas, redes de masa-muelle de tela y la carne y el pelo. • Animación de Cuadros Principales • Splines • Estructuras de datos para rotaciones, como cuaterniones. • Animación de Cámara. • Captura de Movimiento. 	<ul style="list-style-type: none"> • Calcular la localización y orientación de partes de un modelo usando un enfoque de cinemática hacia delante [Usar] • Implementar el método de interpolación <i>spline</i> para producir las posiciones y orientaciones en medio [Usar] • Implementar algoritmos para el modelamiento físico de partículas dinámicas usando simplemente la mecánica de Newton, por ejemplo Witkin & Kass, serpientes y gusanos, Euler simpléctica, Stormer/Verlet, o métodos de punto medio de Euler [Usar] • Discutir las ideas básicas detrás de algunos métodos para dinámica de fluidos para el modelamiento de trayectorias balísticas, por ejemplo salpicaduras, polvo, fuego, o humo [Familiarizarse] • Usar el software de animación común para construir formas orgánicas simples usando <i>metaball</i> y el esqueleto [Usar]
Lecturas: Marschner and Shirley (2016)	

8. Metodología

El profesor del curso presentará clases teóricas de los temas señalados en el programa propiciando la intervención de los alumnos.

El profesor del curso presentará demostraciones para fundamentar clases teóricas.

El profesor y los alumnos realizarán prácticas

Los alumnos deberán asistir a clase habiendo leído lo que el profesor va a presentar. De esta manera se facilitará la comprensión y los estudiantes estarán en mejores condiciones de hacer consultas en clase.

9. Evaluar

Evaluación Continua 1 : 20 %

Examen parcial : 30 %

Evaluación Continua 2 : 20 %

Examen final : 30 %

References

Hearn, Donald and Pauline Baker (1990). *Computer Graphics in C*. Prentice Hall.

Marschner, Steve and Peter Shirley (2016). *Fundamentals of Computer Graphics*. Fourth Edition. CRC Press. ISBN: ISBN-10: 1482229390.



Universidad Nacional de Colombia (UNAL) Sede
Manizales
Programa Profesional de
Administración de Sistemas Informáticos
SILABO

FG350. Liderazgo (Obligatorio)

2022-II

1. Información general	
1.1 Escuela	: Sistemas de Información
1.2 Curso	: FG350. Liderazgo
1.3 Semestre	: 7 ^{mo} Semestre.
1.4 Prerrequisitos	: FG203. Oratoria. (6 ^{to} Sem)
1.5 Condición	: Obligatorio
1.6 Modalidad de aprendizaje	: Presencial
1.7 horas	: 1 HT; 2 HP;
1.8 Créditos	: 2
2. Profesores	
3. Fundamentación del curso	
En la actualidad las diferentes organizaciones en el mundo exigen a sus integrantes el ejercicio de liderazgo, esto significa asumir los retos asignados con eficacia y afán de servicio, siendo estas exigencias necesarias para la búsqueda de una sociedad más justa y reconciliada. Este desafío, pasa por la necesidad de formar a nuestros alumnos con un recto conocimiento de sí mismos, con capacidad de juzgar objetivamente la realidad y de proponer orientaciones que busquen modificar positivamente el entorno.	
4. Resumen	
1. Primera Unidad: Fundamentos del liderazgo 2. 3.	
5. Objetivos Generales	
<ul style="list-style-type: none">• Desarrollar conocimientos, criterios, capacidades y actitudes para ejercer liderazgo, con el objeto de lograr la eficacia y servicio en los retos asignados, contribuyendo así en la construcción de una mejor sociedad.	
6. Contribución a los resultados (Outcomes)	
Esta disciplina contribuye al logro de los siguientes resultados de la carrera:	
3) Comunicarse efectivamente en diversos contextos profesionales. (Usar)	
4) Reconocer las responsabilidades profesionales y hacer juicios informados en el campo profesional de computación con principios éticos. (Usar)	
5) Funcionar efectivamente como miembro o líder de un equipo involucrado en actividades apropiadas a la disciplina del programa. (Usar)	
7) Desarrollar tecnología computacional buscando el bien común, aportando con formación humana, capacidades científicas, tecnológicas y profesionales para solucionar problemas sociales de nuestro entorno. (Usar)	
7. Contenido	

UNIDAD 1: Primera Unidad: Fundamentos del liderazgo (15)	
Competencias:	
Contenido	Objetivos Generales
<ul style="list-style-type: none"> • Teorías de Liderazgo: • Definición de Liderazgo. • Fundamentos de Liderazgo. • Visión integral del Ser Humano y Motivos de la acción. • La práctica de la Virtud en el ejercicio de Liderazgo. 	<ul style="list-style-type: none"> • Analizar y comprender las bases teóricas del ejercicio de Liderazgo.[Familiarizarse] • En base a lo comprendido, asumir la actitud correcta para llevarlo a la práctica.[Familiarizarse] • Iniciar un proceso de autoconocimiento orientado a descubrir rasgos de liderazgo en sí mismo.[Familiarizarse]
Lecturas: Pilar (2002), Manuel. (2009), Alexandre. (2009), D' Souza (n.d.), Alfred. (2010)	

UNIDAD 2: (15)	
Competencias:	
Contenido	Objetivos Generales
<ul style="list-style-type: none"> • Teoría de las Competencias • Reconocimiento de Competencias • Plan de Desarrollo • Modelos Mentales • Necesidades Emocionales • Perfiles Emocionales • Vicios Motivacionales 	<ul style="list-style-type: none"> • Conocer y Desarrollar competencias de Liderazgo, centradas en lograr la eficacia, sin dejar de lado el deber de servicio con los demás.[Familiarizarse] • Reconocer las tendencias personales y grupales necesarias para el ejercicio de Liderazgo.[Familiarizarse]
Lecturas: Wilkinson. (2009), Luis. (2008), Pilar (2002), Maruja. (2007)	

UNIDAD 3: (18)	
Competencias:	
Contenido	Objetivos Generales
<ul style="list-style-type: none"> • La relación personal con el equipo • Liderazgo integral • Acompañamiento y discipulado • Fundamentos de unidad 	<ul style="list-style-type: none"> • Desarrollar habilidades para el trabajo en equipo[Familiarizarse]
Lecturas: Goleman (2012), CardonaP, Hersey, Hunsaker (2010), Hawkins (2012), Ginebra	

8. Metodología
<p>El profesor del curso presentará clases teóricas de los temas señalados en el programa propiciando la intervención de los alumnos.</p> <p>El profesor del curso presentará demostraciones para fundamentar clases teóricas.</p> <p>El profesor y los alumnos realizarán prácticas</p> <p>Los alumnos deberán asistir a clase habiendo leído lo que el profesor va a presentar. De esta manera se facilitará la comprensión y los estudiantes estarán en mejores condiciones de hacer consultas en clase.</p>

9. Evaluar

Evaluación Continua 1 : 20 %

Examen parcial : 30 %

Evaluación Continua 2 : 20 %

Examen final : 30 %

References

- Alexandre., Dianine-Havard (2009). *Perfil del Líder. Hacia un Liderazgo Virtuoso*. Ediciones Urano S.A.
- Alfred., Sonnenfeld (2010). *Liderazgo Ético. La Sabiduría de decidir bien*. Ediciones Encuentro S.A Madrid y Nueva Revista de Madrid.
- D' Souza, SJ Anthony. (n.d.). *Descubre tu Liderazgo*. Editorial Sal Terrae.
- Goleman, D. (2012). *Inteligencia emocional*. Editorial Kairós.
- Hawkins, Peter. (2012). *Coaching y liderazgo de equipos: coaching para un liderazgo con capacidad de transformación*. Ediciones Granica.
- Hunsaker, Phil. (2010). *El nuevo arte de gestionar equipos: Un enfoque actual para guiar y motivar con éxito*.
- Luis., Huete (2008). *Construye tu Sueño*. LID Editorial Empresarial.
- Manuel., Ferreiro Pablo/Alcázar (2009). *Gobierno de Personas en la Empresa*. Ediciones Universidad de Navarra EUNSA.
- Maruja., Chinchilla Nuria/Moragas (2007). *Dueños de Nuestro Destino*. Editorial Ariel.
- Pilar, Cardona Pablo/García Lombardi (2002). *Cómo desarrollar las Competencias de Liderazgo*. PAD Lima- Perú, Tercera Edición.
- Wilkinson., Cardona Pablo/ Helen (2009). *Creciendo como Líder*. Ediciones Universidad de Navarra S.A (EUNSA), Primera Edición.



Universidad Nacional de Colombia (UNAL) Sede
Manizales
Programa Profesional de
Administración de Sistemas Informáticos
SILABO

CS281. Computación en la Sociedad (Obligatorio)

2022-II

1. Información general	
1.1 Escuela	: Sistemas de Información
1.2 Curso	: CS281. Computación en la Sociedad
1.3 Semestre	: 8 ^{vo} Semestre.
1.4 Prerrequisitos	: Ninguno
1.5 Condición	: Obligatorio
1.6 Modalidad de aprendizaje	: Presencial
1.7 horas	: 2 HT;
1.8 Créditos	: 2
2. Profesores	
3. Fundamentación del curso	
Ofrece una visión amplia de los aspectos éticos y profesionales relacionados con la computación. Los tópicos que se incluyen abarcan los aspectos éticos, sociales y políticos. Las dimensiones morales de la computación. Los métodos y herramientas de análisis. Administración de los recursos computacionales. Seguridad y control de los sistemas computacionales. Responsabilidades profesionales y éticas. Propiedad intelectual.	
4. Resumen	
1. Historia 2. Contexto Social 3. Herramientas de Análisis 4. Ética Profesional 5. Propiedad Intelectual 6. Privacidad y Libertades Civiles 7. Políticas de seguridad, Leyes y crímenes computacionales 8. Economía de la Computación	
5. Objetivos Generales	
<ul style="list-style-type: none">• Hacer que el alumno entienda la importancia del cuidado y la ética en la transferencia y uso de la información.• Inculcar en el alumno que las tendencias de mejoramiento de la tecnología, no debe ser llevada a degradar la moral de la sociedad.	
6. Contribución a los resultados (Outcomes)	
Esta disciplina contribuye al logro de los siguientes resultados de la carrera:	
3) Comunicarse efectivamente en diversos contextos profesionales. (Familiarizarse)	
4) Reconocer las responsabilidades profesionales y hacer juicios informados en el campo profesional de computación con principios éticos. (Usar)	
6) Aplicar fundamentos de teoría de ciencias de la computación y desarrollo de software para producir soluciones basados en computación. (Usar)	
7) Desarrollar tecnología computacional buscando el bien común, aportando con formación humana, capacidades científicas, tecnológicas y profesionales para solucionar problemas sociales de nuestro entorno. (Usar)	

7. Contenido

UNIDAD 1: Historia (2)

Competencias:

Contenido

- Pre-historia – El mundo antes de 1946.
- Historia del hardware, software, redes.
- Pioneros de la Computación.
- Historia de Internet.

Objetivos Generales

- Identificar importantes tendencias en la historia del campo de la computación [Familiarizarse]
- Identificar las contribuciones de varios pioneros en el campo de la computación [Familiarizarse]
- Discutir el contexto histórico de los paradigmas de diversos lenguajes de programación [Familiarizarse]
- Comparar la vida diaria antes y después de la llegada de los ordenadores personales y el Internet [Familiarizarse]

Lecturas: K. C. Laudon and J. P. Laudon (2004), McLeod Jr (2000)

UNIDAD 2: Contexto Social (4)	
Competencias:	
Contenido	Objetivos Generales
<ul style="list-style-type: none"> • Implicancias sociales de la computación en un mundo conectado en red. • Impacto de los medios sociales en el individualismo, colectivismo y en la cultura. • Crecimiento y control de la Internet • A menudo se refiere como la brecha digital, las diferencias en el acceso a los recursos de la tecnología digital y sus ramificaciones resultantes para el género, la clase, la etnia, la geografía, y/o los países subdesarrollados. • Los problemas de accesibilidad, incluyendo los requisitos legales. • Computación consciente del contexto. 	<ul style="list-style-type: none"> • Describir las formas positivas y negativas en las que la tecnología computacional (redes, computación móvil, <i>cloud computing</i>) altera los modos de interacción social en el plano personal [Familiarizarse] • Identificar los supuestos y valores incorporados en el hardware y el software de diseño de los desarrolladores, especialmente lo que se refiere a la facilidad de uso para diversas poblaciones incluyendo minorías poblaciones y los discapacitados [Usar] • Interpretar el contexto social de un determinado diseño y su aplicación [Evaluar] • Evaluar la eficacia de un diseño y aplicación dada a partir de datos empíricos [Familiarizarse] • Resumir las implicaciones de los medios sociales en el individualismo frente al colectivismo y la cultura [Familiarizarse] • Discuta cómo el acceso a Internet sirve como una fuerza liberadora para las personas que viven bajo las formas opresivas de gobierno; explicar la utilización los límites al acceso a Internet como herramientas de represión política y social [Familiarizarse] • Analizar los pros y los contras de la dependencia de la computación en la implementación de la democracia (por ejemplo, prestación de servicios sociales, votación electrónica) [Familiarizarse] • Describir el impacto de la escasa representación de las diversas poblaciones en la profesión (por ejemplo, la cultura de la industria, la diversidad de productos) [Usar] • Explicar las consecuencias de la sensibilidad al contexto en los sistemas de computación ubicua [Familiarizarse]
Lecturas: K. C. Laudon and J. P. Laudon (2004), McLeod Jr (2000)	

UNIDAD 3: Herramientas de Análisis (2)**Competencias:****Contenido**

- Argumentación ética.
- Teorías éticas y toma de decisiones.
- Suposiciones morales y valores.

Objetivos Generales

- Evaluar las posiciones de las partes interesadas en una situación dada [Familiarizarse]
- Analizar errores lógicos básicos en una discusión [Usar]
- Analizar un argumento para identificar premisas y la conclusión [Familiarizarse]
- Ilustrar el uso de ejemplo y analogía en el argumento ético [Familiarizarse]
- Evaluar compensaciones éticos / sociales en las decisiones técnicas [Familiarizarse]

Lecturas: K. C. Laudon and J. P. Laudon (2004), McLeod Jr (2000)

UNIDAD 4: Ética Profesional (4)	
Competencias:	
Contenido	Objetivos Generales
<ul style="list-style-type: none"> • Community values and the laws by which we live. • La naturaleza del profesionalismo incluido el cuidado, la atención y la disciplina, la responsabilidad fiduciaria y mentoría. • Mantenerse al día como profesional de computación en términos de familiaridad, herramientas, habilidades, marco legal y profesional, así como la capacidad de autoevaluarse y avances en el campo de la computación. • La certificación profesional, códigos de ética, conducta y práctica, como la ACM / IEEE-CS, SE, AITP, IFIP y las sociedades internacionales. • Rendición de cuentas, la responsabilidad y la confiabilidad (por ejemplo, la corrección de software, fiabilidad y seguridad, así como la confidencialidad ética de los profesionales de seguridad cibernética) • El papel del profesional de de computación en las políticas públicas. • Mantenimiento de la conciencia en relación a las consecuencias. • Disidencia ética y la denuncia de irregularidades. • La relación entre la cultura regional y dilemas éticos. • Tratar con el acoso y la discriminación. • Formas de credenciamiento profesional. • Políticas de uso aceptable para la computación en el lugar de trabajo. • Ergonomía y entornos de trabajo computacionales saludables. • Consideraciones a tiempos de entrega de mercado vs estándares de calidad profesional. 	<ul style="list-style-type: none"> • Identificar los problemas éticos que se plantean en el desarrollo de software y determinar cómo abordarlos técnica y éticamente [Usar] • Explicar la responsabilidad ética de velar por la corrección de software, confiabilidad y seguridad [Evaluar] • Describir los mecanismos que normalmente existen para que profesional se mantenga al día [Familiarizarse] • Describir las fortalezas y debilidades de códigos profesionales relevantes como expresiones de profesionalismo y guías para la toma de decisiones [Familiarizarse] • Analizar un problema mundial de computación, observando el papel de los profesionales y funcionarios del gobierno en el manejo de este problema [Familiarizarse] • Evaluar los códigos de ética profesional de la ACM, la Sociedad de Computación de la IEEE, y otras organizaciones [Familiarizarse] • Describir las formas en que los profesionales pueden contribuir a las políticas públicas [Familiarizarse] • Describir las consecuencias de la conducta profesional inadecuada [Usar] • Identificar las etapas progresivas en un incidente de denuncia de irregularidades [Usar] • Identificar ejemplos de cómo interactúa la cultura regional con dilemas éticos [Familiarizarse] • Investigar las formas de acoso, discriminación y formas de ayuda [Usar] • Examine las diversas formas de acreditación de profesionales [Usar] • Explicar la relación entre la ergonomía en los ambientes y la salud de las personas de computación [Usar] • Desarrollar un uso del computador/política de uso aceptable con medidas coercitivas [Familiarizarse] • Describir los problemas asociados con la presión de la industrias para centrarse en el tiempo de comercialización en comparación con la aplicación de normas de calidad profesional [Usar]
Lecturas: K. C. Laudon and J. P. Laudon (2004), McLeod Jr (2000), Ediciones (2009a), Ediciones (2009b), Ediciones (2010)	

UNIDAD 5: Propiedad Intelectual (4)	
Competencias:	
Contenido	Objetivos Generales
<ul style="list-style-type: none"> • Fundamentos filosóficos de propiedad intelectual. • Derechos de propiedad intelectual. • Propiedad intelectual digital intangible (IDIP). • Fundamentos legales para protección de la propiedad intelectual. • Gestión de derechos digitales. • Copyrights, patentes, secretos de comercio, marcas registradas. • Plagiarismo. • Fundamentos del movimiento Open Source. • Piratería de Software. 	<ul style="list-style-type: none"> • Discute las bases filosóficas de la propiedad intelectual [Evaluar] • Discute la racionalidad de la protección legal de la propiedad intelectual [Familiarizarse] • Describe la legislación orientada a los delitos de derechos de autor digitales [Evaluar] • Critica la legislación orientada a los delitos digitales de derechos de autor [Familiarizarse] • Identifica ejemplos contemporáneos de propiedad intelectual digital intangible [Evaluar] • Justifica el uso de material con derechos de autor [Evaluar] [Familiarizarse] • Evalúa los asuntos éticos inherentes a diversos mecanismos de detección de plagio [Familiarizarse] • Interpreta el intento y la implementación de licencias de software [Familiarizarse] • Discute asuntos que involucran la seguridad de patentes en software [Familiarizarse] • Caracteriza y contrasta los conceptos de derechos de autor, patentes y de marcas comerciales [Familiarizarse] • Identifica los objetivos del movimiento de software libre [Evaluar] • Identifica la naturaleza global de la piratería de software [Familiarizarse]
Lecturas: K. C. Laudon and J. P. Laudon (2004), McLeod Jr (2000), Ediciones (2009a), Ediciones (2009b), Ediciones (2010)	

UNIDAD 6: Privacidad y Libertades Civiles (4)	
Competencias:	
Contenido	Objetivos Generales
<ul style="list-style-type: none"> • Fundamentos filosóficos de derechos de privacidad. • Fundamentos legales de protección de privacidad. • Implicaciones de privacidad de recopilación de datos generalizada de bases de datos transaccionales, almacenes de datos, sistemas de vigilancia y la computación en la nube. • Ramificaciones de privacidad diferencial. • Soluciones basadas en la tecnología para la protección de la privacidad. • Legislación de privacidad en áreas de práctica. • Libertades civiles y diferencias culturales. • Libertad de expresión y sus limitaciones. 	<ul style="list-style-type: none"> • Discute las bases filosóficas para la protección legal de la privacidad personal [Familiarizarse] • Evalúa soluciones para amenazas a la privacidad en bases de datos transaccionales y almacenes de datos [Familiarizarse] • Describe los roles de la recolección de datos en la implementación de sistemas de vigilancia intrusiva (ejm. RFID, reconocimiento de rostro, cobro electrónico, computación móvil) [Familiarizarse] • Describe las ramificaciones de la privacidad diferenciada [Familiarizarse] • Investiga el impacto de soluciones tecnológicas a los problemas de privacidad [Familiarizarse] • Critica la intención, el valor potencial y la implementación de las diversas formas de legislación en privacidad [Familiarizarse] • Identifica estrategias que permitan la apropiada libertad de expresión [Familiarizarse]
Lecturas: K. C. Laudon and J. P. Laudon (2004), McLeod Jr (2000), Ediciones (2009a), Ediciones (2009b), Ediciones (2010)	

UNIDAD 7: Políticas de seguridad, Leyes y crímenes computacionales (2)**Competencias:****Contenido**

- Ejemplos de delitos informáticos y reparación legal para delincuentes informáticos.
- Ingeniería social, robo de identidad y recuperación.
- Tópicos relacionados al uso de acceso indebido y las infracciones y materia de seguridad.
- Motivaciones y ramificaciones del ciberterrorismo y el hacking criminal, cracking.
- Efectos de malware, como virus, worms y Trojan horses.
- Estrategias de prevención de Crimen.
- Políticas de Seguridad.

Objetivos Generales

- Listar ejemplos clásicos de delitos informáticos y incidentes de ingeniería social con impacto social [Familiarizarse]
- Identificar leyes que se aplican a delitos informáticos [Familiarizarse]
- Describir la motivación y ramificaciones de ciberterrorismo y hackeo criminal [Familiarizarse]
- Examinar los problemas éticos y legales relacionados con el mal uso de accesos y diversas violaciones en la seguridad [Familiarizarse]
- Discutir el rol del profesional en seguridad y los problemas que están envueltos [Familiarizarse]
- Investigar medidas que puedan ser consideradas por personas y organizaciones incluyendo al gobierno para prevenir o mitigar efectos indeseables de los delitos informáticos y robo de identidad [Familiarizarse]
- Escribir una política de seguridad de una empresa, la cual incluye procedimientos para administrar contraseñas y monitorizar a los empleados [Familiarizarse]

Lecturas: K. C. Laudon and J. P. Laudon (2004), McLeod Jr (2000), Ediciones (2009a), Ediciones (2009b), Ediciones (2010)

UNIDAD 8: Economía de la Computación (2)	
Competencias:	
Contenido	Objetivos Generales
<ul style="list-style-type: none"> • Monopolio y sus implicaciones económicas. • Efecto del suministro de mano de obra calificada y la demanda sober la calidad de los productos de computación. • Estrategias de precio en el dominio de la computación. • El fenomeno del desarrollo de software outsourcing y off-shoring; impactos en el empleo y la economía. • Consecuencias de la globalización para la profesión de Ciencias de la Computación. • Diferencias en acceso a recursos de computación y el posible efecto de los mismos. • Analisis costo/beneficio de trabajos con consideraciones para manufactura, hardware, software e implicaciones de ingeniería. • Costo estimado versus costo actual in relacion al costo total. • Emprendimiento: perspectivas y entrampamientos. • Efectos de red o economias de escala del lado de la demanda. • El uso de la ingeniería económica para hacer frente a las finanzas. 	<ul style="list-style-type: none"> • Resumir los fundamentos para los esfuerzos antimonopolio [Familiarizarse] • Identificar diversas maneras en que la industria de la tecnología de la información está afectada por la escasez de la oferta de trabajo [Familiarizarse] • Identificar la evolución de la estrategia de precios para el cálculo de los bienes y servicios [Familiarizarse] • Discutir los beneficios, los inconvenientes y las implicaciones de <i>off-shoring</i> y <i>outsourcing</i> [Familiarizarse] • Investigar y defender maneras de tratar las limitaciones en el acceso a la computación. [Usar] • Describir los beneficios económicos de efectos de la red [Usar]
Lecturas: K. C. Laudon and J. P. Laudon (2004), McLeod Jr (2000), Ediciones (2009a), Ediciones (2009b), Ediciones (2010)	

8. Metodología
<p>El profesor del curso presentará clases teóricas de los temas señalados en el programa propiciando la intervención de los alumnos.</p> <p>El profesor del curso presentará demostraciones para fundamentar clases teóricas.</p> <p>El profesor y los alumnos realizarán prácticas</p> <p>Los alumnos deberán asistir a clase habiendo leído lo que el profesor va a presentar. De esta manera se facilitará la comprensión y los estudiantes estarán en mejores condiciones de hacer consultas en clase.</p>

9. Evaluar
<p>Evaluación Continua 1 : 20 %</p> <p>Examen parcial : 30 %</p> <p>Evaluación Continua 2 : 20 %</p> <p>Examen final : 30 %</p>

References

- Ediciones, Datamation, ed. (2009a). *Revista Datamation MC Ediciones*.
- Ediciones, Datamation, ed. (2009b). *Understanding the Digital Economy*.
- Ediciones, Datamation, ed. (2010). *Financial Times Mastering Information Management*.
- Laudon, Kenneth C. and Jane P. Laudon (2004). *Sistemas de Información Gerencial*. Prentice Hall.
- McLeod Jr, Raymond (2000). *Sistemas de Información Gerencial*. Prentice Hall.



Universidad Nacional de Colombia (UNAL) Sede
Manizales
Programa Profesional de
Administración de Sistemas Informáticos
SILABO

CS3I1. Seguridad en Computación (Obligatorio)

2022-II

1. Información general	
1.1 Escuela	: Sistemas de Información
1.2 Curso	: CS3I1. Seguridad en Computación
1.3 Semestre	: 8 ^{vo} Semestre.
1.4 Prerrequisitos	: CS231. Redes y Comunicación. (7 ^{mo} Sem)
1.5 Condición	: Obligatorio
1.6 Modalidad de aprendizaje	: Presencial
1.7 horas	: 1 HT; 2 HP; 2 HL;
1.8 Créditos	: 3
2. Profesores	
3. Fundamentación del curso	
Hoy en día la información es uno de los activos más preciados en cualquier organización. Este cursos está orientado a poder brindar al alumno los elementos de seguridad orientados a proteger la información de la organización y principalmente poder preveer los posibles problemas relacionados con este rubro. Esta materia involucra el desarrollo de una actitud preventiva por parte del alumno en todas las áreas relacionadas al desarrollo de software.	
4. Resumen	
1. Fundamentos y Conceptos en Seguridad 2. Principios de Diseño Seguro 3. Programación Defensiva 4. Ataques y Amenazas 5. Seguridad de Red 6. Criptografía 7. Seguridad en la Web 8. Seguridad de plataformas 9. Investigación digital (Digital Forensics) 10. Seguridad en Ingeniería de Software	
5. Objetivos Generales	
<ul style="list-style-type: none">• Discutir a un nivel intermedio avanzado los los fundamentos de la Seguridad Informática.• Brindar los diferentes aspectos que presenta el código malicioso.• Que el alumno conozca los conceptos de criptografía y seguridad en redes de computadoras.• Discutir y analizar junto con el alumno los aspectos de la Seguridad en Internet.	

6. Contribución a los resultados (*Outcomes*)

Esta disciplina contribuye al logro de los siguientes resultados de la carrera:

- 1) Analizar un problema computacional complejo y aplicar los principios computacionales y otras disciplinas relevantes para identificar soluciones. (**Evaluar**)
- 2) Diseñar, implementar y evaluar una solución basada en computación para cumplir con un conjunto determinado de requisitos computacionales en el contexto de las disciplinas del programa. (**Evaluar**)
- 5) Funcionar efectivamente como miembro o líder de un equipo involucrado en actividades apropiadas a la disciplina del programa. (**Usar**)
- 6) Aplicar fundamentos de teoría de ciencias de la computación y desarrollo de software para producir soluciones basados en computación. (**Evaluar**)
- 7) Desarrollar tecnología computacional buscando el bien común, aportando con formación humana, capacidades científicas, tecnológicas y profesionales para solucionar problemas sociales de nuestro entorno. (**Evaluar**)

7. Contenido

UNIDAD 1: Fundamentos y Conceptos en Seguridad (25)

Competencias:

Contenido	Objetivos Generales
<ul style="list-style-type: none">• CIA (Confidencialidad, Integridad, Disponibilidad)• Conceptos de riesgo, amenazas, vulnerabilidades, y los tipos de ataque .• Autenticación y autorización, control de acceso (vs. obligatoria discrecional)• Concepto de la confianza y la honradez .• Ética (revelación responsable)	<ul style="list-style-type: none">• Analizar las ventajas y desventajas de equilibrar las propiedades clave de seguridad (Confidenciabilidad, Integridad, Disponibilidad) [Familiarizarse]• Describir los conceptos de riesgo, amenazas, vulnerabilidades y vectores de ataque (incluyendo el hecho de que no existe tal cosa como la seguridad perfecta) [Familiarizarse]• Explicar los conceptos de autenticación, autorización, control de acceso [Familiarizarse]• Explicar el concepto de confianza y confiabilidad [Familiarizarse]• Reconocer de que hay problemas éticos más importantes que considerar en seguridad computacional, incluyendo problemas éticos asociados a arreglar o no arreglar vulnerabilidades y revelar o no revelar vulnerabilidades [Familiarizarse]
Lecturas: W and L (2014)	

UNIDAD 2: Principios de Diseño Seguro (25)**Competencias:****Contenido**

- Menor privilegio y aislamiento.
- Valores predeterminados a prueba de fallos.
- Diseño abierto.
- La seguridad de extremo a extremo.
- La defensa en profundidad (por ejemplo, la programación defensiva, defensa en capas)
- Diseño de seguridad.
- Las tensiones entre la seguridad y otros objetivos de diseño.
- Mediación completa.
- El uso de componentes de seguridad vetados.
- Economía del mecanismo (la reducción de la base informática de confianza, minimizar la superficie de ataque)
- Seguridad utilizable.
- Componibilidad de seguridad.
- Prevención, detección y disuasión.

Objetivos Generales

- Describir el principio de privilegios mínimos y el aislamiento que se aplican al diseño del sistema [Familiarizarse]
- Resumir el principio de prueba de fallos y negar por defecto [Familiarizarse]
- Discutir las implicaciones de depender de diseño abierto o secreto de diseño para la seguridad [Familiarizarse]
- Explicar los objetivos de seguridad de datos de extremo a extremo [Familiarizarse]
- Discutir los beneficios de tener múltiples capas de defensas [Familiarizarse]
- Por cada etapa en el ciclo de vida de un producto, describir que consideraciones de seguridad deberían ser evaluadas [Familiarizarse]
- Describir el costo y ventajas y desventajas asociadas con el diseño de seguridad de un producto. [Familiarizarse]
- Describir el concepto de mediación y el principio de mediación completa [Familiarizarse]
- Conocer los componentes estándar para las operaciones de seguridad, en lugar de reinventar las operaciones fundamentales [Familiarizarse]
- Explicar el concepto de computación confiable incluyendo base informática confiable y de la superficie de ataque y el principio de minimización de base informática confiable [Familiarizarse]
- Discutir la importancia de la usabilidad en el diseño de mecanismos de seguridad [Familiarizarse]
- Describir problemas de seguridad que surgen en los límites entre varios componentes [Familiarizarse]
- Identificar los diferentes roles de mecanismos de prevención y mecanismos de eliminación/disuasión [Familiarizarse]

Lecturas: W and L (2014)

UNIDAD 3: Programación Defensiva (25)**Competencias:****Contenido****Objetivos Generales**

- Validación de datos de entrada y sanitización
- Elección del lenguaje de programación y lenguajes con tipos de datos seguro.
- Ejemplos de validación de entrada de datos y sanitización de errores.
 - Desbordamiento de búfer
 - Errores enteros
 - Inyección SQL
 - Vulnerabilidad XSS
- Las condiciones de carrera.
- Manejo correcto de las excepciones y comportamientos inesperados.
- Uso correcto de los componentes de terceros.
- Desplegar eficazmente las actualizaciones de seguridad.
- Información de control de flujo.
- Generando correctamente el azar con fines de seguridad.
- Mecanismos para la detección y mitigación de datos de entrada y errores de sanitización.
- Fuzzing
- El análisis estático y análisis dinámico.
- Programa de verificación.
- Soporte del sistema operativo (por ejemplo, la asignación al azar del espacio de direcciones, canarios)
- El soporte de hardware (por ejemplo, el DEP, TPM)

- Explicar por que la validación de entrada y desinfección de datos es necesario en el frente del control contencioso del canal de entrada [Usar]
- Explicar por que uno debería escoger para desallorar un programa en un lenguaje tipo seguro como Java, en contraste con un lenguaje de programación no seguro como C/C++ [Usar]
- Clasificar los errores de validación de entrada común, y escribir correctamente el código de validación de entrada [Usar]
- Demostrar el uso de un lenguaje de programación de alto nivel cómo prevenir una condición de competencia que ocurran y cómo manejar una excepción [Usar]
- Demostrar la identificación y el manejo elegante de las condiciones de error [Familiarizarse]
- Explique los riesgos de mal uso de las interfaces con código de terceros y cómo utilizar correctamente el código de terceros [Familiarizarse]
- Discutir la necesidad de actualizar el software para corregir las vulnerabilidades de seguridad y la gestión del ciclo de vida de la corrección [Familiarizarse]

Lecturas: W and L (2014)

UNIDAD 4: Ataques y Amenazas (25)	
Competencias:	
Contenido	Objetivos Generales
<ul style="list-style-type: none"> • Atacante metas, capacidades y motivaciones (como economía sumergida, el espionaje digital, la guerra cibernética, las amenazas internas, hacktivismo, las amenazas persistentes avanzadas) • Los ejemplos de malware (por ejemplo, virus, gusanos, spyware, botnets, troyanos o rootkits) • Denegación de Servicio (DoS) y Denegación de Servicio Distribuida (DDoS) • Ingeniería social (por ejemplo, pescando) • Los ataques a la privacidad y el anonimato . • El malware / comunicaciones no deseadas, tales como canales encubiertos y esteganografía. 	<ul style="list-style-type: none"> • Describir tipos de ataques similares en contra de un sistema en particular [Familiarizarse] • Discutir los limitantes de las medidas en contra del malware (ejm. detección basada en firmas, detección de comportamiento) [Familiarizarse] • Identificar las instancias de los ataques de ingeniería social y de los ataques de negación de servicios [Familiarizarse] • Discutir como los ataques de negación de servicios puede ser identificados y reducido [Familiarizarse] • Describir los riesgos de la privacidad y del anonimato en aplicaciones comunmente usadas [Familiarizarse] • Discutir los conceptos de conversión de canales y otros procedimientos de filtrado de datos [Familiarizarse]
Lecturas: W and L (2014)	

UNIDAD 5: Seguridad de Red (25)	
Competencias:	
Contenido	Objetivos Generales
<ul style="list-style-type: none"> • Red de amenazas y tipos de ataques específicos (por ejemplo, la denegación de servicio, spoofing, olfateando y la redirección del tráfico, el hombre en el medio, ataques integridad de los mensajes, los ataques de enrutamiento, y el análisis de tráfico) • El uso de cifrado de datos y seguridad de la red . • Arquitecturas para redes seguras (por ejemplo, los canales seguros, los protocolos de enrutamiento seguro, DNS seguro, VPN, protocolos de comunicación anónimos, aislamiento) • Los mecanismos de defensa y contramedidas (por ejemplo, monitoreo de red, detección de intrusos, firewalls, suplantación de identidad y protección DoS, honeypots, seguimientos) • Seguridad para redes inalámbricas, celulares . • Otras redes no cableadas (por ejemplo, ad hoc, sensor, y redes vehiculares) • Resistencia a la censura. • Gestión de la seguridad operativa de la red (por ejemplo, control de acceso a la red configure) 	<ul style="list-style-type: none"> • Describir las diferentes categorías de amenazas y ataques en redes [Familiarizarse] • Describir las arquitecturas de criptografía de clave pública y privada y cómo las ICP brindan apoyo a la seguridad en redes [Familiarizarse] • Describir ventajas y limitaciones de las tecnologías de seguridad en cada capa de una torre de red [Familiarizarse] • Identificar los adecuados mecanismos de defensa y sus limitaciones dada una amenaza de red [Usar]
Lecturas: W and L (2014)	

UNIDAD 6: Criptografía (25)	
Competencias:	
Contenido	Objetivos Generales
<ul style="list-style-type: none"> • Terminología básica de criptografía cubriendo las nociones relacionadas con los diferentes socios (comunicación), canal seguro / inseguro, los atacantes y sus capacidades, cifrado, descifrado, llaves y sus características, firmas. • Tipos de cifrado (por ejemplo, cifrado César, cifrado affine), junto con los métodos de ataque típicas como el análisis de frecuencia. • Apoyo a la infraestructura de clave pública para la firma digital y el cifrado y sus desafíos. • Criptografía de clave simétrica: <ul style="list-style-type: none"> – El secreto perfecto y el cojín de una sola vez – Modos de funcionamiento para la seguridad semántica y encriptación autenticada (por ejemplo, cifrar-entonces-MAC, OCB, GCM) – Integridad de los mensajes (por ejemplo, CMAC, HMAC) • La criptografía de clave pública: <ul style="list-style-type: none"> – Permutación de trampa, por ejemplo, RSA – Cifrado de clave pública, por ejemplo, el cifrado RSA, cifrado El Gamal – Las firmas digitales – Infraestructura de clave pública (PKI) y certificados – Supuestos de dureza, por ejemplo, Diffie-Hellman, factoring entero • Protocolos de intercambio de claves autenticadas, por ejemplo, TLS . • Primitivas criptográficas: <ul style="list-style-type: none"> – generadores pseudo-aleatorios y cifrados de flujo – cifrados de bloque (permutaciones pseudo-aleatorios), por ejemplo, AES – funciones de pseudo-aleatorios – funciones de hash, por ejemplo, SHA2, resistencia colisión – códigos de autenticación de mensaje – funciones derivaciones clave 	<ul style="list-style-type: none"> • Describir el propósito de la Criptografía y listar formas en las cuales es usada en comunicación de datos [Familiarizarse] • Definir los siguientes términos: Cifrado, Criptoanálisis, Algoritmo Criptográfico, y Criptología y describe dos métodos básicos (cifrados) para transformar texto plano en un texto cifrado [Familiarizarse] • Discutir la importancia de los números primos en criptografía y explicar su uso en algoritmos criptográficos [Familiarizarse] • Ilustrar como medir la entropía y como generar aleatoriedad criptográfica [Usar] • Usa primitivas de clave pública y sus aplicaciones [Usar] • Explicar como los protocolos de intercambio de claves trabajan y como es que pueden fallar [Familiarizarse] • Discutir protocolos criptográficos y sus propiedades [Familiarizarse]
Lecturas: W and L (2014)	

UNIDAD 7: Seguridad en la Web (25)**Competencias:****Contenido**

- Modelo de seguridad Web
 - Modelo de seguridad del navegador incluida la política de mismo origen
 - Los límites de confianza de cliente-servidor, por ejemplo, no pueden depender de la ejecución segura en el cliente
- Gestión de sesiones, la autenticación:
 - Single Sign-On
 - HTTPS y certificados
- Vulnerabilidades de las aplicaciones y defensas :
 - Inyección SQL
 - XSS
 - CSRF
- Seguridad del lado del cliente :
 - Política de seguridad Cookies
 - Extensiones de seguridad HTTP, por ejemplo HSTS
 - Plugins, extensiones y aplicaciones web
 - Seguimiento de los usuarios Web
- Herramientas de seguridad del lado del servidor, por ejemplo, los cortafuegos de aplicación Web (WAFS) y fuzzers

Objetivos Generales

- Describe el modelo de seguridad de los navegadores incluyendo las políticas del mismo origen y modelos de amenazas en seguridad web [Familiarizarse]
- Discutir los conceptos de sesiones web, canales de comunicación seguros tales como Seguridad en la Capa de Transporte(*TLS*) y la importancia de certificados de seguridad, autenticación incluyendo inicio de sesión único, como OAuth y Lenguaje de Marcado para Confirmaciones de Seguridad(*SAML*) [Familiarizarse]
- Investigar los tipos comunes de vulnerabilidades y ataques en las aplicaciones web, y defensas contra ellos [Familiarizarse]
- Utilice las funciones de seguridad del lado del cliente [Usar]

Lecturas: W and L (2014)

UNIDAD 8: Seguridad de plataformas (25)**Competencias:****Contenido**

- Integridad de código y firma de código.
- Arranque seguro, arranque medido, y la raíz de confianza.
- Testimonio.
- TPM y coprocesadores seguros.
- Las amenazas de seguridad de los periféricos, por ejemplo, DMA, IOMMU.
- Ataques físicos: troyanos de hardware, sondas de memoria, ataques de arranque en frío.
- Seguridad de dispositivos integrados, por ejemplo, dispositivos médicos, automóviles.
- Ruta confiable.

Objetivos Generales

- Explica el concepto de integridad de código y firma de códigos, así como el alcance al cual se aplica [Familiarizarse]
- Discute los conceptos del origen de la confidencialidad y el de los procesos de arranque y carga segura [Familiarizarse]
- Describe los mecanismos de arresto remoto de la integridad de un sistema [Familiarizarse]
- Resume las metas y las primitivas claves de los modelos de plataforma confiable (TPM) [Familiarizarse]
- Identifica las amenazas de conectar periféricos en un dispositivo [Familiarizarse]
- Identifica ataques físicos y sus medidas de control [Familiarizarse]
- Identifica ataques en plataformas con hardware que no son del tipo PC [Familiarizarse]
- Discute los conceptos y la importancia de ruta confiable [Familiarizarse]

Lecturas: W and L (2014)

UNIDAD 9: Investigación digital (Digital Forensics) (25)**Competencias:**

Contenido	Objetivos Generales
<ul style="list-style-type: none">● Principios básicos y metodologías de análisis digital forense.● Diseñar sistemas con necesidades forenses en mente.● Reglas de Evidencia - conceptos generales y las diferencias entre las jurisdicciones y la Cadena de Custodia.● Búsqueda y captura de comprobación: requisitos legales y de procedimiento.● Métodos y normas de evidencia digital.● Las técnicas y los estándares para la conservación de los datos.● Cuestiones legales y reportes incluyendo el trabajo como perito.● Investigación digital de los sistema de archivos.● Los forenses de aplicación.● Investigación digital en la web.● Investigación digital en redes.● Investigación digital en dispositivos móviles.● Ataques al computador/red/sistema.● Detección e investigación de ataque.● Contra investigación digital.	<ul style="list-style-type: none">● Describe qué es una investigación digital, las fuentes de evidencia digital, y los límites de técnicas forenses [Familiarizarse]● Explica como diseñar software de apoyo a técnicas forenses [Familiarizarse]● Describe los requisitos legales para usar datos recuperados [Familiarizarse]● Describe el proceso de recolección de evidencia desde el tiempo en que se identifico el requisito hasta la colocación de los datos [Familiarizarse]● Describe como se realiza la recolección de datos y el adecuado almacenamiento de los datos originales y de la copia forense [Familiarizarse]● Realiza recolección de datos en un disco duro [Usar]● Describe la responsabilidad y obligación de una persona mientras testifica como un examinador forense [Familiarizarse]● Recupera datos basados en un determinado término de búsqueda en una imagen del sistema [Usar]● Reconstruye el historial de una aplicación a partir de los artefactos de la aplicación [Familiarizarse]● Reconstruye el historial de navegación web de los artefactos web [Familiarizarse]● Captura e interpreta el tráfico de red [Familiarizarse]● Discute los retos asociados con técnicas forenses de dispositivos móviles [Familiarizarse]

Lecturas: W and L (2014)

UNIDAD 10: Seguridad en Ingeniería de Software (25)	
Competencias:	
Contenido	Objetivos Generales
<ul style="list-style-type: none"> • La construcción de la seguridad en el ciclo de vida de desarrollo de software. • Principios y patrones de diseño seguros. • Especificaciones de software seguros y requisitos. • Prácticas de desarrollo de software de seguros. • Asegure probar el proceso de las pruebas de que se cumplan los requisitos de seguridad (incluyendo análisis estático y dinámico) 	<ul style="list-style-type: none"> • Describir los requisitos para la integración de la seguridad en el SDL [Familiarizarse] • Aplicar los conceptos de los principios de diseño para mecanismos de protección, los principios para seguridad de software (Viega and McGraw) y los principios de diseño de seguridad (Morrie Gasser) en un proyecto de desarrollo de software [Familiarizarse] • Desarrollar especificaciones para un esfuerzo de desarrollo de software que especifica completamente los requisitos funcionales y se identifican las rutas de ejecución esperadas [Familiarizarse]
Lecturas: W and L (2014)	

8. Metodología
<p>El profesor del curso presentará clases teóricas de los temas señalados en el programa propiciando la intervención de los alumnos.</p> <p>El profesor del curso presentará demostraciones para fundamentar clases teóricas.</p> <p>El profesor y los alumnos realizarán prácticas</p> <p>Los alumnos deberán asistir a clase habiendo leído lo que el profesor va a presentar. De esta manera se facilitará la comprensión y los estudiantes estarán en mejores condiciones de hacer consultas en clase.</p>

9. Evaluar
<p>Evaluación Continua 1 : 20 %</p> <p>Examen parcial : 30 %</p> <p>Evaluación Continua 2 : 20 %</p> <p>Examen final : 30 %</p>

References

W, Stallings. and Brown. L (2014). *Computer Security: Principles and Practice*. Pearson Education, Limited. ISBN: 9780133773927.



Universidad Nacional de Colombia (UNAL) Sede
Manizales
Programa Profesional de
Administración de Sistemas Informáticos
SILABO

CS3P1. Computación Paralela y Distribuida
(Obligatorio)

2022-II

1. Información general	
1.1 Escuela	: Sistemas de Información
1.2 Curso	: CS3P1. Computación Paralela y Distribuida
1.3 Semestre	: 8 ^{vo} Semestre.
1.4 Prerrequisitos	: <ul style="list-style-type: none">• CS212. Análisis y Diseño de Algoritmos. (5^{to} Sem)• CS231. Redes y Comunicación. (7^{mo} Sem)
1.5 Condición	: Obligatorio
1.6 Modalidad de aprendizaje	: Presencial
1.7 horas	: 2 HT; 2 HP; 2 HL;
1.8 Créditos	: 4

2. Profesores

3. Fundamentación del curso
<p>La última década ha traído un crecimiento explosivo en computación con multiprocesadores, incluyendo los procesadores de varios núcleos y centros de datos distribuidos. Como resultado, la computación paralela y distribuida se ha convertido de ser un tema ampliamente electivo para ser uno de los principales componentes en la malla estudios en ciencia de la computación de pregrado. Tanto la computación paralela como la distribuida implica la ejecución simultánea de múltiples procesos, cuyas operaciones tienen el potencial para intercalarse de manera compleja. La computación paralela y distribuida construye sobre cimientos en muchas áreas, incluyendo la comprensión de los conceptos fundamentales de los sistemas, tales como: concurrencia y ejecución en paralelo, consistencia en el estado/manipulación de la memoria, y latencia. La comunicación y la coordinación entre los procesos tiene sus cimientos en el paso de mensajes y modelos de memoria compartida de la computación y conceptos algorítmicos como atomicidad, el consenso y espera condicional. El logro de aceleración en la práctica requiere una comprensión de algoritmos paralelos, estrategias para la descomposición problema, arquitectura de sistemas, estrategias de implementación y análisis de rendimiento. Los sistemas distribuidos destacan los problemas de la seguridad y tolerancia a fallos, hacen hincapié en el mantenimiento del estado replicado e introducen problemas adicionales en el campo de las redes de computadoras.</p>

4. Resumen
1. Fundamentos de paralelismo 2. Arquitecturas paralelas 3. Descomposición en paralelo 4. Comunicación y coordinación 5. Análisis y programación de algoritmos paralelos 6. Desempeño en paralelo

5. Objetivos Generales

- Que el alumno sea capaz de crear aplicaciones paralelas de mediana complejidad aprovechando eficientemente máquinas con múltiples núcleos.
- Que el alumno sea capaz de comparar aplicaciones secuenciales y paralelas.
- Que el alumno sea capaz de convertir, cuando la situación lo amerite, aplicaciones secuenciales a paralelas de forma eficiente.

6. Contribución a los resultados (*Outcomes*)

Esta disciplina contribuye al logro de los siguientes resultados de la carrera:

- 1) Analizar un problema computacional complejo y aplicar los principios computacionales y otras disciplinas relevantes para identificar soluciones. (**Usar**)
- 6) Aplicar fundamentos de teoría de ciencias de la computación y desarrollo de software para producir soluciones basados en computación. (**Usar**)

7. Contenido

UNIDAD 1: Fundamentos de paralelismo (18)

Competencias:

Contenido	Objetivos Generales
<ul style="list-style-type: none">• Procesamiento Simultáneo Múltiple.• Metas del Paralelismo (ej. rendimiento) frente a Concurrencia (ej. control de acceso a recursos compartidos)• Paralelismo, comunicación, y coordinación:<ul style="list-style-type: none">– Paralelismo, comunicación, y coordinación– Necesidad de Sincronización• Errores de Programación ausentes en programación secuencial:<ul style="list-style-type: none">– Tipos de Datos (lectura/escritura simultánea o escritura/escritura compartida)– Tipos de Nivel más alto (interleavings violating program intention, no determinismo no deseado)– Falta de vida/progreso (deadlock, starvation)	<ul style="list-style-type: none">• Distinguir el uso de recursos computacionales para una respuesta mas rápida para administrar el acceso eficiente a un recurso compartido [Familiarizarse]• Distinguir múltiples estructuras de programación suficientes para la sincronización que pueden ser interimplementables pero tienen ventajas complementarias [Familiarizarse]• Distinguir datos de carrera (<i>data races</i>) a partir de carreras de mas alto nivel [Familiarizarse]

Lecturas: Pacheco (2011), Matloff (2014), **quinnz**, Georg Hager (2010)

UNIDAD 2: Arquitecturas paralelas (12)	
Competencias:	
Contenido	Objetivos Generales
<ul style="list-style-type: none"> • Procesadores mutlinúcleo. • Memoria compartida vs memoria distribuida. • Multiprocesamiento simétrico. • SIMD, procesamiento de vectores. • GPU, coprocesamiento. • Taxonomía de Flynn. • Soporte a nivel de instrucciones para programación paralela. <ul style="list-style-type: none"> – Instrucciones atómicas como Compare/Set (Comparar / Establecer) • Problemas de Memoria: <ul style="list-style-type: none"> – Caches multiprocesador y coherencia de cache – Acceso a Memoria no uniforme (NUMA) • Topologías. <ul style="list-style-type: none"> – Interconexiones – Clusters – Compartir recursos (p.e., buses e interconexiones) 	<ul style="list-style-type: none"> • Explicar las diferencias entre memoria distribuida y memoria compartida [Evaluar] • Describir la arquitectura SMP y observar sus principales características [Evaluar] • Distinguir los tipos de tareas que son adecuadas para máquinas SIMD [Usar] • Describir las ventajas y limitaciones de GPUs vs CPUs [Usar] • Explicar las características de cada clasificación en la taxonomía de Flynn [Usar] • Describir los desafíos para mantener la coherencia de la caché [Familiarizarse] • Describir los desafíos clave del desempeño en diferentes memorias y topologías de sistemas distribuidos [Familiarizarse]
Lecturas: Pacheco (2011), Kirk and Hwu (2013), Sanders and Kandrot (2010), Georg Hager (2010)	

UNIDAD 3: Descomposición en paralelo (18)	
Competencias:	
Contenido	Objetivos Generales
<ul style="list-style-type: none"> • Necesidad de Comunicación y coordinación/sincronización. • Independencia y Particionamiento. • Conocimiento Básico del Concepto de Descomposición Paralela. • Descomposición basada en tareas: <ul style="list-style-type: none"> – Implementación de estrategias como hebras • Descomposición de Información Paralela <ul style="list-style-type: none"> – Estrategias como SIMD y MapReduce • Actores y Procesos Reactivos (solicitud de gestores) 	<ul style="list-style-type: none"> • Explicar por qué la sincronización es necesaria en un programa paralelo específico [Usar] • Identificar oportunidades para particionar un programa serial en módulos paralelos independientes [Familiarizarse] • Escribir un algoritmo paralelo correcto y escalable [Usar] • Paralelizar un algoritmo mediante la aplicación de descomposición basada en tareas [Usar] • Paralelizar un algoritmo mediante la aplicación de descomposición datos en paralelo [Usar] • Escribir un programa usando actores y/o procesos reactivos [Usar]
Lecturas: Pacheco (2011), Matloff (2014), Quinn (2003), Georg Hager (2010)	

UNIDAD 4: Comunicación y coordinación (18)	
Competencias:	
Contenido	Objetivos Generales
<ul style="list-style-type: none"> • Memoria Compartida. • La consistencia, y su papel en los lenguaje de programación garantias para los programas de carrera libre. • Pasos de Mensaje: <ul style="list-style-type: none"> – Mensajes Punto a Punto versus multicast (o basados en eventos) – Estilos para enviar y recibir mensajes Blocking vs non-blocking – Buffering de mensajes • Atomicidad: <ul style="list-style-type: none"> – Especificar y probar atomicidad y requerimientos de seguridad – Granularidad de accesos atómicos y actualizaciones, y uso de estructuras como secciones críticas o transacciones para describirlas – Exclusión mutua usando bloques, semáforos, monitores o estructuras relacionadas <ul style="list-style-type: none"> * Potencial para fallas y bloqueos (<i>deadlock</i>) (causas, condiciones, prevención) – Composición <ul style="list-style-type: none"> * Componiendo acciones atómicas granulares más grandes usando sincronización * Transacciones, incluyendo enfoques optimistas y conservadores • Consensos: <ul style="list-style-type: none"> – (Cíclicos) barreras, contadores y estructuras relacionadas • Acciones condicionales: <ul style="list-style-type: none"> – Espera condicional (p.e., empleando variables de condición) 	<ul style="list-style-type: none"> • Usar exclusión mútua para evitar una condición de carrera [Usar] • Dar un ejemplo de una ordenación de accesos entre actividades concurrentes (por ejemplo, un programa con condición de carrera) que no son secuencialmente consistentes [Familiarizarse] • Dar un ejemplo de un escenario en el que el bloqueo de mensajes enviados pueden dar <i>deadlock</i> [Usar] • Explicar cuándo y por qué mensajes de multidifusión (<i>multicast</i>) o basado en eventos puede ser preferible a otras alternativas [Familiarizarse] • Escribir un programa que termine correctamente cuando todo el conjunto de procesos concurrentes hayan sido completados [Usar] • Dar un ejemplo de un escenario en el que un intento optimista de actualización puede nunca completarse [Familiarizarse] • Usar semaforos o variables de condición para bloquear hebras hasta una necesaria precondition de mantenga [Usar]
Lecturas: Pacheco (2011), Matloff (2014), Quinn (2003), Georg Hager (2010)	

UNIDAD 5: Análisis y programación de algoritmos paralelos (18)**Competencias:****Contenido****Objetivos Generales**

- Caminos críticos, el trabajo y la duración y la relación con la ley de Amdahl.
- Aceleración y escalabilidad.
- Naturalmente (vergonzosamente) algoritmos paralelos.
- Patrones Algoritmicos paralelos (divide-y-conquista, map/reduce, amos-trabajadores, otros)
 - Algoritmos específicos (p.e., MergeSort paralelo)
- Algoritmos de grafos paralelo (por ejemplo, la ruta más corta en paralelo, árbol de expansión paralela)
- Cálculos de matriz paralelas.
- Productor-consumidor y algoritmos paralelos segmentados.
- Ejemplos de algoritmos paralelos no-escalables.

- Definir: camino crítico, trabajo y *span* [Familiarizarse]
- Calcular el trabajo y el *span* y determinar el camino crítico con respecto a un diagrama de ejecución paralela. [Usar]
- Definir *speed-up* y explicar la noción de escalabilidad de un algoritmo en este sentido [Familiarizarse]
- Identificar tareas independientes en un programa que debe ser paralelizado [Usar]
- Representar características de una carga de trabajo que permita o evite que sea naturalmente paralelizable [Familiarizarse]
- Implementar un algoritmo dividir y conquistar paralelo (y/o algoritmo de un grafo) y medir empíricamente su desempeño relativo a su analogo secuencial [Usar]
- Descomponer un problema (por ejemplo, contar el número de ocurrencias de una palabra en un documento) via operaciones *map* y *reduce* [Usar]
- Proporcionar un ejemplo de un problema que se corresponda con el paradigma productor-consumidor [Usar]
- Dar ejemplos de problemas donde el uso de *pipelining* sería un medio eficaz para la paralelización [Usar]
- Implementar un algoritmo de matriz paralela [Usar]
- Identificar los problemas que surgen en los algoritmos del tipo productor-consumidor y los mecanismos que pueden utilizarse para superar dichos problemas [Usar]

Lecturas: Matloff (2014), Quinn (2003), Georg Hager (2010)

UNIDAD 6: Desempeño en paralelo (18)	
Competencias:	
Contenido	Objetivos Generales
<ul style="list-style-type: none"> • Equilibrio de carga. • La medición del desempeño. • Programación y contención. • Evaluación de la comunicación de arriba. • Gestión de datos: <ul style="list-style-type: none"> – Costos de comunicación no uniforme debidos a proximidad – Efectos de Cache (p.e., false sharing) – Manteniendo localidad espacial • Consumo de energía y gestión. 	<ul style="list-style-type: none"> • Detectar y corregir un desbalanceo de carga [Usar] • Calcular las implicaciones de la ley de Amdahl para un algoritmo paralelo particular [Usar] • Describir como la distribución/disposición de datos puede afectar a los costos de comunicación de un algoritmo [Familiarizarse] • Detectar y corregir una instancia de uso compartido falso (<i>false sharing</i>) [Usar] • Explicar el impacto de la planificación en el desempeño paralelo [Familiarizarse] • Explicar el impacto en el desempeño de la localidad de datos [Familiarizarse] • Explicar el impacto y los puntos de equilibrio relacionados al uso de energía en el desempeño paralelo [Familiarizarse]
Lecturas: Pacheco (2011), Matloff (2014), Kirk and Hwu (2013), Sanders and Kandrot (2010), Georg Hager (2010)	

8. Metodología
<p>El profesor del curso presentará clases teóricas de los temas señalados en el programa propiciando la intervención de los alumnos.</p> <p>El profesor del curso presentará demostraciones para fundamentar clases teóricas.</p> <p>El profesor y los alumnos realizarán prácticas</p> <p>Los alumnos deberán asistir a clase habiendo leído lo que el profesor va a presentar. De esta manera se facilitará la comprensión y los estudiantes estarán en mejores condiciones de hacer consultas en clase.</p>

9. Evaluar
<p>Evaluación Continua 1 : 20 %</p> <p>Examen parcial : 30 %</p> <p>Evaluación Continua 2 : 20 %</p> <p>Examen final : 30 %</p>

References

- Georg Hager, Gerhard Wellein (2010). *Introduction to High Performance Computing for Scientists and Engineers (Chapman & Hall/CRC Computational Science)*. Ed. by CRC Press. 1st. ISBN: 978-1439811924.
- Kirk, David B. and Wen-mei W. Hwu (2013). *Programming Massively Parallel Processors: A Hands-on Approach*. 2nd. Morgan Kaufmann. ISBN: 978-0-12-415992-1.
- Matloff, Norm (2014). *Programming on Parallel Machines*. University of California, Davis.
- Pacheco, Peter S. (2011). *An Introduction to Parallel Programming*. 1st. Morgan Kaufmann. ISBN: 978-0-12-374260-5.
- Quinn, Michael J. (2003). *Parallel Programming in C with MPI and OpenMP*. 1st. McGraw-Hill Education Group. ISBN: 0071232656.

Sanders, Jason and Edward Kandrot (2010). *CUDA by Example: An Introduction to General-Purpose GPU Programming*.
1st. Addison-Wesley Professional. ISBN: 0131387685, 9780131387683.



Universidad Nacional de Colombia (UNAL) Sede
Manizales
Programa Profesional de
Administración de Sistemas Informáticos
SILABO

CS402. Proyecto de Final de Carrera I (Obligatorio)

2022-II

1. Información general	
1.1 Escuela	: Sistemas de Información
1.2 Curso	: CS402. Proyecto de Final de Carrera I
1.3 Semestre	: 8 ^{vo} Semestre.
1.4 Prerrequisitos	: CS401. Metodología de la Investigación en Computación. (7 ^{mo} Sem)
1.5 Condición	: Obligatorio
1.6 Modalidad de aprendizaje	: Presencial
1.7 horas	: 2 HT; 2 HP;
1.8 Créditos	: 3
2. Profesores	
3. Fundamentación del curso	
Este curso tiene por objetivo que el alumno pueda realizar un estudio del estado del arte de un que el alumno ha elegido como tema para su tesis.	
4. Resumen	
1. Levantamiento del estado del arte	
5. Objetivos Generales	
<ul style="list-style-type: none">• Que el alumno realice una investigación inicial en un tema específico realizando el estudio del estado del arte del tema elegido.• Que el alumno muestre dominio en el tema de la línea de investigación elegida.• Que el alumno elija un docente que domine el de investigación elegida como asesor.• Los entregables de este curso son: Avance parcial: Bibliografía sólida y avance de un Reporte Técnico. Final: Reporte Técnico con experimentos preliminares comparativos que demuestren que el alumno ya conoce las técnicas existentes en el área de su proyecto y elegir a un docente que domine el área de su proyecto como asesor de su proyecto.	

6. Contribución a los resultados (*Outcomes*)

Esta disciplina contribuye al logro de los siguientes resultados de la carrera:

- 1) Analizar un problema computacional complejo y aplicar los principios computacionales y otras disciplinas relevantes para identificar soluciones. (**Evaluar**)
- 2) Diseñar, implementar y evaluar una solución basada en computación para cumplir con un conjunto determinado de requisitos computacionales en el contexto de las disciplinas del programa. (**Usar**)
- 3) Comunicarse efectivamente en diversos contextos profesionales. (**Usar**)
- 4) Reconocer las responsabilidades profesionales y hacer juicios informados en el campo profesional de computación con principios éticos. (**Evaluar**)
- 5) Funcionar efectivamente como miembro o líder de un equipo involucrado en actividades apropiadas a la disciplina del programa. (**Usar**)
- 6) Aplicar fundamentos de teoría de ciencias de la computación y desarrollo de software para producir soluciones basados en computación. (**Evaluar**)
- 7) Desarrollar tecnología computacional buscando el bien común, aportando con formación humana, capacidades científicas, tecnológicas y profesionales para solucionar problemas sociales de nuestro entorno. (**Usar**)

7. Contenido

UNIDAD 1: Levantamiento del estado del arte (60)

Competencias:

Contenido	Objetivos Generales
<ul style="list-style-type: none">• Realizar un estudio profundo del estado del arte en un determinado tópico del área de Computación.• Redacción de artículos técnicos en computación.	<ul style="list-style-type: none">• Hacer un levantamiento bibliográfico del estado del arte del tema escogido (esto significa muy probablemente 1 o 2 capítulos de marco teórico además de la introducción que es el capítulo I de la tesis) [Usar]• Redactar un documento en latex en formato artículo (<i>paper</i>) con mayor calidad que en Proyecto I (dominar tablas, figuras, ecuaciones, índices, bibtex, referencias cruzadas, citas, pstricks) [Usar]• Tratar de hacer las presentaciones utilizando prosper [Usar]• Mostrar experimentos básicos [Usar]• Elegir un asesor que domine el área de investigación realizada [Usar]

Lecturas: IEEE-Computer Society (2008), Association for Computing Machinery (2008), CiteSeer.IST (2008)

8. Metodología

El profesor del curso presentará clases teóricas de los temas señalados en el programa propiciando la intervención de los alumnos.

El profesor del curso presentará demostraciones para fundamentar clases teóricas.

El profesor y los alumnos realizarán prácticas

Los alumnos deberán asistir a clase habiendo leído lo que el profesor va a presentar. De esta manera se facilitará la comprensión y los estudiantes estarán en mejores condiciones de hacer consultas en clase.

9. Evaluar

Evaluación Continua 1 : 20 %

Examen parcial : 30 %

Evaluación Continua 2 : 20 %

Examen final : 30 %

References

Association for Computing Machinery (2008). *Digital Libray*. <http://portal.acm.org/dl.cfm>. Association for Computing Machinery.

CiteSeer.IST (2008). *Scientific Literature Digital Libray*. <http://citeseer.ist.psu.edu>. College of Information Sciences and Technology, Penn State University.

IEEE-Computer Society (2008). *Digital Libray*. <http://www.computer.org/publications/dlib>. IEEE-Computer Society.



Universidad Nacional de Colombia (UNAL) Sede
Manizales
Programa Profesional de
Administración de Sistemas Informáticos
SILABO

ET201. Formación de Empresas de Base Tecnológica I
(Obligatorio)

2022-II

1. Información general

1.1 Escuela	:	Sistemas de Información
1.2 Curso	:	ET201. Formación de Empresas de Base Tecnológica I
1.3 Semestre	:	8 ^{vo} Semestre.
1.4 Prerrequisitos	:	Ninguno
1.5 Condición	:	Obligatorio
1.6 Modalidad de aprendizaje	:	Presencial
1.7 horas	:	2 HT; 2 HP;
1.8 Créditos	:	3

2. Profesores

3. Fundamentación del curso

Este es el primer curso dentro del área de formación de empresas de base tecnológica, tiene como objetivo dotar al futuro profesional de conocimientos, actitudes y aptitudes que le permitan elaborar un plan de negocio para una empresa de base tecnológica. El curso está dividido en las siguientes unidades: Introducción, Creatividad, De la idea a la oportunidad, el modelo Canvas, Customer Development y Lean Startup, Aspectos Legales y Marketing, Finanzas de la empresa y Presentación.

Se busca aprovechar el potencial creativo e innovador y el esfuerzo de los alumnos en la creación de nuevas empresas.

4. Resumen

1. 2. 3. 4. 5. 6. 7. 8.

5. Objetivos Generales

- Que el alumno conozca como elaborar un plan de negocio para dar inicio a una empresa de base tecnológica.
- Que el alumno sea capaz de realizar, usando modelos de negocio, la concepción y presentación de una propuesta de negocio.

6. Contribución a los resultados (*Outcomes*)

Esta disciplina contribuye al logro de los siguientes resultados de la carrera:

- 1) Analizar un problema computacional complejo y aplicar los principios computacionales y otras disciplinas relevantes para identificar soluciones. (**Usar**)
- 2) Diseñar, implementar y evaluar una solución basada en computación para cumplir con un conjunto determinado de requisitos computacionales en el contexto de las disciplinas del programa. (**Evaluar**)
- 3) Comunicarse efectivamente en diversos contextos profesionales. (**Usar**)
- 5) Funcionar efectivamente como miembro o líder de un equipo involucrado en actividades apropiadas a la disciplina del programa. (**Usar**)
- 6) Aplicar fundamentos de teoría de ciencias de la computación y desarrollo de software para producir soluciones basados en computación. (**Usar**)
- 7) Desarrollar tecnología computacional buscando el bien común, aportando con formación humana, capacidades científicas, tecnológicas y profesionales para solucionar problemas sociales de nuestro entorno. (**Evaluar**)

7. Contenido

UNIDAD 1: (5)

Competencias:

Contenido

- Emprendedor, emprendedurismo e innovación tecnológica
- Modelos de negocio
- Formación de equipos

Objetivos Generales

- Identificar características de los emprendedores [Familiarizarse]
- Introducir modelos de negocio [Familiarizarse]

Lecturas: Byers, R. Dorf, and Nelson (2010), Osterwalder and Pigneur (2010), Garzozi-Pincay et al. (2014)

UNIDAD 2: (5)

Competencias:

Contenido

- Visión
- Misión
- La Propuesta de valor
- Creatividad e invención
- Tipos y fuentes de innovación
- Estrategia y Tecnología
- Escala y ámbito

Objetivos Generales

- Plantear correctamente la visión y misión de empresa [Usar]
- Caracterizar una propuesta de valor innovadora [Evaluar]
- Identificar los diversos tipos y fuentes de innovación [Familiarizarse]

Lecturas: Byers, R. Dorf, and Nelson (2010), Blank and B. Dorf (2012), Garzozi-Pincay et al. (2014)

UNIDAD 3: (5)	
Competencias:	
Contenido	Objetivos Generales
<ul style="list-style-type: none"> • Estrategia de la Empresa • Barreras • Ventaja competitiva sostenible • Alianzas • Aprendizaje organizacional • Desarrollo y diseño de productos 	<ul style="list-style-type: none"> • Conocer estrategias empresariales [Familiarizarse] • Caracterizar barreras y ventajas competitivas [Familiarizarse]
Lecturas: Byers, R. Dorf, and Nelson (2010), Osterwalder and Pigneur (2010), Ries (2011), Garzozzi-Pincay et al. (2014)	

UNIDAD 4: (20)	
Competencias:	
Contenido	Objetivos Generales
<ul style="list-style-type: none"> • Creación de un nuevo negocio • El plan de negocio • Canvas • Elementos del Canvas 	<ul style="list-style-type: none"> • Conocer los elementos del modelo Canvas [Usar] • Elaborar un plan de negocio basado en el modelo Canvas [Usar]
Lecturas: Osterwalder and Pigneur (2010), Blank and B. Dorf (2012), Garzozzi-Pincay et al. (2014)	

UNIDAD 5: (20)	
Competencias:	
Contenido	Objetivos Generales
<ul style="list-style-type: none"> • Aceleración versus incubación • Customer Development • Lean Startup 	<ul style="list-style-type: none"> • Conocer y aplicar el modelo Customer Development [Usar] • Conocer y aplicar el modelo Lean Startup [Usar]
Lecturas: Blank and B. Dorf (2012), Ries (2011), Garzozzi-Pincay et al. (2014)	

UNIDAD 6: (5)	
Competencias:	
Contenido	Objetivos Generales
<ul style="list-style-type: none"> • Aspectos Legales y tributarios para la constitución de la empresa • Propiedad intelectual • Patentes • Copyrights y marca registrada • Objetivos de marketing y segmentos de mercado • Investigación de mercado y búsqueda de clientes 	<ul style="list-style-type: none"> • Conocer los aspectos legales necesarios para la formación de una empresa tecnológica [Familiarizarse] • Identificar segmentos de mercado y objetivos de marketing [Familiarizarse]
Lecturas: Byers, R. Dorf, and Nelson (2010), Ries (2011), Congreso de la Republica del Perú (1996), Republica del Peru (1997), Garzozzi-Pincay et al. (2014)	

UNIDAD 7: (5)	
Competencias:	
Contenido	Objetivos Generales
<ul style="list-style-type: none"> • Modelo de costos • Modelo de utilidades • Precio • Plan financiero • Formas de financiamiento • Fuentes de capital • Capital de riesgo 	<ul style="list-style-type: none"> • Definir um modelo de costos y utilidades [Evaluar] • Conocer las diversas fuentes de financiamiento [Familiarizarse]
Lecturas: Byers, R. Dorf, and Nelson (2010), Blank and B. Dorf (2012), Garzozzi-Pincay et al. (2014)	

UNIDAD 8: (5)	
Competencias:	
Contenido	Objetivos Generales
<ul style="list-style-type: none"> • The Elevator Pitch • Presentación • Negociación 	<ul style="list-style-type: none"> • Conocer las diversas formas de presentar propuestas de negocio [Familiarizarse] • Realizar la presentación de una propuesta de negocio [Usar]
Lecturas: Byers, R. Dorf, and Nelson (2010), Blank and B. Dorf (2012), Garzozzi-Pincay et al. (2014)	

8. Metodología
<p>El profesor del curso presentará clases teóricas de los temas señalados en el programa propiciando la intervención de los alumnos.</p> <p>El profesor del curso presentará demostraciones para fundamentar clases teóricas.</p> <p>El profesor y los alumnos realizarán prácticas</p> <p>Los alumnos deberán asistir a clase habiendo leído lo que el profesor va a presentar. De esta manera se facilitará la comprensión y los estudiantes estarán en mejores condiciones de hacer consultas en clase.</p>

9. Evaluar

Evaluación Continua 1 : 20 %

Examen parcial : 30 %

Evaluación Continua 2 : 20 %

Examen final : 30 %

References

- Blank, Steve and Bob Dorf (2012). *The Startup Owner's Manual: The Step-By-Step Guide for Building a Great Company*. K and S Ranch.
- Byers, Thomas, Richard Dorf, and Andrew Nelson (2010). *Technology Ventures: From Idea to Enterprise*. McGraw-Hill Science.
- Congreso de la Republica del Perú (1996). *Decreto Legislativo N°823. Ley de la Propiedad Industrial*. El Peruano.
- Garzozi-Pincay, René et al. (2014). *Planes de Negocios para Emprendedores*. Iniciativa Latinoamericana de Libros de Texto Abiertos (LATIn).
- Osterwalder, Alexander and Yves Pigneur (2010). *Business Model Generation: A Handbook for Visionaries, Game Changers, and Challengers*. Wiley.
- Republica del Peru, Congreso de la (1997). *Ley N°26887. Ley General de Sociedades*. El Peruano.
- Ries, Eric (2011). *The Lean Startup: How Today's Entrepreneurs Use Continuous Innovation to Create Radically Successful Businesses*. Crown Business.



Universidad Nacional de Colombia (UNAL) Sede
Manizales
Programa Profesional de
Administración de Sistemas Informáticos
SILABO

CS370. Big Data (Obligatorio)

2022-II

1. Información general	
1.1 Escuela	: Sistemas de Información
1.2 Curso	: CS370. Big Data
1.3 Semestre	: 9 ^{no} Semestre.
1.4 Prerrequisitos	: <ul style="list-style-type: none">• CS272. Bases de Datos II. (5^{to} Sem)• CS3P1. Computación Paralela y Distribuída. (8^{vo} Sem)
1.5 Condición	: Obligatorio
1.6 Modalidad de aprendizaje	: Presencial
1.7 horas	: 1 HT; 2 HP; 2 HL;
1.8 Créditos	: 3
2. Profesores	
3. Fundamentación del curso	
En la actualidad conocer enfoques escalables para procesar y almacenar grande volúmenes de información (terabytes, petabytes e inclusive exabytes) es fundamental en cursos de ciencia de la computación. Cada día, cada hora, cada minuto se genera gran cantidad de información la cual necesita ser procesada, almacenada, analizada.	
4. Resumen	
1. Introducción a Big Data 2. Hadoop 3. Procesamiento de Grafos en larga escala	
5. Objetivos Generales	
<ul style="list-style-type: none">• Que el alumno sea capaz de crear aplicaciones paralelas para procesar grandes volúmenes de información.• Que el alumno sea capaz de comparar las alternativas para el procesamiento de big data.• Que el alumno sea capaz de proponer arquitecturas para una aplicación escalable.	
6. Contribución a los resultados (Outcomes)	
Esta disciplina contribuye al logro de los siguientes resultados de la carrera: <ol style="list-style-type: none">1) Analizar un problema computacional complejo y aplicar los principios computacionales y otras disciplinas relevantes para identificar soluciones. (Usar)6) Aplicar fundamentos de teoría de ciencias de la computación y desarrollo de software para producir soluciones basados en computación. (Usar)	
7. Contenido	

UNIDAD 1: Introducción a Big Data (15)	
Competencias:	
Contenido	Objetivos Generales
<ul style="list-style-type: none"> • Visión global sobre Cloud Computing • Visión global sobre Sistema de Archivos Distribuidos • Visión global sobre el modelo de programación MapReduce 	<ul style="list-style-type: none"> • Explicar el concepto de Cloud Computing desde el punto de vista de Big Data[Familiarizarse] • Explicar el concepto de los Sistema de Archivos Distribuidos [Familiarizarse] • Explicar el concepto del modelo de programación MapReduce[Familiarizarse]
Lecturas: Coulouris et al. (2011)	

UNIDAD 2: Hadoop (15)	
Competencias:	
Contenido	Objetivos Generales
<ul style="list-style-type: none"> • Visión global de Hadoop. • Historia. • Estructura de Hadoop. • HDFS, Hadoop Distributed File System. • Modelo de Programación MapReduce 	<ul style="list-style-type: none"> • Entender y explicar la suite de Hadoop. [Familiarizarse] • Implementar soluciones usando el modelo de programación MapReduce. [Usar] • Entender la forma como se guardan los datos en el HDFS. [Familiarizarse]
Lecturas: Hwang, Dongarra, and Fox (2011), Buyya, Vecchiola, and Selvi (2013)	

UNIDAD 3: Procesamiento de Grafos en larga escala (10)	
Competencias:	
Contenido	Objetivos Generales
<ul style="list-style-type: none"> • Pregel: A System for Large-scale Graph Processing. • Distributed GraphLab: A Framework for Machine Learning and Data Mining in the Cloud. • Apache Giraph is an iterative graph processing system built for high scalability. 	<ul style="list-style-type: none"> • Entender y explicar la arquitectura del proyecto Pregel. [Familiarizarse] • Entender la arquitectura del proyecto GraphLab. [Familiarizarse] • Entender la arquitectura del proyecto Giraph. [Familiarizarse] • Implementar soluciones usando Pregel, GraphLab o Giraph. [Usar]
Lecturas: Low et al. (2012), Malewicz et al. (2010), Baluja et al. (2008)	

8. Metodología
<p>El profesor del curso presentará clases teóricas de los temas señalados en el programa propiciando la intervención de los alumnos.</p> <p>El profesor del curso presentará demostraciones para fundamentar clases teóricas.</p> <p>El profesor y los alumnos realizarán prácticas</p> <p>Los alumnos deberán asistir a clase habiendo leído lo que el profesor va a presentar. De esta manera se facilitará la comprensión y los estudiantes estarán en mejores condiciones de hacer consultas en clase.</p>

9. Evaluar

Evaluación Continua 1 : 20 %

Examen parcial : 30 %

Evaluación Continua 2 : 20 %

Examen final : 30 %

References

- Baluja, Shumeet et al. (2008). “Video Suggestion and Discovery for Youtube: Taking Random Walks Through the View Graph”. In: *Proceedings of the 17th International Conference on World Wide Web*. WWW '08. ACM: Beijing, China, pp. 895–904. ISBN: 978-1-60558-085-2. DOI: 10.1145/1367497.1367618.
- Buyya, Rajkumar, Christian Vecchiola, and S. Thamarai Selvi (2013). *Mastering Cloud Computing: Foundations and Applications Programming*. 1st. Morgan Kaufmann Publishers Inc.: San Francisco, CA, USA. ISBN: 9780124095397, 9780124114548.
- Coulouris, George et al. (2011). *Distributed Systems: Concepts and Design*. 5th. Addison-Wesley Publishing Company: USA. ISBN: 0132143011, 9780132143011.
- Hwang, Kai, Jack Dongarra, and Geoffrey C. Fox (2011). *Distributed and Cloud Computing: From Parallel Processing to the Internet of Things*. 1st. Morgan Kaufmann Publishers Inc.: San Francisco, CA, USA. ISBN: 0123858801, 9780123858801.
- Low, Yucheng et al. (Apr. 2012). “Distributed GraphLab: A Framework for Machine Learning and Data Mining in the Cloud”. In: *Proc. VLDB Endow.* 5(8), pp. 716–727. ISSN: 2150-8097. DOI: 10.14778/2212351.2212354.
- Malewicz, Grzegorz et al. (2010). “Pregel: A System for Large-scale Graph Processing”. In: *ACM SIGMOD Record*. SIGMOD '10, pp. 135–146. DOI: 10.1145/1807167.1807184.



Universidad Nacional de Colombia (UNAL) Sede
Manizales
Programa Profesional de
Administración de Sistemas Informáticos
SILABO

CS403. Proyecto de Final de Carrera II (Obligatorio)

2022-II

1. Información general	
1.1 Escuela	: Sistemas de Información
1.2 Curso	: CS403. Proyecto de Final de Carrera II
1.3 Semestre	: 9 ^{no} Semestre.
1.4 Prerrequisitos	: CS402. Proyecto de Final de Carrera I. (8 ^{vo} Sem)
1.5 Condición	: Obligatorio
1.6 Modalidad de aprendizaje	: Presencial
1.7 horas	: 2 HT; 2 HP;
1.8 Créditos	: 3
2. Profesores	
3. Fundamentación del curso	
Este curso tiene por objetivo que el alumno concluya su proyecto de tesis.	
4. Resumen	
1. Proyecto de Tesis 2. Avance de Tesis	
5. Objetivos Generales	
<ul style="list-style-type: none">• Que el alumno este en la capacidad de presentar formalmente su proyecto de tesis con el marco teórico y levantamiento bibliográfico completo.• Que el alumno domine el estado del arte de su área de investigación.• Los entregables de este curso son: Avance parcial: Avance del plan de tesis incluyendo motivación y contexto, definición del problema, objetivos, cronograma de actividades hasta el proyecto final de tesis y el estado del arte del tema abordado. Final: Plan de tesis completo y Avance de la Tesis incluyendo los capítulos de marco teórico, trabajos relacionados y resultados (formales o estadísticos) preliminares orientados a su tema de tesis.	

6. Contribución a los resultados (*Outcomes*)

Esta disciplina contribuye al logro de los siguientes resultados de la carrera:

- 1) Analizar un problema computacional complejo y aplicar los principios computacionales y otras disciplinas relevantes para identificar soluciones. (**Evaluar**)
- 2) Diseñar, implementar y evaluar una solución basada en computación para cumplir con un conjunto determinado de requisitos computacionales en el contexto de las disciplinas del programa. (**Evaluar**)
- 3) Comunicarse efectivamente en diversos contextos profesionales. (**Evaluar**)
- 4) Reconocer las responsabilidades profesionales y hacer juicios informados en el campo profesional de computación con principios éticos. (**Evaluar**)
- 5) Funcionar efectivamente como miembro o líder de un equipo involucrado en actividades apropiadas a la disciplina del programa. (**Evaluar**)
- 6) Aplicar fundamentos de teoría de ciencias de la computación y desarrollo de software para producir soluciones basados en computación. (**Evaluar**)
- 7) Desarrollar tecnología computacional buscando el bien común, aportando con formación humana, capacidades científicas, tecnológicas y profesionales para solucionar problemas sociales de nuestro entorno. (**Evaluar**)

7. Contenido

UNIDAD 1: Proyecto de Tesis (30)

Competencias:

Contenido	Objetivos Generales
<ul style="list-style-type: none">• Proyecto de Tesis.	<ul style="list-style-type: none">• Descripción del formato utilizado por la Universidad para el plan de tesis [Evaluar]• Concluir el plan del proyecto de tesis[Evaluar]• Presentar el estado del arte del tema de tesis (50%)[Evaluar]

Lecturas: IEEE-Computer Society (2008), Association for Computing Machinery (2008), CiteSeer.IST (2008)

UNIDAD 2: Avance de Tesis (30)

Competencias:

Contenido	Objetivos Generales
<ul style="list-style-type: none">• Avance de Tesis.	<ul style="list-style-type: none">• Descripción del formato utilizado por la Universidad para la tesis[Evaluar]• Concluir el capítulo del Marco Teórico de la Tesis[Evaluar]• Concluir el capítulo de Trabajos Relacionados (35%)[Evaluar]• Planear, desarrollar y presentar resultados (formales o estadísticos) de experimentos orientados a su tema de tesis (35%)[Evaluar]

Lecturas: IEEE-Computer Society (2008), Association for Computing Machinery (2008), CiteSeer.IST (2008)

8. Metodología

El profesor del curso presentará clases teóricas de los temas señalados en el programa propiciando la intervención de los alumnos.

El profesor del curso presentará demostraciones para fundamentar clases teóricas.

El profesor y los alumnos realizarán prácticas

Los alumnos deberán asistir a clase habiendo leído lo que el profesor va a presentar. De esta manera se facilitará la comprensión y los estudiantes estarán en mejores condiciones de hacer consultas en clase.

9. Evaluar

Evaluación Continua 1 : 20 %

Examen parcial : 30 %

Evaluación Continua 2 : 20 %

Examen final : 30 %

References

Association for Computing Machinery (2008). *Digital Libray*. <http://portal.acm.org/dl.cfm>. Association for Computing Machinery.

CiteSeer.IST (2008). *Scientific Literature Digital Libray*. <http://citeseer.ist.psu.edu>. College of Information Sciences and Technology, Penn State University.

IEEE-Computer Society (2008). *Digital Libray*. <http://www.computer.org/publications/dlib>. IEEE-Computer Society.



Universidad Nacional de Colombia (UNAL) Sede
Manizales
Programa Profesional de
Administración de Sistemas Informáticos
SILABO

CS351. Tópicos en Computación Gráfica (Electivo)

2022-II

1. Información general	
1.1 Escuela	: Sistemas de Información
1.2 Curso	: CS351. Tópicos en Computación Gráfica
1.3 Semestre	: 9 ^{no} Semestre.
1.4 Prerrequisitos	: CS251. Computación Gráfica. (7 ^{mo} Sem)
1.5 Condición	: Electivo
1.6 Modalidad de aprendizaje	: Presencial
1.7 horas	: 2 HT; 2 HP; 2 HL;
1.8 Créditos	: 4
2. Profesores	
3. Fundamentación del curso	
En este curso se puede profundizar en alguno de los tópicos mencionados en el área de Computación Gráfica (<i>Graphics and Visual Computing - GV</i>). Éste curso está destinado a realizar algún curso avanzado sugerido por la curricula de la ACM/IEEE. Hughes et al. (2013); Hearn and Baker (1990)	
4. Resumen	
1. Topics Avanzados en Computación Gráfica	
5. Objetivos Generales	
<ul style="list-style-type: none">• Que el alumno utilice técnicas de computación gráfica más sofisticadas que involucren estructuras de datos y algoritmos complejos.• Que el alumno aplique los conceptos aprendidos para crear una aplicación sobre un problema real.• Que el alumno investigue la posibilidad de crear un nuevo algoritmo y/o técnica nueva para resolver un problema real.	
6. Contribución a los resultados (<i>Outcomes</i>)	
Esta disciplina contribuye al logro de los siguientes resultados de la carrera:	
1) Analizar un problema computacional complejo y aplicar los principios computacionales y otras disciplinas relevantes para identificar soluciones. (Usar)	
6) Aplicar fundamentos de teoría de ciencias de la computación y desarrollo de software para producir soluciones basados en computación. (Usar)	
7. Contenido	

UNIDAD 1: Topics Avanzados en Computación Gráfica (0)	
Competencias:	
Contenido	Objetivos Generales
<ul style="list-style-type: none"> • CS355. Advanced Computer Graphics • CS356. Computer animation • CS313. Geometric Algorithms • CS357. Visualización • CS358. Virtual reality 	<ul style="list-style-type: none"> • Tópicos Avanzados en Computación Gráfica
Lecturas: Marschner and Shirley (2016)	

8. Metodología
<p>El profesor del curso presentará clases teóricas de los temas señalados en el programa propiciando la intervención de los alumnos.</p> <p>El profesor del curso presentará demostraciones para fundamentar clases teóricas.</p> <p>El profesor y los alumnos realizarán prácticas</p> <p>Los alumnos deberán asistir a clase habiendo leído lo que el profesor va a presentar. De esta manera se facilitará la comprensión y los estudiantes estarán en mejores condiciones de hacer consultas en clase.</p>

9. Evaluar
<p>Evaluación Continua 1 : 20 %</p> <p>Examen parcial : 30 %</p> <p>Evaluación Continua 2 : 20 %</p> <p>Examen final : 30 %</p>

References

- Hearn, Donald and Pauline Baker (1990). *Computer Graphics in C*. Prentice Hall.
- Hughes, John F. et al. (2013). *Computer Graphics - Principles and Practice 3rd Edition*. Addison-Wesley.
- Marschner, Steve and Peter Shirley (2016). *Fundamentals of Computer Graphics*. Fourth Edition. CRC Press. ISBN: ISBN-10: 1482229390.



Universidad Nacional de Colombia (UNAL) Sede
Manizales
Programa Profesional de
Administración de Sistemas Informáticos
SILABO

CS361. Tópicos en Inteligencia Artificial (Electivo)

2022-II

1. Información general	
1.1 Escuela	: Sistemas de Información
1.2 Curso	: CS361. Tópicos en Inteligencia Artificial
1.3 Semestre	: 9 ^{no} Semestre.
1.4 Prerrequisitos	: CS261. Inteligencia Artificial. (7 ^{mo} Sem)
1.5 Condición	: Electivo
1.6 Modalidad de aprendizaje	: Presencial
1.7 horas	: 2 HT; 2 HP; 2 HL;
1.8 Créditos	: 4
2. Profesores	
3. Fundamentación del curso	
Provee una serie de herramientas para resolver problemas que son difíciles de solucionar con los métodos algorítmicos tradicionales. Incluyendo heurísticas, planeamiento, formalismos en la representación del conocimiento y del razonamiento, técnicas de aprendizaje en máquinas, técnicas aplicables a los problemas de acción y reacción: así como el aprendizaje de lenguaje natural, visión artificial y robótica entre otros.	
4. Resumen	
1.	
5. Objetivos Generales	
<ul style="list-style-type: none">• Realizar algún curso avanzado de Inteligencia Artificial sugerido por el currículo de la ACM/IEEE.	
6. Contribución a los resultados (Outcomes)	
Esta disciplina contribuye al logro de los siguientes resultados de la carrera:	
1) Analizar un problema computacional complejo y aplicar los principios computacionales y otras disciplinas relevantes para identificar soluciones. (Evaluar)	
5) Funcionar efectivamente como miembro o líder de un equipo involucrado en actividades apropiadas a la disciplina del programa. (Usar)	
6) Aplicar fundamentos de teoría de ciencias de la computación y desarrollo de software para producir soluciones basados en computación. (Evaluar)	
7. Contenido	

UNIDAD 1: (60)	
Competencias:	
Contenido	Objetivos Generales
<ul style="list-style-type: none"> • CS360. Sistemas Inteligentes • CS361. Razonamiento automatizado • CS362. Sistemas Basados en Conocimiento • CS363. Aprendizaje de Maquina Russell and Norvig (2003), Haykin (1999) • CS364. Sistemas de Planeamiento • CS365. Procesamiento de Lenguaje Natural • CS366. Agentes • CS367. Robótica • CS368. Computación Simbólica • CS369. Algoritmos Genéticos Goldberg (1989) 	<ul style="list-style-type: none"> • Profundizar en diversas técnicas relacionadas a la Inteligencia Artificial [Usar]
Lecturas: Russell and Norvig (2003), Haykin (1999), Goldberg (1989)	

8. Metodología
<p>El profesor del curso presentará clases teóricas de los temas señalados en el programa propiciando la intervención de los alumnos.</p> <p>El profesor del curso presentará demostraciones para fundamentar clases teóricas.</p> <p>El profesor y los alumnos realizarán prácticas</p> <p>Los alumnos deberán asistir a clase habiendo leído lo que el profesor va a presentar. De esta manera se facilitará la comprensión y los estudiantes estarán en mejores condiciones de hacer consultas en clase.</p>

9. Evaluar
<p>Evaluación Continua 1 : 20 %</p> <p>Examen parcial : 30 %</p> <p>Evaluación Continua 2 : 20 %</p> <p>Examen final : 30 %</p>

References

Goldberg, David (1989). *Genetic Algorithms in Search, Optimization and Machine Learning*. Addison Wesley.

Haykin, Simon (1999). *Neural networks: A Comprehensive Foundation*. Prentice Hall.

Russell, Stuart and Peter Norvig (2003). *Inteligencia Artificial: Un enfoque moderno*. Prentice Hall.



Universidad Nacional de Colombia (UNAL) Sede
Manizales
Programa Profesional de
Administración de Sistemas Informáticos
SILABO

CS392. Tópicos Avanzados en Ingeniería de Software
(Electivo)

2022-II

1. Información general

1.1 Escuela	:	Sistemas de Información
1.2 Curso	:	CS392. Tópicos Avanzados en Ingeniería de Software
1.3 Semestre	:	9 ^{mo} Semestre.
1.4 Prerrequisitos	:	CS391. Ingeniería de Software III. (7 ^{mo} Sem)
1.5 Condición	:	Electivo
1.6 Modalidad de aprendizaje	:	Presencial
1.7 horas	:	2 HT; 2 HP; 2 HL;
1.8 Créditos	:	4

2. Profesores

3. Fundamentación del curso

El desarrollo de software requiere del uso de mejores prácticas de desarrollo, gestión de proyectos de TI, manejo de equipos y uso eficiente y racional de frameworks de aseguramiento de la calidad y de Gobierno de Portfolios, estos elemento son pieza clave y transversal para el éxito del proceso productivo. Este curso explora el diseño, selección, implementación y gestión de soluciones TI en las Organizaciones. El foco está en las aplicaciones y la infraestructura y su aplicación en el negocio.

4. Resumen

1. Diseño de Software 2. Gestión de Proyectos de Software 3. 4.

5. Objetivos Generales

- Entender una variedad de frameworks para el análisis de arquitectura empresarial y la toma de decisiones
- Utilizar técnicas para la evaluación y gestión del riesgo en el portfollio de la empresa
- Evaluar y planificar la integración de tecnologías emergentes
- Entender el papel y el potencial de las TI para a apoyar la gestión de procesos empresariales
- Entender los difentes enfoques para modelar y mejorar los procesos de negocio
- Describir y comprender modelos de aseguramiento de la calidad como marco clave para el éxitos de los proyectos de TI.
- Comprender y aplicar el framework de IT Governance como elemento clave para la gestión del portfollio de aplicaciones Empresariales

6. Contribución a los resultados (*Outcomes*)

Esta disciplina contribuye al logro de los siguientes resultados de la carrera:

- 1) Analizar un problema computacional complejo y aplicar los principios computacionales y otras disciplinas relevantes para identificar soluciones. (**E**valuar)
- 2) Diseñar, implementar y evaluar una solución basada en computación para cumplir con un conjunto determinado de requisitos computacionales en el contexto de las disciplinas del programa. (**E**valuar)
- 3) Comunicarse efectivamente en diversos contextos profesionales. (**U**sar)
- 5) Funcionar efectivamente como miembro o líder de un equipo involucrado en actividades apropiadas a la disciplina del programa. (**U**sar)
- 6) Aplicar fundamentos de teoría de ciencias de la computación y desarrollo de software para producir soluciones basados en computación. (**E**valuar)
- 7) Desarrollar tecnología computacional buscando el bien común, aportando con formación humana, capacidades científicas, tecnológicas y profesionales para solucionar problemas sociales de nuestro entorno. (**E**valuar)

7. Contenido

UNIDAD 1: Diseño de Software (18)

Competencias:

Contenido

Objetivos Generales

- Principios de diseño del sistema: niveles de abstracción (diseño arquitectónico y el diseño detallado), separación de intereses, ocultamiento de información, de acoplamiento y de cohesión, de reutilización de estructuras estándar.
- Diseño de paradigmas tales como diseño estructurado (descomposición funcional de arriba hacia abajo), el análisis orientado a objetos y diseño, orientado a eventos de diseño, diseño de nivel de componente, centrado datos estructurada, orientada a aspectos, orientado a la función, orientado al servicio.
- Modelos estructurales y de comportamiento de los diseños de software.
- Diseño de patrones.
- Relaciones entre los requisitos y diseños: La transformación de modelos, el diseño de los contratos, invariantes.
- Conceptos de arquitectura de software y arquitecturas estándar (por ejemplo, cliente-servidor, n-capas, transforman centrados, tubos y filtros).
- El uso de componentes de diseño: selección de componentes, diseño, adaptación y componentes de ensamblaje, componentes y patrones, componentes y objetos (por ejemplo, construir una GUI usando un standard widget set)
- Diseños de refactorización utilizando patrones de diseño
- Calidad del diseño interno, y modelos para: eficiencia y desempeño, redundancia y tolerancia a fallos, trazabilidad de los requerimientos.
- Medición y análisis de la calidad de un diseño.
- Compensaciones entre diferentes aspectos de la calidad.
- Aplicaciones en frameworks.
- Middleware: El paradigma de la orientación a objetos con middleware, requerimientos para correr y clasificar objetos, monitores de procesamiento de transacciones y el sistema de flujo de trabajo.
- Principales diseños de seguridad y codificación (cross-reference IAS/Principles of secure design).
 - Principio de privilegios mínimos
 - Principio de falla segura por defecto
 - Principio de aceptabilidad psicológica

- Formular los principios de diseño, incluyendo la separación de problemas, ocultación de información, acoplamiento y cohesión, y la encapsulación [Usar]
- Usar un paradigma de diseño para diseñar un sistema de software básico y explicar cómo los principios de diseño del sistema se han aplicado en este diseño [Usar]
- Construir modelos del diseño de un sistema de software simple los cuales son apropiado para el paradigma utilizado para diseñarlo [Usar]
- En el contexto de un paradigma de diseño simple, describir uno o más patrones de diseño que podrían ser aplicables al diseño de un sistema de software simple [Usar]
- Para un sistema simple adecuado para una situación dada, discutir y seleccionar un paradigma de diseño apropiado [Usar]
- Crear modelos apropiados para la estructura y el comportamiento de los productos de software desde la especificaciones de requisitos [Usar]
- Explicar las relaciones entre los requisitos para un producto de software y su diseño, utilizando los modelos apropiados [Usar]
- Para el diseño de un sistema de software simple dentro del contexto de un único paradigma de diseño, describir la arquitectura de software de ese sistema [Usar]
- Dado un diseño de alto nivel, identificar la arquitectura de software mediante la diferenciación entre las arquitecturas comunes de software, tales como 3 capas (*3-tier*), *pipe-and-filter*, y cliente-servidor [Usar]
- Investigar el impacto de la selección arquitecturas de software en el diseño de un sistema simple [Usar]
- Aplicar ejemplos simples de patrones en un diseño de software [Usar]
- Describir una manera de refactorar y discutir cuando esto debe ser aplicado [Usar]
- Seleccionar componentes adecuados para el uso en un diseño de un producto de software [Usar]
- Explicar cómo los componentes deben ser adaptados para ser usados en el diseño de un producto de software [Usar]
- Diseñar un contrato para un típico componente de software pequeño para el uso de un dado sistema [Usar]
- Discutir y seleccionar la arquitectura de software adecuada para un sistema de software simple para un dado escenario [Usar]

UNIDAD 2: Gestión de Proyectos de Software (14)**Competencias:****Contenido****Objetivos Generales**

- La participación del equipo:
 - Procesos elemento del equipo, incluyendo responsabilidades de tarea, la estructura de reuniones y horario de trabajo
 - Roles y responsabilidades en un equipo de software
 - Equipo de resolución de conflictos
 - Los riesgos asociados con los equipos virtuales (comunicación, la percepción, la estructura)
- Estimación de esfuerzo (a nivel personal)
- Riesgo.
 - El papel del riesgo en el ciclo de vida
 - Categorías elemento de riesgo, incluyendo la seguridad, la seguridad, mercado, finanzas, tecnología, las personas, la calidad, la estructura y el proceso de
- Gestión de equipos:
 - Organización de equipo y la toma de decisiones
 - Roles de identificación y asignación
 - Individual y el desempeño del equipo de evaluación
- Gestión de proyectos:
 - Programación y seguimiento de elementos
 - Herramientas de gestión de proyectos
 - Análisis de Costo/Beneficio
- Software de medición y técnicas de estimación.
- Aseguramiento de la calidad del software y el rol de las mediciones.
- Riesgo.
 - El papel del riesgo en el ciclo de vida
 - Categorías elemento de riesgo, incluyendo la seguridad, la seguridad, mercado, finanzas, tecnología, las personas, la calidad, la estructura y el proceso de
- En todo el sistema de aproximación al riesgo, incluyendo riesgos asociados con herramientas.

- Discutir los comportamientos comunes que contribuyen al buen funcionamiento de un equipo [Usar]
- Crear y seguir un programa para una reunión del equipo [Usar]
- Identificar y justificar las funciones necesarias en un equipo de desarrollo de software [Usar]
- Entender las fuentes, obstáculos y beneficios potenciales de un conflicto de equipo [Usar]
- Aplicar una estrategia de resolución de conflictos en un ambiente de equipo [Usar]
- Utilizar un método ad hoc para estimar el esfuerzo de desarrollo del software (ejemplo, tiempo) y comparar con el esfuerzo actual requerido [Usar]
- Listar varios ejemplos de los riesgos del software [Usar]
- Describir el impacto del riesgo en el ciclo de vida de desarrollo de software [Usar]
- Describir las diferentes categorías de riesgo en los sistemas de software [Usar]
- Demostrar a través de la colaboración de proyectos de equipo los elementos centrales de la construcción de equipos y gestión de equipos [Usar]
- Describir como la elección de modelos de procesos afectan la estructura organizacional de equipos y procesos de toma de decisiones [Usar]
- Crear un equipo mediante la identificación de los roles apropiados y la asignación de funciones a los miembros del equipo [Usar]
- Evaluar y retroalimentar a los equipos e individuos sobre su desempeño en un ambiente de equipo [Usar]
- Usando un software particular procesar, describir los aspectos de un proyecto que necesita ser planeado y monitoreado, (ejemplo, estimar el tamaño y esfuerzo, un horario, reasignación de recursos, control de configuración, gestión de cambios, identificación de riesgos en un proyecto y gestión) [Usar]
- Realizar el seguimiento del progreso de alguna etapa de un proyecto que utiliza métricas de proyectos apropiados [Usar]
- Comparar las técnicas simples de tamaño de software y estimación de costos [Usar]
- Usar una herramienta de gestión de proyectos para ayudar en la asignación y rastreo de tareas en un proyecto de desarrollo de software [Usar]
- Describir el impacto de la tolerancia de riesgos en el proceso de desarrollo de software [Usar]

UNIDAD 3: (14)	
Competencias:	
Contenido	Objetivos Generales
<ul style="list-style-type: none"> • Administración del servicio como práctica. • Ciclo de vida del servicio. • Definiciones y conceptos genéricos. • Modelos y principios claves. • Procesos. • Tecnología y arquitectura. • Competencia y entrenamiento. 	<ul style="list-style-type: none"> • Utilizar y aplicar correctamente ITIL en el proceso de software. [Usar]
Lecturas: Sommerville (2017), Pressman and Maxim (2015)	

UNIDAD 4: (14)	
Competencias:	
Contenido	Objetivos Generales
<ul style="list-style-type: none"> • Fundamentos e Introducción. • Frameworks de Control y IT Governance. 	<ul style="list-style-type: none"> • Utilizar y aplicar correctamente COBIT en el proceso de software. [Usar]
Lecturas: Sommerville (2017), Pressman and Maxim (2015)	

8. Metodología
<p>El profesor del curso presentará clases teóricas de los temas señalados en el programa propiciando la intervención de los alumnos.</p> <p>El profesor del curso presentará demostraciones para fundamentar clases teóricas.</p> <p>El profesor y los alumnos realizarán prácticas</p> <p>Los alumnos deberán asistir a clase habiendo leído lo que el profesor va a presentar. De esta manera se facilitará la comprensión y los estudiantes estarán en mejores condiciones de hacer consultas en clase.</p>

9. Evaluar
<p>Evaluación Continua 1 : 20 %</p> <p>Examen parcial : 30 %</p> <p>Evaluación Continua 2 : 20 %</p> <p>Examen final : 30 %</p>

References

Pressman, Roger S. and Bruce Maxim (Jan. 2015). *Software Engineering: A Practitioner's Approach*. 8th. McGraw-Hill.
 Sommerville, Ian (Mar. 2017). *Software Engineering*. 10th. Pearson.



Universidad Nacional de Colombia (UNAL) Sede
Manizales
Programa Profesional de
Administración de Sistemas Informáticos
SILABO

CB309. Bioinformática (Obligatorio)

2022-II

1. Información general	
1.1 Escuela	: Sistemas de Información
1.2 Curso	: CB309. Bioinformática
1.3 Semestre	: 9 ^{no} Semestre.
1.4 Prerrequisitos	: <ul style="list-style-type: none">• CS212. Análisis y Diseño de Algoritmos. (5^{to} Sem)• MA307. Matemática aplicada a la computación. (6^{to} Sem)
1.5 Condición	: Obligatorio
1.6 Modalidad de aprendizaje	: Presencial
1.7 horas	: 1 HT; 2 HL;
1.8 Créditos	: 2

2. Profesores

3. Fundamentación del curso
<p>El uso de métodos computacionales en las ciencias biológicas se ha convertido en una de las herramientas claves para el campo de la biología molecular, siendo parte fundamental en las investigaciones de esta área.</p> <p>En Biología Molecular, existen diversas aplicaciones que involucran tanto al ADN, al análisis de proteínas o al secuenciamiento del genoma humano, que dependen de métodos computacionales. Muchos de estos problemas son realmente complejos y tratan con grandes conjuntos de datos.</p> <p>Este curso puede ser aprovechado para ver casos de uso concretos de varias áreas de conocimiento de Ciencia de la Computación como: Lenguajes de Programación (PL), Algoritmos y Complejidad (AL), Probabilidades y Estadística, Manejo de Información (IM), Sistemas Inteligentes (IS).</p>

4. Resumen
1. Introducción a la Biología Molecular 2. Comparación de Secuencias 3. Árboles Filogenéticos 4. Ensamblaje de Secuencias de ADN 5. Estructuras secundarias y terciarias 6. Modelos Probabilísticos en Biología Molecular

5. Objetivos Generales
<ul style="list-style-type: none">• Que el alumno tenga un conocimiento sólido de los problemas biológicos moleculares que desafían a la computación.• Que el alumno sea capaz de abstraer la esencia de los diversos problemas biológicos para plantear soluciones usando sus conocimientos de Ciencia de la Computación

6. Contribución a los resultados (*Outcomes*)

Esta disciplina contribuye al logro de los siguientes resultados de la carrera:

- 1) Analizar un problema computacional complejo y aplicar los principios computacionales y otras disciplinas relevantes para identificar soluciones. (**Evaluar**)
- 2) Diseñar, implementar y evaluar una solución basada en computación para cumplir con un conjunto determinado de requisitos computacionales en el contexto de las disciplinas del programa. (**Usar**)
- 6) Aplicar fundamentos de teoría de ciencias de la computación y desarrollo de software para producir soluciones basados en computación. (**Usar**)
- 7) Desarrollar tecnología computacional buscando el bien común, aportando con formación humana, capacidades científicas, tecnológicas y profesionales para solucionar problemas sociales de nuestro entorno. (**Usar**)

7. Contenido

UNIDAD 1: Introducción a la Biología Molecular (4)

Competencias:

Contenido

- Revisión de la química orgánica: moléculas y macromoléculas, azúcares, ácidos nucleicos, nucleótidos, ARN, ADN, proteínas, aminoácidos y niveles de estructura en las proteínas.
- El dogma de la vida: del ADN a las proteínas, transcripción, traducción, síntesis de proteínas
- Estudio del genoma: Mapas y secuencias, técnicas específicas

Objetivos Generales

- Lograr un conocimiento general de los tópicos más importantes en Biología Molecular. [Familiarizarse]
- Entender que los problemas biológicos son un desafío al mundo computacional. [Evaluar]

Lecturas: Clote and Backofen (2000), Setubal and Meidanis (1997)

UNIDAD 2: Comparación de Secuencias (4)

Competencias:

Contenido

- Secuencias de nucleótidos y secuencias de aminoácidos.
- Alineamiento de secuencias, el problema de alineamiento por pares, búsqueda exhaustiva, Programación dinámica, alineamiento global, alineamiento local, penalización por gaps
- Comparación de múltiples secuencias: suma de pares, análisis de complejidad por programación dinámica, heurísticas de alineamiento, algoritmo estrella, algoritmos de alineamiento progresivo.

Objetivos Generales

- Entender y solucionar el problema de alineamiento de un par de secuencias. [Usar]
- Comprender y solucionar el problema de alineamiento de múltiples secuencias. [Usar]
- Conocer los diversos algoritmos de alineamiento de secuencias existentes en la literatura. [Familiarizarse]

Lecturas: Clote and Backofen (2000), Setubal and Meidanis (1997), Pevzner (2000)

UNIDAD 3: Árboles Filogenéticos (4)	
Competencias:	
Contenido	Objetivos Generales
<ul style="list-style-type: none"> • Filogenia: Introducción y relaciones filogenéticas. • Árboles Filogenéticos: definición, tipo de árboles, problema de búsqueda y reconstrucción de árboles • Métodos de Reconstrucción: métodos por parsimonia, métodos por distancia, métodos por máxima verosimilitud, confianza de los árboles reconstruidos 	<ul style="list-style-type: none"> • Comprender el concepto de filogenia, árboles filogenéticos y la diferencia metodológica entre biología y biología molecular. [Familiarizarse] • Comprender el problema de reconstrucción de árboles filogenéticos, conocer y aplicar los principales algoritmos para reconstrucción de árboles filogenéticos. [Evaluar]
Lecturas: Clote and Backofen (2000), Setubal and Meidanis (1997), Pevzner (2000)	

UNIDAD 4: Ensamblaje de Secuencias de ADN (4)	
Competencias:	
Contenido	Objetivos Generales
<ul style="list-style-type: none"> • Fundamento biológico: caso ideal, dificultades, métodos alternativos para secuenciamiento de ADN • Modelos formales de ensamblaje: <i>Shortest Common Superstring, Reconstruction, Multicontig</i> • Algoritmos para ensamblaje de secuencias: representación de overlaps, caminos para crear <i>superstrings</i>, algoritmo voraz, grafos acíclicos. • Heurísticas para ensamblaje: búsqueda de sobreposiciones, ordenación de fragmentos, alineamientos y consenso. 	<ul style="list-style-type: none"> • Comprender el desafío computacional que ofrece el problema de Ensamblaje de Secuencias. [Familiarizarse] • Entender el principio de modelo formal para ensamblaje. [Evaluar] • Conocer las principales heurísticas para el problema de ensamblaje de secuencias ADN [Usar]
Lecturas: Setubal and Meidanis (1997), Aluru (2006)	

UNIDAD 5: Estructuras secundarias y terciarias (4)	
Competencias:	
Contenido	Objetivos Generales
<ul style="list-style-type: none"> • Estructuras moleculares: primaria, secundaria, terciaria, cuaternaria. • Predicción de estructuras secundarias de ARN: modelo formal, energía de pares, estructuras con bases independientes, solución con Programación Dinámica, estructuras con bucles. • <i>Protein folding</i>: Estructuras en proteínas, problema de <i>protein folding</i>. • <i>Protein Threading</i>: Definiciones, Algoritmo <i>Branch & Bound</i>, <i>Branch & Bound</i> para <i>protein threading</i>. • <i>Structural Alignment</i>: definiciones, algoritmo DALI 	<ul style="list-style-type: none"> • Conocer las estructuras protéicas y la necesidad de métodos computacionales para la predicción de la geometría. [Familiarizarse] • Conocer los algoritmos de solución de problemas de predicción de estructuras secundarias ARN, y de estructuras en proteínas. [Evaluar]
Lecturas: Setubal and Meidanis (1997), Clote and Backofen (2000), Aluru (2006)	

UNIDAD 6: Modelos Probabilísticos en Biología Molecular (4)	
Competencias:	
Contenido	Objetivos Generales
<ul style="list-style-type: none"> • Probabilidad: Variables aleatorias, Cadenas de Markov, Algoritmo de Metropoli-Hasting, Campos Aleatorios de Markov y Muestreador de Gibbs, Máxima Verosimilitud. • Modelos Ocultos de Markov (HMM), estimación de parámetros, algoritmo de Viterbi y método Baul-Welch, Aplicación en alineamientos de pares y múltiples, en detección de Motifs en proteínas, en ADN eucariótico, en familias de secuencias. • Filogenia Probabilística: Modelos probabilísticos de evolución, verosimilitud de alineamientos, verosimilitud para inferencia, comparación de métodos probailísticos y no probabilísticos 	<ul style="list-style-type: none"> • Revisar conceptos de Modelos Probabilísticos y comprender su importancia en Biología Molecular Computacional. [Evaluar] • Conocer y aplicar Modelos Ocultos de Markov para varios análisis en Biología Molecular. [Usar] • Conocer la aplicación de modelos probabilísticos en Filogenia y comparalos con modelos no probabilísticos [Evaluar]
Lecturas: Durbin et al. (1998), Clote and Backofen (2000), Aluru (2006), Krogh et al. (1994)	

8. Metodología
<p>El profesor del curso presentará clases teóricas de los temas señalados en el programa propiciando la intervención de los alumnos.</p> <p>El profesor del curso presentará demostraciones para fundamentar clases teóricas.</p> <p>El profesor y los alumnos realizarán prácticas</p> <p>Los alumnos deberán asistir a clase habiendo leído lo que el profesor va a presentar. De esta manera se facilitará la comprensión y los estudiantes estarán en mejores condiciones de hacer consultas en clase.</p>

9. Evaluar
<p>Evaluación Continua 1 : 20 %</p> <p>Examen parcial : 30 %</p> <p>Evaluación Continua 2 : 20 %</p> <p>Examen final : 30 %</p>

References

- Aluru, Srinivas, ed. (2006). *Handbook of Computational Molecular Biology*. Computer and Information Science Series. Chapman & Hall, CRC: Boca Raton, FL.
- Clote, P. and R. Backofen (2000). *Computational Molecular Biology: An Introduction*. 279 pages. John Wiley & Sons Ltd.
- Durbin, R. et al. (1998). *Biological Sequence Analysis: Probabilistic Models of Proteins and Nucleic Acids*. Cambridge University Press, p. 357. ISBN: 9780521629713.
- Krogh, Anders et al. (1994). "Hidden Markov Models in Computational Biology, Applications to Protein Modeling". In: *J Molecular Biology* 235, pp. 1501–1531.
- Pevzner, Pavel A. (2000). *Computational Molecular Biology: an Algorithmic Approach*. The MIT Press: Cambridge, Massachusetts.
- Setubal, João Carlos and João Meidanis (1997). *Introduction to computational molecular biology*. Boston: PWS Publishing Company, pp. I–XIII, 1–296. ISBN: 978-0-534-95262-4.



Universidad Nacional de Colombia (UNAL) Sede
Manizales
Programa Profesional de
Administración de Sistemas Informáticos
SILABO

ET301. Formación de Empresas de Base Tecnológica II
(Obligatorio)

2022-II

1. Información general

1.1 Escuela	:	Sistemas de Información
1.2 Curso	:	ET301. Formación de Empresas de Base Tecnológica II
1.3 Semestre	:	9 ^{mo} Semestre.
1.4 Prerrequisitos	:	ET201. Formación de Empresas de Base Tecnológica I. (8 ^{vo} Sem)
1.5 Condición	:	Obligatorio
1.6 Modalidad de aprendizaje	:	Presencial
1.7 horas	:	2 HT; 2 HP;
1.8 Créditos	:	3

2. Profesores

3. Fundamentación del curso

Este curso tiene como objetivo dotar al futuro profesional de conocimientos, actitudes y aptitudes que le permitan formar su propia empresa de desarrollo de software y/o consultoría en informática. El curso está dividido en tres unidades: Valorización de Proyectos, Marketing de Servicios y Negociaciones. En la primera unidad se busca que el alumno pueda analizar y tomar decisiones en relación a la viabilidad de un proyecto y/o negocio.

En la segunda unidad se busca preparar al alumno para que este pueda llevar a cabo un plan de marketing satisfactorio del bien o servicio que su empresa pueda ofrecer al mercado. La tercera unidad busca desarrollar la capacidad negociadora de los participantes a través del entrenamiento vivencial y práctico y de los conocimientos teóricos que le permitan cerrar contrataciones donde tanto el cliente como el proveedor resulten ganadores. Consideramos estos temas sumamente críticos en las etapas de lanzamiento, consolidación y eventual relanzamiento de una empresa de base tecnológica.

4. Resumen

1. 2. 3.

5. Objetivos Generales

- Que el alumno comprenda y aplique la terminología y conceptos fundamentales de ingeniería económica que le permitan valorizar un proyecto para tomar la mejor decisión económica.
- Que el alumno adquiera las bases para formar su propia empresa de base tecnológica.

6. Contribución a los resultados (*Outcomes*)

Esta disciplina contribuye al logro de los siguientes resultados de la carrera:

- 7) Desarrollar tecnología computacional buscando el bien común, aportando con formación humana, capacidades científicas, tecnológicas y profesionales para solucionar problemas sociales de nuestro entorno. (**Familiarizarse**)
- 7) Desarrollar tecnología computacional buscando el bien común, aportando con formación humana, capacidades científicas, tecnológicas y profesionales para solucionar problemas sociales de nuestro entorno. (**Usar**)
- 3) Comunicarse efectivamente en diversos contextos profesionales. (**Usar**)
- 5) Funcionar efectivamente como miembro o líder de un equipo involucrado en actividades apropiadas a la disciplina del programa. (**Usar**)
- 7) Desarrollar tecnología computacional buscando el bien común, aportando con formación humana, capacidades científicas, tecnológicas y profesionales para solucionar problemas sociales de nuestro entorno. (**Evaluar**)

7. Contenido

UNIDAD 1: (20)

Competencias:

Contenido	Objetivos Generales
<ul style="list-style-type: none">• Introducción• Proceso de toma de decisiones• El valor del dinero en el tiempo• Tasa de interés y tasa de rendimiento• Interés simple e interés compuesto• Identificación de costos• Flujo de Caja Neto• Tasa de Retorno de Inversión (TIR)• Valor Presente Neto (VPN)• Valorización de Proyectos	<ul style="list-style-type: none">• Permitir al alumno tomar decisiones sobre como invertir mejor los fondos disponibles, fundamentadas en el análisis de los factores tanto económicos como no económicos que determinen la viabilidad de un emprendimiento. [Evaluar]

Lecturas: Blank and Tarkin (2006)

UNIDAD 2: (30)	
Competencias:	
Contenido	Objetivos Generales
<ul style="list-style-type: none"> • Introducción • Importancia del marketing en las empresas de servicios • El Proceso estratégico. • El Plan de Marketing • Marketing estratégico y marketing operativo • Segmentación, targeting y posicionamiento de servicios en mercados competitivos • Ciclo de vida del producto • Aspectos a considerar en la fijación de precios en servicios • El rol de la publicidad, las ventas y otras formas de comunicación • El comportamiento del consumidor en servicios • Fundamentos de marketing de servicios • Creación del modelo de servicio • Gestión de la calidad de servicio 	<ul style="list-style-type: none"> • Brindar las herramientas al alumno para que pueda identificar, analizar y aprovechar las oportunidades de marketing que generan valor en un emprendimiento. [Usar] • Lograr que el alumno conozca, entienda e identifique criterios, habilidades, métodos y procedimientos que permitan una adecuada formulación de estrategias de marketing en sectores y medios específicos como lo es una empresa de base tecnológica. [Usar]
Lecturas: Kotler and Keller (2006), Lovelock and Wirtz (2009)	

UNIDAD 3: (10)	
Competencias:	
Contenido	Objetivos Generales
<ul style="list-style-type: none"> • Introducción. ¿Qué es una negociación? • Teoría de las necesidades de la negociación • La proceso de la negociación • Estilos de negociación • Teoría de juegos • El método Harvard de negociación 	<ul style="list-style-type: none"> • Conocer los puntos clave en el proceso de negociación. [Usar] • Establecer una metodología de negociación eficaz. [Usar] • Desarrollar destrezas y habilidades que permitan llevar a cabo una negociación exitosa. [Usar]
Lecturas: Fisher, Ury, and Patton (1996), Manuel Dasí and Martínez (2006)	

8. Metodología
<p>El profesor del curso presentará clases teóricas de los temas señalados en el programa propiciando la intervención de los alumnos.</p> <p>El profesor del curso presentará demostraciones para fundamentar clases teóricas.</p> <p>El profesor y los alumnos realizarán prácticas</p> <p>Los alumnos deberán asistir a clase habiendo leído lo que el profesor va a presentar. De esta manera se facilitará la comprensión y los estudiantes estarán en mejores condiciones de hacer consultas en clase.</p>

9. Evaluar

Evaluación Continua 1 : 20 %

Examen parcial : 30 %

Evaluación Continua 2 : 20 %

Examen final : 30 %

References

- Blank, Leland and Anthony Tarkin (2006). *Ingeniería Económica*. McGraw Hill, México D.F., México.
- Fisher, Roger, William Ury, and Bruce Patton (1996). *Si... ¡de acuerdo! Cómo negociar sin ceder*. Norma, Barcelona.
- Kotler, Philip and Kevin L. Keller (2006). *Dirección de Marketing*. Prentice Hall, México.
- Lovelock, Christopher and Jochen Wirtz (2009). *Marketing de servicios. Personal, tecnología y estrategia*. Prentice Hall, México.
- Manuel Dasí, Fernando de and Rafael Martínez-Vilanova Martínez (2006). *Técnicas de Negociación. Un método práctico*. Esic, Madrid.



Universidad Nacional de Colombia (UNAL) Sede
Manizales
Programa Profesional de
Administración de Sistemas Informáticos
SILABO

CS3P2. Cloud Computing (Obligatorio)

2022-II

1. Información general	
1.1 Escuela	: Sistemas de Información
1.2 Curso	: CS3P2. Cloud Computing
1.3 Semestre	: 10 ^{mo} Semestre.
1.4 Prerrequisitos	: CS370. Big Data. (9 ^{no} Sem)
1.5 Condición	: Obligatorio
1.6 Modalidad de aprendizaje	: Presencial
1.7 horas	: 1 HT; 2 HP; 2 HL;
1.8 Créditos	: 3
2. Profesores	
3. Fundamentación del curso	
Para entender las técnicas computacionales avanzadas, los estudiantes deberán tener un fuerte conocimiento de las diversas estructuras discretas, estructuras que serán implementadas y usadas en laboratorio en el lenguaje de programación.	
4. Resumen	
1. Sistemas distribuidos 2. Cloud Computing 3. Centros de Procesamiento de Datos 4. Cloud Computing 5. Cloud Computing 6. Modelos de Programación	
5. Objetivos Generales	
<ul style="list-style-type: none">• Que el alumno sea capaz de modelar problemas de ciencia de la computación usando grafos y árboles relacionados con estructuras de datos.• Que el alumno aplicar eficientemente estrategias de recorrido para poder buscar datos de una manera óptima.	
6. Contribución a los resultados (Outcomes)	
Esta disciplina contribuye al logro de los siguientes resultados de la carrera:	
1) Analizar un problema computacional complejo y aplicar los principios computacionales y otras disciplinas relevantes para identificar soluciones. (Usar)	
6) Aplicar fundamentos de teoría de ciencias de la computación y desarrollo de software para producir soluciones basados en computación. (Usar)	
7. Contenido	

UNIDAD 1: Sistemas distribuidos (15)	
Competencias:	
Contenido	Objetivos Generales
<ul style="list-style-type: none"> • Fallos: <ul style="list-style-type: none"> – Fallos basados en red (incluyendo particiones) y fallos basados en nodos – Impacto en garantías a nivel de sistema (p.e., disponibilidad) • Envío de mensajes distribuido: <ul style="list-style-type: none"> – Conversión y transmisión de datos – Sockets – Secuenciamiento de mensajes – Almacenando <i>Buffering</i>, reenviando y desechando mensajes • Compensaciones de diseño para Sistemas Distribuidos: <ul style="list-style-type: none"> – Latencia versus rendimiento – Consistencia, disponibilidad, tolerancia de particiones • Diseño de Servicio Distribuido: <ul style="list-style-type: none"> – Protocolos y servicios Stateful versus stateless – Diseños de Sesión (basados en la conexión) – Diseños reactivos (provocados por E/S) y diseños de múltiples hilos • Algoritmos de Distribución de Núcleos: <ul style="list-style-type: none"> – Elección, descubrimiento 	<ul style="list-style-type: none"> • Distinguir las fallas de red de otros tipos de fallas [Familiarizarse] • Explicar por qué estructuras de sincronización como cerraduras simples (<i>locks</i>) no son útiles en la presencia de fallas distribuidas [Familiarizarse] • Escribir un programa que realiza cualquier proceso de <i>marshalling</i> requerido y la conversión en unidades de mensajes, tales como paquetes, para comunicar datos importantes entre dos <i>hosts</i> [Usar] • Medir el rendimiento observado y la latencia de la respuesta a través de los <i>hosts</i> en una red dada [Usar] • Explicar por qué un sistema distribuido no puede ser simultáneamente Consistente (<i>Consistent</i>), Disponible (<i>Available</i>) y Tolerante a fallas (<i>Partition tolerant</i>). [Familiarizarse] • Implementar un servidor sencillo - por ejemplo, un servicio de corrección ortográfica [Usar] • Explicar las ventajas y desventajas entre: <i>overhead</i>, escalabilidad y tolerancia a fallas entre escoger un diseño sin estado (<i>stateless</i>) y un diseño con estado (<i>stateful</i>) para un determinado servicio [Familiarizarse] • Describir los desafíos en la escalabilidad, asociados con un servicio creciente para soportar muchos clientes, así como los asociados con un servicio que tendrá transitoriamente muchos clientes [Familiarizarse] • Dar ejemplos de problemas donde algoritmos de consenso son requeridos, por ejemplo, la elección de líder [Usar]
Lecturas: Coulouris et al. (2011)	

UNIDAD 2: Cloud Computing (15)	
Competencias:	
Contenido	Objetivos Generales
<ul style="list-style-type: none"> • Visión global de <i>Cloud Computing</i>. • Historia. • Visión global de las tecnologías que envuelve. • Beneficios, riesgos y aspectos económicos. • Servicios en la nube. <ul style="list-style-type: none"> – Infraestructura como servicio <ul style="list-style-type: none"> * Elasticidad de recursos * APIs de la Plataforma – Software como servicio – Seguridad – Administración del Costo • Computación a Escala de Internet: <ul style="list-style-type: none"> – Particionamiento de Tareas – Acceso a datos – Clusters, grids y mallas 	<ul style="list-style-type: none"> • Explicar el concepto de Cloud Computing. [Familiarizarse] • Listar algunas tecnologías relacionadas con Cloud Computing. [Familiarizarse] • Explicar las estrategias para sincronizar una vista común de datos compartidos a través de una colección de dispositivos [Familiarizarse] • Discutir las ventajas y desventajas del paradigma de Cloud Computing. [Familiarizarse] • Expresar los beneficios económicos así como las características y riesgos del paradigma de Cloud para negocios y proveedores de cloud. [Familiarizarse] • Diferenciar entre los modelos de servicio. [Usar]
Lecturas: Hwang, Dongarra, and Fox (2011), Buyya, Vecchiola, and Selvi (2013)	

UNIDAD 3: Centros de Procesamiento de Datos (10)	
Competencias:	
Contenido	Objetivos Generales
<ul style="list-style-type: none"> • Visión global de un centro de procesamiento de datos. • Consideraciones en el diseño. • Comparación de actuales grandes centros de procesamiento de datos. 	<ul style="list-style-type: none"> • Describir la evolución de los Data Centers. [Familiarizarse] • Esbozar la arquitectura de un data center en detalle. [Familiarizarse] • Indicar consideraciones de diseño y discutir su impacto. [Familiarizarse]
Lecturas: Hwang, Dongarra, and Fox (2011), Buyya, Vecchiola, and Selvi (2013)	

UNIDAD 4: Cloud Computing (20)	
Competencias:	
Contenido	Objetivos Generales
<ul style="list-style-type: none"> • Virtualización. <ul style="list-style-type: none"> – Gestión de recursos compartidos – Migración de procesos • Seguridad, recursos y aislamiento de fallas. • Almacenamiento como servicio. • Elasticidad. • Xen y VMware. • Amazon EC2. 	<ul style="list-style-type: none"> • Virtualización. <ul style="list-style-type: none"> – Gestión de recursos compartidos – Migración de procesos . [Familiarizarse] • Explicar las ventajas y desventajas de usar una infraestructura virtualizada. [Familiarizarse] • Identificar las razones por qué la virtualización está llegando a ser enormemente útil, especialmente en la cloud. [Familiarizarse] • Explicar diferentes tipos de aislamiento como falla, recursos y seguridad proporcionados por la virtualización y utilizado por la cloud. [Familiarizarse] • Explicar la complejidad que puede tener el administrar en términos de niveles de abstracción y interfaces bien definidas y su aplicabilidad para la virtualización en la cloud. [Familiarizarse] • Definir virtualización y identificar diferentes tipos de máquinas virtuales. [Familiarizarse] • Identificar condiciones de virtualización de CPU, reconocer la diferencia entre <i>full virtualization</i> y <i>paravirtualization</i>, explicar emulación como mayor técnica para virtualización del CPU y examinar planificación virtual del CPU en Xen. [Familiarizarse] • Esbozar la diferencia entre la clásica memoria virtual del SO y la virtualización de memoria. Explicar los múltiples niveles de mapeamiento de páginas en oposición a la virtualización de la memoria. Definir memoria <i>over-commitment</i> e ilustrar sobre VMware <i>memory ballooning</i> como técnica de reclamo para sistemas virtualizados con memoria <i>over-committed</i>. [Familiarizarse]
Lecturas: Hwang, Dongarra, and Fox (2011), Buyya, Vecchiola, and Selvi (2013)	

UNIDAD 5: Cloud Computing (12)	
Competencias:	
Contenido	Objetivos Generales
<ul style="list-style-type: none"> • Almacenamiento de datos en la nube: <ul style="list-style-type: none"> – Acceso compartido a data stores de consistencia débil – Sincronización de datos – Particionamiento de datos – Sistemas de Archivos Distribuidos – Replicación • Visión global sobre tecnologías de almacenamiento. • Conceptos fundamentales sobre almacenamiento en la cloud. • Amazon S3 y EBS. • Sistema de archivos distribuidos. • Sistema de bases de datos NoSQL. 	<ul style="list-style-type: none"> • Describir la organización general de datos y almacenamiento. [Familiarizarse] • Identificar los problemas de escalabilidad y administración de la big data. Discutir varias abstracciones en almacenamiento. [Familiarizarse] • Comparar y contrastar diferentes tipos de sistema de archivos. Comparar y contrastar el Sistema de Archivos Distribuido de Hadoop (HDFS) y el Sistema de Archivos Paralelo Virtual (PVFS). [Usar] • Comparar y contrastar diferentes tipos de bases de datos. Discutir las ventajas y desventajas sobre las bases de datos NoSQL. [Usar] • Discutir los conceptos de almacenamiento en la cloud. [Familiarizarse]
Lecturas: Hwang, Dongarra, and Fox (2011), Buyya, Vecchiola, and Selvi (2013)	

UNIDAD 6: Modelos de Programación (12)	
Competencias:	
Contenido	Objetivos Generales
<ul style="list-style-type: none"> • Visión global de los modelos de programación basados en cloud computing. • Modelo de Programación MapReduce. • Modelo de programación para aplicaciones basadas en Grafos. 	<ul style="list-style-type: none"> • Explicar los aspectos fundamentales de los modelos de programación paralela y distribuida. [Familiarizarse] • Diferencias entre los modelos de programación: MapReduce, Pregel, GraphLab y Giraph. [Usar] • Explicar los principales conceptos en el modelo de programación MapReduce. [Usar]
Lecturas: Hwang, Dongarra, and Fox (2011), Buyya, Vecchiola, and Selvi (2013), Low et al. (2012), Malewicz et al. (2010), Baluja et al. (2008)	

8. Metodología
<p>El profesor del curso presentará clases teóricas de los temas señalados en el programa propiciando la intervención de los alumnos.</p> <p>El profesor del curso presentará demostraciones para fundamentar clases teóricas.</p> <p>El profesor y los alumnos realizarán prácticas</p> <p>Los alumnos deberán asistir a clase habiendo leído lo que el profesor va a presentar. De esta manera se facilitará la comprensión y los estudiantes estarán en mejores condiciones de hacer consultas en clase.</p>

9. Evaluar

Evaluación Continua 1 : 20 %

Examen parcial : 30 %

Evaluación Continua 2 : 20 %

Examen final : 30 %

References

- Baluja, Shumeet et al. (2008). “Video Suggestion and Discovery for Youtube: Taking Random Walks Through the View Graph”. In: *Proceedings of the 17th International Conference on World Wide Web*. WWW '08. ACM: Beijing, China, pp. 895–904. ISBN: 978-1-60558-085-2. DOI: 10.1145/1367497.1367618.
- Buyya, Rajkumar, Christian Vecchiola, and S. Thamarai Selvi (2013). *Mastering Cloud Computing: Foundations and Applications Programming*. 1st. Morgan Kaufmann Publishers Inc.: San Francisco, CA, USA. ISBN: 9780124095397, 9780124114548.
- Coulouris, George et al. (2011). *Distributed Systems: Concepts and Design*. 5th. Addison-Wesley Publishing Company: USA. ISBN: 0132143011, 9780132143011.
- Hwang, Kai, Jack Dongarra, and Geoffrey C. Fox (2011). *Distributed and Cloud Computing: From Parallel Processing to the Internet of Things*. 1st. Morgan Kaufmann Publishers Inc.: San Francisco, CA, USA. ISBN: 0123858801, 9780123858801.
- Low, Yucheng et al. (Apr. 2012). “Distributed GraphLab: A Framework for Machine Learning and Data Mining in the Cloud”. In: *Proc. VLDB Endow.* 5(8), pp. 716–727. ISSN: 2150-8097. DOI: 10.14778/2212351.2212354.
- Malewicz, Grzegorz et al. (2010). “Pregel: A System for Large-scale Graph Processing”. In: *Proc. ACM SIGMOD*. SIGMOD '10, pp. 135–146. DOI: 10.1145/1807167.1807184.



Universidad Nacional de Colombia (UNAL) Sede
Manizales
Programa Profesional de
Administración de Sistemas Informáticos
SILABO

CS3P3. Internet de las Cosas (Obligatorio)

2022-II

1. Información general	
1.1 Escuela	: Sistemas de Información
1.2 Curso	: CS3P3. Internet de las Cosas
1.3 Semestre	: 10 ^{mo} Semestre.
1.4 Prerrequisitos	: CS3P1. Computación Paralela y Distribuída. (8 ^{vo} Sem)
1.5 Condición	: Obligatorio
1.6 Modalidad de aprendizaje	: Presencial
1.7 horas	: 1 HT; 2 HP; 2 HL;
1.8 Créditos	: 3
2. Profesores	
3. Fundamentación del curso	
La última década ha traído un crecimiento explosivo en computación con multiprocesadores, incluyendo los procesadores de varios núcleos y centros de datos distribuidos. Como resultado, la computación paralela y distribuida se ha convertido de ser un tema ampliamente electivo para ser uno de los principales componentes en la malla estudios en ciencia de la computación de pregrado. Tanto la computación paralela como la distribuida implica la ejecución simultánea de múltiples procesos en diferentes dispositivos que cambian de posición.	
4. Resumen	
1. Fundamentos de paralelismo 2. Arquitecturas paralelas 3. Descomposición en paralelo 4. Comunicación y coordinación 5. Análisis y programación de algoritmos paralelos 6. Desempeño en paralelo	
5. Objetivos Generales	
<ul style="list-style-type: none">• Que el alumno sea capaz de crear aplicaciones paralelas de mediana complejidad aprovechando eficientemente distintos dispositivos móviles.	
6. Contribución a los resultados (Outcomes)	
Esta disciplina contribuye al logro de los siguientes resultados de la carrera:	
1) Analizar un problema computacional complejo y aplicar los principios computacionales y otras disciplinas relevantes para identificar soluciones. (Usar)	
2) Diseñar, implementar y evaluar una solución basada en computación para cumplir con un conjunto determinado de requisitos computacionales en el contexto de las disciplinas del programa. (Usar)	
6) Aplicar fundamentos de teoría de ciencias de la computación y desarrollo de software para producir soluciones basados en computación. (Usar)	
7. Contenido	

UNIDAD 1: Fundamentos de paralelismo (18)	
Competencias:	
Contenido	Objetivos Generales
<ul style="list-style-type: none"> • Procesamiento Simultáneo Múltiple. • Metas del Paralelismo (ej. rendimiento) frente a Concurrencia (ej. control de acceso a recursos compartidos) • Paralelismo, comunicación, y coordinación: <ul style="list-style-type: none"> – Paralelismo, comunicación, y coordinación – Necesidad de Sincronización • Errores de Programación ausentes en programación secuencial: <ul style="list-style-type: none"> – Tipos de Datos (lectura/escritura simultánea o escritura/escritura compartida) – Tipos de Nivel más alto (interleavings violating program intention, no determinismo no deseado) – Falta de vida/progreso (deadlock, starvation) 	<ul style="list-style-type: none"> • Distinguir el uso de recursos computacionales para una respuesta mas rápida para administrar el acceso eficiente a un recurso compartido [Familiarizarse] • Distinguir múltiples estructuras de programación suficientes para la sincronización que pueden ser interimplementables pero tienen ventajas complementarias [Familiarizarse] • Distinguir datos de carrera (<i>data races</i>) a partir de carreras de mas alto nivel [Familiarizarse]
Lecturas: Pacheco (2011), Matloff (2014), Quinn (2003)	

UNIDAD 2: Arquitecturas paralelas (12)	
Competencias:	
Contenido	Objetivos Generales
<ul style="list-style-type: none"> • Procesadores mutlinúcleo. • Memoria compartida vs memoria distribuida. • Multiprocesamiento simétrico. • SIMD, procesamiento de vectores. • GPU, coprocesamiento. • Taxonomía de Flynn. • Soporte a nivel de instrucciones para programación paralela. <ul style="list-style-type: none"> – Instrucciones atómicas como Compare/Set (Comparar / Establecer) • Problemas de Memoria: <ul style="list-style-type: none"> – Caches multiprocesador y coherencia de cache – Acceso a Memoria no uniforme (NUMA) • Topologías. <ul style="list-style-type: none"> – Interconexiones – Clusters – Compartir recursos (p.e., buses e interconexiones) 	<ul style="list-style-type: none"> • Explicar las diferencias entre memoria distribuida y memoria compartida [Evaluar] • Describir la arquitectura SMP y observar sus principales características [Evaluar] • Distinguir los tipos de tareas que son adecuadas para máquinas SIMD [Usar] • Describir las ventajas y limitaciones de GPUs vs CPUs [Usar] • Explicar las características de cada clasificación en la taxonomía de Flynn [Usar] • Describir los desafíos para mantener la coherencia de la caché [Familiarizarse] • Describir los desafíos clave del desempeño en diferentes memorias y topologías de sistemas distribuidos [Familiarizarse]
Lecturas: Pacheco (2011), Kirk and Hwu (2013), Sanders and Kandrot (2010)	

UNIDAD 3: Descomposición en paralelo (18)	
Competencias:	
Contenido	Objetivos Generales
<ul style="list-style-type: none"> • Necesidad de Comunicación y coordinación/sincronización. • Independencia y Particionamiento. • Conocimiento Básico del Concepto de Descomposición Paralela. • Descomposición basada en tareas: <ul style="list-style-type: none"> – Implementación de estrategias como hebras • Descomposición de Información Paralela <ul style="list-style-type: none"> – Estrategias como SIMD y MapReduce • Actores y Procesos Reactivos (solicitud de gestores) 	<ul style="list-style-type: none"> • Explicar por qué la sincronización es necesaria en un programa paralelo específico [Usar] • Identificar oportunidades para particionar un programa serial en módulos paralelos independientes [Familiarizarse] • Escribir un algoritmo paralelo correcto y escalable [Usar] • Paralelizar un algoritmo mediante la aplicación de descomposición basada en tareas [Usar] • Paralelizar un algoritmo mediante la aplicación de descomposición datos en paralelo [Usar] • Escribir un programa usando actores y/o procesos reactivos [Usar]
Lecturas: Pacheco (2011), Matloff (2014), Quinn (2003)	

UNIDAD 4: Comunicación y coordinación (18)	
Competencias:	
Contenido	Objetivos Generales
<ul style="list-style-type: none"> • Memoria Compartida. • La consistencia, y su papel en los lenguaje de programación garantias para los programas de carrera libre. • Pasos de Mensaje: <ul style="list-style-type: none"> – Mensajes Punto a Punto versus multicast (o basados en eventos) – Estilos para enviar y recibir mensajes Blocking vs non-blocking – Buffering de mensajes • Atomicidad: <ul style="list-style-type: none"> – Especificar y probar atomicidad y requerimientos de seguridad – Granularidad de accesos atómicos y actualizaciones, y uso de estructuras como secciones críticas o transacciones para describirlas – Exclusión mutua usando bloques, semáforos, monitores o estructuras relacionadas <ul style="list-style-type: none"> * Potencial para fallas y bloqueos (<i>deadlock</i>) (causas, condiciones, prevención) – Composición <ul style="list-style-type: none"> * Componiendo acciones atómicas granulares más grandes usando sincronización * Transacciones, incluyendo enfoques optimistas y conservadores • Consensos: <ul style="list-style-type: none"> – (Cíclicos) barreras, contadores y estructuras relacionadas • Acciones condicionales: <ul style="list-style-type: none"> – Espera condicional (p.e., empleando variables de condición) 	<ul style="list-style-type: none"> • Usar exclusión mútua para evitar una condición de carrera [Usar] • Dar un ejemplo de una ordenación de accesos entre actividades concurrentes (por ejemplo, un programa con condición de carrera) que no son secuencialmente consistentes [Familiarizarse] • Dar un ejemplo de un escenario en el que el bloqueo de mensajes enviados pueden dar <i>deadlock</i> [Usar] • Explicar cuándo y por qué mensajes de multidifusión (<i>multicast</i>) o basado en eventos puede ser preferible a otras alternativas [Familiarizarse] • Escribir un programa que termine correctamente cuando todo el conjunto de procesos concurrentes hayan sido completados [Usar] • Dar un ejemplo de un escenario en el que un intento optimista de actualización puede nunca completarse [Familiarizarse] • Usar semaforos o variables de condición para bloquear hebras hasta una necesaria precondition de mantenga [Usar]
Lecturas: Pacheco (2011), Matloff (2014), Quinn (2003)	

UNIDAD 5: Análisis y programación de algoritmos paralelos (18)**Competencias:****Contenido****Objetivos Generales**

- Caminos críticos, el trabajo y la duración y la relación con la ley de Amdahl.
- Aceleración y escalabilidad.
- Naturalmente (vergonzosamente) algoritmos paralelos.
- Patrones Algoritmicos paralelos (divide-y-conquista, map/reduce, amos-trabajadores, otros)
 - Algoritmos específicos (p.e., MergeSort paralelo)
- Algoritmos de grafos paralelo (por ejemplo, la ruta más corta en paralelo, árbol de expansión paralela)
- Cálculos de matriz paralelas.
- Productor-consumidor y algoritmos paralelos segmentados.
- Ejemplos de algoritmos paralelos no-escalables.

- Definir: camino crítico, trabajo y *span* [Familiarizarse]
- Calcular el trabajo y el *span* y determinar el camino crítico con respecto a un diagrama de ejecución paralela. [Usar]
- Definir *speed-up* y explicar la noción de escalabilidad de un algoritmo en este sentido [Familiarizarse]
- Identificar tareas independientes en un programa que debe ser paralelizado [Usar]
- Representar características de una carga de trabajo que permita o evite que sea naturalmente paralelizable [Familiarizarse]
- Implementar un algoritmo dividir y conquistar paralelo (y/o algoritmo de un grafo) y medir empíricamente su desempeño relativo a su analogo secuencial [Usar]
- Descomponer un problema (por ejemplo, contar el número de ocurrencias de una palabra en un documento) via operaciones *map* y *reduce* [Usar]
- Proporcionar un ejemplo de un problema que se corresponda con el paradigma productor-consumidor [Usar]
- Dar ejemplos de problemas donde el uso de *pipelining* sería un medio eficaz para la paralelización [Usar]
- Implementar un algoritmo de matriz paralela [Usar]
- Identificar los problemas que surgen en los algoritmos del tipo productor-consumidor y los mecanismos que pueden utilizarse para superar dichos problemas [Usar]

Lecturas: Matloff (2014), Quinn (2003)

UNIDAD 6: Desempeño en paralelo (18)	
Competencias:	
Contenido	Objetivos Generales
<ul style="list-style-type: none"> • Equilibrio de carga. • La medición del desempeño. • Programación y contención. • Evaluación de la comunicación de arriba. • Gestión de datos: <ul style="list-style-type: none"> – Costos de comunicación no uniforme debidos a proximidad – Efectos de Cache (p.e., false sharing) – Manteniendo localidad espacial • Consumo de energía y gestión. 	<ul style="list-style-type: none"> • Detectar y corregir un desbalanceo de carga [Usar] • Calcular las implicaciones de la ley de Amdahl para un algoritmo paralelo particular [Usar] • Describir como la distribución/disposición de datos puede afectar a los costos de comunicación de un algoritmo [Familiarizarse] • Detectar y corregir una instancia de uso compartido falso (<i>false sharing</i>) [Usar] • Explicar el impacto de la planificación en el desempeño paralelo [Familiarizarse] • Explicar el impacto en el desempeño de la localidad de datos [Familiarizarse] • Explicar el impacto y los puntos de equilibrio relacionados al uso de energía en el desempeño paralelo [Familiarizarse]
Lecturas: Pacheco (2011), Matloff (2014), Kirk and Hwu (2013), Sanders and Kandrot (2010)	

8. Metodología
<p>El profesor del curso presentará clases teóricas de los temas señalados en el programa propiciando la intervención de los alumnos.</p> <p>El profesor del curso presentará demostraciones para fundamentar clases teóricas.</p> <p>El profesor y los alumnos realizarán prácticas</p> <p>Los alumnos deberán asistir a clase habiendo leído lo que el profesor va a presentar. De esta manera se facilitará la comprensión y los estudiantes estarán en mejores condiciones de hacer consultas en clase.</p>

9. Evaluar
<p>Evaluación Continua 1 : 20 %</p> <p>Examen parcial : 30 %</p> <p>Evaluación Continua 2 : 20 %</p> <p>Examen final : 30 %</p>

References

- Kirk, David B. and Wen-mei W. Hwu (2013). *Programming Massively Parallel Processors: A Hands-on Approach*. 2nd. Morgan Kaufmann. ISBN: 978-0-12-415992-1.
- Matloff, Norm (2014). *Programming on Parallel Machines*. University of California, Davis.
- Pacheco, Peter S. (2011). *An Introduction to Parallel Programming*. 1st. Morgan Kaufmann. ISBN: 978-0-12-374260-5.
- Quinn, Michael J. (2003). *Parallel Programming in C with MPI and OpenMP*. 1st. McGraw-Hill Education Group. ISBN: 0071232656.
- Sanders, Jason and Edward Kandrot (2010). *CUDA by Example: An Introduction to General-Purpose GPU Programming*. 1st. Addison-Wesley Professional. ISBN: 0131387685, 9780131387683.



Universidad Nacional de Colombia (UNAL) Sede
Manizales
Programa Profesional de
Administración de Sistemas Informáticos
SILABO

CS404. Proyecto de Final de Carrera III (Obligatorio)

2022-II

1. Información general	
1.1 Escuela	: Sistemas de Información
1.2 Curso	: CS404. Proyecto de Final de Carrera III
1.3 Semestre	: 10 ^{mo} Semestre.
1.4 Prerrequisitos	: CS403. Proyecto de Final de Carrera II. (9 ^{no} Sem)
1.5 Condición	: Obligatorio
1.6 Modalidad de aprendizaje	: Presencial
1.7 horas	: 2 HT; 8 HP;
1.8 Créditos	: 6
2. Profesores	
3. Fundamentación del curso	
Este curso tiene por objetivo que el alumno logre finalizar adecuadamente su borrador de tesis.	
4. Resumen	
1. Escritura del Borrador del trabajo de final de carrera (tesis)	
5. Objetivos Generales	
<ul style="list-style-type: none">• Que el alumno complete este curso con su tesis elaborada en calidad suficiente como para una inmediata sustentación.• Que el alumno presente formalmente el borrador de tesis ante las autoridades de la facultad.• Los entregables de este curso son: Parcial: Avance del proyecto de tesis incluyendo en el documento: introducción, marco teórico, estado del arte, propuesta, análisis y/o experimentos y bibliografía sólida. Final: Documento de tesis completo y listo para sustentar en un plazo no mayor de quince días.	

6. Contribución a los resultados (*Outcomes*)

Esta disciplina contribuye al logro de los siguientes resultados de la carrera:

- 1) Analizar un problema computacional complejo y aplicar los principios computacionales y otras disciplinas relevantes para identificar soluciones. (**Evaluar**)
- 2) Diseñar, implementar y evaluar una solución basada en computación para cumplir con un conjunto determinado de requisitos computacionales en el contexto de las disciplinas del programa. (**Evaluar**)
- 3) Comunicarse efectivamente en diversos contextos profesionales. (**Evaluar**)
- 4) Reconocer las responsabilidades profesionales y hacer juicios informados en el campo profesional de computación con principios éticos. (**Evaluar**)
- 5) Funcionar efectivamente como miembro o líder de un equipo involucrado en actividades apropiadas a la disciplina del programa. (**Evaluar**)
- 6) Aplicar fundamentos de teoría de ciencias de la computación y desarrollo de software para producir soluciones basados en computación. (**Evaluar**)
- 7) Desarrollar tecnología computacional buscando el bien común, aportando con formación humana, capacidades científicas, tecnológicas y profesionales para solucionar problemas sociales de nuestro entorno. (**Evaluar**)

7. Contenido

UNIDAD 1: Escritura del Borrador del trabajo de final de carrera (tesis) (60)

Competencias:

Contenido	Objetivos Generales
<ul style="list-style-type: none">• Redacción y corrección del trabajo de final de carrera	<ul style="list-style-type: none">• Parte experimental concluida (si fuese adecuado al proyecto) [Evaluar]• Verificar que el documento cumpla con el formato de tesis de la carrera [Evaluar]• Entrega del borrador de tesis finalizado y considerado listo para una sustentación pública del mismo (requisito de aprobación) [Evaluar]

Lecturas: IEEE-Computer Society (2008), Association for Computing Machinery (2008), CiteSeer.IST (2008)

8. Metodología

El profesor del curso presentará clases teóricas de los temas señalados en el programa propiciando la intervención de los alumnos.

El profesor del curso presentará demostraciones para fundamentar clases teóricas.

El profesor y los alumnos realizarán prácticas

Los alumnos deberán asistir a clase habiendo leído lo que el profesor va a presentar. De esta manera se facilitará la comprensión y los estudiantes estarán en mejores condiciones de hacer consultas en clase.

9. Evaluar

Evaluación Continua 1 : 20 %

Examen parcial : 30 %

Evaluación Continua 2 : 20 %

Examen final : 30 %

References

- Association for Computing Machinery (2008). *Digital Libray*. <http://portal.acm.org/dl.cfm>. Association for Computing Machinery.
- CiteSeer.IST (2008). *Scientific Literature Digital Libray*. <http://citeseer.ist.psu.edu>. College of Information Sciences and Technology, Penn State University.
- IEEE-Computer Society (2008). *Digital Libray*. <http://www.computer.org/publications/dlib>. IEEE-Computer Society.



Universidad Nacional de Colombia (UNAL) Sede
Manizales
Programa Profesional de
Administración de Sistemas Informáticos
SILABO

CS362. Robótica (Electivo)

2022-II

1. Información general	
1.1 Escuela	: Sistemas de Información
1.2 Curso	: CS362. Robótica
1.3 Semestre	: 10 ^{mo} Semestre.
1.4 Prerrequisitos	: CS361. Tópicos en Inteligencia Artificial. (9 ^{no} Sem)
1.5 Condición	: Electivo
1.6 Modalidad de aprendizaje	: Presencial
1.7 horas	: 2 HT; 4 HL;
1.8 Créditos	: 4
2. Profesores	
3. Fundamentación del curso	
Que el alumno conozca y comprenda los conceptos y principios fundamentales de control, planificación de caminos y definición de estrategias en robótica móvil así como conceptos de percepción robótica de forma que entienda el potencial de los sistemas robóticos actuales	
4. Resumen	
1. Robótica 2. Robótica 3. Robótica 4. Visión y percepción por computador 5. Robótica	
5. Objetivos Generales	
<ul style="list-style-type: none">• Sintetizar el potencial y las limitaciones del estado del arte de los sistemas robóticos actuales.• Implementar algoritmos de planeamiento de movimientos simples.• Explicar las incertezas asociadas con sensores y la forma de tratarlas.• Diseñar una arquitectura de control simple• Describir varias estrategias de navegación• Entender el rol y las aplicaciones de la percepción robótica• Describir la importancia del reconocimiento de imágenes y objetos en sistemas inteligentes• Delinear las principales técnicas de reconocimiento de objetos• Describir las diferentes características de las tecnologías usadas en percepción	

6. Contribución a los resultados (Outcomes)

Esta disciplina contribuye al logro de los siguientes resultados de la carrera:

- 1) Analizar un problema computacional complejo y aplicar los principios computacionales y otras disciplinas relevantes para identificar soluciones. (**Usar**)
- 5) Funcionar efectivamente como miembro o líder de un equipo involucrado en actividades apropiadas a la disciplina del programa. (**Usar**)
- 6) Aplicar fundamentos de teoría de ciencias de la computación y desarrollo de software para producir soluciones basados en computación. (**Usar**)

7. Contenido**UNIDAD 1: Robótica (5)****Competencias:****Contenido****Objetivos Generales**

- Vision general: problemas y progreso
 - Estado del arte de los sistemas robóticos, incluyendo sus sensores y una visión general de su procesamiento
 - Arquitecturas de control robótico, ejem., deliverado vs. control reactivo y vehiculos Braitenberg
 - Modelando el mundo y modelos de mundo
 - Incertidumbre inherente en detección y control
- Configuración de espacio y mapas de entorno.

- Listar capacidades y limitaciones de sistemas del estado del arte en robótica de hoy , incluyendo sus sensores y el procesamiento del sensor crucial que informa a esos sistemas [Familiarizarse]
- Integrar sensores, actuadores y software en un robot diseñado para emprender alguna tarea [Usar]

Lecturas: Siegwart and Nourbakhsh (2004), S, W, and D (2005), Stone (2000)

UNIDAD 2: Robótica (15)**Competencias:****Contenido****Objetivos Generales**

- Interpretando datos del sensor con incertidumbre.
- Localización y mapeo.

- Programar un robot para llevar a cabo tareas simples usando arquitecturas de control deliverativo, reactivo y/o híbrido [Usar]
- Implementar algoritmos de planificación de movimientos fundamentales dentro del espacio de configuración de un robot [Usar]

Lecturas: Siegwart and Nourbakhsh (2004), S, W, and D (2005)

UNIDAD 3: Robótica (20)	
Competencias:	
Contenido	Objetivos Generales
<ul style="list-style-type: none"> • Navegación y control. • Planeando el movimiento. 	<ul style="list-style-type: none"> • Caracterizar las incertidumbres asociadas con sensores y actuadores de robot comunes; articular estrategias para mitigar esas incertidumbres. [Usar] • Listar las diferencias entre representaciones de los robot de su entorno externo, incluyendo sus fortalezas y defectos [Usar]
Lecturas: Siegwart and Nourbakhsh (2004)	

UNIDAD 4: Visión y percepción por computador (10)	
Competencias:	
Contenido	Objetivos Generales
<ul style="list-style-type: none"> • Visión Computacional <ul style="list-style-type: none"> – Adquisición de imágenes, representación, procesamiento y propiedades – Representación de formas, reconocimiento y segmentación de objetos – Análisis de movimiento • Modularidad en reconocimiento. 	<ul style="list-style-type: none"> • Resumir la importancia del reconocimiento de imágenes y objetos en Inteligencia Artificial (AI) e indicar varias aplicaciones significativas de esta tecnología [Usar] • Implementar reconocimiento de objetos en 2d basados en la representación del contorno y/o regiones basadas en formas [Usar]
Lecturas: M, V, and B. R (2007), G. R. C. and W. R. E. (2007)	

UNIDAD 5: Robótica (10)	
Competencias:	
Contenido	Objetivos Generales
<ul style="list-style-type: none"> • Coordinación multi-robots. 	<ul style="list-style-type: none"> • Comparar y contrastar al menos tres estrategias para la navegación de robots dentro de entornos conocidos y/o no conocidos, incluyendo sus fortalezas y defectos [Familiarizarse] • Describir al menos una aproximación para la coordinación de acciones y detección de varios robots para realizar una simple tarea [Familiarizarse]
Lecturas: Stone (2000)	

8. Metodología
<p>El profesor del curso presentará clases teóricas de los temas señalados en el programa propiciando la intervención de los alumnos.</p> <p>El profesor del curso presentará demostraciones para fundamentar clases teóricas.</p> <p>El profesor y los alumnos realizarán prácticas</p> <p>Los alumnos deberán asistir a clase habiendo leído lo que el profesor va a presentar. De esta manera se facilitará la comprensión y los estudiantes estarán en mejores condiciones de hacer consultas en clase.</p>

9. Evaluar

Evaluación Continua 1 : 20 %

Examen parcial : 30 %

Evaluación Continua 2 : 20 %

Examen final : 30 %

References

- M, Sonka., Hlavac. V, and Boile. R (2007). *Image Processing, Analysis and Machine Vision*. Cengage-Engineering.
- R C, Gonzales. and Woods. R E (2007). *Digital Image Processing*. Prentice Hall. ISBN: 013168728X,978013168728B.
- S, Thrun., Burgard. W, and Fox. D (2005). *Probabilistic Robotics*. Intelligent Robots and Autonomous Agents. The MIT Press.
- Siegwart, R. and I. Nourbakhsh (2004). *Introduction to Autonomous Mobile Robots*. The MIT Press. ISBN: 0-262-19502-X.
- Stone, Peter (2000). *Layered Learning in Multiagent Systems*. Intelligent Robots and Autonomous Agents. The MIT Press. ISBN: 9780262194389.



Universidad Nacional de Colombia (UNAL) Sede
Manizales
Programa Profesional de
Administración de Sistemas Informáticos
SILABO

FG211. Ética Profesional (Obligatorio)

2022-II

1. Información general	
1.1 Escuela	: Sistemas de Información
1.2 Curso	: FG211. Ética Profesional
1.3 Semestre	: 10 ^{mo} Semestre.
1.4 Prerrequisitos	: Ninguno
1.5 Condición	: Obligatorio
1.6 Modalidad de aprendizaje	: Presencial
1.7 horas	: 2 HT; 2 HP;
1.8 Créditos	: 3
2. Profesores	
3. Fundamentación del curso	
La ética es una parte constitutiva inherente al ser humano, y como tal debe plasmarse en el actuar cotidiano y profesional de la persona humana. Es indispensable que la persona asuma su rol activo en la sociedad pues los sistemas económico-industrial, político y social no siempre están en función de valores y principios, siendo éstos en realidad los pilares sobre los que debería basarse todo el actuar de los profesionales.	
4. Resumen	
1. Objetividad moral 2. 3. 4.	
5. Objetivos Generales	
<ul style="list-style-type: none">• Que el alumno amplíe sus propios criterios personales de discernimiento moral en el quehacer profesional, de forma que no sólo tome en cuenta los criterios técnicos pertinentes sino que incorpore a sí mismo cuestionamientos de orden moral y se adhiera a una ética profesional correcta, de forma que sea capaz de aportar positivamente en el desarrollo económico y social de la ciudad, región, país y comunidad global.[Usar]	
6. Contribución a los resultados (Outcomes)	
Esta disciplina contribuye al logro de los siguientes resultados de la carrera:	
4) Reconocer las responsabilidades profesionales y hacer juicios informados en el campo profesional de computación con principios éticos. (Usar)	
6) Aplicar fundamentos de teoría de ciencias de la computación y desarrollo de software para producir soluciones basados en computación. (Usar)	
7) Desarrollar tecnología computacional buscando el bien común, aportando con formación humana, capacidades científicas, tecnológicas y profesionales para solucionar problemas sociales de nuestro entorno. (Usar)	
7. Contenido	

UNIDAD 1: Objetividad moral (12)	
Competencias:	
Contenido	Objetivos Generales
<ul style="list-style-type: none"> • Ser profesional y ser moral. • La objetividad moral y la formulación de principios morales. • El profesional y sus valores. • La conciencia moral de la persona. • El aporte de la DSI en el quehacer profesional. • El bien común y el principio de subsidiaridad. • Principios morales y propiedad privada. • Justicia: Algunos conceptos básicos. 	<ul style="list-style-type: none"> • Presentar al alumno la importancia de tener principios y valores en la sociedad actual.[Usar] • Presentar algunos de los principios de podrían contribuir en la sociedad de ser aplicados y vividos día a día. [Usar] • Presentar a los alumnos el aporte de la Doctrina Social de la Iglesia en el quehacer profesional. [Usar]
Lecturas: Computing Machinery (ACM) (1992), Schmidt (1995), Loza (2000), Argandoña (2006)	

UNIDAD 2: (12)	
Competencias:	
Contenido	Objetivos Generales
<ul style="list-style-type: none"> • La responsabilidad individual del trabajador en la empresa. • Liderazgo y ética profesional en el entorno laboral. • Principios generales sobre la colaboración en hechos inmorales. • El profesional frente al soborno: ¿víctima o colaboración? 	<ul style="list-style-type: none"> • Presentar al alumno el rol de la responsabilidad social individual y del liderazgo en la empresa. [Familiarizarse] • Conocer el juicio de la ética frente a la corrupción y sobornos como forma de relación laboral. [Familiarizarse] • Presentar la profesión como una forma de realización personal, y como consecuencia. []
Lecturas: Computing Machinery (ACM) (1992), Manzone (2007), Schmidt (1995), Pérez López (1998), Nieburh (2003)	

UNIDAD 3: (12)	
Competencias:	
Contenido	Objetivos Generales
<ul style="list-style-type: none"> • La ética profesional frente a la ética general. • Trabajo y profesión en los tiempos actuales. • Ética, ciencia y tecnología. • Valores éticos en organizaciones relacionadas con el uso de la información. • Valores éticos en la era de la Sociedad de la Información. 	<ul style="list-style-type: none"> • Presentar al alumno las interrelaciones entre ética y las disciplinas de la última era tecnológica.[Familiarizarse]
Lecturas: Computing Machinery (ACM) (1992), IEEE (2004), Hernández (2006)	

UNIDAD 4: (12)	
Competencias:	
Contenido	Objetivos Generales
<ul style="list-style-type: none"> • Ética informática. <ul style="list-style-type: none"> – Ética y software. – El software libre. • Regulación y ética de telecomunicaciones. <ul style="list-style-type: none"> – Ética en Internet. • Derechos de autor y patentes. • Ética en los servicios de consultoría. • Ética en los procesos de innovación tecnológica. • Ética en la gestión tecnológica y en empresas de base tecnológica. 	<ul style="list-style-type: none"> • Presentar al alumno algunos aspectos que confrontan la ética con el quehacer de las disciplinas emergentes en la sociedad de la información.[Familiarizarse]
Lecturas: Comunicaciones Sociales (2002), Hernández (2006), Computing Machinery (ACM) (1992)	

8. Metodología

El profesor del curso presentará clases teóricas de los temas señalados en el programa propiciando la intervención de los alumnos.

El profesor del curso presentará demostraciones para fundamentar clases teóricas.

El profesor y los alumnos realizarán prácticas

Los alumnos deberán asistir a clase habiendo leído lo que el profesor va a presentar. De esta manera se facilitará la comprensión y los estudiantes estarán en mejores condiciones de hacer consultas en clase.

9. Evaluar

Evaluación Continua 1 : 20 %

Examen parcial : 30 %

Evaluación Continua 2 : 20 %

Examen final : 30 %

References

- Argandoña (2006). "La identidad Cristiana del Directivo de Empresa". In: *IESE*.
Computing Machinery (ACM), Association for (1992). "ACM Code of Ethics and Professional Conduct". In. *Comunicaciones Sociales*, Pontificio Consejo para las (2002). *Ética en Internet*.
Hernández, A. (2006). *Ética Actual y Profesional. Lecturas para la Convivencia Global en el Siglo XXI*. Ed. Thomson.
IEEE (2004). "IEEE Code of Ethics". In: *IEE*.
Loza, C (2000). "El aporte de la Doctrina Social de la Iglesia a la Toma de Decisiones Empresariales". In: *Separata ofrecida por el profesor*.
Manzone, G. (2007). *La Responsabilidad de la Empresa, Business Ethics y Doctrina Social de la Iglesia en Diálogo*. Universidad Católica San Pablo.
Nieburh, R. (2003). *El Yo Responsable. Ensayo de Filosofía Moral Cristiana*. Bilbao.
Pérez López, J. A. (1998). *Liderazgo y Ética en la Dirección de Empresas*. Bilbao.
Schmidt, E. (1995). *Ética y Negocios para América Latina*. Universidad del Pacífico.