

## 1. CURSO

CS111. Introducción a la programación (Obligatorio)

## 2. INFORMACIÓN GENERAL

- 2.1 Curso : CS111. Introducción a la programación  
 2.2 Semestre : 1<sup>er</sup> Semestre.  
 2.3 Créditos : 3  
 2.4 horas : 3 HT; 3 HP;  
 2.5 Duración del periodo : 16 semanas  
 2.6 Condición : Obligatorio  
 2.7 Modalidad de aprendizaje : Presencial  
 2.8 Prerrequisitos : Ninguno Ninguno

## 3. PROFESORES

Atención previa coordinación con el profesor

## 4. INTRODUCCIÓN AL 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. Lo tópicos incluyen tipos de datos, estructuras de control, funciones, listas, recursividad y la mecánica de la ejecución, prueba y depuración.

## 5. OBJETIVOS

- Introducir los conceptos fundamentales de programación.
- Desarrollar su capacidad de abstracción utilizar un lenguaje de programación.

## 6. RESULTADOS DEL ESTUDIANTE

- 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 la teoría de la computación y los fundamentos del desarrollo de software para producir soluciones basadas en computación. (**Usar**)

## 7. TEMAS

Unidad 1: Historia de los Lenguajes de Programación (5)	
Resultados esperados:	
Temas	Objetivos de Aprendizaje
<ul style="list-style-type: none"> <li>• Perspectiva histórica de los lenguajes de programación.</li> <li>• Conceptos de Programación tradicionales.</li> </ul>	<ul style="list-style-type: none"> <li>• Identificar importantes tendencias en la historia del campo de la computación [Familiarizarse]</li> <li>• Discutir el contexto histórico de los paradigmas de diversos lenguajes de programación [Familiarizarse]</li> </ul>
Lecturas : [BB19]	

Unidad 2: Conceptos Fundamentales de Programación (9)	
Resultados esperados: 1	
Temas	Objetivos de Aprendizaje
<ul style="list-style-type: none"> <li>• Sintaxis y semántica básica de un lenguaje de alto nivel.</li> <li>• Variables y tipos de datos primitivos (ej., números, caracteres, booleanos)</li> <li>• Expresiones y asignaciones.</li> <li>• Operaciones básicas I/O.</li> <li>• Estructuras de control condicional e iterativas.</li> <li>• Funciones definidas por el usuario.</li> <li>• Paso de funciones y parámetros.</li> <li>• Concepto de recursividad.</li> </ul>	<ul style="list-style-type: none"> <li>• 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]</li> <li>• Identifica y describe el uso de tipos de datos primitivos [Familiarizarse]</li> <li>• Escribe programas que usan tipos de datos primitivos [Usar]</li> <li>• Modifica y expande programas cortos que usen estructuras de control condicionales e iterativas así como funciones [Usar]</li> <li>• 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]</li> <li>• Elige estructuras de condición y repetición adecuadas para una tarea de programación dada [Familiarizarse]</li> <li>• Describe el concepto de recursividad y da ejemplos de su uso [Evaluar]</li> <li>• Identifica el caso base y el caso general de un problema basado en recursividad [Familiarizarse]</li> </ul>
<b>Lecturas :</b> [Gut13], [Zel10]	

Unidad 3: Algoritmos y Estructuras de Datos fundamentales (8)	
Resultados esperados: 1	
Temas	Objetivos de Aprendizaje
<ul style="list-style-type: none"> <li>• Algoritmos numéricos simples, tales como el cálculo de la media de una lista de números, encontrar el mínimo y máximo.</li> <li>• Algoritmos de ordenamiento de peor caso cuadrático (selección, inserción)</li> <li>• Algoritmos de ordenamiento con peor caso o caso promedio en <math>O(N \lg N)</math></li> <li>• Algoritmos de búsqueda secuencial y binaria.</li> </ul>	<ul style="list-style-type: none"> <li>• Implementar algoritmos numéricos básicos [Usar]</li> <li>• Implementar algoritmos de búsqueda simple y explicar las diferencias en sus tiempos de complejidad [Evaluar]</li> <li>• Ser capaz de implementar algoritmos de ordenamiento comunes cuadráticos y <math>O(N \log N)</math> [Usar]</li> <li>• Discutir el tiempo de ejecución y eficiencia de memoria de los principales algoritmos de ordenamiento y búsqueda.</li> <li>• 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]</li> <li>• 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]</li> <li>• Trazar y/o implementar un algoritmo de comparación de string [Usar]</li> </ul>
<b>Lecturas :</b> [Gut13], [Zel10]	

Unidad 4: Programación orientada a objetos (4)	
Resultados esperados: 1	
Temas	Objetivos de Aprendizaje
<ul style="list-style-type: none"> <li>• Lenguajes orientados a objetos para la encapsulación: Privacidad y visibilidad de miembros de la clase.</li> <li>• Definición de las categorías, campos, métodos y constructores.</li> <li>• Subclases y herencia.</li> <li>• Asignación dinámica: definición de método de llamada.</li> </ul>	<ul style="list-style-type: none"> <li>• Diseñar e implementar una clase [Usar]</li> <li>• Usar subclase para diseñar una jerarquía simple de clases que permita al código ser reusable por diferentes subclases [Familiarizarse]</li> <li>• Comparar y contrastar (1) el enfoque procedurar/funcional- definiendo una función por cada operación con el cuerdo 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 [Familiarizarse]</li> <li>• Explicar la relación entre la herencia orientada a objetos (codigo compartido y <i>overriding</i>) y subtipificación (la idea de un subtipo es ser utilizable en un contexto en el que espera al supertipo) [Familiarizarse]</li> <li>• Usar mecanismos de encapsulación orientada a objetos [Familiarizarse].</li> </ul>
Lecturas : [Gut13], [Zel10]	

Unidad 5: Programación de Video Juegos ()	
Resultados esperados: 1	
Temas	Objetivos de Aprendizaje
<ul style="list-style-type: none"> <li>• Uso de una librería de video juegos.</li> </ul>	<ul style="list-style-type: none"> <li>• Aplicar los fundamentos y los conceptos de lenguajes de programación para el desarrollo de un video juego simple. [Usar]</li> </ul>
Lecturas : [sweigart2012making]	

## 8. PLAN DE TRABAJO

### 8.1 Metodología

Se fomenta la participación individual y en equipo para exponer sus ideas, motivándolos con puntos adicionales en las diferentes etapas de la evaluación del curso.

### 8.2 Sesiones Teóricas

Las sesiones de teoría se llevan a cabo en clases magistrales donde se realizarán actividades que propicien un aprendizaje activo, con dinámicas que permitan a los estudiantes interiorizar los conceptos.

### 8.3 Sesiones Prácticas

Las sesiones prácticas se llevan en clase donde se desarrollan una serie de ejercicios y/o conceptos prácticos mediante planteamiento de problemas, la resolución de problemas, ejercicios puntuales y/o en contextos aplicativos.

## 9. SISTEMA DE EVALUACIÓN

\*\*\*\*\* EVALUATION MISSING \*\*\*\*\*

## 10. BIBLIOGRAFÍA BÁSICA

- [BB19] J. Glenn Brookshear and Dennis Brylow. *Computer Science: An Overview*. Ed. by PEARSON. Global Edition. Pearson, 2019. ISBN: 1292263423. URL: <http://www.pearsonhighered.com/brookshear>.
- [Gut13] John V Guttag. . *Introduction To Computation And Programming Using Python*. MIT Press, 2013.
- [Zel10] John Zelle. *Python Programming: An Introduction to Computer Science*. Franklin, Beedle & Associates Inc, 2010.