UNIVERSIDAD
NACIONAL
DE COLOMBIA

Book of Syllabi

Undergraduate Program in Information
Systems

– 2022-II–

Sede Manizales: November 15, 2022

Equipo de trabajo

Alex Cuadros-Vargas
Director of the Department of Computer Science, UCSP, Arequipa-Peru
Member of the Peruvian Computer Society (SPC)
email: *alex@ucsp.edu.pe*
Kelly Vizconde La Motta
Profesor of the Department of Computer Science, UCSP, Arequipa-Peru
email: *kvizconde@ucsp.edu.pe*

Ernesto Cuadros-Vargas (Editor) *<ecuadros@spc.org.pe>*
President of the Peruvian Computer Society (SPC) 2001-2007, 2009
Member of the Steering Committee de ACM/IEEE-CS Computing Curricula
for Computer Science (CS2013)
Member of Steering Committee de ACM/IEEE-CS Computing Curricula 2020
(CS2020)
http://socios.spc.org.pe/ecuadros

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

Contents

Universidad Nacional de Colombia (UNAL) Sede Manizales
Undergraduate Program in Information Systems
SILABO

CS111. Computing Foundations (Mandatory)

2022-II

1. General information		
1.1 School	:	Sistemas de Información
1.2 Course	:	CS111. Computing Foundations
1.3 Semester	:	1er Semestre.
1.4 Prerrequisites	:	None
1.5 Type of course	:	Mandatory
1.6 Learning modality	:	Face to face
1.7 Horas	:	2 HT; 2 HP; 2 HL;
1.8 Credits	:	4

2. Professors

3. Course foundation
This is the first course in the sequence of introductory courses to Computer Science. This course is intended to cover the concepts outlined by the Computing Curricula IEEE-CS/ACM 2013. Programming is one of the pillars of Computer Science; any professional of the area, will need to program to materialize their models and proposals. This course introduces participants to the fundamental concepts of this art. Topics include data types, control structures, functions, lists, recursion, and the mechanics of execution, testing, and debugging.

4. Summary
1. History 2. Basic Type Systems 3. Fundamental Programming Concepts 4. Basic Analysis 5. Fundamental Data Structures and Algorithms 6. Algorithms and Design 7. Development Methods

5. Generales Goals
• Introduce the fundamental concepts of programming.   • Develop the ability of abstraction using programming language

6. Contribution to Outcomes
This discipline contributes to the achievement of the following outcomes: **1)** Analyze a complex computing problem and to apply principles of computing and other relevant disciplines to identify solutions. (**Usage**)   **2)** Design, implement and evaluate a computing-based solution to meet a given set of computing requirements in the context of the program's discipline. (**Usage**)   **6)** Apply computer science theory and software development fundamentals to produce computing-based solutions. (**Usage**)

7. Content

UNIT 1: History (5)	
Competences:	
Content	**Generales Goals**
- Prehistory, the world before 1946 - History of computer hardware, software, networking - Pioneers of computing - History of the Internet	- Identify significant continuing trends in the history of the computing field [Familiarity] - Identify the contributions of several pioneers in the computing field [Familiarity] - Discuss the historical context for several programming language paradigms [Familiarity] - Compare daily life before and after the advent of personal computers and the Internet [Assessment]
Readings: Brookshear and Brylow (2019), Guttag (2013), Zelle (2010)	

UNIT 2: Basic Type Systems (2)	
Competences:	
Content	**Generales Goals**
- A type as a set of values together with a set of operations - Primitive types (e.g., numbers, Booleans) - Compound types built from other types (e.g., records, unions, arrays, lists, functions, references) - Association of types to variables, arguments, results, and fields - Type safety and errors caused by using values inconsistently given their intended types	- For both a primitive and a compound type, informally describe the values that have that type [Familiarity] - For a language with a static type system, describe the operations that are forbidden statically, such as passing the wrong type of value to a function or method [Familiarity] - Describe examples of program errors detected by a type system [Familiarity] - For multiple programming languages, identify program properties checked statically and program properties checked dynamically [Usage] - Use types and type-error messages to write and debug programs [Usage] - Define and use program pieces (such as functions, classes, methods) that use generic types, including for collections [Usage]
Readings: Guttag (2013), Zelle (2010)	

UNIT 3: Fundamental Programming Concepts (9)	
Competences:	
Content	**Generales Goals**
- Basic syntax and semantics of a higher-level language - Variables and primitive data types (e.g., numbers, characters, Booleans) - Expressions and assingments - Simple I/O including file I/O - Conditional and iterative control structures - Functions and parameter passing - The concept of recursion	- Analyze and explain the behavior of simple programs involving the fundamental programming constructs variables, expressions, assignments, I/O, control constructs, functions, parameter passing, and recursion. [Assessment] - Identify and describe uses of primitive data types [Familiarity] - Write programs that use primitive data types [Usage] - Modify and expand short programs that use standard conditional and iterative control structures and functions [Usage] - Design, implement, test, and debug a program that uses each of the following fundamental programming constructs: basic computation, simple I/O, standard conditional and iterative structures, the definition of functions, and parameter passing [Usage] - Write a program that uses file I/O to provide persistence across multiple executions [Usage] - Choose appropriate conditional and iteration constructs for a given programming task [Familiarity] - Describe the concept of recursion and give examples of its use [Assessment] - Identify the base case and the general case of a recursively-defined problem [Familiarity]
Readings: Guttag (2013), Zelle (2010)	

UNIT 4: Basic Analysis (2)	
Competences:	
Content	**Generales Goals**
- Differences among best, expected, and worst case behaviors of an algorithm - Big O notation: formal definition - Complexity classes, such as constant, logarithmic, linear, quadratic, and exponential - Big O notation: use - Analysis of iterative and recursive algorithms	- Explain what is meant by "best", "expected", and "worst" case behavior of an algorithm [Familiarity] - In the context of specific algorithms, identify the characteristics of data and/or other conditions or assumptions that lead to different behaviors [Familiarity] - State the formal definition of big O [Familiarity] - Use big O notation formally to give asymptotic upper bounds on time and space complexity of algorithms [Usage] - Use big O notation formally to give expected case bounds on time complexity of algorithms [Usage]
Readings: Guttag (2013), Zelle (2010)	

UNIT 5: Fundamental Data Structures and Algorithms (8)	
Competences:	
Content	**Generales Goals**
- Simple numerical algorithms, such as computing the average of a list of numbers, finding the min, max, - Sequential and binary search algorithms - Worst case quadratic sorting algorithms (selection, insertion) - Worst or average case O(N log N) sorting algorithms (quicksort, heapsort, mergesort) - Hash tables, including strategies for avoiding and resolving collisions - Binary search trees - Common operations on binary search trees such as select min, max, insert, delete, iterate over tree - Graphs and graph algorithms - Representations of graphs (e.g., adjacency list, adjacency matrix) - Depth- and breadth-first traversals - Heaps - Graphs and graph algorithms - Maximum and minimum cut problem - Local search - Pattern matching and string/text algorithms (e.g., substring matching, regular expression matching, longest common subsequence algorithms)	- Implement basic numerical algorithms [Usage] - Implement simple search algorithms and explain the differences in their time complexities [Assessment] - Be able to implement common quadratic and O(N log N) sorting algorithms [Usage] - Describe the implementation of hash tables, including collision avoidance and resolution [Familiarity] - Discuss the runtime and memory efficiency of principal algorithms for sorting, searching, and hashing [Familiarity] - Discuss factors other than computational efficiency that influence the choice of algorithms, such as programming time, maintainability, and the use of application-specific patterns in the input data [Familiarity] - Explain how tree balance affects the efficiency of various binary search tree operations [Familiarity] - Solve problems using fundamental graph algorithms, including depth-first and breadth-first search [Usage] - Demonstrate the ability to evaluate algorithms, to select from a range of possible options, to provide justification for that selection, and to implement the algorithm in a particular context [Assessment] - Describe the heap property and the use of heaps as an implementation of priority queues [Familiarity] - Solve problems using graph algorithms, including single-source and all-pairs shortest paths, and at least one minimum spanning tree algorithm [Usage] - Trace and/or implement a string-matching algorithm [Usage]
Readings: Guttag (2013), Zelle (2010)	

UNIT 6: Algorithms and Design (9)	
Competences:	
Content	**Generales Goals**
- The concept and properties of algorithms - Informal comparison of algorithm efficiency (e.g., operation counts) - The role of algorithms in the problem-solving process - Problem-solving strategies - Iterative and recursive mathematical functions - Iterative and recursive traversal of data structures - Divide-and-conquer strategies - Fundamental design concepts and principles - Abstraction - Program decomposition - Encapsulation and information hiding - Separation of behaivor and implementation	- Discuss the importance of algorithms in the problem-solving process [Familiarity] - Discuss how a problem may be solved by multiple algorithms, each with different properties [Familiarity] - Create algorithms for solving simple problems [Usage] - Use a programming language to implement, test, and debug algorithms for solving simple problems [Usage] - Implement, test, and debug simple recursive functions and procedures [Usage] - Determine whether a recursive or iterative solution is most appropriate for a problem [Assessment] - Implement a divide-and-conquer algorithm for solving a problem [Usage] - Apply the techniques of decomposition to break a program into smaller pieces [Usage] - Identify the data components and behaviors of multiple abstract data types [Usage] - Implement a coherent abstract data type, with loose coupling between components and behaviors [Usage] - Identify the relative strengths and weaknesses among multiple designs or implementations for a problem [Assessment]
Readings: Guttag (2013), Zelle (2010)	

UNIT 7: Development Methods (1)	
Competences:	
Content	**Generales Goals**
- Modern programming enviroments - Code search - Programming using library components and their APIs	Construct and debug programs using the standard libraries available with a chosen programming language [Familiarity]
Readings: Guttag (2013), Zelle (2010)	

8. Methodology

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. Assessment

Continuous Assessment 1 : 20 %

Partial Exam : 30 %

Continuous Assessment 2 : 20 %

Final exam : 30 %

References

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

Guttag, 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
Undergraduate Program in Information Systems
SILABO

CS1D1. Discrete Structures I (Mandatory)

2022-II

1. General information		
1.1 School	:	Sistemas de Información
1.2 Course	:	CS1D1. Discrete Structures I
1.3 Semester	:	1er Semestre.
1.4 Prerrequisites	:	None
1.5 Type of course	:	Mandatory
1.6 Learning modality	:	Face to face
1.7 Horas	:	2 HT; 4 HP;
1.8 Credits	:	4

2. Professors

3. Course foundation
Discrete structures provide the theoretical foundations necessary for computation. These fundamentals are not only useful to develop computation from a theoretical point of view as it happens in the course of computational theory, but also is useful for the practice of computing; In particular in applications such as verification, cryptography, formal methods, etc.

4. Summary
1. Sets, Relations, and Functions 2. Basic Logic 3. Proof Techniques 4. Data Representation

5. Generales Goals

- Apply Properly concepts of finite mathematics (sets, relations, functions) to represent data of real problems.

- Model real situations described in natural language, using propositional logic and predicate logic.

- Determine the abstract properties of binary relations.

- Choose the most appropriate demonstration method to determine the veracity of a proposal and construct correct mathematical arguments.

- Interpret mathematical solutions to a problem and determine their reliability, advantages and disadvantages.

- Express the operation of a simple electronic circuit using Boolean algebra.

6. Contribution to Outcomes

This discipline contributes to the achievement of the following outcomes:

1) Analyze a complex computing problem and to apply principles of computing and other relevant disciplines to identify solutions. (**Assessment**)

6) Apply computer science theory and software development fundamentals to produce computing-based solutions. (**Assessment**)

7. Content

UNIT 1: Sets, Relations, and Functions (22)	
Competences:	
Content	**Generales Goals**
- Sets - Venn diagrams - Union, intersection, complement - Cartesian product - Power sets - Cardinality of finite sets - Relations: - Reflexivity, simmetry, transitivity - Equivalence relations - Partial order relations and sets - Extremal elements of a partially ordered sets - Functions - Surjections, injections, bijections - Inverses - Composition	- Explain with examples the basic terminology of functions, relations, and sets [Assessment] - Perform the operations associated with sets, functions, and relations [Assessment] - Relate practical examples to the appropriate set, function, or relation model, and interpret the associated operations and terminology in context [Assessment]
Readings: Grimaldi (2003), Rosen (2007), Velleman (2006)	

UNIT 2: Basic Logic (14)	
Competences:	
Content	**Generales Goals**
• Propositional logic • Logical connectives • Truth tables • Normal forms (conjunctive and disjunctive) • Validity of well-formed formula • Propositional inference rules (concepts of modus ponens and modus tollens) • Predicate logic – Universal and existential quantification • Limitations of propositional and predicate logic (e.g., expressiveness issues)	• Convert logical statements from informal language to propositional and predicate logic expressions [Usage] • Apply formal methods of symbolic propositional and predicate logic, such as calculating validity of formulae and computing normal forms [Usage] • Use the rules of inference to construct proofs in propositional and predicate logic [Usage] • Describe how symbolic logic can be used to model real-life situations or applications, including those arising in computing contexts such as software analysis (eg, program correctness), database queries, and algorithms [Familiarity] • Apply formal logic proofs and/or informal, but rigorous, logical reasoning to real problems, such as predicting the behavior of software or solving problems such as puzzles [Usage] • Describe the strengths and limitations of propositional and predicate logic [Usage]
Readings: Rosen (2007), Grimaldi (2003), Velleman (2006)	

UNIT 3: Proof Techniques (14)	
Competences:	
Content	**Generales Goals**
• Notions of implication, equivalence, converse, inverse, contrapositive, negation, and contradiction • The structure of mathematical proofs • Direct proofs • Disproving by counterexample • Proof by contradiction • Induction over natural numbers • Structural induction • Weak and strong induction (i.e., First and Second Principle of Induction) • Recursive mathematical definitions • Well orderings	• Identify the proof technique used in a given proof [Assessment] • Outline the basic structure of each proof technique (direct proof, proof by contradiction, and induction) described in this unit [Usage] • Apply each of the proof techniques (direct proof, proof by contradiction, and induction) correctly in the construction of a sound argument [Usage] • Determine which type of proof is best for a given problem [Assessment] • Explain the parallels between ideas of mathematical and/or structural induction to recursion and recursively defined structures [Familiarity] • Explain the relationship between weak and strong induction and give examples of the appropriate use of each [Assessment] • State the well-ordering principle and its relationship to mathematical induction [Familiarity]
Readings: Rosen (2007), **Vel06**, Scheinerman (2012), Velleman (2006)	

UNIT 4: Data Representation (10)	
Competences:	
Content	**Generales Goals**
• Numerical representation: sign-magnitude, floating point. • Representation of other objects: sets, relations, functions.	• Explain numerical representations such as sign-magnitude and floating point. [Assessment]. • Carry out arithmetic operations using different kinds of representations. [Assessment]. • Explain the floating point standard IEEE-754 [Familiarity].
Readings: Rosen (2007), Grimaldi (2003), Velleman (2006)	

8. Methodology
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. Assessment
Continuous Assessment 1 : 20 % **Partial Exam** : 30 % **Continuous Assessment 2** : 20 % **Final exam** : 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
Undergraduate Program in Information Systems
SILABO

MA100. Mathematics I (Mandatory)

2022-II

1. General information		
1.1 School	:	Sistemas de Información
1.2 Course	:	MA100. Mathematics I
1.3 Semester	:	1er Semestre.
1.4 Prerrequisites	:	None
1.5 Type of course	:	Mandatory
1.6 Learning modality	:	Face to face
1.7 Horas	:	2 HT; 6 HP;
1.8 Credits	:	5

2. Professors

3. Course foundation
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. Summary
1. 2. 3. 4.

5. Generales Goals
.   .   .

6. Contribution to Outcomes
This discipline contributes to the achievement of the following outcomes:

1) Analyze a complex computing problem and to apply principles of computing and other relevant disciplines to identify solutions. (**Assessment**)

6) Apply computer science theory and software development fundamentals to produce computing-based solutions. (**Assessment**)

7. Content

UNIT 1: (20)	
Competences:	
Content	**Generales Goals**
• . • .	• . • .
Readings: Stewart (2012), ión (2014)	

UNIT 2: (10)	
Competences:	
Content	**Generales Goals**
• . • . • . • . • . • .	• . • . • . • . • . • .
Readings: Stewart (2012), ión (2014)	

UNIT 3: (20)	
Competences:	
Content	**Generales Goals**
• . • . • . • . • .	• . • . • . • . • . • . • . • . • . • . • .
Readings: Stewart (2012), ión (2014)	

UNIT 4: (22)	
Competences:	
Content	Generales Goals
 • . • . • . • .	 • . • . • . • . • . • . • . • . • . • . • .
Readings: Stewart (2012), ión (2014)	

---

8. Methodology

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. Assessment

Continuous Assessment 1 : 20 %

Partial Exam : 30 %

Continuous Assessment 2 : 20 %

Final exam : 30 %

References

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

Universidad Nacional de Colombia (UNAL) Sede Manizales
Undergraduate Program in Information Systems
SILABO

FG101. Integral Communication (Mandatory)

2022-II

1. General information		
1.1 School	:	Sistemas de Información
1.2 Course	:	FG101. Integral Communication
1.3 Semester	:	1er Semestre.
1.4 Prerrequisites	:	None
1.5 Type of course	:	Mandatory
1.6 Learning modality	:	Face to face
1.7 Horas	:	2 HT; 2 HP;
1.8 Credits	:	3

2. Professors

3. Course foundation
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 In order to achieve effective communication in the personal and professional field, the proper handling of the Language in oral and written form is a priority. It is therefore justified that the students of UTEC University know, understand and apply the conceptual and operational aspects of their language, for the development of their fundamental communicative skills: Listening, speaking, reading and writing. Consequently the permanent exercise and the contribution of the fundamentals contribute greatly in the academic formation and, in the future, in the performance of his profession.

4. Summary
1. 2. 3. 4.

5. Generales Goals
- 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.   - Develop communicative skills through the theory and practice of language that help the student to overcome the academic requirements of the undergraduate and contribute to his humanistic training and human person.

6. Contribution to Outcomes

This discipline contributes to the achievement of the following outcomes:

3) Communicate effectively in a variety of professional contexts. (**Usage**)

5) Function effectively as a member or leader of a team engaged in activities appropriate to the program's discipline. (**Usage**)

6) Apply computer science theory and software development fundamentals to produce computing-based solutions. (**Usage**)

7) Develop computational technology for the well-being of all, contributing with human formation, scientific, technological and professional skills to solve social problems of our community. (**Usage**)

---

7. Content

UNIT 1: (16)

Competences:

Content	Generales Goals
- 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. - The communication, definition, relevance. Elements. Process. Functions. Classification. Oral and written communication. - The language: definition. Features and functions. Language: levels. System. Rule. Speaks. The linguistic sign: definition, characteristics. - Multilingualism in Peru. Dialect variations in Peru. - The word: definition, classes and structure. The monemas: lexema and morpheme. The morpheme: classes. Etymology. - The Academic Article: Definition, structure, choice of topic, delimitation of the topic.	- Reconocer y valorar la comunicación como un proceso de comprensión e intercambio de mensajes, diferenciando sus elementos, funciones y clasificación [Usage]. - Analizar las características, funciones y elementos del lenguaje y de la lengua [Usage]. - Identificar las características del multilingüismo en el Perú, valorando su riqueza idiomática [Usage]. - Identificar las cualidades de la palabra y sus clases [Usage]. - Recognize and value communication as a process of understanding and exchanging messages, differentiating its elements, functions and classification[Usage]. - Analyze the characteristics, functions and elements of language and language [Usage]. - Identify the characteristics of multilingualism in Peru, valuing its idiomatic richness [Usage]. - Identify the qualities of the word and its classes [Usage].

Readings: Lengua Española (2010)

UNIT 2: (16)	
Competences:	
Content	**Generales Goals**

Content

- Párrafo: Idea principal, secundaria y global.

- El texto: definición, características. Cohesión y coherencia.

- Organización del texto: La referencia (deixis); anáfora, catáfora, elipsis. Conectores lógicos y textuales.

- Tipos de texto: descriptivo (procesos), expositivo, argumentativo.

- Funciones de elocución en el texto: generalización, identificación, nominalización, clasificación, ejemplificación, definición.

- Textos discontinuos: gráficos, tablas y diagramas.

- Búsqueda de información. Fuentes de información. Referencias y citas. Registro de información: fichas, notas, resúmenes, etc. Aparato crítico: concepto y finalidad. Normas APA u otro.

- Paragraph: Main, secondary and global idea.

- The text: definition, characteristics. Cohesion and coherence.

- Organization of the text: The reference (dejis); Anaphora, cataphora, ellipsis. Logical and textual connectors.

- Types of text: descriptive (processes), expository, argumentative.

- Functions of elocution in the text: generalization, identification, nominalization, classification, exemplification, definition.

- Discontinuous texts: graphs, tables and diagrams.

- Search for information. Information sources. References and citations. Record of information: index cards, notes, summaries, etc. Critical apparatus: concept and purpose. APA Standards or other.

Generales Goals

- Redactar textos expositivos resaltando la idea principal y secundaria [Usage].

- Redactar textos expositivos con adecuada cohesión y coherencia, haciendo uso de referencias y conectores textuales [Usage].

- Interpretar textos discontinuos valorando su importancia para la comprensión del mensaje [Usage].

- Redactar textos expositivos resaltando la idea principal y secundaria [Usage].

- Redactar textos expositivos con adecuada cohesión y coherencia, haciendo uso de referencias y conectores textuales [Usage].

- Interpretar textos discontinuos valorando su importancia para la comprensión del mensaje [Usage].

Readings: Lengua Española (2010), Gatti Muriel (2007)

UNIT 3: (12)

Competences:

Content	Generales Goals
- La oración: definición y clases. La oración enunciativa, interrogativa, imperativa, exclamativa, optativa. La proposición y la frase. La oración simple y compuesta. Coordinación y subordinación. El sintagma: estructura y clases: nominal, verbal, adjetival, preposicional, adverbial. - Elaboración de un glosario de términos técnicos, abreviaturas y siglas relacionadas con la especialidad (actividad permanente a lo largo del semestre). - Redacción del artículo académico: Resumen, palabras clave, introducción, desarrollo, conclusiones, bibliografiaTecnologia (Normas APA u otro que la Escuela profesional requiera).	- Reconocer y analizar la estructura oracional valorando su importancia y utilidad en la redacción de textos [Usage]. - Registrar y emplear terminología propia de la especialidad [Usage].

Readings: Sanchez Lobato (2005)

UNIT 4: (12)

Competences:

Content	Generales Goals
- Redacción de correspondencia: carta - solicitud, informe, memorando, hoja de vida. - El discurso oral: propósitos, partes. Escuchar: propósitos y condiciones. Vicios de dicción: barbarismo, solecismo, cacofonía, redundancia, anfibología, monotonía. Régimen preposicional. - Comunicación en grupo Proceso, dinámica, estructura Formas (Técnicas): Mesa redonda, panel, foro y debate. - Revisión final del artículo académico. Presentación y exposición oral de trabajos de producción intelectual.	- Redactar textos académicos y funcionales atendiendo los distintos momentos de su producción, su estructura, finalidad y formalidad [Usage]. - Demostrar habilidades como emisor o receptor en distintas situaciones de comunicación con corrección idiomática [Usage]. - Aplicar las diferentes formas (técnicas) de comunicación en grupo reconociendo su importancia para la solución de problemas, toma de decisiones o discusión [Usage].

Readings: Martin Vivaldi (2006)

8. Methodology
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. Assessment

Continuous Assessment 1 : 20 %

Partial Exam : 30 %

Continuous Assessment 2 : 20 %

Final exam : 30 %

References

Gatti Muriel, Carlos (2007). *Elementos de la gramática española*. Lima, Universidad del Pacífico.

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

Martin Vivaldi, Gonzalo (2006). *Teoría y práctica de la composición y estilo*. Thompson.

Sanchez Lobato, J (2005). *Saber Escribir*. España, Instituto Cervantes.

Universidad Nacional de Colombia (UNAL) Sede Manizales
Undergraduate Program in Information Systems
SILABO

FG101. Integral Communication (Mandatory)

2022-II

1. General information		
1.1 School	:	Sistemas de Información
1.2 Course	:	FG101. Integral Communication
1.3 Semester	:	1er Semestre.
1.4 Prerrequisites	:	None
1.5 Type of course	:	Mandatory
1.6 Learning modality	:	Face to face
1.7 Horas	:	2 HT; 2 HP;
1.8 Credits	:	3

2. Professors

3. Course foundation
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 In order to achieve effective communication in the personal and professional field, the proper handling of the Language in oral and written form is a priority. It is therefore justified that the students of UTEC University know, understand and apply the conceptual and operational aspects of their language, for the development of their fundamental communicative skills: Listening, speaking, reading and writing. Consequently the permanent exercise and the contribution of the fundamentals contribute greatly in the academic formation and, in the future, in the performance of his profession.

4. Summary
1. 2. 3. 4.

5. Generales Goals
- 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.   - Develop communicative skills through the theory and practice of language that help the student to overcome the academic requirements of the undergraduate and contribute to his humanistic training and human person.

6. Contribution to Outcomes

This discipline contributes to the achievement of the following outcomes:

3) Communicate effectively in a variety of professional contexts. (**Usage**)

5) Function effectively as a member or leader of a team engaged in activities appropriate to the program's discipline. (**Usage**)

6) Apply computer science theory and software development fundamentals to produce computing-based solutions. (**Usage**)

7) Develop computational technology for the well-being of all, contributing with human formation, scientific, technological and professional skills to solve social problems of our community. (**Usage**)

---

7. Content

UNIT 1: (16)

Competences:

Content	Generales Goals
- 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. - The communication, definition, relevance. Elements. Process. Functions. Classification. Oral and written communication. - The language: definition. Features and functions. Language: levels. System. Rule. Speaks. The linguistic sign: definition, characteristics. - Multilingualism in Peru. Dialect variations in Peru. - The word: definition, classes and structure. The monemas: lexema and morpheme. The morpheme: classes. Etymology. - The Academic Article: Definition, structure, choice of topic, delimitation of the topic.	- Reconocer y valorar la comunicación como un proceso de comprensión e intercambio de mensajes, diferenciando sus elementos, funciones y clasificación [Usage]. - Analizar las características, funciones y elementos del lenguaje y de la lengua [Usage]. - Identificar las características del multilingüismo en el Perú, valorando su riqueza idiomática [Usage]. - Identificar las cualidades de la palabra y sus clases [Usage]. - Recognize and value communication as a process of understanding and exchanging messages, differentiating its elements, functions and classification[Usage]. - Analyze the characteristics, functions and elements of language and language [Usage]. - Identify the characteristics of multilingualism in Peru, valuing its idiomatic richness [Usage]. - Identify the qualities of the word and its classes [Usage].

Readings: Lengua Española (2010)

UNIT 2: (16)

Competences:

Content	**Generales Goals**
- Párrafo: Idea principal, secundaria y global. - El texto: definición, características. Cohesión y coherencia. - Organización del texto: La referencia (deixis); anáfora, catáfora, elipsis. Conectores lógicos y textuales. - Tipos de texto: descriptivo (procesos), expositivo, argumentativo. - Funciones de elocución en el texto: generalización, identificación, nominalización, clasificación, ejemplificación, definición. - Textos discontinuos: gráficos, tablas y diagramas. - Búsqueda de información. Fuentes de información. Referencias y citas. Registro de información: fichas, notas, resúmenes, etc. Aparato crítico: concepto y finalidad. Normas APA u otro. - Paragraph: Main, secondary and global idea. - The text: definition, characteristics. Cohesion and coherence. - Organization of the text: The reference (dejis); Anaphora, cataphora, ellipsis. Logical and textual connectors. - Types of text: descriptive (processes), expository, argumentative. - Functions of elocution in the text: generalization, identification, nominalization, classification, exemplification, definition. - Discontinuous texts: graphs, tables and diagrams. - Search for information. Information sources. References and citations. Record of information: index cards, notes, summaries, etc. Critical apparatus: concept and purpose. APA Standards or other.	- Redactar textos expositivos resaltando la idea principal y secundaria [Usage]. - Redactar textos expositivos con adecuada cohesión y coherencia, haciendo uso de referencias y conectores textuales [Usage]. - Interpretar textos discontinuos valorando su importancia para la comprensión del mensaje [Usage]. - Redactar textos expositivos resaltando la idea principal y secundaria [Usage]. - Redactar textos expositivos con adecuada cohesión y coherencia, haciendo uso de referencias y conectores textuales [Usage]. - Interpretar textos discontinuos valorando su importancia para la comprensión del mensaje [Usage].

Readings: Lengua Española (2010), Gatti Muriel (2007)	

UNIT 3: (12)	
Competences:	
Content	**Generales Goals**
- La oración: definición y clases. La oración enunciativa, interrogativa, imperativa, exclamativa, optativa. La proposición y la frase. La oración simple y compuesta. Coordinación y subordinación. El sintagma: estructura y clases: nominal, verbal, adjetival, preposicional, adverbial. - Elaboración de un glosario de términos técnicos, abreviaturas y siglas relacionadas con la especialidad (actividad permanente a lo largo del semestre). - Redacción del artículo académico: Resumen, palabras clave, introducción, desarrollo, conclusiones, bibliografiaTecnologia (Normas APA u otro que la Escuela profesional requiera).	- Reconocer y analizar la estructura oracional valorando su importancia y utilidad en la redacción de textos [Usage]. - Registrar y emplear terminología propia de la especialidad [Usage].
Readings: Sanchez Lobato (2005)	

UNIT 4: (12)	
Competences:	
Content	**Generales Goals**
- Redacción de correspondencia: carta - solicitud, informe, memorando, hoja de vida. - El discurso oral: propósitos, partes. Escuchar: propósitos y condiciones. Vicios de dicción: barbarismo, solecismo, cacofonía, redundancia, anfibología, monotonía. Régimen preposicional. - Comunicación en grupo Proceso, dinámica, estructura Formas (Técnicas): Mesa redonda, panel, foro y debate. - Revisión final del artículo académico. Presentación y exposición oral de trabajos de producción intelectual.	- Redactar textos académicos y funcionales atendiendo los distintos momentos de su producción, su estructura, finalidad y formalidad [Usage]. - Demostrar habilidades como emisor o receptor en distintas situaciones de comunicación con corrección idiomática [Usage]. - Aplicar las diferentes formas (técnicas) de comunicación en grupo reconociendo su importancia para la solución de problemas, toma de decisiones o discusión [Usage].
Readings: Martin Vivaldi (2006)	

8. Methodology
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. Assessment

Continuous Assessment 1 : 20 %

Partial Exam : 30 %

Continuous Assessment 2 : 20 %

Final exam : 30 %

References

Gatti Muriel, Carlos (2007). *Elementos de la gramática española.* Lima, Universidad del Pacífico.

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

Martin Vivaldi, Gonzalo (2006). *Teoría y práctica de la composición y estilo.* Thompson.

Sanchez Lobato, J (2005). *Saber Escribir.* España, Instituto Cervantes.

Universidad Nacional de Colombia (UNAL) Sede Manizales
Undergraduate Program in Information Systems
SILABO

FG102. Study Methodology (Mandatory)

1. General information		
1.1 School	:	Sistemas de Información
1.2 Course	:	FG102. Study Methodology
1.3 Semester	:	1er Semestre.
1.4 Prerrequisites	:	None
1.5 Type of course	:	Mandatory
1.6 Learning modality	:	Face to face
1.7 Horas	:	2 HT; 2 HP;
1.8 Credits	:	3

2. Professors

3. Course foundation
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. Summary
1. 2. 3. 4.

5. Generales Goals
• 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. Contribution to Outcomes

This discipline contributes to the achievement of the following outcomes:

1) Analyze a complex computing problem and to apply principles of computing and other relevant disciplines to identify solutions. (**Familiarity**)

2) Design, implement and evaluate a computing-based solution to meet a given set of computing requirements in the context of the program's discipline. (**Familiarity**)

3) Communicate effectively in a variety of professional contexts. (**Usage**)

5) Function effectively as a member or leader of a team engaged in activities appropriate to the program's discipline. (**Usage**)

6) Apply computer science theory and software development fundamentals to produce computing-based solutions. (**Usage**)

7) Develop computational technology for the well-being of all, contributing with human formation, scientific, technological and professional skills to solve social problems of our community. (**Familiarity**)

7. Content

UNIT 1: (12)

Competences:

Content	Generales Goals
- El subrayado. - Toma de puntes. - La vocación, hábitos de la vida universitaria. - Interacción humana. - La voluntad como requisito para el aprendizaje. - La plantificación y el tiempo	- Analizar la documentación normativa de la Universidad valorando su importancia para la convivencia y desempeño académico. [Usage] - Comprender y valorar la exigencia de la vida universitaria como parte de la formación personal y profesional.[Usage] - Planificar adecuadamente el tiempo en función de sus metas personales y académicas.[Usage] - Elaborar un plan de mejora personal a partir del conocimiento de sí mismo.[Usage]

Readings: bibliografiaTecnologia

UNIT 2: (12)	
Competences:	

Content	Generales Goals
- 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.	- Identificar los procesos mentales relacionándolos con el aprendizaje [Usage]. - Comprender el proceso del aprendizaje para determinar el estilo propio e incorporarlo en su actividad académica [Usage]. - Desarrollar estrategias para el análisis de textos potenciando la comprensión lectora [Usage]. - Diseñar un programa estratégico para afrontar con éxito los exámenes[Usage].

Readings: Rodríguez (2007), Perez (2010), Quintana (2007)

UNIT 3: (12)	
Competences:	

Content	Generales Goals
- Los mapas conceptuales. Características y elementos. - Los derechos de autor y el plagio. Derechos personales o morales. Derechos patrimoniales. "Copyrigth". - 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.	- Aplicar las técnicas de estudio atendiendo a sus particularidades y adecuándolas a las distintas situaciones que demanda el aprendizaje [Usage]. - Reconocer la importancia del respeto a la propiedad Intelectual [Usage]. - Reconocer la importancia de la Inteligencia Emocional, la conducta asertiva, la autoestima y la resiliencia valorándolas como fortalezas para el desempeño universitario [Usage].

Readings: Chávez (2011), Velazco (1999)

UNIT 4: (12)	
Competences:	
Content	**Generales Goals**
- 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.	- Aplicar las técnicas de estudio atendiendo a sus particularidades y adecuándolas a las distintas situaciones que demanda el aprendizaje [Usage]. - Asumir el manejo de conductas y actitudes para el aprendizaje cooperativo y el desempeño en los equipos de trabajo [Usage]. - Formular un proyecto de método personal de estudio, de acuerdo a su estilo y necesidades, que incluya técnicas y estrategias [Usage].
Readings: Rodríguez (2007), Chávez (2011)	

8. Methodology
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. Assessment
Continuous Assessment 1 : 20 %   **Partial Exam** : 30 %   **Continuous Assessment 2** : 20 %   **Final exam** : 30 %

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
Undergraduate Program in Information Systems
SILABO

CS100. Introduction to Computer Science (Mandatory)

2022-II

1. General information		
1.1 School	:	Sistemas de Información
1.2 Course	:	CS100. Introduction to Computer Science
1.3 Semester	:	2do Semestre.
1.4 Prerrequisites	:	None
1.5 Type of course	:	Mandatory
1.6 Learning modality	:	Face to face
1.7 Horas	:	2 HT; 2 HP;
1.8 Credits	:	3

2. Professors

3. Course foundation
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 de Datos, Sistemas Operativos, Bases de Datos, etc.

4. Summary
1. 2. Basic Logic

5. Generales Goals
• Brindar un panorama del área del conocimiento que es cubierta en la ciencia de la computación.

6. Contribution to Outcomes
This discipline contributes to the achievement of the following outcomes:

1) Analyze a complex computing problem and to apply principles of computing and other relevant disciplines to identify solutions. (**Familiarity**)

4) Recognize professional responsabilities and make informed judgments in computing practice based on legal and ethical principles. (**Familiarity**)

5) Function effectively as a member or leader of a team engaged in activities appropriate to the program's discipline. (**Familiarity**)

6) Apply computer science theory and software development fundamentals to produce computing-based solutions. (**Familiarity**)

7) Develop computational technology for the well-being of all, contributing with human formation, scientific, technological and professional skills to solve social problems of our community. (**Familiarity**)

7. Content

UNIT 1: (2)	
Competences:	
Content	**Generales Goals**
• Introducción a la computación. • Historia de la computación.	• Encourage students to study Computer Science. [Familiarity]
Readings: Brookshear (2015)	

UNIT 2: Basic Logic (2)	
Competences:	
Content	**Generales Goals**
• Propositional logic • Logical connectives • Truth tables • Normal forms (conjunctive and disjunctive)	• Convert logical statements from informal language to propositional and predicate logic expressions [Familiarity] • Apply formal methods of symbolic propositional and predicate logic, such as calculating validity of formulae and computing normal forms [Familiarity]
Readings: Brookshear (2015)	

8. Methodology
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. Assessment
Continuous Assessment 1 : 20 %   **Partial Exam** : 30 %   **Continuous Assessment 2** : 20 %   **Final exam** : 30 %

References

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

Universidad Nacional de Colombia (UNAL) Sede Manizales
Undergraduate Program in Information Systems
SILABO

CS112. Computer Science I (Mandatory)

2022-II

1. General information		
1.1 School	:	Sistemas de Información
1.2 Course	:	CS112. Computer Science I
1.3 Semester	:	2do Semestre.
1.4 Prerrequisites	:	CS111. Computing Foundations. (1st Sem)
1.5 Type of course	:	Mandatory
1.6 Learning modality	:	Face to face
1.7 Horas	:	2 HT; 2 HP; 4 HL;
1.8 Credits	:	5

2. Professors

3. Course foundation
This is the second course in the sequence of introductory courses in computer science. The course will introduce students in the various topics of the area of computing such as: Algorithms, Data Structures, Software Engineering, etc.

4. Summary
1. General overwiew of Programming Languages 2. Virtual Machines 3. Basic Type Systems 4. Fundamental Programming Concepts 5. Object-Oriented Programming 6. Algorithms and Design 7. Algorithmic Strategies 8. Basic Analysis 9. Fundamental Data Structures and Algorithms

5. Generales Goals
• Introduce the student to the foundations of the object orientation paradigm, allowing the assimilation of concepts necessary to develop information systems.

6. Contribution to Outcomes
This discipline contributes to the achievement of the following outcomes: **1)** Analyze a complex computing problem and to apply principles of computing and other relevant disciplines to identify solutions. (**Assessment**)   **2)** Design, implement and evaluate a computing-based solution to meet a given set of computing requirements in the context of the program's discipline. (**Assessment**)   **5)** Function effectively as a member or leader of a team engaged in activities appropriate to the program's discipline. (**Familiarity**)   **6)** Apply computer science theory and software development fundamentals to produce computing-based solutions. (**Usage**)

7. Content

UNIT 1: General overwiew of Programming Languages (1)

Competences:

Content	Generales Goals
- Brief review of programming paradigms. - Comparison between functional programming and imperative programming. - History of programming languages.	Discuss the historical context for several programming language paradigms [Familiarity]

Readings: Stroustrup2013, Deitel17

UNIT 2: Virtual Machines (1)

Competences:

Content	Generales Goals
- The virtual machine concept. - Types of virtualization (including Hardware/Software, OS, Server, Service, Network). - Intermediate languages.	- Explain the concept of virtual memory and how it is realized in hardware and software [Familiarity] - Differentiate emulation and isolation [Familiarity] - Evaluate virtualization trade-offs [Assessment]

Readings: Stroustrup2013, Deitel17

UNIT 3: Basic Type Systems (2)	
Competences:	
Content	**Generales Goals**
- A type as a set of values together with a set of operations - Primitive types (e.g., numbers, Booleans) - Compound types built from other types (e.g., records, unions, arrays, lists, functions, references) - Model statement (link, visibility, scope and life time). - General view of type checking.	- For both a primitive and a compound type, informally describe the values that have that type [Familiarity] - For a language with a static type system, describe the operations that are forbidden statically, such as passing the wrong type of value to a function or method [Familiarity] - Describe examples of program errors detected by a type system [Familiarity] - For multiple programming languages, identify program properties checked statically and program properties checked dynamically [Usage] - Give an example program that does not type-check in a particular language and yet would have no error if run [Familiarity] - Use types and type-error messages to write and debug programs [Usage] - Explain how typing rules define the set of operations that are legal for a type [Familiarity] - Write down the type rules governing the use of a particular compound type [Usage] - Explain why undecidability requires type systems to conservatively approximate program behavior [Familiarity] - Define and use program pieces (such as functions, classes, methods) that use generic types, including for collections [Usage] - Discuss the differences among generics, subtyping, and overloading [Familiarity] - Explain multiple benefits and limitations of static typing in writing, maintaining, and debugging software [Familiarity]
Readings: Stroustrup2013, **Deitel17**	

UNIT 4: Fundamental Programming Concepts (6)	
Competences:	
Content	**Generales Goals**
- Basic syntax and semantics of a higher-level language - Variables and primitive data types (e.g., numbers, characters, Booleans) - Expressions and assingments - Simple I/O including file I/O - Conditional and iterative control structures - Functions and parameter passing	- Analyze and explain the behavior of simple programs involving the fundamental programming constructs variables, expressions, assignments, I/O, control constructs, functions, parameter passing, and recursion. [Assessment] - Identify and describe uses of primitive data types [Familiarity] - Write programs that use primitive data types [Usage] - Modify and expand short programs that use standard conditional and iterative control structures and functions [Usage] - Design, implement, test, and debug a program that uses each of the following fundamental programming constructs: basic computation, simple I/O, standard conditional and iterative structures, the definition of functions, and parameter passing [Usage] - Write a program that uses file I/O to provide persistence across multiple executions [Usage] - Choose appropriate conditional and iteration constructs for a given programming task [Assessment] - Describe the concept of recursion and give examples of its use [Familiarity] - Identify the base case and the general case of a recursively-defined problem [Assessment]
Readings: Stroustrup2013, Deitel17	

UNIT 5: Object-Oriented Programming (10)	
Competences:	
Content	**Generales Goals**

• Object-oriented design
 	
  – Decomposition into objects carrying state and having behavior
  – Class-hierarchy design for modeling
 	
• Object-oriented idioms for encapsulation
 	
  – Privacy and visibility of class members
  – Interfaces revealing only method signatures
  – Abstract base classes
 	
• Definition of classes: fields, methods, and constructors
 	
• Subclasses, inheritance, and method overriding
 	
• Subtyping
 	
  – Subtype polymorphism; implicit upcasts in typed languages
  – Notion of behavioral replacement: subtypes acting like supertypes
  – Relationship between subtyping and inheritance
 	
• Using collection classes, iterators, and other common library components
 	
• Dynamic dispatch: definition of method-call
	• Design and implement a class [Usage]
	
	• Use subclassing to design simple class hierarchies that allow code to be reused for distinct subclasses [Usage]
	
	• Correctly reason about control flow in a program using dynamic dispatch [Usage]
	
	• Compare and contrast (1) the procedural/functional approach—defining a function for each operation with the function body providing a case for each data variant—and (2) the object-oriented approach—defining a class for each data variant with the class definition providing a method for each operation Understand both as defining a matrix of operations and variants [Assessment]
	
	• Explain the relationship between object-oriented inheritance (code-sharing and overriding) and subtyping (the idea of a subtype being usable in a context that expects the supertype) [Familiarity]
	
	• Use object-oriented encapsulation mechanisms such as interfaces and private members [Usage]
	
	• Define and use iterators and other operations on aggregates, including operations that take functions as arguments, in multiple programming languages, selecting the most natural idioms for each language [Usage]

Readings: Stroustrup2013, **Deitel17**	

UNIT 6: Algorithms and Design (3)	
Competences:	
Content	**Generales Goals**
- Problem-solving strategies - Iterative and recursive mathematical functions - Iterative and recursive traversal of data structures - Divide-and-conquer strategies - The role of algorithms in the problem-solving process - Problem-solving strategies - Iterative and recursive mathematical functions - Iterative and recursive traversal of data structures - Divide-and-conquer strategies - Fundamental design concepts and principles - Abstraction - Program decomposition - Encapsulation and information hiding - Separation of behaivor and implementation	- Discuss the importance of algorithms in the problem-solving process [Familiarity] - Discuss how a problem may be solved by multiple algorithms, each with different properties [Familiarity] - Create algorithms for solving simple problems [Usage] - Use a programming language to implement, test, and debug algorithms for solving simple problems [Usage] - Implement, test, and debug simple recursive functions and procedures [Usage] - Determine whether a recursive or iterative solution is most appropriate for a problem [Assessment] - Implement a divide-and-conquer algorithm for solving a problem [Usage] - Apply the techniques of decomposition to break a program into smaller pieces [Usage] - Identify the data components and behaviors of multiple abstract data types [Usage] - Implement a coherent abstract data type, with loose coupling between components and behaviors [Usage] - Identify the relative strengths and weaknesses among multiple designs or implementations for a problem [Assessment]
Readings: Stroustrup2013, **Deitel17**	

UNIT 7: Algorithmic Strategies (3)	
Competences:	
Content	**Generales Goals**
• Brute-force algorithms • Greedy algorithms • Divide-and-conquer • Recursive backtracking • Dynamic Programming	• For each of the strategies (brute-force, greedy, divide-and-conquer, recursive backtracking, and dynamic programming), identify a practical example to which it would apply [Familiarity] • Use a greedy approach to solve an appropriate problem and determine if the greedy rule chosen leads to an optimal solution [Assessment] • Use a divide-and-conquer algorithm to solve an appropriate problem [Usage] • Use recursive backtracking to solve a problem such as navigating a maze [Usage] • Use dynamic programming to solve an appropriate problem [Usage] • Determine an appropriate algorithmic approach to a problem [Assessment] • Describe various heuristic problem-solving methods [Familiarity]
Readings: Stroustrup2013, **Deitel17**	

UNIT 8: Basic Analysis (2)	
Competences:	
Content	**Generales Goals**
• Differences among best, expected, and worst case behaviors of an algorithm	• Explain what is meant by "best", "expected", and "worst" case behavior of an algorithm [Familiarity]
Readings: Stroustrup2013, **Deitel17**	

UNIT 9: Fundamental Data Structures and Algorithms (6)	
Competences:	
Content	**Generales Goals**
- Simple numerical algorithms, such as computing the average of a list of numbers, finding the min, max, - Sequential and binary search algorithms - Worst case quadratic sorting algorithms (selection, insertion) - Worst or average case O(N log N) sorting algorithms (quicksort, heapsort, mergesort)	- Implement basic numerical algorithms [Usage] - Implement simple search algorithms and explain the differences in their time complexities [Assessment] - Be able to implement common quadratic and O(N log N) sorting algorithms [Usage] - Discuss the runtime and memory efficiency of principal algorithms for sorting, searching, and hashing [Familiarity] - Discuss factors other than computational efficiency that influence the choice of algorithms, such as programming time, maintainability, and the use of application-specific patterns in the input data [Familiarity] - Explain how tree balance affects the efficiency of various binary search tree operations [Familiarity] - Demonstrate the ability to evaluate algorithms, to select from a range of possible options, to provide justification for that selection, and to implement the algorithm in a particular context [Assessment] - Trace and/or implement a string-matching algorithm [Usage]
Readings: Stroustrup2013, **Deitel17**	

---

8. Methodology

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. Assessment

Continuous Assessment 1 : 20 %

Partial Exam : 30 %

Continuous Assessment 2 : 20 %

Final exam : 30 %

Universidad Nacional de Colombia (UNAL) Sede Manizales
Undergraduate Program in Information Systems
SILABO

CS1D2. Discrete Structures II (Mandatory)

2022-II

1. General information		
1.1 School	:	Sistemas de Información
1.2 Course	:	CS1D2. Discrete Structures II
1.3 Semester	:	2do Semestre.
1.4 Prerrequisites	:	CS1D1. Discrete Structures I. (1st Sem)
1.5 Type of course	:	Mandatory
1.6 Learning modality	:	Face to face
1.7 Horas	:	2 HT; 2 HP; 2 HL;
1.8 Credits	:	4

2. Professors

3. Course foundation

In order to understand the advanced computational techniques, the students must have a strong knowledge of the Various discrete structures, structures that will be implemented and used in the laboratory in the programming language..

4. Summary

1. Digital Logic and Data Representation 2. Basics of Counting 3. Graphs and Trees

5. Generales Goals

- That the student is able to model computer science problems using graphs and trees related to data structures.

- That the student applies efficient travel strategies to be able to search data in an optimal way.

- That the student uses the various counting techniques to solve computational problems.

6. Contribution to Outcomes

This discipline contributes to the achievement of the following outcomes:

1) Analyze a complex computing problem and to apply principles of computing and other relevant disciplines to identify solutions. (**Familiarity**)

6) Apply computer science theory and software development fundamentals to produce computing-based solutions. (**Familiarity**)

7. Content

UNIT 1: Digital Logic and Data Representation (10)	
Competences:	
Content	**Generales Goals**
- Reticles: Types and properties. - Boolean algebras. - Boolean Functions and Expressions. - Representation of Boolean Functions: Normal Disjunctive and Conjunctive Form. - Logical gates. - Circuit Minimization.	- Explain the importance of Boolean algebra as a unification of set theory and propositional logic [Assessment]. - Explain the algebraic structures of reticulum and its types [Assessment]. - Explain the relationship between the reticulum and the ordinate set and the wise use to show that a set is a reticulum [Assessment]. - Explain the properties that satisfies a Boolean algebra [Assessment]. - Demonstrate if a terna formed by a set and two internal operations is or not Boolean algebra [Assessment]. - Find the canonical forms of a Boolean function [Assessment]. - Represent a Boolean function as a Boolean circuit using logic gates [Assessment]. - Minimize a Boolean function. [Assessment].
Readings: Rosen (2007), Grimaldi (2003)	

UNIT 2: Basics of Counting (40)	
Competences:	
Content	**Generales Goals**

• Counting arguments
 	
  – Set cardinality and counting
 	
  – Sum and product rule
 	
  – Inclusion-exclusion principle
 	
  – Arithmetic and geometric progressions
 	
• The pigeonhole principle
 	
• Permutations and combinations
 	
  – Basic definitions
 	
  – Pascal's identity
 	
  – The binomial theorem
 	
• Solving recurrence relations
 	
  – An example of a simple recurrence relation, such as Fibonacci numbers
 	
  – Other examples, showing a variety of solutions
 	
• Basic modular arithmetic
	• Apply counting arguments, including sum and product rules, inclusion-exclusion principle and arithmetic/geometric progressions [Familiarity]
	
	• Apply the pigeonhole principle in the context of a formal proof [Familiarity]
	
	• Compute permutations and combinations of a set, and interpret the meaning in the context of the particular application [Familiarity]
	
	• Map real-world applications to appropriate counting formalisms, such as determining the number of ways to arrange people around a table, subject to constraints on the seating arrangement, or the number of ways to determine certain hands in cards (eg, a full house) [Familiarity]
	
	• Solve a variety of basic recurrence relations [Familiarity]
	
	• Analyze a problem to determine underlying recurrence relations [Familiarity]
	
	• Perform computations involving modular arithmetic [Familiarity]

Readings: Grimaldi (1997)	

UNIT 3: Graphs and Trees (40)	
Competences:	
Content	**Generales Goals**

• Trees
 	
  – Properties
 	
  – Traversal strategies
 	
• Undirected graphs
 	
• Directed graphs
 	
• Weighted graphs
 	
• Spanning trees/forests
 	
• Graph isomorphism
	• Illustrate by example the basic terminology of graph theory, and some of the properties and special cases of each type of graph/tree [Familiarity]
	
	• Demonstrate different traversal methods for trees and graphs, including pre, post, and in-order traversal of trees [Familiarity]
	
	• Model a variety of real-world problems in computer science using appropriate forms of graphs and trees, such as representing a network topology or the organization of a hierarchical file system [Familiarity]
	
	• Show how concepts from graphs and trees appear in data structures, algorithms, proof techniques (structural induction), and counting [Familiarity]
	
	• Explain how to construct a spanning tree of a graph [Familiarity]
	
	• Determine if two graphs are isomorphic [Familiarity]

Readings: Johnsonbaugh (1999)	

8. Methodology
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. Assessment
Continuous Assessment 1 : 20 %   **Partial Exam** : 30 %   **Continuous Assessment 2** : 20 %   **Final exam** : 30 %

References

Grimaldi, R. (1997). *Matemáticas Discretas y Combinatoria*. Addison Wesley Iberoamericana.

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
Undergraduate Program in Information Systems
SILABO

MA101. Mathematics II (Mandatory)

2022-II

1. General information		
1.1 School	:	Sistemas de Información
1.2 Course	:	MA101. Mathematics II
1.3 Semester	:	2do Semestre.
1.4 Prerrequisites	:	MA100. Mathematics I. (1st Sem)
1.5 Type of course	:	Mandatory
1.6 Learning modality	:	Face to face
1.7 Horas	:	2 HT; 4 HP;
1.8 Credits	:	4

2. Professors

3. Course foundation
The course develops in students the skills to deal with models of science and engineering skills. In the first part of the course a study of the functions of several variables, partial derivatives, multiple integrals and an introduction to vector fields is performed. Then the student will use the basic concepts of calculus to model and solve ordinary differential equations using techniques such as Laplace transforms and Fourier series.

4. Summary
1. Multi-Variable Function Differential 2. Multi-Variable function Integral 3. Series 4. Ordinary Differential Equations

5. Generales Goals
- Apply derivation rules and partial differentation in functions of several variables.   - Apply techniques for calculating multiple integrals.   - Understand and use the concepts of vector calculus.   - Understand the importance of series.   - Identify and solve differential equations of the first order and their applications in chemical and physical problems.

6. Contribution to Outcomes
This discipline contributes to the achievement of the following outcomes:

1) Analyze a complex computing problem and to apply principles of computing and other relevant disciplines to identify solutions. (**Assessment**)

6) Apply computer science theory and software development fundamentals to produce computing-based solutions. (**Assessment**)

7. Content

UNIT 1: Multi-Variable Function Differential (24)	
Competences:	
Content	**Generales Goals**
- Concept of multi-variable functions. - Directional Derivates - Tangent line, normal plane to curve line and tangent plane, normal line to a curve plan. Know to calculate their equations. - Concept of extreme value and conditional extreme value of multi-variable functions - Applications problems such as modeling total production of an economic system, speed of sound through the ocean, thickener optimization, etc.	- Understand the concept of multi-variable functions. - Master the concept and calculation method of the direction derivative and gradient of the guide. - Master the calculation method of the first order and second order partial derivative of composite functions. - Master the calculation method of the partial derivatives for implicit functions. - Understand tangent line, normal plane to curve line and tangent plane, normal line to a curve plan. Know to calculate their equations. - Learn the concept of extreme value and conditional extreme value of multi-variable functions; know to find out the binary function extreme value. - Be able to solve simple applications problems.
Readings: Stewart (2012), Zill (2013)	

UNIT 2: Multi-Variable function Integral (12)	
Competences:	
Content	**Generales Goals**
- Double integral, triple integral and nature of the multiple integral. - Method of double integral - Line Integral - The Divergence, Rotation and Laplacian	- Understand the double integral, triple integral, and understand the nature of the multiple integral. - Master the calculation method of double integral (Cartesian coordinates, polar coordinates) the triple integral (Cartesian coordinates, cylindrical coordinates, spherical coordinates). - Understand the concept of line Integral, their properties and relationships. - Know to calculate the line integral. - Master the calculation the rotational, divergence and Laplacian.
Readings: Stewart (2012), Zill (2013)	

UNIT 3: Series (24)	
Competences:	

Content	**Generales Goals**
• Convergent series • Taylor and McLaurin series • Orthogonal functions	• Master to calculation if series is convergent, and if convergent, find the sum of the series trying to find the radius of convergence and the interval of convergence of a power series. • Represent a function as a power series and find the Taylor and McLaurin Series to estimate function values to a desired accuracy. • Understand the concepts of orthogonal functions and the expansion of a given function f to find its Fourier series.

Readings: Stewart (2012), Zill (2013)	

UNIT 4: Ordinary Differential Equations (30)	
Competences:	

Content	**Generales Goals**
• Concept of differential equations • Methods to resolve differential equations • Methods to resolve the secod order linear differential equations • Higher order linear ordinary differential equations • Applications problems using Laplace transforms	• Understand differential equations, solutions, order, general solution, initial conditions and special solutions etc. • Master the calculation method for variables separable equation and first order linear equations. Known to solve homogeneous equation and Bernoulli (Bernoulli) equations; understand variable substitution to solve the equation. • Master to solve total differential equations. • Be able to use reduced order method to solve equations. • Understand the structure of the second order linear differential equation. • Master calculation method for the constant coefficient homogeneous linear differential equations; and understand calculation method for the higher order homogeneous linear differential equations. • Know to apply the differential equation calculation method to solve simple geometric and physic application problems. • Solve properly certain types of differential equations using Laplace transforms.

Readings: Stewart (2012), Zill (2013)	

8. Methodology

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. Assessment

Continuous Assessment 1 : 20 %

Partial Exam : 30 %

Continuous Assessment 2 : 20 %

Final exam : 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
Undergraduate Program in Information Systems
SILABO

CS113. Computer Science II (Mandatory)

2022-II

1. General information		
1.1 School	:	Sistemas de Información
1.2 Course	:	CS113. Computer Science II
1.3 Semester	:	3er Semestre.
1.4 Prerrequisites	:	

- CS112. Computer Science I. (2nd Sem)

- CS112. Computer Science I. (2nd Sem)

1.5 Type of course	:	Mandatory
1.6 Learning modality	:	Face to face
1.7 Horas	:	2 HT; 2 HP; 4 HL;
1.8 Credits	:	5

2. Professors

3. Course foundation

This is the third course in the sequence of introductory courses in computer science. This course is intended to cover Concepts indicated by the Computing Curriculum IEEE (c) -ACM 2001, under the functional-first approach. The object-oriented paradigm allows us to combat complexity by making models from abstractions of the problem elements and using techniques such as encapsulation, modularity, polymorphism and inheritance. The Dominion of these topics will enable participants to provide computational solutions to design problems simple of the real world.

4. Summary

1. Fundamental Programming Concepts 2. Object-Oriented Programming 3. Algorithms and Design 4. Basic Analysis 5. Basic Type Systems 6. Fundamental Data Structures and Algorithms 7. Event-Driven and Reactive Programming 8. Graphs and Trees 9. Software Design 10. Requirements Engineering

5. Generales Goals

- Introduce the student in the fundaments of the paradigm of object orientation, allowing the assimilation of concepts necessary to develop an information system

6. Contribution to Outcomes

This discipline contributes to the achievement of the following outcomes:

1) Analyze a complex computing problem and to apply principles of computing and other relevant disciplines to identify solutions. (**Usage**)

3) Communicate effectively in a variety of professional contexts. (**Usage**)

5) Function effectively as a member or leader of a team engaged in activities appropriate to the program's discipline. (**Usage**)

7. Content

UNIT 1: Fundamental Programming Concepts (5)
Competences:

Content	Generales Goals
• Basic syntax and semantics of a higher-level language • Variables and primitive data types (e.g., numbers, characters, Booleans) • Expressions and assingments • Simple I/O including file I/O • Conditional and iterative control structures • Functions and parameter passing • The concept of recursion	• Analyze and explain the behavior of simple programs involving the fundamental programming constructs variables, expressions, assignments, I/O, control constructs, functions, parameter passing, and recursion. [Usage] • Identify and describe uses of primitive data types [Usage] • Write programs that use primitive data types [Usage] • Modify and expand short programs that use standard conditional and iterative control structures and functions [Usage] • Design, implement, test, and debug a program that uses each of the following fundamental programming constructs: basic computation, simple I/O, standard conditional and iterative structures, the definition of functions, and parameter passing [Usage] • Write a program that uses file I/O to provide persistence across multiple executions [Usage] • Choose appropriate conditional and iteration constructs for a given programming task [Usage] • Describe the concept of recursion and give examples of its use [Usage] • Identify the base case and the general case of a recursively-defined problem [Usage]
Readings: stroustrup2013, Vandervoorde (2002), Lippman and E.Moo (2013)	

2

UNIT 2: Object-Oriented Programming (7)	
Competences:	
Content	**Generales Goals**
- Object-oriented design - Decomposition into objects carrying state and having behavior - Class-hierarchy design for modeling - Definition of classes: fields, methods, and constructors - Subclasses, inheritance, and method overriding - Dynamic dispatch: definition of method-call - Subtyping - Subtype polymorphism; implicit upcasts in typed languages - Notion of behavioral replacement: subtypes acting like supertypes - Relationship between subtyping and inheritance - Object-oriented idioms for encapsulation - Privacy and visibility of class members - Interfaces revealing only method signatures - Abstract base classes - Using collection classes, iterators, and other common library components	- Design and implement a class [Usage] - Use subclassing to design simple class hierarchies that allow code to be reused for distinct subclasses [Usage] - Correctly reason about control flow in a program using dynamic dispatch [Usage] - Compare and contrast (1) the procedural/functional approach—defining a function for each operation with the function body providing a case for each data variant—and (2) the object-oriented approach—defining a class for each data variant with the class definition providing a method for each operation Understand both as defining a matrix of operations and variants [Usage] - Explain the relationship between object-oriented inheritance (code-sharing and overriding) and subtyping (the idea of a subtype being usable in a context that expects the supertype) [Usage] - Use object-oriented encapsulation mechanisms such as interfaces and private members [Usage] - Define and use iterators and other operations on aggregates, including operations that take functions as arguments, in multiple programming languages, selecting the most natural idioms for each language [Usage]
Readings: stroustrup2013	

UNIT 3: Algorithms and Design (5)	
Competences:	
Content	**Generales Goals**
- The concept and properties of algorithms - Informal comparison of algorithm efficiency (e.g., operation counts) - The role of algorithms in the problem-solving process - Problem-solving strategies - Iterative and recursive mathematical functions - Iterative and recursive traversal of data structures - Divide-and-conquer strategies - Fundamental design concepts and principles - Abstraction - Program decomposition - Encapsulation and information hiding - Separation of behaivor and implementation	- Discuss the importance of algorithms in the problem-solving process [Usage] - Discuss how a problem may be solved by multiple algorithms, each with different properties [Usage] - Create algorithms for solving simple problems [Usage] - Use a programming language to implement, test, and debug algorithms for solving simple problems [Usage] - Implement, test, and debug simple recursive functions and procedures [Usage] - Determine whether a recursive or iterative solution is most appropriate for a problem [Usage] - Implement a divide-and-conquer algorithm for solving a problem [Usage] - Apply the techniques of decomposition to break a program into smaller pieces [Usage] - Identify the data components and behaviors of multiple abstract data types [Usage] - Implement a coherent abstract data type, with loose coupling between components and behaviors [Usage] - Identify the relative strengths and weaknesses among multiple designs or implementations for a problem [Usage]
Readings: stroustrup2013, **Weert16**, Lippman and E.Moo (2013)	

UNIT 4: Basic Analysis (3)	
Competences:	
Content	**Generales Goals**
- Differences among best, expected, and worst case behaviors of an algorithm - Asymptotic analysis of upper and expected complexity bounds - Big O notation: formal definition - Complexity classes, such as constant, logarithmic, linear, quadratic, and exponential - Empirical measurements of performance - Time and space trade-offs in algorithms - Big O notation: use - Little o, big omega and big theta notation - Recurrence relations - Analysis of iterative and recursive algorithms - Master Theorem and Recursion Trees	- Explain what is meant by "best", "expected", and "worst" case behavior of an algorithm [Usage] - In the context of specific algorithms, identify the characteristics of data and/or other conditions or assumptions that lead to different behaviors [Usage] - Determine informally the time and space complexity of different algorithms [Usage] - State the formal definition of big O [Usage] - List and contrast standard complexity classes [Usage] - Perform empirical studies to validate hypotheses about runtime stemming from mathematical analysis Run algorithms on input of various sizes and compare performance [Usage] - Give examples that illustrate time-space trade-offs of algorithms [Usage] - Use big O notation formally to give asymptotic upper bounds on time and space complexity of algorithms [Usage] - Use big O notation formally to give expected case bounds on time complexity of algorithms [Usage] - Explain the use of big omega, big theta, and little o notation to describe the amount of work done by an algorithm [Usage] - Use recurrence relations to determine the time complexity of recursively defined algorithms [Usage] - Solve elementary recurrence relations, eg, using some form of a Master Theorem [Usage]
Readings: stroustrup2013	

UNIT 5: Basic Type Systems (5)	
Competences:	
Content	**Generales Goals**
- A type as a set of values together with a set of operations - Primitive types (e.g., numbers, Booleans) - Compound types built from other types (e.g., records, unions, arrays, lists, functions, references) - Association of types to variables, arguments, results, and fields - Type safety and errors caused by using values inconsistently given their intended types - Goals and limitations of static typing - Eliminating some classes of errors without running the program - Undecidability means static analysis must conservatively approximate program behavior - Generic types (parametric polymorphism) - Definition - Use for generic libraries such as collections - Comparison with ad hoc polymorphism (overloading) and subtype polymorphism - Complementary benefits of static and dynamic typing - Errors early vs. errors late/avoided - Enforce invariants during code development and code maintenance vs. postpone typing decisions while prototyping and conveniently allow flexible coding patterns such as heterogeneous collections - Avoid misuse of code vs. allow more code reuse - Detect incomplete programs vs. allow incomplete programs to run	- For both a primitive and a compound type, informally describe the values that have that type [Usage] - For a language with a static type system, describe the operations that are forbidden statically, such as passing the wrong type of value to a function or method [Usage] - Describe examples of program errors detected by a type system [Usage] - For multiple programming languages, identify program properties checked statically and program properties checked dynamically [Usage] - Give an example program that does not type-check in a particular language and yet would have no error if run [Usage] - Use types and type-error messages to write and debug programs [Usage] - Explain how typing rules define the set of operations that are legal for a type [Usage] - Write down the type rules governing the use of a particular compound type [Usage] - Explain why undecidability requires type systems to conservatively approximate program behavior [Usage] - Define and use program pieces (such as functions, classes, methods) that use generic types, including for collections [Usage] - Discuss the differences among generics, subtyping, and overloading [Usage] - Explain multiple benefits and limitations of static typing in writing, maintaining, and debugging software [Usage]
Readings: stroustrup2013	

UNIT 6: Fundamental Data Structures and Algorithms (3)	
Competences:	
Content	Generales Goals

• Simple numerical algorithms, such as computing the average of a list of numbers, finding the min, max,
 	
• Sequential and binary search algorithms
 	
• Worst case quadratic sorting algorithms (selection, insertion)
 	
• Worst or average case O(N log N) sorting algorithms (quicksort, heapsort, mergesort)
 	
• Hash tables, including strategies for avoiding and resolving collisions
 	
• Binary search trees
 	
   – Common operations on binary search trees such as select min, max, insert, delete, iterate over tree
 	
• Graphs and graph algorithms
 	
   – Representations of graphs (e.g., adjacency list, adjacency matrix)
 	
   – Depth- and breadth-first traversals
 	
• Heaps
 	
• Graphs and graph algorithms
 	
   – Maximum and minimum cut problem
 	
   – Local search
 	
• Pattern matching and string/text algorithms (e.g., substring matching, regular expression matching, longest common subsequence algorithms)
	• Implement basic numerical algorithms [Usage]
	
	• Implement simple search algorithms and explain the differences in their time complexities [Usage]
	
	• Be able to implement common quadratic and O(N log N) sorting algorithms [Usage]
	
	• Describe the implementation of hash tables, including collision avoidance and resolution [Usage]
	
	• Discuss the runtime and memory efficiency of principal algorithms for sorting, searching, and hashing [Usage]
	
	• Discuss factors other than computational efficiency that influence the choice of algorithms, such as programming time, maintainability, and the use of application-specific patterns in the input data [Usage]
	
	• Explain how tree balance affects the efficiency of various binary search tree operations [Usage]
	
	• Solve problems using fundamental graph algorithms, including depth-first and breadth-first search [Usage]
	
	• Demonstrate the ability to evaluate algorithms, to select from a range of possible options, to provide justification for that selection, and to implement the algorithm in a particular context [Usage]
	
	• Describe the heap property and the use of heaps as an implementation of priority queues [Usage]
	
	• Solve problems using graph algorithms, including single-source and all-pairs shortest paths, and at least one minimum spanning tree algorithm [Usage]
	
	• Trace and/or implement a string-matching algorithm [Usage]

Readings: stroustrup2013, Pai and Abraham (2018)	

UNIT 7: Event-Driven and Reactive Programming (2)	
Competences:	
Content	**Generales Goals**
- Events and event handlers - Canonical uses such as GUIs, mobile devices, robots, servers - Using a reactive framework - Defining event handlers/listeners - Main event loop not under event-handler-writer's control - Externally-generated events and program-generated events - Separation of model, view, and controller	- Write event handlers for use in reactive systems, such as GUIs [Usage] - Explain why an event-driven programming style is natural in domains where programs react to external events [Usage] - Describe an interactive system in terms of a model, a view, and a controller [Usage]
Readings: stroustrup2013, Williams (2011)	

UNIT 8: Graphs and Trees (7)	
Competences:	
Content	**Generales Goals**
- Trees - Properties - Traversal strategies - Undirected graphs - Directed graphs - Weighted graphs - Spanning trees/forests - Graph isomorphism	- Illustrate by example the basic terminology of graph theory, and some of the properties and special cases of each type of graph/tree [Usage] - Demonstrate different traversal methods for trees and graphs, including pre, post, and in-order traversal of trees [Usage] - Model a variety of real-world problems in computer science using appropriate forms of graphs and trees, such as representing a network topology or the organization of a hierarchical file system [Usage] - Show how concepts from graphs and trees appear in data structures, algorithms, proof techniques (structural induction), and counting [Usage] - Explain how to construct a spanning tree of a graph [Usage] - Determine if two graphs are isomorphic [Usage]
Readings: Nakariakov (2013)	

UNIT 9: Software Design (6)	
Competences:	
Content	**Generales Goals**

Content	Generales Goals

• System design principles: levels of abstraction (architectural design and detailed design), separation of concerns, information hiding, coupling and cohesion , re-use of standard structures
 	
• Design Paradigms such as structured design (top-down functional decomposition), object-oriented analysis and design, event driven design, component-level design, data-structured centered, aspect oriented, function oriented, service oriented
 	
• Structural and behavioral models of software designs
 	
• Design patterns
 	
• Relationships between requirements and designs: transformation of models, design of contracts, invariants
 	
• Software architecture concepts and standard architectures (e.g. client-server, n-layer, transform centered, pipes-and-filters)
 	
• The use of component desing: component selection, design, adaptation and assembly of components, component and patterns, components and objects (for example, building a GUI using a standar widget set)
 	
• Refactoring designs using design patterns
 	
• Internal design qualities, and models for them: efficiency and performance, redundancy and fault tolerance, traceability of requeriments
 	
• Measurement and analysis of design quality
 	
• Tradeoffs between different aspects of quality
 	
• Application frameworks
 	
• Middleware: the object-oriented paradigm within middleware, object request brokers and marshalling, transaction processing monitors, workflow systems
 	
• Principles of secure design and coding
 	
   – Principle of least privilege
   – Principle of fail-safe defaults
   – Principle of psychological acceptability
	• Articulate design principles including separation of concerns, information hiding, coupling and cohesion, and encapsulation [Usage]
	
	• Use a design paradigm to design a simple software system, and explain how system design principles have been applied in this design [Usage]
	
	• Construct models of the design of a simple software system that are appropriate for the paradigm used to design it [Usage]
	
	• Within the context of a single design paradigm, describe one or more design patterns that could be applicable to the design of a simple software system [Usage]
	
	• For a simple system suitable for a given scenario, discuss and select an appropriate design paradigm [Usage]
	
	• Create appropriate models for the structure and behavior of software products from their requirements specifications [Usage]
	
	• Explain the relationships between the requirements for a software product and its design, using appropriate models [Usage]
	
	• For the design of a simple software system within the context of a single design paradigm, describe the software architecture of that system [Usage]
	
	• Given a high-level design, identify the software architecture by differentiating among common software architectures such as 3-tier, pipe-and-filter, and client-server [Usage]
	
	• Investigate the impact of software architectures selection on the design of a simple system [Usage]
	
	• Apply simple examples of patterns in a software design [Usage]
	
	• Describe a form of refactoring and discuss when it may be applicable [Usage]
	
	• Select suitable components for use in the design of a software product [Usage]
	
	• Explain how suitable components might need to be adapted for use in the design of a software product [Usage]
	
	• Design a contract for a typical small software component for use in a given system [Usage]
	
	• Discuss and select appropriate software architecture for a simple system suitable for a given scenario [Usage]
	
	• Apply models for internal and external qualities in designing software components to achieve an acceptable tradeoff between conflicting quality aspects [Us-

UNIT 10: Requirements Engineering (1)	
Competences:	
Content	**Generales Goals**
- Describing functional requirements using, for example, use cases or users stories - Properties of requirements including consistency, validity, completeness, and feasibility - Software requirements elicitation - Describing system data using, for example, class diagrams or entity-relationship diagrams - Non functional requirements and their relationship to software quality - Evaluation and use of requirements specifications - Requirements analysis modeling techniques - Acceptability of certainty / uncertainty considerations regarding software / system behavior - Prototyping - Basic concepts of formal requirements specification - Requirements specification - Requirements validation - Requirements tracing	- List the key components of a use case or similar description of some behavior that is required for a system [Usage] - Describe how the requirements engineering process supports the elicitation and validation of behavioral requirements [Usage] - Interpret a given requirements model for a simple software system [Usage] - Describe the fundamental challenges of and common techniques used for requirements elicitation [Usage] - List the key components of a data model (eg, class diagrams or ER diagrams) [Usage] - Identify both functional and non-functional requirements in a given requirements specification for a software system [Usage] - Conduct a review of a set of software requirements to determine the quality of the requirements with respect to the characteristics of good requirements [Usage] - Apply key elements and common methods for elicitation and analysis to produce a set of software requirements for a medium-sized software system [Usage] - Compare the plan-driven and agile approaches to requirements specification and validation and describe the benefits and risks associated with each [Usage] - Use a common, non-formal method to model and specify the requirements for a medium-size software system [Usage] - Translate into natural language a software requirements specification (eg, a software component contract) written in a formal specification language [Usage] - Create a prototype of a software system to mitigate risk in requirements [Usage] - Differentiate between forward and backward tracing and explain their roles in the requirements validation process [Usage]
Readings: stroustrup2013	

8. Methodology

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. Assessment

Continuous Assessment 1 : 20 %

Partial Exam : 30 %

Continuous Assessment 2 : 20 %

Final exam : 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 Platforml.

Pai, Praseed and Peter Abraham (2018). *C++ Reactive Programming*. 1st. Packt.

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
Undergraduate Program in Information Systems
SILABO

CS113. Computer Science II (Mandatory)

2022-II

1. General information		
1.1 School	:	Sistemas de Información
1.2 Course	:	CS113. Computer Science II
1.3 Semester	:	3er Semestre.
1.4 Prerrequisites	:	

- CS112. Computer Science I. (2nd Sem)

- CS112. Computer Science I. (2nd Sem)

1.5 Type of course	:	Mandatory
1.6 Learning modality	:	Face to face
1.7 Horas	:	2 HT; 2 HP; 4 HL;
1.8 Credits	:	5

2. Professors

3. Course foundation

This is the third course in the sequence of introductory courses in computer science. This course is intended to cover Concepts indicated by the Computing Curriculum IEEE (c) -ACM 2001, under the functional-first approach. The object-oriented paradigm allows us to combat complexity by making models from abstractions of the problem elements and using techniques such as encapsulation, modularity, polymorphism and inheritance. The Dominion of these topics will enable participants to provide computational solutions to design problems simple of the real world.

4. Summary

1. Fundamental Programming Concepts 2. Object-Oriented Programming 3. Algorithms and Design 4. Basic Analysis 5. Basic Type Systems 6. Fundamental Data Structures and Algorithms 7. Event-Driven and Reactive Programming 8. Graphs and Trees 9. Software Design 10. Requirements Engineering

5. Generales Goals

- Introduce the student in the fundaments of the paradigm of object orientation, allowing the assimilation of concepts necessary to develop an information system

6. Contribution to Outcomes

This discipline contributes to the achievement of the following outcomes:

1) Analyze a complex computing problem and to apply principles of computing and other relevant disciplines to identify solutions. (**Usage**)

3) Communicate effectively in a variety of professional contexts. (**Usage**)

5) Function effectively as a member or leader of a team engaged in activities appropriate to the program's discipline. (**Usage**)

7. Content

UNIT 1: Fundamental Programming Concepts (5)	
Competences:	
Content	**Generales Goals**
- Basic syntax and semantics of a higher-level language - Variables and primitive data types (e.g., numbers, characters, Booleans) - Expressions and assingments - Simple I/O including file I/O - Conditional and iterative control structures - Functions and parameter passing - The concept of recursion	- Analyze and explain the behavior of simple programs involving the fundamental programming constructs variables, expressions, assignments, I/O, control constructs, functions, parameter passing, and recursion. [Usage] - Identify and describe uses of primitive data types [Usage] - Write programs that use primitive data types [Usage] - Modify and expand short programs that use standard conditional and iterative control structures and functions [Usage] - Design, implement, test, and debug a program that uses each of the following fundamental programming constructs: basic computation, simple I/O, standard conditional and iterative structures, the definition of functions, and parameter passing [Usage] - Write a program that uses file I/O to provide persistence across multiple executions [Usage] - Choose appropriate conditional and iteration constructs for a given programming task [Usage] - Describe the concept of recursion and give examples of its use [Usage] - Identify the base case and the general case of a recursively-defined problem [Usage]
Readings: stroustrup2013, Vandervoorde (2002), Lippman and E.Moo (2013)	

UNIT 2: Object-Oriented Programming (7)	
Competences:	
Content	**Generales Goals**

• Object-oriented design
 	
  – Decomposition into objects carrying state and having behavior
  – Class-hierarchy design for modeling
 	
• Definition of classes: fields, methods, and constructors
 	
• Subclasses, inheritance, and method overriding
 	
• Dynamic dispatch: definition of method-call
 	
• Subtyping
 	
  – Subtype polymorphism; implicit upcasts in typed languages
  – Notion of behavioral replacement: subtypes acting like supertypes
  – Relationship between subtyping and inheritance
 	
• Object-oriented idioms for encapsulation
 	
  – Privacy and visibility of class members
  – Interfaces revealing only method signatures
  – Abstract base classes
 	
• Using collection classes, iterators, and other common library components
	• Design and implement a class [Usage]
	
	• Use subclassing to design simple class hierarchies that allow code to be reused for distinct subclasses [Usage]
	
	• Correctly reason about control flow in a program using dynamic dispatch [Usage]
	
	• Compare and contrast (1) the procedural/functional approach—defining a function for each operation with the function body providing a case for each data variant—and (2) the object-oriented approach—defining a class for each data variant with the class definition providing a method for each operation Understand both as defining a matrix of operations and variants [Usage]
	
	• Explain the relationship between object-oriented inheritance (code-sharing and overriding) and subtyping (the idea of a subtype being usable in a context that expects the supertype) [Usage]
	
	• Use object-oriented encapsulation mechanisms such as interfaces and private members [Usage]
	
	• Define and use iterators and other operations on aggregates, including operations that take functions as arguments, in multiple programming languages, selecting the most natural idioms for each language [Usage]

Readings: stroustrup2013	

UNIT 3: Algorithms and Design (5)	
Competences:	
Content	**Generales Goals**

• The concept and properties of algorithms
 	
   – Informal comparison of algorithm efficiency (e.g., operation counts)
 	
• The role of algorithms in the problem-solving process
 	
• Problem-solving strategies
 	
   – Iterative and recursive mathematical functions
 	
   – Iterative and recursive traversal of data structures
 	
   – Divide-and-conquer strategies
 	
• Fundamental design concepts and principles
 	
   – Abstraction
 	
   – Program decomposition
 	
   – Encapsulation and information hiding
 	
   – Separation of behaivor and implementation
	• Discuss the importance of algorithms in the problem-solving process [Usage]
	
	• Discuss how a problem may be solved by multiple algorithms, each with different properties [Usage]
	
	• Create algorithms for solving simple problems [Usage]
	
	• Use a programming language to implement, test, and debug algorithms for solving simple problems [Usage]
	
	• Implement, test, and debug simple recursive functions and procedures [Usage]
	
	• Determine whether a recursive or iterative solution is most appropriate for a problem [Usage]
	
	• Implement a divide-and-conquer algorithm for solving a problem [Usage]
	
	• Apply the techniques of decomposition to break a program into smaller pieces [Usage]
	
	• Identify the data components and behaviors of multiple abstract data types [Usage]
	
	• Implement a coherent abstract data type, with loose coupling between components and behaviors [Usage]
	
	• Identify the relative strengths and weaknesses among multiple designs or implementations for a problem [Usage]

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

UNIT 4: Basic Analysis (3)	
Competences:	
Content	**Generales Goals**
- Differences among best, expected, and worst case behaviors of an algorithm - Asymptotic analysis of upper and expected complexity bounds - Big O notation: formal definition - Complexity classes, such as constant, logarithmic, linear, quadratic, and exponential - Empirical measurements of performance - Time and space trade-offs in algorithms - Big O notation: use - Little o, big omega and big theta notation - Recurrence relations - Analysis of iterative and recursive algorithms - Master Theorem and Recursion Trees	- Explain what is meant by "best", "expected", and "worst" case behavior of an algorithm [Usage] - In the context of specific algorithms, identify the characteristics of data and/or other conditions or assumptions that lead to different behaviors [Usage] - Determine informally the time and space complexity of different algorithms [Usage] - State the formal definition of big O [Usage] - List and contrast standard complexity classes [Usage] - Perform empirical studies to validate hypotheses about runtime stemming from mathematical analysis Run algorithms on input of various sizes and compare performance [Usage] - Give examples that illustrate time-space trade-offs of algorithms [Usage] - Use big O notation formally to give asymptotic upper bounds on time and space complexity of algorithms [Usage] - Use big O notation formally to give expected case bounds on time complexity of algorithms [Usage] - Explain the use of big omega, big theta, and little o notation to describe the amount of work done by an algorithm [Usage] - Use recurrence relations to determine the time complexity of recursively defined algorithms [Usage] - Solve elementary recurrence relations, eg, using some form of a Master Theorem [Usage]
Readings: stroustrup2013	

UNIT 5: Basic Type Systems (5)	
Competences:	
Content	**Generales Goals**
- A type as a set of values together with a set of operations - Primitive types (e.g., numbers, Booleans) - Compound types built from other types (e.g., records, unions, arrays, lists, functions, references) - Association of types to variables, arguments, results, and fields - Type safety and errors caused by using values inconsistently given their intended types - Goals and limitations of static typing - Eliminating some classes of errors without running the program - Undecidability means static analysis must conservatively approximate program behavior - Generic types (parametric polymorphism) - Definition - Use for generic libraries such as collections - Comparison with ad hoc polymorphism (overloading) and subtype polymorphism - Complementary benefits of static and dynamic typing - Errors early vs. errors late/avoided - Enforce invariants during code development and code maintenance vs. postpone typing decisions while prototyping and conveniently allow flexible coding patterns such as heterogeneous collections - Avoid misuse of code vs. allow more code reuse - Detect incomplete programs vs. allow incomplete programs to run	- For both a primitive and a compound type, informally describe the values that have that type [Usage] - For a language with a static type system, describe the operations that are forbidden statically, such as passing the wrong type of value to a function or method [Usage] - Describe examples of program errors detected by a type system [Usage] - For multiple programming languages, identify program properties checked statically and program properties checked dynamically [Usage] - Give an example program that does not type-check in a particular language and yet would have no error if run [Usage] - Use types and type-error messages to write and debug programs [Usage] - Explain how typing rules define the set of operations that are legal for a type [Usage] - Write down the type rules governing the use of a particular compound type [Usage] - Explain why undecidability requires type systems to conservatively approximate program behavior [Usage] - Define and use program pieces (such as functions, classes, methods) that use generic types, including for collections [Usage] - Discuss the differences among generics, subtyping, and overloading [Usage] - Explain multiple benefits and limitations of static typing in writing, maintaining, and debugging software [Usage]
Readings: stroustrup2013	

UNIT 6: Fundamental Data Structures and Algorithms (3)	
Competences:	
Content	**Generales Goals**

• Simple numerical algorithms, such as computing the average of a list of numbers, finding the min, max,
 	
• Sequential and binary search algorithms
 	
• Worst case quadratic sorting algorithms (selection, insertion)
 	
• Worst or average case O(N log N) sorting algorithms (quicksort, heapsort, mergesort)
 	
• Hash tables, including strategies for avoiding and resolving collisions
 	
• Binary search trees
 	
   – Common operations on binary search trees such as select min, max, insert, delete, iterate over tree
 	
• Graphs and graph algorithms
 	
   – Representations of graphs (e.g., adjacency list, adjacency matrix)
 	
   – Depth- and breadth-first traversals
 	
• Heaps
 	
• Graphs and graph algorithms
 	
   – Maximum and minimum cut problem
 	
   – Local search
 	
• Pattern matching and string/text algorithms (e.g., substring matching, regular expression matching, longest common subsequence algorithms)
	• Implement basic numerical algorithms [Usage]
	
	• Implement simple search algorithms and explain the differences in their time complexities [Usage]
	
	• Be able to implement common quadratic and O(N log N) sorting algorithms [Usage]
	
	• Describe the implementation of hash tables, including collision avoidance and resolution [Usage]
	
	• Discuss the runtime and memory efficiency of principal algorithms for sorting, searching, and hashing [Usage]
	
	• Discuss factors other than computational efficiency that influence the choice of algorithms, such as programming time, maintainability, and the use of application-specific patterns in the input data [Usage]
	
	• Explain how tree balance affects the efficiency of various binary search tree operations [Usage]
	
	• Solve problems using fundamental graph algorithms, including depth-first and breadth-first search [Usage]
	
	• Demonstrate the ability to evaluate algorithms, to select from a range of possible options, to provide justification for that selection, and to implement the algorithm in a particular context [Usage]
	
	• Describe the heap property and the use of heaps as an implementation of priority queues [Usage]
	
	• Solve problems using graph algorithms, including single-source and all-pairs shortest paths, and at least one minimum spanning tree algorithm [Usage]
	
	• Trace and/or implement a string-matching algorithm [Usage]

Readings: stroustrup2013, Pai and Abraham (2018)	

UNIT 7: Event-Driven and Reactive Programming (2)	
Competences:	
Content	**Generales Goals**
- Events and event handlers - Canonical uses such as GUIs, mobile devices, robots, servers - Using a reactive framework - Defining event handlers/listeners - Main event loop not under event-handler-writer's control - Externally-generated events and program-generated events - Separation of model, view, and controller	- Write event handlers for use in reactive systems, such as GUIs [Usage] - Explain why an event-driven programming style is natural in domains where programs react to external events [Usage] - Describe an interactive system in terms of a model, a view, and a controller [Usage]
Readings: stroustrup2013, Williams (2011)	

UNIT 8: Graphs and Trees (7)	
Competences:	
Content	**Generales Goals**
- Trees - Properties - Traversal strategies - Undirected graphs - Directed graphs - Weighted graphs - Spanning trees/forests - Graph isomorphism	- Illustrate by example the basic terminology of graph theory, and some of the properties and special cases of each type of graph/tree [Usage] - Demonstrate different traversal methods for trees and graphs, including pre, post, and in-order traversal of trees [Usage] - Model a variety of real-world problems in computer science using appropriate forms of graphs and trees, such as representing a network topology or the organization of a hierarchical file system [Usage] - Show how concepts from graphs and trees appear in data structures, algorithms, proof techniques (structural induction), and counting [Usage] - Explain how to construct a spanning tree of a graph [Usage] - Determine if two graphs are isomorphic [Usage]
Readings: Nakariakov (2013)	

UNIT 9: Software Design (6)	
Competences:	
Content	**Generales Goals**

Content

- System design principles: levels of abstraction (architectural design and detailed design), separation of concerns, information hiding, coupling and cohesion , re-use of standard structures

- Design Paradigms such as structured design (top-down functional decomposition), object-oriented analysis and design, event driven design, component-level design, data-structured centered, aspect oriented, function oriented, service oriented

- Structural and behavioral models of software designs

- Design patterns

- Relationships between requirements and designs: transformation of models, design of contracts, invariants

- Software architecture concepts and standard architectures (e.g. client-server, n-layer, transform centered, pipes-and-filters)

- The use of component desing: component selection, design, adaptation and assembly of components, component and patterns, components and objects (for example, building a GUI using a standar widget set)

- Refactoring designs using design patterns

- Internal design qualities, and models for them: efficiency and performance, redundancy and fault tolerance, traceability of requeriments

- Measurement and analysis of design quality

- Tradeoffs between different aspects of quality

- Application frameworks

- Middleware: the object-oriented paradigm within middleware, object request brokers and marshalling, transaction processing monitors, workflow systems

- Principles of secure design and coding

  - Principle of least privilege
  - Principle of fail-safe defaults
  - Principle of psychological acceptability

Generales Goals

- Articulate design principles including separation of concerns, information hiding, coupling and cohesion, and encapsulation [Usage]

- Use a design paradigm to design a simple software system, and explain how system design principles have been applied in this design [Usage]

- Construct models of the design of a simple software system that are appropriate for the paradigm used to design it [Usage]

- Within the context of a single design paradigm, describe one or more design patterns that could be applicable to the design of a simple software system [Usage]

- For a simple system suitable for a given scenario, discuss and select an appropriate design paradigm [Usage]

- Create appropriate models for the structure and behavior of software products from their requirements specifications [Usage]

- Explain the relationships between the requirements for a software product and its design, using appropriate models [Usage]

- For the design of a simple software system within the context of a single design paradigm, describe the software architecture of that system [Usage]

- Given a high-level design, identify the software architecture by differentiating among common software architectures such as 3-tier, pipe-and-filter, and client-server [Usage]

- Investigate the impact of software architectures selection on the design of a simple system [Usage]

- Apply simple examples of patterns in a software design [Usage]

- Describe a form of refactoring and discuss when it may be applicable [Usage]

- Select suitable components for use in the design of a software product [Usage]

- Explain how suitable components might need to be adapted for use in the design of a software product [Usage]

- Design a contract for a typical small software component for use in a given system [Usage]

- Discuss and select appropriate software architecture for a simple system suitable for a given scenario [Usage]

- Apply models for internal and external qualities in designing software components to achieve an accept-able tradeoff between conflicting quality aspects [Us-

9

UNIT 10: Requirements Engineering (1)	
Competences:	
Content	**Generales Goals**
• Describing functional requirements using, for example, use cases or users stories • Properties of requirements including consistency, validity, completeness, and feasibility • Software requirements elicitation • Describing system data using, for example, class diagrams or entity-relationship diagrams • Non functional requirements and their relationship to software quality • Evaluation and use of requirements specifications • Requirements analysis modeling techniques • Acceptability of certainty / uncertainty considerations regarding software / system behavior • Prototyping • Basic concepts of formal requirements specification • Requirements specification • Requirements validation • Requirements tracing	• List the key components of a use case or similar description of some behavior that is required for a system [Usage] • Describe how the requirements engineering process supports the elicitation and validation of behavioral requirements [Usage] • Interpret a given requirements model for a simple software system [Usage] • Describe the fundamental challenges of and common techniques used for requirements elicitation [Usage] • List the key components of a data model (eg, class diagrams or ER diagrams) [Usage] • Identify both functional and non-functional requirements in a given requirements specification for a software system [Usage] • Conduct a review of a set of software requirements to determine the quality of the requirements with respect to the characteristics of good requirements [Usage] • Apply key elements and common methods for elicitation and analysis to produce a set of software requirements for a medium-sized software system [Usage] • Compare the plan-driven and agile approaches to requirements specification and validation and describe the benefits and risks associated with each [Usage] • Use a common, non-formal method to model and specify the requirements for a medium-size software system [Usage] • Translate into natural language a software requirements specification (eg, a software component contract) written in a formal specification language [Usage] • Create a prototype of a software system to mitigate risk in requirements [Usage] • Differentiate between forward and backward tracing and explain their roles in the requirements validation process [Usage]
Readings: stroustrup2013	

8. Methodology

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. Assessment

Continuous Assessment 1 : 20 %

Partial Exam : 30 %

Continuous Assessment 2 : 20 %

Final exam : 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 Platforml.
Pai, Praseed and Peter Abraham (2018). *C++ Reactive Programming*. 1st. Packt.
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
Undergraduate Program in Information Systems
SILABO

CS1D3. Abstract Algebra (Mandatory)

2022-II

1. General information		
1.1 School	:	Sistemas de Información
1.2 Course	:	CS1D3. Abstract Algebra
1.3 Semester	:	3er Semestre.
1.4 Prerrequisites	:	

- CS1D1. Discrete Structures I. (1st Sem)

- CS112. Computer Science I. (2nd Sem)

1.5 Type of course	:	Mandatory
1.6 Learning modality	:	Face to face
1.7 Horas	:	2 HT; 2 HL;
1.8 Credits	:	3

2. Professors

3. Course foundation
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. Summary
1. 2. 3. Cryptography 4.

5. Generales Goals

- 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. Contribution to Outcomes

This discipline contributes to the achievement of the following outcomes:

1) Analyze a complex computing problem and to apply principles of computing and other relevant disciplines to identify solutions. (**Assessment**)

6) Apply computer science theory and software development fundamentals to produce computing-based solutions. (**Assessment**)

7. Content

UNIT 1: (16)	
Competences:	
Content	**Generales Goals**
• Número enteros, algoritmos de la división, máximo común divisor, algoritmo de Euclides y algoritmo extendido de Euclides. Ecuaciones diofánticas • Aritmética Modular y Operaciones en Zn: 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.	• Realizar cálculos que involucren aritmética modular [Usage] • Describir algoritmos numérico teóricos básicos eficientes, incluyendo el algoritmo de Euclides y el algoritmo extendido de Euclides. [Assessment] • Establecer la importancia del estudio de la teoría de números. [Familiarity] • Discuss the importance of prime numbers in cryptography and explain their use in cryptographic algorithms [Familiarity]
Readings: Rosen (2011), Grimaldi (2003), Koshy (2007)	

UNIT 2: (14)	
Competences:	
Content	**Generales Goals**
• 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.	• Adquirir habilidad en la resolución de problemas abstractos y en la formulación de conjeturas . [Familiarity] • Argumentar como los principales teoremas y algoritmos permiten resolver problemas criptográficos. [Assessment]
Readings: Grimaldi (2003), Gallian (2012), Koshy (2007)	

UNIT 3: Cryptography (20)	
Competences:	
Content	**Generales Goals**

Content

- Basic Cryptography Terminology covering notions pertaining to the different (communication) partners, secure/unsecure channel, attackers and their capabilities, encryption, decryption, keys and their characteristics, signatures

- Cipher types (e.g., Caesar cipher, affine cipher) together with typical attack methods such as frequency analysis

- Public Key Infrastructure support for digital signature and encryption and its challenges

- Mathematical Preliminaries essential for cryptography, including topics in linear algebra, number theory, probability theory, and statistics

- Cryptographic primitives:

  - pseudo-random generators and stream ciphers

  - block ciphers (pseudo-random permutations), e.g., AES

  - pseudo-random functions

  - hash functions, e.g., SHA2, collision resistance

  - message authentication codes

  - key derivations functions

- Symmetric key cryptography

  - Perfect secrecy and the one time pad

  - Modes of operation for semantic security and authenticated encryption (e.g., encrypt-then-MAC, OCB, GCM)

  - Message integrity (e.g., CMAC, HMAC)

- Public key cryptography:

  - Trapdoor permutation, e.g., RSA

  - Public key encryption, e.g., RSA encryption, EI Gamal encryption

  - Digital signatures

  - Public-key infrastructure (PKI) and certificates

  - Hardness assumptions, e.g., Diffie-Hellman, integer factoring

- Authenticated key exchange protocols, e.g., TLS

- Cryptographic protocols: challenge-response authentication, zero-knowledge protocols, commitment, oblivious transfer, secure 2-party or multi-party computation, secret sharing, and applications

- Motivate concepts using real-world applications, e.g., electronic cash, secure channels between clients and servers, secure electronic mail, entity authentication, device pairing, voting systems.

- Security definitions and attacks on cryptographic primitives:

Generales Goals

- Describe the purpose of Cryptography and list ways it is used in data communications [Familiarity]

- Define the following terms: Cipher, Cryptanalysis, Cryptographic Algorithm, and Cryptology and describe the two basic methods (ciphers) for transforming plain text in cipher text [Familiarity]

- Discuss the importance of prime numbers in cryptography and explain their use in cryptographic algorithms [Familiarity]

- Explain how Public Key Infrastructure supports digital signing and encryption and discuss the limitations/vulnerabilities [Familiarity]

- Use cryptographic primitives and their basic properties [Familiarity]

- Illustrate how to measure entropy and how to generate cryptographic randomness [Familiarity]

- Use public-key primitives and their applications [Familiarity]

- Explain how key exchange protocols work and how they fail [Familiarity]

- Discuss cryptographic protocols and their properties [Familiarity]

- Describe real-world applications of cryptographic primitives and protocols [Familiarity]

- Summarize precise security definitions, attacker capabilities and goals [Familiarity]

- Apply appropriate known cryptographic techniques for a given scenario [Familiarity]

- Appreciate the dangers of inventing one's own cryptographic methods [Familiarity]

- Describe quantum cryptography and the impact of quantum computing on cryptographic algorithms [Familiarity]

3

UNIT 4: (10)	
Competences:	
Content	**Generales Goals**
• Elementos, proceso de transmitir una palabra • Esquemas de codificación: paridad, triple repetición, verificación de paridad y generación de códigos de grupo.	• Utilizar las propiedades de las estructuras algebraicas en el estudio de la teoria algebraica de los códigos. [Familiarity] • Aplicar técnicas que permitan la detección de errores, y si es necesario, proveer de métodos para reconstruir palabras originales. [Usage]
Readings: Grimaldi (2003), W.Trappe and Washington (2005)	

8. Methodology

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. Assessment

Continuous Assessment 1 : 20 %

Partial Exam : 30 %

Continuous Assessment 2 : 20 %

Final exam : 30 %

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
Undergraduate Program in Information Systems
SILABO

CS221. Computer Architecture (Mandatory)

2022-II

1. General information		
1.1 School	:	Sistemas de Información
1.2 Course	:	CS221. Computer Architecture
1.3 Semester	:	3er Semestre.
1.4 Prerrequisites	:	CS1D2. Discrete Structures II. (2nd Sem)
1.5 Type of course	:	Mandatory
1.6 Learning modality	:	Face to face
1.7 Horas	:	2 HT; 2 HL;
1.8 Credits	:	3

2. Professors

3. Course foundation

A computer scientist must have a solid knowledge of the organization and design principles of diverse computer systems, by understanding the limitations of modern systems they could propose next-gen paradigms. This course teaches the basics and principles of Computer Architecture. This class addresses digital logic design, basics of Computer Architecture and processor design (Instruction Set architecture, microarchitecture, out-of-order execution, branch prediction), execution paradigms (superscalar, dataflow, VLIW, SIMD, GPUs, systolic, multithreading) and memory system organization.

4. Summary

1. Digital logic and digital systems 2. Machine level representation of data 3. Assembly level machine organization 4. Functional organization 5. Memory system organization and architecture 6. Interfacing and communication 7. Multiprocessing and alternative architectures 8. Performance enhancements

5. Generales Goals

- Provide a first approach in Computer Architecture.

- Study the design and evolution of computer architectures, which lead to modern approaches and implementations in computing systems.

- Provide fine-grained details of computer hardware, and its relation with software execution.

- Implement a simple microprocessor using Verilog language.

6. Contribution to Outcomes

This discipline contributes to the achievement of the following outcomes:

1) Analyze a complex computing problem and to apply principles of computing and other relevant disciplines to identify solutions. (**Usage**)

6) Apply computer science theory and software development fundamentals to produce computing-based solutions. (**Assessment**)

7. Content

UNIT 1: Digital logic and digital systems (18)	
Competences:	
Content	**Generales Goals**

Content	Generales Goals
- Overview and history of computer architecture - Combinational and sequential logic/Field programmable gate arrays as a fundamental combinational + sequential logic building block - Abstraction models - Computer-aided design tools that process hardware and architectural representations - Register transfer notation/Hardware Description Language (Verilog/VHDL) - Physical constraints (gate delays, fan-in, fan-out, energy/power)	- Describe the progression of technology devices from vacuum tubes to VLSI, from mainframe computer architectures to the organization of warehouse-scale computers [Familiarity] - Comprehend the trend of modern computer architectures towards multi-core and that parallelism is inherent in all hardware systems [Usage] - Explain the implications of the "power wall" in terms of further processor performance improvements and the drive towards harnessing parallelism [Usage] - Articulate that there are many equivalent representations of computer functionality, including logical expressions and gates, and be able to use mathematical expressions to describe the functions of simple combinational and sequential circuits [Familiarity] - Design the basic building blocks of a computer: arithmetic-logic unit (gate-level), registers (gate-level), central processing unit (register transfer-level), memory (register transfer-level) [Usage] - Use CAD tools for capture, synthesis, and simulation to evaluate simple building blocks (eg, arithmetic-logic unit, registers, movement between registers) of a simple computer design [Familiarity] - Evaluate the functional and timing diagram behavior of a simple processor implemented at the logic circuit level [Assessment]
Readings: 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)	

UNIT 2: Machine level representation of data (8)	
Competences:	
Content	Generales Goals
- Bits, bytes, and words - Numeric data representation and number bases - Fixed- and floating-point systems - Signed and twos-complement representations - Representation of non-numeric data (character codes, graphical data) - Representation of registers and arrays	- Explain why everything is data, including instructions, in computers [Assessment] - Explain the reasons for using alternative formats to represent numerical data [Familiarity] - Describe how negative integers are stored in sign-magnitude and twos-complement representations [Usage] - Explain how fixed-length number representations affect accuracy and precision [Usage] - Describe the internal representation of non-numeric data, such as characters, strings, records, and arrays [Usage] - Convert numerical data from one format to another [Usage]
Readings: 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)	

UNIT 3: Assembly level machine organization (8)	
Competences:	
Content	**Generales Goals**
- Basic organization of the von Neumann machine - Control unit; instruction fetch, decode, and execution - Instruction sets and types (data manipulation, control, I/O) - Assembly/machine language programming - Instruction formats - Addressing modes - Subroutine call and return mechanisms - I/O and interrupts - Heap vs. Static vs. Stack vs. Code segments	- Explain the organization of the classical von Neumann machine and its major functional units [Familiarity] - Describe how an instruction is executed in a classical von Neumann machine, with extensions for threads, multiprocessor synchronization, and SIMD execution [Familiarity] - Describe instruction level parallelism and hazards, and how they are managed in typical processor pipelines [Familiarity] - Summarize how instructions are represented at both the machine level and in the context of a symbolic assembler [Familiarity] - Demonstrate how to map between high-level language patterns into assembly/machine language notations [Usage] - Explain different instruction formats, such as addresses per instruction and variable length vs fixed length formats [Usage] - Explain how subroutine calls are handled at the assembly level [Usage] - Explain the basic concepts of interrupts and I/O operations [Familiarity] - Write simple assembly language program segments [Usage] - Show how fundamental high-level programming constructs are implemented at the machine-language level [Usage]
Readings: 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)	

UNIT 4: Functional organization (8)	
Competences:	
Content	**Generales Goals**
• Implementation of simple datapaths, including instruction pipelining, hazard detection and resolution • Control unit: microprogrammed • Instruction pipelining • Introduction to instruction-level parallelism (ILP)	• Compare alternative implementation of datapaths [Assessment] • Discuss the concept of control points and the generation of control signals using hardwired or microprogrammed implementations [Familiarity] • Explain basic instruction level parallelism using pipelining and the major hazards that may occur [Usage] • Design and implement a complete processor, including datapath and control [Usage] • Determine, for a given processor and memory system implementation, the average cycles per instruction [Assessment]
Readings: 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)	

UNIT 5: Memory system organization and architecture (8)	
Competences:	
Content	**Generales Goals**
• Storage systems and their technology • Memory hierarchy: importance of temporal and spatial locality • Main memory organization and operations • Latency, cycle time, bandwidth, and interleaving • Cache memories (address mapping, block size, replacement and store policy) • Multiprocessor cache consistency/Using the memory system for inter-core synchronization/atomic memory operations • Virtual memory (page table, TLB) • Fault handling and reliability • Error coding, data compression, and data integrity	• Identify the main types of memory technology (eg, SRAM, DRAM, Flash, magnetic disk) and their relative cost and performance [Familiarity] • Explain the effect of memory latency on running time [Familiarity] • Describe how the use of memory hierarchy (cache, virtual memory) is used to reduce the effective memory latency [Usage] • Describe the principles of memory management [Usage] • Explain the workings of a system with virtual memory management [Usage] • Compute Average Memory Access Time under a variety of cache and memory configurations and mixes of instruction and data references [Assessment]
Readings: 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)	

UNIT 6: Interfacing and communication (8)	
Competences:	
Content	**Generales Goals**
• I/O fundamentals: handshaking, buffering, programmed I/O, interrupt-driven I/O • Interrupt structures: vectored and prioritized, interrupt acknowledgment • External storage, physical organization, and drives • Buses: bus protocols, arbitration, direct-memory access (DMA) • Introduction to networks: communications networks as another layer of remote access • Multimedia support • RAID architectures	• Explain how interrupts are used to implement I/O control and data transfers [Familiarity] • Identify various types of buses in a computer system [Familiarity] • Describe data access from a magnetic disk drive [Usage] • Compare common network organizations, such as ethernet/bus, ring, switched vs routed [Assessment] • Identify the cross-layer interfaces needed for multimedia access and presentation, from image fetch from remote storage, through transport over a communications network, to staging into local memory, and final presentation to a graphical display [Familiarity] • Describe the advantages and limitations of RAID architectures [Familiarity]
Readings: 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)	

UNIT 7: Multiprocessing and alternative architectures (8)	
Competences:	
Content	**Generales Goals**
• Power Law • Example SIMD and MIMD instruction sets and architectures • Interconnection networks (hypercube, shuffle-exchange, mesh, crossbar) • Shared multiprocessor memory systems and memory consistency • Multiprocessor cache coherence	• Discuss the concept of parallel processing beyond the classical von Neumann model [Assessment] • Describe alternative parallel architectures such as SIMD and MIMD [Familiarity] • Explain the concept of interconnection networks and characterize different approaches [Usage] • Discuss the special concerns that multiprocessing systems present with respect to memory management and describe how these are addressed [Familiarity] • Describe the differences between memory backplane, processor memory interconnect, and remote memory via networks, their implications for access latency and impact on program performance [Assessment]
Readings: 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)	

UNIT 8: Performance enhancements (8)	
Competences:	
Content	**Generales Goals**
- Superscalar architecture - Branch prediction, Speculative execution, Out-of-order execution - Prefetching - Vector processors and GPUs - Hardware support for multithreading - Scalability - Alternative architectures, such as VLIW/EPIC, and Accelerators and other kinds of Special-Purpose Processors	- Describe superscalar architectures and their advantages [Familiarity] - Explain the concept of branch prediction and its utility [Usage] - Characterize the costs and benefits of prefetching [Assessment] - Explain speculative execution and identify the conditions that justify it [Assessment] - Discuss the performance advantages that multithreading offered in an architecture along with the factors that make it difficult to derive maximum benefits from this approach [Assessment] - Describe the relevance of scalability to performance [Assessment]
Readings: 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. Methodology
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. Assessment
Continuous Assessment 1 : 20 %
Partial Exam : 30 %
Continuous Assessment 2 : 20 %
Final exam : 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
Undergraduate Program in Information Systems
SILABO

CS2B1. Platform Based Development (Mandatory)

2022-II

1. General information		
1.1 School	:	Sistemas de Información
1.2 Course	:	CS2B1. Platform Based Development
1.3 Semester	:	3er Semestre.
1.4 Prerrequisites	:	CS112. Computer Science I. (2nd Sem)
1.5 Type of course	:	Mandatory
1.6 Learning modality	:	Face to face
1.7 Horas	:	1 HT; 2 HP; 2 HL;
1.8 Credits	:	3

2. Professors

3. Course foundation
The world has changed due to the use of fabric and related technologies, rapid, timely and personalized access to the information, through web technology, ubiquitous and pervasive; they have changed the way we do things, how do we think? and how does the industry develop? Web technologies, ubiquitous and pervasive are based on the development of web services, web applications and mobile applications, which are necessary to understand the architecture, design, and implementation of web services, web applications and mobile applications.

4. Summary
1. Introduction 2. Web Platforms 3. Desarrollo de servicios y aplicaciones web 4. Mobile Platforms 5. Mobile Applications for Android Handheld Systems

5. Generales Goals
- That the student is able to design and implement services, web applications using tools and languages such as HTML, CSS, JavaScript (including AJAX), back-end scripting and a database, at an intermediate level.   - That the student is able to develop mobile applications, administration of web servers in a Unix system and an introduction to web security, at an intermediate level.

6. Contribution to Outcomes

This discipline contributes to the achievement of the following outcomes:

2) Design, implement and evaluate a computing-based solution to meet a given set of computing requirements in the context of the program's discipline. (**Usage**)

3) Communicate effectively in a variety of professional contexts. (**Usage**)

5) Function effectively as a member or leader of a team engaged in activities appropriate to the program's discipline. (**Usage**)

6) Apply computer science theory and software development fundamentals to produce computing-based solutions. (**Usage**)

7) Develop computational technology for the well-being of all, contributing with human formation, scientific, technological and professional skills to solve social problems of our community. (**Usage**)

7. Content

UNIT 1: Introduction (5)	
Competences:	
Content	**Generales Goals**
• Overview of platforms (e.g., Web, Mobile, Game, Industrial) • Programming via platform-specific APIs • Overview of Platform Languages (e.g., Objective C, HTML5) • Programming under platform constraints	• Describe how platform-based development differs from general purpose programming [Familiarity] • List characteristics of platform languages [Familiarity] • Write and execute a simple platform-based program [Familiarity] • List the advantages and disadvantages of programming with platform constraints [Familiarity]
Readings: fielding2000fielding, grove2009web, annuzzi2013introduction, Cornez2015	

UNIT 2: Web Platforms (5)	
Competences:	
Content	**Generales Goals**
• Web programming languages (e.g., HTML5, Java Script, PHP, CSS) • • Web Platform constraints: Client-Server, Stateless-Stateful, Cache, Uniform Interface, Layered System, Code on Demand, ReST. • Web platform constraints • Software as a Service (SaaS) • Web standards	• Design and Implement a simple web application [Familiarity] • Describe the constraints that the web puts on developers [Familiarity] • Compare and contrast web programming with general purpose programming [Familiarity] • Describe the differences between Software-as-a-Service and traditional software products [Familiarity] • Discuss how web standards impact software development [Familiarity] • Review an existing web application against a current web standard [Familiarity]
Readings: fielding2000fielding	

UNIT 3: Desarrollo de servicios y aplicaciones web (25)	
Competences:	
Content	**Generales Goals**
- Describe, identify and debug issues related to web application development - Design and development of interactive web applications using HTML5 and Python - Use MySQL for data management and manipulate MySQL with Python - Design and development of asynchronous web applications using Ajax techniques - Using dynamic client side Javascript scripting language and server side python scripting language with Ajax - Apply XML / JSON technologies for data management with Ajax - Use framework, services and Ajax web APIs and apply design patterns to web application development	- Server-side python scripting language: variables, data types, operations, strings, functions, control statements, arrays, files and directory access, maintain state. [Usage] - Web programming approach using embedded python. [Usage] - Accessing and Manipulating MySQL. [Usage] - The Ajax web application development approach. [Usage] - DOM and CSS used in JavaScript. [Usage] - Asynchronous Content Update Technologies. [Usage] - XMLHttpRequest objects use to communicate between clients and servers. [Usage] - XML and JSON. [Usage] - XSLT and XPath as mechanisms for transforming XML documents. [Usage] - Web services and APIs (especially Google Maps). [Usage] - Macros Ajax for the development of contemporary web applications. [Usage] - Design patterns used in web applications. [Usage]
Readings: freeman2011head	

UNIT 4: Mobile Platforms (5)	
Competences:	
Content	**Generales Goals**
- Mobile programming languages - Design Principles: Segregation of Interfaces, Single Responsability, Separation of concerns, Dependency Inversion. - Challenges with mobility and wireless communication - Location-aware applications - Performance / power tradeoffs - Mobile platform constraints - Emerging technologies	- Design and implement a mobile application for a given mobile platform [Familiarity] - Discuss the constraints that mobile platforms put on developers [Familiarity] - Discuss the performance vs power tradeoff [Familiarity] - Compare and Contrast mobile programming with general purpose programming [Familiarity]
Readings: martin2017clean, annuzzi2013introduction	

UNIT 5: Mobile Applications for Android Handheld Systems (25)	
Competences:	
Content	**Generales Goals**
- 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	- Students identify necessary software and install it on their personal computers. - Students perform various tasks to familiarize themselves with the Android platform and Environment for development. [Usage] - Students build applications that trace the lifecycle callback methods emitted by the Android platform and demonstrate the behavior of Android when device configuration changes (for example, when the device moves from vertical to horizontal and vice versa). [Usage] - Students build applications that require starting multiple activities through both standard and custom methods. [Usage] - Students build applications that require standard and custom permissions. [Usage] - Students build an application that uses a single code base, but creates different user interfaces depending on the screen size of a device. [Usage] - Students construct a to-do list manager using the user interface elements discussed in class. The application allows users to create new items and to display them in a ListView. [Usage] - Students build an application that uses location information to collect latitude, length of places they visit. [Usage]
Readings: annuzzi2013introduction, **Cornez2015**	

8. Methodology
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. Assessment
Continuous Assessment 1 : 20 %
Partial Exam : 30 %
Continuous Assessment 2 : 20 %
Final exam : 30 %

Universidad Nacional de Colombia (UNAL) Sede Manizales
Undergraduate Program in Information Systems
SILABO

MA102. Calculus I (Mandatory)

2022-II

1. General information		
1.1 School	:	Sistemas de Información
1.2 Course	:	MA102. Calculus I
1.3 Semester	:	3er Semestre.
1.4 Prerrequisites	:	MA100. Mathematics I. (1st Sem)
1.5 Type of course	:	Mandatory
1.6 Learning modality	:	Face to face
1.7 Horas	:	2 HT; 4 HP;
1.8 Credits	:	4

2. Professors

3. Course foundation

This course introduces the first concepts of linear algebra as well as numerical methods with an emphasis on problem solving with the Scilab open source libe package. Mathematical theory is limited to fundamentals, while effective application for problem solving is privileged. In each subject, a few methods of relevance for engineering are taught. Knowledge of these methods prepares students for the search for more advanced alternatives, if required.

4. Summary

1. Introduction 2. Linear Algebra 3. Numerical methods

5. Generales Goals

- Ability to apply knowledge about Mathematics.

- Ability to apply engineering knowledge.

- Ability to apply the modern knowledge, techniques, skills and tools of modern engineering to the practice of engineering

6. Contribution to Outcomes

This discipline contributes to the achievement of the following outcomes:

1) Analyze a complex computing problem and to apply principles of computing and other relevant disciplines to identify solutions. (**Assessment**)

6) Apply computer science theory and software development fundamentals to produce computing-based solutions. (**Assessment**)

7. Content

UNIT 1: Introduction (18)	
Competences:	
Content	**Generales Goals**
• Importance of linear algebra and numerical methods. Examples.	• Be able to understand the basic concepts and importance of Linear Algebra and Numerical Methods.
Readings: Anton and Rorres (2014), Chapra and Canale (12015)	

UNIT 2: Linear Algebra (14)	
Competences:	
Content	**Generales Goals**

• Elementary matrix algebra and determinants
 	
• Null space and exact solutions of systems of linear equations Ax=b:
 	
   – Tridiagonal and triangular systems and Gaussian elimination with and without pivoting.
   – LU factorization and Crout algorithm.
 	
• Basics on eigenvalues and eigenvectors:
 	
   – Characteristic polynomials.
   – Algebraic and geometric multiplicities.
 	
• Least squares estimation.
 	
• Linear transformations.
	• Understanding the basics concepts of Linear Algebra.
	
	• Solve properly linear transformations problems.

Readings: Anton and Rorres (2014), Chapra and Canale (12015)	

UNIT 3: Numerical methods (22)	
Competences:	
Content	**Generales Goals**
- Basics on solutions of systems of linear equations Ax=b: Jacobi and Gauss Seidel methods. - Application of matrix factorizations to the solution of linear systems (singular value decomposition, QR, Cholesky) Numerical computation of null space, rank and condition number. - Root finding: - Bisection. - Fixed-point iteration. - Newton-Raphson methods. - Basics on interpolation: - Newton and Lagrange polynomial interpolations - Spline interpolation - Basics on numerical differentiation and Taylor approximation - Basics on numerical integration: - Trapezium, midpoint and Simpson rule - Gaussian quadrature - Basics on numerical solutions to ODEs: - Finite differences; Euler and Runge-Kutta methods - Converting higher order ODEs into a system of low order ODEs - Runge-Kutta methods for systems of equations - Single shooting method - Short introduction to optimization techniques: overview on linear programming, bounded linear systems, quadratic programming, gradient descent.	- Understanding the basics concepts of Numerical Methods. - Applying the most frequent methods for the resolution of mathematical problems. - Implementing and applying numerical algorithms for the solution of mathematical problems using the Scilab open-source computational package. - Applying Scilab for the solution of mathematical problems and for plotting graphs.
Readings: Anton and Rorres (2014), Chapra and Canale (12015)	

8. Methodology
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. Assessment

Continuous Assessment 1 : 20 %

Partial Exam : 30 %

Continuous Assessment 2 : 20 %

Final exam : 30 %

References

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

Chapra, S.C. and R.P. Canale (12015). *Numerical Methods for Engineers,* 7th. Vol. 1. McGraw-Hill.

Universidad Nacional de Colombia (UNAL) Sede Manizales
Undergraduate Program in
Information Systems
SILABO

CS210. Algorithms and Data Structures (Mandatory)

2022-II

1. General information		
1.1 School	:	Sistemas de Información
1.2 Course	:	CS210. Algorithms and Data Structures
1.3 Semester	:	4to Semestre.
1.4 Prerrequisites	:	CS113. Computer Science II. (3rd Sem)
1.5 Type of course	:	Mandatory
1.6 Learning modality	:	Face to face
1.7 Horas	:	2 HT; 2 HP; 2 HL;
1.8 Credits	:	4

2. Professors

3. Course foundation
The theoretical foundation of all branches of computing rests on algorithms and data structures, this course will provide participants with an introduction to these topics, thus forming a basis that will serve for the following courses in the career.

4. Summary
1. Graphs 2. Scatter Matrices 3. Balanced Trees

5. Generales Goals
- Make the student understand the importance of algorithms for solving problems.   - Introduce the student to the field of application of data structures.

6. Contribution to Outcomes
This discipline contributes to the achievement of the following outcomes:

1) Analyze a complex computing problem and to apply principles of computing and other relevant disciplines to identify solutions. (**Usage**)

2) Design, implement and evaluate a computing-based solution to meet a given set of computing requirements in the context of the program's discipline. (**Usage**)

6) Apply computer science theory and software development fundamentals to produce computing-based solutions. (**Usage**)

7. Content

UNIT 1: Graphs (12)	
Competences:	
Content	**Generales Goals**
• Graph Concept • Directed Graphs and Non-directed Graphs. • Using Graphs. • Measurement of efficiency ,in time and space. • Adjacency matrices. • Tag adjacent matrices. • Adjacency Lists. • Implementation of graphs using adjacency matrices. • Graph Implementation using adjacency lists • Insertion, search and deletion of nodes and edges. • Graph search algorithms.	• Acquire Dexterity to Perform Correct Implementation. [Usage] • Develop knowledge to decide when it is better to use one implementation technique than another. [Usage]
Readings: Cormen et al. (2009), Fager et al. (2014), Knuth (1997), Knuth (1998)	

UNIT 2: Scatter Matrices (8)	
Competences:	
Content	**Generales Goals**
• Initial concepts. • Dense Matrices • Measurement of Efficiency in Time and Space • Static scatter vs. dynamic matrix creation. • Insert, search, and delete methods.	• Understand the use and implementation of scatter matrices.[Assessment]
Readings: Cormen et al. (2009), Fager et al. (2014), Knuth (1997), Knuth (1998)	

UNIT 3: Balanced Trees (16)	
Competences:	
Content	**Generales Goals**
• AVL Trees. • Measurement of Efficiency. • Simple and Composite Rotations • Insertion, deletion and search. • Trees B , B+ B* y Patricia.	• Understand the basic functions of these complex structures in order to acquire the capacity for their implementation. [Assessment]
Readings: Cormen et al. (2009), Fager et al. (2014), Knuth (1997), Knuth (1998)	

8. Methodology
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. Assessment
Continuous Assessment 1 : 20 %   **Partial Exam** : 30 %   **Continuous Assessment 2** : 20 %   **Final exam** : 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
Undergraduate Program in Information Systems
SILABO

CS211. Computer Science Theory (Mandatory)

2022-II

1. General information		
1.1 School	:	Sistemas de Información
1.2 Course	:	CS211. Computer Science Theory
1.3 Semester	:	4to Semestre.
1.4 Prerrequisites	:	CS1D2. Discrete Structures II. (2nd Sem)
1.5 Type of course	:	Mandatory
1.6 Learning modality	:	Face to face
1.7 Horas	:	2 HT; 2 HP; 2 HL;
1.8 Credits	:	4

2. Professors

3. Course foundation
This course emphasizes formal languages, computer models and computability, as well as the fundamentals of computational complexity and complete NP problems.

4. Summary
1. Basic Automata Computability and Complexity 2. Advanced Computational Complexity 3. Advanced Automata Theory and Computability

5. Generales Goals
• That the student learn the fundamental concepts of the theory of formal languages.

6. Contribution to Outcomes
This discipline contributes to the achievement of the following outcomes:

1) Analyze a complex computing problem and to apply principles of computing and other relevant disciplines to identify solutions. (**Assessment**)

6) Apply computer science theory and software development fundamentals to produce computing-based solutions. (**Assessment**)

7. Content

UNIT 1: Basic Automata Computability and Complexity (20)	
Competences:	
Content	**Generales Goals**
- Finite-state machines - Regular expressions - The halting problem - Context-free grammars - Introduction to the P and NP classes and the P vs. NP problem - Introduction to the NP-complete class and exemplary NP-complete problems (e.g., SAT, Knapsack) - Turing machines, or an equivalent formal model of universal computation - Nondeterministic Turing machines - Chomsky hierarchy - The Church-Turing thesis - Computability - Rice's Theorem - Examples of uncomputable functions - Implications of uncomputability	- Discuss the concept of finite state machines [Assessment] - Design a deterministic finite state machine to accept a specified language [Assessment] - Generate a regular expression to represent a specified language [Assessment] - Explain why the halting problem has no algorithmic solution [Assessment] - Design a context-free grammar to represent a specified language [Assessment] - Define the classes P and NP [Assessment] - Explain the significance of NP-completeness [Assessment] - Explain the Church-Turing thesis and its significance [Familiarity] - Explain Rice's Theorem and its significance [Familiarity] - Provide examples of uncomputable functions [Familiarity] - Prove that a problem is uncomputable by reducing a classic known uncomputable problem to it [Familiarity]
Readings: Martin (2010), Linz (2011), Sipser (2012)	

UNIT 2: Advanced Computational Complexity (20)	
Competences:	
Content	**Generales Goals**
- Review of the classes P and NP; introduce P-space and EXP - Polynomial hierarchy - NP-completeness (Cook's theorem) - Classic NP-complete problems - Reduction Techniques	- Define the classes P and NP (Also appears in AL/Basic Automata, Computability, and Complexity) [Assessment] - Define the P-space class and its relation to the EXP class [Assessment] - Explain the significance of NP-completeness (Also appears in AL/Basic Automata, Computability, and Complexity) [Assessment] - Provide examples of classic NP-complete problems [Assessment] - Prove that a problem is NP-complete by reducing a classic known NP-complete problem to it [Assessment]
Readings: Martin (2010), Linz (2011), Sipser (2012), Hopcroft and Ullman (2013)	

UNIT 3: Advanced Automata Theory and Computability (20)	
Competences:	
Content	**Generales Goals**
- Sets and languages - Regular languages - Review of deterministic finite automata (DFAs) - Nondeterministic finite automata (NFAs) - Equivalence of DFAs and NFAs - Review of regular expressions; their equivalence to finite automata - Closure properties - Proving languages non-regular, via the pumping lemma or alternative means - Context-free languages - Push-down automata (PDAs) - Relationship of PDAs and context-free grammars - Properties of context-free languages	- Determine a language's place in the Chomsky hierarchy (regular, context-free, recursively enumerable) [Assessment] - Convert among equivalently powerful notations for a language, including among DFAs, NFAs, and regular expressions, and between PDAs and CFGs [Assessment]
Readings: Hopcroft and Ullman (2013), Brookshear (1993)	

---

8. Methodology

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. Assessment

Continuous Assessment 1 : 20 %

Partial Exam : 30 %

Continuous Assessment 2 : 20 %

Final exam : 30 %

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
Undergraduate Program in Information Systems
SILABO

CS271. Databases I (Mandatory)

2022-II

1. General information		
1.1 School	:	Sistemas de Información
1.2 Course	:	CS271. Databases I
1.3 Semester	:	4to Semestre.
1.4 Prerrequisites	:	CS1D3. Abstract Algebra. (3rd Sem)
1.5 Type of course	:	Mandatory
1.6 Learning modality	:	Face to face
1.7 Horas	:	2 HT; 4 HL;
1.8 Credits	:	4

2. Professors

3. Course foundation
Information management (IM) plays a major role in almost all areas where computers are used. This area includes the capture, digitization, representation, organization, transformation and presentation of information; Algorithms to improve the efficiency and effectiveness of accessing and updating stored information, data modeling and abstraction, and physical file storage techniques.It also covers information security, privacy, integrity and protection in a shared environment. Students need to be able to develop conceptual and physical data models, determine which (IM) methods and techniques are appropriate for a given problem, and be able to select and implement an appropriate IM solution that reflects all applicable restrictions, including Scalability and usability.

4. Summary
1. Database Systems 2. Data Modeling 3. Indexing 4. Relational Databases 5. Query Languages

5. Generales Goals
- That the student learn to represent information in a database prioritizing the efficiency in the recovery of the same.   - That the student learn the fundamental concepts of the management of databases. This includes the design of databases, database languages and the realization of databases.   - Discuss the database model with the base in relational algebra, relational calculus and the study of SQL statements.

6. Contribution to Outcomes

This discipline contributes to the achievement of the following outcomes:

1) Analyze a complex computing problem and to apply principles of computing and other relevant disciplines to identify solutions. (**Assessment**)

3) Communicate effectively in a variety of professional contexts. (**Usage**)

4) Recognize professional responsabilities and make informed judgments in computing practice based on legal and ethical principles. (**Usage**)

5) Function effectively as a member or leader of a team engaged in activities appropriate to the program's discipline. (**Usage**)

6) Apply computer science theory and software development fundamentals to produce computing-based solutions. (**Assessment**)

7) Develop computational technology for the well-being of all, contributing with human formation, scientific, technological and professional skills to solve social problems of our community. (**Usage**)

7. Content

UNIT 1: Database Systems (14)

Competences:

Content	Generales Goals
- Approaches to and evolution of database systems - Components of database systems - Design of core DBMS functions (e.g., query mechanisms, transaction management, buffer management, access methods) - Database architecture and data independence - Use of a declarative query language - Systems supporting structured and/or stream content - Approaches for managing large volumes of data (e.g., noSQL database systems, use of MapReduce).	- Explain the characteristics that distinguish the database approach from the approach of programming with data files [Usage] - Describe the most common designs for core database system components including the query optimizer, query executor, storage manager, access methods, and transaction processor [Usage] - Cite the basic goals, functions, and models of database systems [Usage] - Describe the components of a database system and give examples of their use [Usage] - Identify major DBMS functions and describe their role in a database system [Usage] - Explain the concept of data independence and its importance in a database system [Usage] - Use a declarative query language to elicit information from a database [Usage] - Describe facilities that datatbases provide supporting structures and/or stream (sequence) data, eg, text [Usage] - Describe major approaches to storing and processing large volumes of data [Usage]

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

UNIT 2: Data Modeling (14)	
Competences:	
Content	**Generales Goals**
- Data modeling - Conceptual models (e.g., entity-relationship, UML diagrams) - Spreadsheet models - Relational data models - Object-oriented models - Semi-structured data model (expressed using DTD or XML Schema, for example)	- Compare and contrast appropriate data models, including internal structures, for different types of data [Usage] - Describe concepts in modeling notation (eg, Entity-Relation Diagrams or UML) and how they would be used [Usage] - Define the fundamental terminology used in the relational data model [Usage] - Describe the basic principles of the relational data model [Usage] - Apply the modeling concepts and notation of the relational data model [Usage] - Describe the main concepts of the OO model such as object identity, type constructors, encapsulation, inheritance, polymorphism, and versioning [Usage] - Describe the differences between relational and semi-structured data models [Usage] - Give a semi-structured equivalent (eg, in DTD or XML Schema) for a given relational schema [Usage]
Readings: Simsion and Witt (2004), Elmasri and Navathe (2004), Korth and Silberschatz (2002)	

UNIT 3: Indexing (4)	
Competences:	
Content	**Generales Goals**
- The impact of indices on query performance - The basic structure of an index - Keeping a buffer of data in memory - Creating indexes with SQL - Indexing text - Indexing the web (e.g., web crawling)	- Generate an index file for a collection of resources [Usage] - Explain the role of an inverted index in locating a document in a collection [Usage] - Explain how stemming and stop words affect indexing [Usage] - Identify appropriate indices for given relational schema and query set [Usage] - Estimate time to retrieve information, when indices are used compared to when they are not used [Usage] - Describe key challenges in web crawling, eg, detecting duplicate documents, determining the crawling frontier [Usage]
Readings: Whitehorn and Marklyn (2001), Ramakrishnan and Gehrke (2003), Emil Eifrem and Robinson (2015), C.J (2011), Korth and Silberschatz (2002)	

UNIT 4: Relational Databases (14)	
Competences:	
Content	**Generales Goals**
- Mapping conceptual schema to a relational schema - Entity and referential integrity - Relational algebra and relational calculus - Relational Database design - Functional dependency - Decomposition of a schema; lossless-join and dependency-preservation properties of a decomposition - Candidate keys, superkeys, and closure of a set of attributes - Normal forms (BCNF) - Multi-valued dependency (4NF) - Join dependency (PJNF, 5NF) - Representation theory	- Prepare a relational schema from a conceptual model developed using the entity- relationship model [Usage] - Explain and demonstrate the concepts of entity integrity constraint and referential integrity constraint (including definition of the concept of a foreign key) [Usage] - Demonstrate use of the relational algebra operations from mathematical set theory (union, intersection, difference, and Cartesian product) and the relational algebra operations developed specifically for relational databases (select (restrict), project, join, and division) [Usage] - Write queries in the relational algebra [Usage] - Write queries in the tuple relational calculus [Usage] - Determine the functional dependency between two or more attributes that are a subset of a relation [Usage] - Connect constraints expressed as primary key and foreign key, with functional dependencies [Usage] - Compute the closure of a set of attributes under given functional dependencies [Usage] - Determine whether a set of attributes form a superkey and/or candidate key for a relation with given functional dependencies [Usage] - Evaluate a proposed decomposition, to say whether it has lossless-join and dependency-preservation [Usage] - Describe the properties of BCNF, PJNF, 5NF [Usage] - Explain the impact of normalization on the efficiency of database operations especially query optimization [Usage] - Describe what is a multi-valued dependency and what type of constraints it specifies [Usage]
Readings: Whitehorn and Marklyn (2001), Ramakrishnan and Gehrke (2003), Emil Eifrem and Robinson (2015), C.J (2011), Korth and Silberschatz (2002)	

UNIT 5: Query Languages (12)	
Competences:	
Content	**Generales Goals**
- Overview of database languages - SQL (data definition, query formulation, update sublanguage, constraints, integrity) - Selections - Projections - Select-project-join - Aggregates and group-by - Subqueries - QBE and 4th-generation environments - Different ways to invoke non-procedural queries in conventional languages - Introduction to other major query languages (e.g., XPATH, SPARQL) - Stored procedures	- Create a relational database schema in SQL that incorporates key, entity integrity, and referential integrity constraints [Usage] - Use SQL to create tables and retrieve (SELECT) information from a database [Usage] - Evaluate a set of query processing strategies and select the optimal strategy [Usage] - Create a non-procedural query by filling in templates of relations to construct an example of the desired query result [Usage] - Embed object-oriented queries into a stand-alone language such as C++ or Java (eg, SELECT ColMethod() FROM Object) [Usage] - Write a stored procedure that deals with parameters and has some control flow, to provide a given functionality [Usage]
Readings: Dietrich (2001), Elmasri and Navathe (2004), Celko (2005), Korth and Silberschatz (2002)	

8. Methodology
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. Assessment
Continuous Assessment 1 : 20 %
Partial Exam : 30 %
Continuous Assessment 2 : 20 %
Final exam : 30 %

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.

<div align="center">

Universidad Nacional de Colombia (UNAL) Sede Manizales
Undergraduate Program in Information Systems
SILABO

MA201. Calculus II (Mandatory)

</div>

2022-II

1. General information		
1.1 School	:	Sistemas de Información
1.2 Course	:	MA201. Calculus II
1.3 Semester	:	4to Semestre.
1.4 Prerrequisites	:	

- MA101. Mathematics II. (2nd Sem)

- MA102. Calculus I. (3rd Sem)

1.5 Type of course	:	Mandatory
1.6 Learning modality	:	Face to face
1.7 Horas	:	2 HT; 4 HP;
1.8 Credits	:	4

2. Professors

3. Course foundation
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. Summary
1. 2. 3. 4. 5. 6.

5. Generales Goals

- 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. Contribution to Outcomes
This discipline contributes to the achievement of the following outcomes:

1) Analyze a complex computing problem and to apply principles of computing and other relevant disciplines to identify solutions. (**Assessment**)

6) Apply computer science theory and software development fundamentals to produce computing-based solutions. (**Assessment**)

7. Content

UNIT 1: (8)

Competences:

Content	Generales Goals
R^3 como espacio euclídeo y álgebra . - Superficies básicas en el espacio.	- Manejar el álgebra vectorial en R^3[Usage]. - Identificar tipos de superficies en el espacio [Usage]. - Graficar superficies básicas [Usage].

Readings: Apóstol (1973), Simmons (1995)

UNIT 2: (20)

Competences:

Content	Generales Goals
- Funciones vectoriales de variable real. Reparametrizaciones - Diferenciación e integración - Velocidad, aceleración , curvatura, torsión	Describir las diferentes carácteristicas de una curva [Usage].

Readings: Apóstol (1973), Simmons (1995)

UNIT 3: (20)

Competences:

Content	Generales Goals
- Curvas de nivel - Límites y continuidad - Diferenciación	- Graficar campos escalares - Discutir la existencia de un límite y la continuidad de un campo escalar [Usage]. - Calcular derivadas parciales y totales [Usage].

Readings: Apóstol (1973), Bartle (1976), Simmons (1995)

UNIT 4: (12)

Competences:

Content	Generales Goals
- Máximos y mínimos - Multiplicadores de Lagrange	- Interpretar la noción de gradiente en curvas de nivel y en superficies de nivel [Usage]. - Usar técnicas para hallar extremos [Usage].

Readings: Apóstol (1973), Simmons (1995), Bartle (1976)

UNIT 5: (12)	
Competences:	
Content	**Generales Goals**
- Integración de Riemann - Integración sobre regiones - Cambio de coordenadas - Aplicaciones	- Reconocer regiones de integración adecuadas [Usage]. - Realizar cambios de coordenadas adecuados [Usage]. - Aplicar la integración múltiple a problemas [Usage].
Readings: Apóstol (1973)	

UNIT 6: (18)	
Competences:	
Content	**Generales Goals**
- Integrales de linea - Campos conservativos - Integrales de superficie	- Calcular la integral de linea de campos vectoriales[Usage]. - Reconocer campos conservativos[Usage]. - Hallar funciones potenciales de campos conservativos [Usage]. - Hallar integrales de superficies y aplicarlas [Usage].
Readings: Apóstol (1973)	

8. Methodology
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. Assessment
Continuous Assessment 1 : 20 %   **Partial Exam** : 30 %   **Continuous Assessment 2** : 20 %   **Final exam** : 30 %

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
Undergraduate Program in Information Systems
SILABO

MA203. Statistics and Probability (Mandatory)

2022-II

1. General information		
1.1 School	:	Sistemas de Información
1.2 Course	:	MA203. Statistics and Probability
1.3 Semester	:	4to Semestre.
1.4 Prerrequisites	:	MA102. Calculus I. (3rd Sem)
1.5 Type of course	:	Mandatory
1.6 Learning modality	:	Face to face
1.7 Horas	:	2 HT; 2 HP; 2 HL;
1.8 Credits	:	4

2. Professors

3. Course foundation
It provides an introduction to probability theory and statistical inference with applications, needs in data analysis, design of random models and decision making.

4. Summary
1. Variable Type 2. Descriptive Statistics 3. Inferential Statistics

5. Generales Goals

- An ability to design and conduct experiments, as well as to analyze and interpret data.

- An ability to identify, formulate, and solve real problems.

6. Contribution to Outcomes
This discipline contributes to the achievement of the following outcomes:

1) Analyze a complex computing problem and to apply principles of computing and other relevant disciplines to identify solutions. (**Assessment**)

6) Apply computer science theory and software development fundamentals to produce computing-based solutions. (**Assessment**)

7. Content

UNIT 1: Variable Type (6)	
Competences:	

Content	Generales Goals
• Variable Type: Continuous, discrete	• Classify the relevant variables identified according to their type: continuous (interval and ratio), categorical (nominal, ordinal, dichotomous). • Identify the relevant variables of a system using a process approach.

Readings: M.Ross (2014), Mendenhall (2014)

UNIT 2: Descriptive Statistics (6)	
Competences:	

Content	Generales Goals
• Central Tendency (Mean, median, mode) • Dispersion (Range, standard deviation, quartile) • Graphics: histogram, boxplot, etc.: Communication ability.	• Use central tendency measures and dispersion measures to describe the data gathered. • Use graphics to communicate the characteristics of the data gathered.

Readings: M.Ross (2014), Mendenhall (2014)

UNIT 3: Inferential Statistics (6)	
Competences:	

Content	Generales Goals
• Determination of the sample size • Confidence interval • Type I and type II error • Distribution type • Hypothesis test (t-student, means, proportions and ANOVA) • Relationships between variables: correlation, regression.	• Propose questions and hypotheses of interest. • Analyze the data gathered using different statistical tools to answer questions of interest. • Draw conclusions based on the analysis performed.

Readings: M.Ross (2014), Mendenhall (2014)

8. Methodology

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. Assessment

Continuous Assessment 1 : 20 %

Partial Exam : 30 %

Continuous Assessment 2 : 20 %

Final exam : 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
Undergraduate Program in Information Systems
SILABO

CS212. Algorithm Analysis and Design (Mandatory)

2022-II

1. General information		
1.1 School	:	Sistemas de Información
1.2 Course	:	CS212. Algorithm Analysis and Design
1.3 Semester	:	5to Semestre.
1.4 Prerrequisites	:	CS210. Algorithms and Data Structures. (4th Sem)
1.5 Type of course	:	Mandatory
1.6 Learning modality	:	Face to face
1.7 Horas	:	2 HT; 2 HP; 2 HL;
1.8 Credits	:	4

2. Professors

3. Course foundation
An algorithm is, essentially, a well-defined set of rules or instructions that allow solving a computational problem. The theoretical study of the performance of the algorithms and the resources used by them, usually time and space, allows us to evaluate if an algorithm is suitable for solving a specific problem, comparing it with other algorithms for the same problem or even delimiting the boundary between Viable and impossible. This matter is so important that even Donald E. Knuth defined Computer Science as the study of algorithms. This course will present the most common techniques used in the analysis and design of efficient algorithms, with the purpose of learning the fundamental principles of the design, implementation and analysis of algorithms for the solution of computational problems

4. Summary
1. Basic Analysis 2. Algorithmic Strategies 3. Fundamental Data Structures and Algorithms 4. Basic Automata Computability and Complexity 5. Advanced Data Structures Algorithms and Analysis

5. Generales Goals

- Develop the ability to evaluate the complexity and quality of algorithms proposed for a given problem.

- Study the most representative, introductory algorithms of the most important classes of problems treated in computation.

- Develop the ability to solve algorithmic problems using the fundamental principles of algorithm design learneds.

- Be able to answer the following questions when a new algorithm is presented: How good is the performance ?, Is there a better way to solve the problem?

6. Contribution to Outcomes

This discipline contributes to the achievement of the following outcomes:

1) Analyze a complex computing problem and to apply principles of computing and other relevant disciplines to identify solutions. (**Assessment**)

5) Function effectively as a member or leader of a team engaged in activities appropriate to the program's discipline. (**Usage**)

6) Apply computer science theory and software development fundamentals to produce computing-based solutions. (**Usage**)

7. Content

UNIT 1: Basic Analysis (10)

Competences:

Content	Generales Goals
- Differences among best, expected, and worst case behaviors of an algorithm - Asymptotic analysis of upper and expected complexity bounds - Complexity classes, such as constant, logarithmic, linear, quadratic, and exponential - Asymptotic Notation - Analysis of iterative and recursive algorithms - Inductive proofs and correctness of algorithms - Master Theorem and Recursion Trees	- Explain what is meant by "best", "expected", and "worst" case behavior of an algorithm [Assessment] - Determine informally the time and space complexity of different algorithms [Assessment] - List and contrast standard complexity classes [Assessment] - Explain the use of big omega, big theta, and little o notation to describe the amount of work done by an algorithm [Assessment] - Analyze worst-case running times of algorithms using asymptotic analysis [Assessment] - Use recurrence relations to determine the time complexity of recursively defined algorithms [Assessment] - Solve elementary recurrence relations, eg, using some form of a Master Theorem [Assessment] - Argue the correctness of algorithms using inductive proofs [Assessment]

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

UNIT 2: Algorithmic Strategies (30)	
Competences:	
Content	**Generales Goals**
- Brute-force algorithms - Greedy algorithms - Divide-and-conquer - Dynamic Programming	- For each of the strategies (brute-force, greedy, divide-and-conquer, recursive backtracking, and dynamic programming), identify a practical example to which it would apply [Assessment] - Use a greedy approach to solve an appropriate problem and determine if the greedy rule chosen leads to an optimal solution [Assessment] - Use a divide-and-conquer algorithm to solve an appropriate problem [Assessment] - Use dynamic programming to solve an appropriate problem [Assessment] - Determine an appropriate algorithmic approach to a problem [Assessment]
Readings: Kleinberg and Tardos (2005), Dasgupta, Papadimitriou, and Vazirani (2006), Rivest and Stein (2009), Alsuwaiyel (1999)	

UNIT 3: Fundamental Data Structures and Algorithms (6)	
Competences:	
Content	**Generales Goals**
- Graphs and graph algorithms - Maximum and minimum cut problem - Local search - Cache oblivious algorithms - Number theory and cryptography	- Discuss factors other than computational efficiency that influence the choice of algorithms, such as programming time, maintainability, and the use of application-specific patterns in the input data [Familiarity] - Solve problems using fundamental graph algorithms, including depth-first and breadth-first search [Assessment] - Demonstrate the ability to evaluate algorithms, to select from a range of possible options, to provide justification for that selection, and to implement the algorithm in a particular context [Assessment] - Solve problems using graph algorithms, including single-source and all-pairs shortest paths, and at least one minimum spanning tree algorithm [Assessment]
Readings: Kleinberg and Tardos (2005), Dasgupta, Papadimitriou, and Vazirani (2006), Rivest and Stein (2009), Sedgewick and Wayne (2011), Goodrich and Tamassia (2009)	

UNIT 4: Basic Automata Computability and Complexity (2)	
Competences:	
Content	**Generales Goals**
- Introduction to the P and NP classes and the P vs. NP problem - Introduction to the NP-complete class and exemplary NP-complete problems (e.g., SAT, Knapsack) - Reductions	- Define the classes P and NP [Familiarity] - Explain the significance of NP-completeness [Familiarity]
Readings: Kleinberg and Tardos (2005), Dasgupta, Papadimitriou, and Vazirani (2006), Rivest and Stein (2009)	

UNIT 5: Advanced Data Structures Algorithms and Analysis (12)	
Competences:	
Content	**Generales Goals**
- Graphs (e.g, topological sort, finding strongly connected components, matching) - Randomized algorithms - Amortized analysis - Probabilistic analysis - Approximation Algorithms - Linear Programming	- Understand the mapping of real-world problems to algorithmic solutions (eg, as graph problems, linear programs, etc) [Familiarity] - Select and apply advanced analysis techniques (eg, amortized, probabilistic, etc) to algorithms [Usage]
Readings: Kleinberg and Tardos (2005), Dasgupta, Papadimitriou, and Vazirani (2006), Rivest and Stein (2009), Tarjan (1983), Rawlins (1992)	

8. Methodology
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. Assessment
Continuous Assessment 1 : 20 %
Partial Exam : 30 %
Continuous Assessment 2 : 20 %
Final exam : 30 %

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
Undergraduate Program in Information Systems
SILABO

CS272. Databases II (Mandatory)

2022-II

1. General information		
1.1 School	:	Sistemas de Información
1.2 Course	:	CS272. Databases II
1.3 Semester	:	5to Semestre.
1.4 Prerrequisites	:	CS271. Databases I. (4th Sem)
1.5 Type of course	:	Mandatory
1.6 Learning modality	:	Face to face
1.7 Horas	:	1 HT; 2 HP; 2 HL;
1.8 Credits	:	3

2. Professors

3. Course foundation
Information Management (IM) plays a leading role in almost every area where computers are used. This area includes the capture, digitization, representation, organization, transformation and presentation of information; Algorithms to improve the efficiency and effectiveness of access and update of stored information, data modeling and abstraction, and physical file storage techniques.   It also covers information security, privacy, integrity and protection in a shared environment. Students need to be able to develop conceptual and physical data models, determine which IM methods and techniques are appropriate for a given problem, and be able to select and implement an appropriate IM solution that reflects all applicable constraints, including scalability and Usability.

4. Summary
1. Physical Database Design 2. Transaction Processing 3. Information Storage and Retrieval 4. Distributed Databases

5. Generales Goals
- To make the student understand the different applications that the databases have, in the different areas of knowledge.   - Show appropriate ways of storing information based on their various approaches and their subsequent retrieval of information.

6. Contribution to Outcomes

This discipline contributes to the achievement of the following outcomes:

1) Analyze a complex computing problem and to apply principles of computing and other relevant disciplines to identify solutions. (**Assessment**)

4) Recognize professional responsabilities and make informed judgments in computing practice based on legal and ethical principles. (**Assessment**)

6) Apply computer science theory and software development fundamentals to produce computing-based solutions. (**Assessment**)

7) Develop computational technology for the well-being of all, contributing with human formation, scientific, technological and professional skills to solve social problems of our community. (**Assessment**)

7. Content

UNIT 1: Physical Database Design (10)
Competences:

Content	Generales Goals
- Storage and file structure - Indexed files - Hashed files - Signature files - B-trees - Files with dense index - Files with variable length records - Database efficiency and tuning	- Explain the concepts of records, record types, and files, as well as the different techniques for placing file records on disk [Usage] - Give examples of the application of primary, secondary, and clustering indexes [Usage] - Distinguish between a non-dense index and a dense index [Usage] - Implement dynamic multilevel indexes using B-trees [Usage] - Explain the theory and application of internal and external hashing techniques [Usage] - Use hashing to facilitate dynamic file expansion [Usage] - Describe the relationships among hashing, compression, and efficient database searches [Usage] - Evaluate costs and benefits of various hashing schemes [Usage] - Explain how physical database design affects database transaction efficiency [Usage]

Readings: Burleson (2004), Celko (2005)

UNIT 2: Transaction Processing (12)	
Competences:	
Content	**Generales Goals**
• Transactions • Failure and recovery • Concurrency control • Interaction of transaction management with storage, especially buffering	• Create a transaction by embedding SQL into an application program [Usage] • Explain the concept of implicit commits [Usage] • Describe the issues specific to efficient transaction execution [Usage] • Explain when and why rollback is needed and how logging assures proper rollback [Usage] • Explain the effect of different isolation levels on the concurrency control mechanisms [Usage] • Choose the proper isolation level for implementing a specified transaction protocol [Usage] • Identify appropriate transaction boundaries in application programs [Usage]
Readings: Philip A. Bernstein (1997), Ramez Elmasri (2004)	

UNIT 3: Information Storage and Retrieval (10)	
Competences:	
Content	**Generales Goals**
- Documents, electronic publishing, markup, and markup languages - Tries, inverted files, PAT trees, signature files, indexing - Morphological analysis, stemming, phrases, stop lists - Term frequency distributions, uncertainty, fuzziness, weighting - Vector space, probabilistic, logical, and advanced models - Information needs, relevance, evaluation, effectiveness - Thesauri, ontologies, classification and categorization, metadata - Bibliographic information, bibliometrics, citations - Routing and (community) filtering - Multimedia search, information seeking behavior, user modeling, feedback - Information summarization and visualization - Faceted search (e.g., using citations, keywords, classification schemes) - Digital libraries - Digitization, storage, interchange, digital objects, composites, and packages - Metadata and cataloging - Naming, repositories, archives - Archiving and preservation, integrity - Spaces (conceptual, geographical, 2/3D, VR) - Architectures (agents, buses, wrappers/mediators), interoperability - Services (searching, linking, browsing, and so forth) - Intellectual property rights management, privacy, and protection (watermarking)	- Explain basic information storage and retrieval concepts [Usage] - Describe what issues are specific to efficient information retrieval [Usage] - Give applications of alternative search strategies and explain why the particular search strategy is appropriate for the application [Usage] - Design and implement a small to medium size information storage and retrieval system, or digital library [Usage] - Describe some of the technical solutions to the problems related to archiving and preserving information in a digital library [Usage]
Readings: Peter Brusilovsky (1998), Ramez Elmasri (2004)	

UNIT 4: Distributed Databases (36)	
Competences:	
Content	**Generales Goals**
- Distributed DBMS - Distributed data storage - Distributed query processing - Distributed transaction model - Homogeneous and heterogeneous solutions - Client-server distributed databases - Parallel DBMS - Parallel DBMS architectures: shared memory, shared disk, shared nothing; - Speedup and scale-up, e.g., use of the MapReduce processing model - Data replication and weak consistency models	- Explain the techniques used for data fragmentation, replication, and allocation during the distributed database design process [Usage] - Evaluate simple strategies for executing a distributed query to select the strategy that minimizes the amount of data transfer [Usage] - Explain how the two-phase commit protocol is used to deal with committing a transaction that accesses databases stored on multiple nodes [Usage] - Describe distributed concurrency control based on the distinguished copy techniques and the voting method [Usage] - Describe the three levels of software in the client-server model [Usage]
Readings: M. Tamer Ozsu (1999)	

8. Methodology
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. Assessment
Continuous Assessment 1 : 20 %
Partial Exam : 30 %
Continuous Assessment 2 : 20 %
Final exam : 30 %

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
Undergraduate Program in Information Systems
SILABO

CS291. Software Engineering I (Mandatory)

2022-II

1. General information		
1.1 School	:	Sistemas de Información
1.2 Course	:	CS291. Software Engineering I
1.3 Semester	:	5to Semestre.
1.4 Prerrequisites	:	

- CS113. Computer Science II. (3rd Sem)

- CS271. Databases I. (4th Sem)

1.5 Type of course	:	Mandatory
1.6 Learning modality	:	Face to face
1.7 Horas	:	2 HT; 2 HP; 2 HL;
1.8 Credits	:	4

2. Professors

3. Course foundation
The aim of developing software, except for extremely simple applications, requires the execution of a well-defined development process. Professionals in this area require a high degree of knowledge of the different models and development process, so that they are able to choose the most suitable for each development project. On the other hand, the development of medium and large-scale systems requires the use of pattern and component libraries and the mastery of techniques related to component-based design

4. Summary
1. Requirements Engineering 2. Software Design 3. Software Construction

5. Generales Goals

- Provide the student with a theoretical and practical framework for the development of software under quality standards.

- Familiarize the student with the software modeling and construction processes through the use of CASE tools.

- Students should be able to select architectures and ad-hoc technology platforms for deployment scenarios

- Applying component-based modeling to ensure variables such as quality, cost, and time-to-market in development processes.

- Provide students with best practices for software verification and validation.

6. Contribution to Outcomes

This discipline contributes to the achievement of the following outcomes:

1) Analyze a complex computing problem and to apply principles of computing and other relevant disciplines to identify solutions. (**Usage**)

2) Design, implement and evaluate a computing-based solution to meet a given set of computing requirements in the context of the program's discipline. (**Usage**)

3) Communicate effectively in a variety of professional contexts. (**Usage**)

6) Apply computer science theory and software development fundamentals to produce computing-based solutions. (**Assessment**)

7. Content

UNIT 1: Requirements Engineering (18)	
Competences:	
Content	**Generales Goals**
- Describing functional requirements using, for example, use cases or users stories - Properties of requirements including consistency, validity, completeness, and feasibility - Software requirements elicitation - Describing system data using, for example, class diagrams or entity-relationship diagrams - Non functional requirements and their relationship to software quality - Evaluation and use of requirements specifications - Requirements analysis modeling techniques - Acceptability of certainty / uncertainty considerations regarding software / system behavior - Prototyping - Basic concepts of formal requirements specification - Requirements specification - Requirements validation - Requirements tracing	- List the key components of a use case or similar description of some behavior that is required for a system [Assessment] - Describe how the requirements engineering process supports the elicitation and validation of behavioral requirements [Assessment] - Interpret a given requirements model for a simple software system [Assessment] - Describe the fundamental challenges of and common techniques used for requirements elicitation [Assessment] - List the key components of a data model (eg, class diagrams or ER diagrams) [Assessment] - Identify both functional and non-functional requirements in a given requirements specification for a software system [Assessment] - Conduct a review of a set of software requirements to determine the quality of the requirements with respect to the characteristics of good requirements [Assessment] - Apply key elements and common methods for elicitation and analysis to produce a set of software requirements for a medium-sized software system [Assessment] - Compare the plan-driven and agile approaches to requirements specification and validation and describe the benefits and risks associated with each [Assessment] - Use a common, non-formal method to model and specify the requirements for a medium-size software system [Assessment] - Translate into natural language a software requirements specification (eg, a software component contract) written in a formal specification language [Assessment] - Create a prototype of a software system to mitigate risk in requirements [Assessment] - Differentiate between forward and backward tracing and explain their roles in the requirements validation process [Assessment]
Readings: Eric Freeman and Sierra (2014), Hans-Erik Eriksson and Fado (2003)	

UNIT 2: Software Design (18)	
Competences:	
Content	**Generales Goals**

Content	Generales Goals

• System design principles: levels of abstraction (architectural design and detailed design), separation of concerns, information hiding, coupling and cohesion , re-use of standard structures
 	
• Design Paradigms such as structured design (top-down functional decomposition), object-oriented analysis and design, event driven design, component-level design, data-structured centered, aspect oriented, function oriented, service oriented
 	
• Structural and behavioral models of software designs
 	
• Design patterns
 	
• Relationships between requirements and designs: transformation of models, design of contracts, invariants
 	
• Software architecture concepts and standard architectures (e.g. client-server, n-layer, transform centered, pipes-and-filters)
 	
• The use of component desing: component selection, design, adaptation and assembly of components, component and patterns, components and objects (for example, building a GUI using a standar widget set)
 	
• Refactoring designs using design patterns
 	
• Internal design qualities, and models for them: efficiency and performance, redundancy and fault tolerance, traceability of requeriments
 	
• Measurement and analysis of design quality
 	
• Tradeoffs between different aspects of quality
 	
• Application frameworks
 	
• Middleware: the object-oriented paradigm within middleware, object request brokers and marshalling, transaction processing monitors, workflow systems
 	
• Principles of secure design and coding
 	
   – Principle of least privilege
   – Principle of fail-safe defaults
   – Principle of psychological acceptability
	• Articulate design principles including separation of concerns, information hiding, coupling and cohesion, and encapsulation [Familiarity]
	
	• Use a design paradigm to design a simple software system, and explain how system design principles have been applied in this design [Usage]
	
	• Construct models of the design of a simple software system that are appropriate for the paradigm used to design it [Usage]
	
	• Within the context of a single design paradigm, describe one or more design patterns that could be applicable to the design of a simple software system [Familiarity]
	
	• For a simple system suitable for a given scenario, discuss and select an appropriate design paradigm [Usage]
	
	• Create appropriate models for the structure and behavior of software products from their requirements specifications [Usage]
	
	• Explain the relationships between the requirements for a software product and its design, using appropriate models [Assessment]
	
	• For the design of a simple software system within the context of a single design paradigm, describe the software architecture of that system [Familiarity]
	
	• Given a high-level design, identify the software architecture by differentiating among common software architectures such as 3-tier, pipe-and-filter, and client-server [Familiarity]
	
	• Investigate the impact of software architectures selection on the design of a simple system [Assessment]
	
	• Apply simple examples of patterns in a software design [Usage]
	
	• Describe a form of refactoring and discuss when it may be applicable [Familiarity]
	
	• Select suitable components for use in the design of a software product [Usage]
	
	• Explain how suitable components might need to be adapted for use in the design of a software product [Familiarity]
	
	• Design a contract for a typical small software component for use in a given system [Usage]
	
	• Discuss and select appropriate software architecture for a simple system suitable for a given scenario [Usage]
	
	• Apply models for internal and external qualities in designing software components to achieve an acceptable tradeoff between conflicting quality aspects [Us-

UNIT 3: Software Construction (24)	
Competences:	
Content	**Generales Goals**

• Coding practices: techniques, idioms/patterns, mechanisms for building quality programs
 	
   – Defensive coding practices
 	
   – Secure coding practices
 	
   – Using exception handling mechanisms to make programs more robust, fault-tolerant
 	
• Coding standards
 	
• Integration strategies
 	
• Development context: "green field" vs. existing code base
 	
   – Change impact analysis
 	
   – Change actualization
 	
• Potential security problems in programs
 	
   – Buffer and other types of overflows
 	
   – Race conditions
 	
   – Improper initialization, including choice of privileges
 	
   – Checking input
 	
   – Assuming success and correctness
 	
   – Validating assumptions
	• Describe techniques, coding idioms and mechanisms for implementing designs to achieve desired properties such as reliability, efficiency, and robustness [Assessment]
	
	• Build robust code using exception handling mechanisms [Assessment]
	
	• Describe secure coding and defensive coding practices [Assessment]
	
	• Select and use a defined coding standard in a small software project [Assessment]
	
	• Compare and contrast integration strategies including top-down, bottom-up, and sandwich integration [Assessment]
	
	• Describe the process of analyzing and implementing changes to code base developed for a specific project [Assessment]
	
	• Describe the process of analyzing and implementing changes to a large existing code base [Assessment]
	
	• Rewrite a simple program to remove common vulnerabilities, such as buffer overflows, integer overflows and race conditions [Assessment]
	
	• Write a software component that performs some non-trivial task and is resilient to input and run-time errors [Assessment]

Readings: Eric Freeman and Sierra (2014), Hans-Erik Eriksson and Fado (2003)	

8. Methodology
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. Assessment
Continuous Assessment 1 : 20 % **Partial Exam** : 30 % **Continuous Assessment 2** : 20 % **Final exam** : 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
Undergraduate Program in Information Systems
SILABO

CS342. Compilers (Mandatory)

2022-II

1. General information		
1.1 School	:	Sistemas de Información
1.2 Course	:	CS342. Compilers
1.3 Semester	:	5to Semestre.
1.4 Prerrequisites	:	CS211. Computer Science Theory. (4th Sem)
1.5 Type of course	:	Mandatory
1.6 Learning modality	:	Face to face
1.7 Horas	:	2 HT; 2 HP; 2 HL;
1.8 Credits	:	4

2. Professors

3. Course foundation
That the student knows and understands the concepts and fundamental principles of the theory of compilation to realize the construction of a compiler

4. Summary
1. Program Representation 2. Language Translation and Execution 3. Syntax Analysis 4. Compiler Semantic Analysis 5. Code Generation

5. Generales Goals
• Know the basic techniques used during the process of intermediate generation, optimization and code generation.   • Learning to implement small compilers.

6. Contribution to Outcomes
This discipline contributes to the achievement of the following outcomes:

1) Analyze a complex computing problem and to apply principles of computing and other relevant disciplines to identify solutions. (**Assessment**)

6) Apply computer science theory and software development fundamentals to produce computing-based solutions. (**Assessment**)

7. Content

UNIT 1: Program Representation (5)	
Competences:	
Content	**Generales Goals**
- Programs that take (other) programs as input such as interpreters, compilers, type-checkers, documentation generators - Abstract syntax trees; contrast with concrete syntax - Data structures to represent code for execution, translation, or transmission - Just-in-time compilation and dynamic recompilation - Other common features of virtual machines, such as class loading, threads, and security.	- Explain how programs that process other programs treat the other programs as their input data [Familiarity] - Describe an abstract syntax tree for a small language [Familiarity] - Describe the benefits of having program representations other than strings of source code [Familiarity] - Write a program to process some representation of code for some purpose, such as an interpreter, an expression optimizer, or a documentation generator [Familiarity] - Explain the use of metadata in run-time representations of objects and activation records, such as class pointers, array lengths, return addresses, and frame pointers [Familiarity] - Discuss advantages, disadvantages, and difficulties of just-in-time and dynamic recompilation [Familiarity] - Identify the services provided by modern language run-time systems [Familiarity]
Readings: Louden (2004b)	

UNIT 2: Language Translation and Execution (10)	
Competences:	
Content	**Generales Goals**
- Interpretation vs. compilation to native code vs. compilation to portable intermediate representation - Language translation pipeline: parsing, optional type-checking, translation, linking, execution - Execution as native code or within a virtual machine - Alternatives like dynamic loading and dynamic (or "just-in-time") code generation - Run-time representation of core language constructs such as objects (method tables) and first-class functions (closures) - Run-time layout of memory: call-stack, heap, static data - Implementing loops, recursion, and tail calls - Memory management - Manual memory management: allocating, de-allocating, and reusing heap memory - Automated memory management: garbage collection as an automated technique using the notion of reachability	- Distinguish a language definition (what constructs mean) from a particular language implementation (compiler vs interpreter, run-time representation of data objects, etc) [Assessment] - Distinguish syntax and parsing from semantics and evaluation [Assessment] - Sketch a low-level run-time representation of core language constructs, such as objects or closures [Assessment] - Explain how programming language implementations typically organize memory into global data, text, heap, and stack sections and how features such as recursion and memory management map to this memory model [Assessment] - Identify and fix memory leaks and dangling-pointer dereferences [Assessment] - Discuss the benefits and limitations of garbage collection, including the notion of reachability [Assessment]
Readings: Aho et al. (2011), Louden (2004a), Appel (2002), Teufel and Schmidt (1998)	

UNIT 3: Syntax Analysis (10)	
Competences:	
Content	**Generales Goals**
- Scanning (lexical analysis) using regular expressions - Parsing strategies including top-down (e.g., recursive descent, Earley parsing, or LL) and bottom-up (e.g., backtracking or LR) techniques; role of context-free grammars - Generating scanners and parsers from declarative specifications	- Use formal grammars to specify the syntax of languages [Assessment] - Use declarative tools to generate parsers and scanners [Assessment] - Identify key issues in syntax definitions: ambiguity, associativity, precedence [Assessment]
Readings: Aho et al. (2011), Louden (2004a), Appel (2002), Teufel and Schmidt (1998)	

UNIT 4: Compiler Semantic Analysis (15)	
Competences:	
Content	**Generales Goals**
- High-level program representations such as abstract syntax trees - Scope and binding resolution - Type checking - Declarative specifications such as attribute grammars	- Implement context-sensitive, source-level static analyses such as type-checkers or resolving identifiers to identify their binding occurrences [Assessment] - Describe semantic analyses using an attribute grammar [Assessment]
Readings: Aho et al. (2011), Louden (2004a), Appel (2002), Teufel and Schmidt (1998)	

UNIT 5: Code Generation (20)	
Competences:	
Content	**Generales Goals**
- Procedure calls and method dispatching - Separate compilation; linking - Instruction selection - Instruction scheduling - Register allocation - Peephole optimization	- Identify all essential steps for automatically converting source code into assembly or other low-level languages [Assessment] - Generate the low-level code for calling functions/methods in modern languages [Assessment] - Discuss why separate compilation requires uniform calling conventions [Assessment] - Discuss why separate compilation limits optimization because of unknown effects of calls [Assessment] - Discuss opportunities for optimization introduced by naive translation and approaches for achieving optimization, such as instruction selection, instruction scheduling, register allocation, and peephole optimization [Assessment]
Readings: Aho et al. (2011), Louden (2004a), Appel (2002), Teufel and Schmidt (1998)	

8. Methodology
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. Assessment
Continuous Assessment 1 : 20 %
Partial Exam : 30 %
Continuous Assessment 2 : 20 %
Final exam : 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
Undergraduate Program in Information Systems
SILABO

CB111. Computational Physics (Mandatory)

2022-II

1. General information		
1.1 School	:	Sistemas de Información
1.2 Course	:	CB111. Computational Physics
1.3 Semester	:	5to Semestre.
1.4 Prerrequisites	:	MA102. Calculus I. (3rd Sem)
1.5 Type of course	:	Mandatory
1.6 Learning modality	:	Face to face
1.7 Horas	:	2 HT; 2 HP; 2 HL;
1.8 Credits	:	4

2. Professors

3. Course foundation
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 partículas; 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. Summary
1. 2. 3. 4. 5. 6.

5. Generales Goals

- 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. Contribution to Outcomes
This discipline contributes to the achievement of the following outcomes:

1) Analyze a complex computing problem and to apply principles of computing and other relevant disciplines to identify solutions. (**Usage**)

6) Apply computer science theory and software development fundamentals to produce computing-based solutions. (**Usage**)

7. Content

UNIT 1: (6)
Competences:

Content	Generales Goals
- Análisis dimensional. - Vectores. Propiedades. Operaciones. - Caso práctico: Estimación de fuerzas.	- Entender y trabajar con las magnitudes físicas del SI.[Usage] - Abstraer de la naturaleza los conceptos físicos rigurosos y representarlos en modelos vectoriales.[Usage] - Entender y aplicar los conceptos vectoriales a problemas físicos reales.[Usage]

Readings: Burbano (2006), Resnik (2007), Serway (2009), Tipler (2009)

UNIT 2: (6)
Competences:

Content	Generales Goals
- 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.	- 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 equilibro. - Capacidad para resolver problemas de casos prácticos. - Entender el concepto de fricción y resolver problemas.

Readings: Burbano (2006), Resnik (2007), Serway (2009), Tipler (2009)

UNIT 3: (6)
Competences:

Content	Generales Goals
- 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.	- 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.

Readings: Burbano (2006), Resnik (2007), Serway (2009), Tipler (2009)

UNIT 4: (6)	
Competences:	
Content	**Generales Goals**
• Segunda Ley de Newton. • Fuerza y movimiento. • Momento de inercia.	• 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
Readings: Burbano (2006), Resnik (2007), Serway (2009), Tipler (2009)	

UNIT 5: (6)	
Competences:	
Content	**Generales Goals**
• Trabajo. • Fuerzas constantes. • Fuerzas variables. • Potencia. • Caso práctico: Estimación de la potencia de una planta hidroeléctrica.	• Comprender el concepto de Trabajo. • Comprender y aplicar el concepto de Potencia a la resolución de problemas. • Resolver problemas.
Readings: Burbano (2006), Resnik (2007), Serway (2009), Tipler (2009)	

UNIT 6: (6)	
Competences:	
Content	**Generales Goals**
• Tipos de energía. • Conservación de la energía. • Dinámica de un sistema de partículas. • Colisiones.	• 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 partículas interactuantes.
Readings: Burbano (2006), Resnik (2007), Serway (2009), Tipler (2009)	

8. Methodology
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. Assessment

Continuous Assessment 1 : 20 %

Partial Exam : 30 %

Continuous Assessment 2 : 20 %

Final exam : 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
Undergraduate Program in Information Systems
SILABO

FG106. Theater (Mandatory)

2022-II

1. General information		
1.1 School	:	Sistemas de Información
1.2 Course	:	FG106. Theater
1.3 Semester	:	5to Semestre.
1.4 Prerrequisites	:	None
1.5 Type of course	:	Mandatory
1.6 Learning modality	:	Face to face
1.7 Horas	:	1 HT; 2 HP;
1.8 Credits	:	2

2. Professors

3. Course foundation
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. Summary
1. 2. 3. 4. 5. 6.

5. Generales Goals
• 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. Contribution to Outcomes

This discipline contributes to the achievement of the following outcomes:

3) Communicate effectively in a variety of professional contexts. (**Usage**)

4) Recognize professional responsabilities and make informed judgments in computing practice based on legal and ethical principles. (**Usage**)

7) Develop computational technology for the well-being of all, contributing with human formation, scientific, technological and professional skills to solve social problems of our community. (**Usage**)

7. Content

UNIT 1: (6)	
Competences:	

Content	**Generales Goals**
• ¿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.	• Reconocer la vigencia del Arte y la creatividad en el desarrollo personal y social [Usage]. • Relacionar al estudiante con su grupo valorando la importancia de la comunicación humana y del colectivo social [Usage]. • Reconocer nociones básicas del teatro [Usage].

Readings: Majorana (1958), Pavis (1998)

UNIT 2: (6)	
Competences:	

Content	**Generales Goals**
• Juego, luego existo. • El juego del niño y el juego dramático. • Juegos de integración grupal y juegos de creatividad. • La secuencia teatral.	• Reconocer el juego como herramienta fundamental del teatro [Usage]. • Interiorizar y revalorar el juego como aprendizaje creativo [Usage]. • Acercar al estudiante de manera espontánea y natural, a la vivencia teatral [Usage].

Readings: Majorana (1958), Pavis (1998)

UNIT 3: (9)	
Competences:	

Content	**Generales Goals**
• 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.	• Experimentar con nuevas formas de expresión y comunicación [Usage]. • Conocer algunos mecanismos de control y manejo corporal [Usage]. • Brindar caminos para que el alumno pueda desarrollar creativamente su imaginación, su capacidad de relación y captación de estímulos auditivos, rítmicos y visuales [Usage]. • Conocer y desarrollar el manejo de su espacio propio y de sus relaciones espaciales [Usage]. • Experimentar estados emocionales diferentes y climas colectivos nuevos [Usage].

Readings: Majorana (1958), Pavis (1998)

UNIT 4: (12)
Competences:

Content	Generales Goals
• 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.	• Ejercitarse en el manejo de destrezas comunicativas no verbales [Usage]. • Practicar juegos y ejercicios de lenguaje corporal, individual y grupalmente [Usage]. • Expresar libre y creativamente sus emociones y sentimientos y su visión de la sociedad a través de representaciones originales con diversos lenguajes [Usage]. • Conocer los tipos de actuación [Usage].

Readings: Majorana (1958), Pavis (1998)

UNIT 5: (3)
Competences:

Content	Generales Goals
• El orígen 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.	• Conocer la influencia que la sociedad ha ejercido en el teatro y la respuesta de este arte ante los diferentes momentos de la historia [Usage]. • Apreciar el valor y aporte de las obras de dramaturgos importantes [Usage]. • Analizar el contexto social del arte teatral [Usage]. • Reflexionar sobre el Teatro Peruano y arequipeño [Usage].

Readings: Majorana (1958), Pavis (1998)

UNIT 6: (12)
Competences:

Content	Generales Goals
• 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.	• Emplear la creación teatral, como manifestación de ideas y sentimientos propios ante la sociedad [Usage]. • Aplicar las técnicas practicadas y los conocimientos aprendidos en una apreciación y/o expresión teatral concreta que vincule el rol de la educación [Usage]. • Intercambiar experiencias y realizar presentaciones breves de ejercicios teatrales en grupo, frente a público [Usage].

Readings: Majorana (1958), Pavis (1998)

8. Methodology

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. Assessment

Continuous Assessment 1 : 20 %

Partial Exam : 30 %

Continuous Assessment 2 : 20 %

Final exam : 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
Undergraduate Program in Information Systems
SILABO

CS292. Software Engineering II (Mandatory)

2022-II

1. General information		
1.1 School	:	Sistemas de Información
1.2 Course	:	CS292. Software Engineering II
1.3 Semester	:	6to Semestre.
1.4 Prerrequisites	:	CS291. Software Engineering I. (5th Sem)
1.5 Type of course	:	Mandatory
1.6 Learning modality	:	Face to face
1.7 Horas	:	2 HT; 2 HP; 2 HL;
1.8 Credits	:	4

2. Professors

3. Course foundation
The topics of this course extend the ideas of software design and development from the introduction sequence to programming to encompass the problems encountered in large-scale projects. It is a broader and more complete view of Software Engineering appreciated from a Project point of view.

4. Summary
1. Tools and Environments 2. Software Verification and Validation 3. Software Evolution 4. Software Project Management

5. Generales Goals
- Enable students to be part of and define software development teams facing real-world problems.   - familiarize the students with the process of administering a software project in such a way as to be able to create, improve and use tools and metrics that allow them to carry out the estimation and monitoring of a software project   - Create, evaluate and execute a test plan for medium-sized code segments, Distinguish between different types of tests, lay the foundation for creating, improve test procedures and tools for these purposes   - Select with justification an appropriate set of tools to support the development of a range of software products.   - Create, improve and use existing patterns for software maintenance. Disclose features and design patterns for software reuse.   - Identify and discuss different specialized systems, create, improve and use specialized standards for the design, implementation, maintenance and testing of specialized systems.

6. Contribution to Outcomes

This discipline contributes to the achievement of the following outcomes:

1) Analyze a complex computing problem and to apply principles of computing and other relevant disciplines to identify solutions. (**Usage**)

2) Design, implement and evaluate a computing-based solution to meet a given set of computing requirements in the context of the program's discipline. (**Usage**)

3) Communicate effectively in a variety of professional contexts. (**Usage**)

6) Apply computer science theory and software development fundamentals to produce computing-based solutions. (**Assessment**)

7. Content

UNIT 1: Tools and Environments (12)
Competences:

Content	Generales Goals
- Software configuration management and version control - Release management - Requierements analysis and desing modeling tools - Testing tools including static and dynamic analysis tools - Programming enviroments that automate parts of program construction pocesses (e.g., automated builds) – Continuous integration - Tool integration concepts and mechanisms	- Software configuration management and version control [Usage] - Release management [Usage] - Requierements analysis and desing modeling tools [Usage] - Testing tools including static and dynamic analysis tools [Usage] - Programming enviroments that automate parts of program construction pocesses (e.g., automated builds) – Continuous integration [Usage] - Tool integration concepts and mechanisms [Usage]

Readings: 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)

UNIT 2: Software Verification and Validation (12)	
Competences:	
Content	**Generales Goals**

• Verification and validation concepts
 	
• Inspections, reviews, audits
 	
• Testing types, including human computer interface, usability, reliability, security, conformance to specification
 	
• Testing fundamentals
 	
    – Unit, integration, validation, and system testing
    – Test plan creation and test case generation
    – Black-box and white-box testing techniques
    – Regression testing and test automation
 	
• Defect tracking
 	
• Limitations of testing in particular domains, such as parallel or safety-critical systems
 	
• Static approaches and dynamic approaches to verification
 	
• Test-driven development
 	
• Validation planning; documentation for validation
 	
• Object-oriented testing; systems testing
 	
• Verification and validation of non-code artifacts (documentation, help files, training materials)
 	
• Fault logging, fault tracking and technical support for such activities
 	
• Fault estimation and testing termination including defect seeding
	• Distinguish between program validation and verification [Usage]
	
	• Describe the role that tools can play in the validation of software [Usage]
	
	• Undertake, as part of a team activity, an inspection of a medium-size code segment [Usage]
	
	• Describe and distinguish among the different types and levels of testing (unit, integration, systems, and acceptance) [Usage]
	
	• Describe techniques for identifying significant test cases for integration, regression and system testing [Usage]
	
	• Create and document a set of tests for a medium-size code segment [Usage]
	
	• Describe how to select good regression tests and automate them [Usage]
	
	• Use a defect tracking tool to manage software defects in a small software project [Usage]
	
	• Discuss the limitations of testing in a particular domain [Usage]
	
	• Evaluate a test suite for a medium-size code segment [Usage]
	
	• Compare static and dynamic approaches to verification [Usage]
	
	• Identify the fundamental principles of test-driven development methods and explain the role of automated testing in these methods [Usage]
	
	• Discuss the issues involving the testing of object-oriented software [Usage]
	
	• Describe techniques for the verification and validation of non-code artifacts [Usage]
	
	• Describe approaches for fault estimation [Usage]
	
	• Estimate the number of faults in a small software application based on fault density and fault seeding [Usage]
	
	• Conduct an inspection or review of software source code for a small or medium sized software project [Usage]

Readings: 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)	

UNIT 3: Software Evolution (12)	
Competences:	
Content	**Generales Goals**
- Software development in the context of large, pre-existing code bases – Software change – Concerns and concernlocation – Refactoring - Software evolution - Characteristics of maintainable software - Reengineering systems - Software reuse – Code segments – Libraries and frameworks – Components – Product lines	- Identify the principal issues associated with software evolution and explain their impact on the software lifecycle [Usage] - Estimate the impact of a change request to an existing product of medium size [Usage] - Use refactoring in the process of modifying a software component [Usage] - Discuss the challenges of evolving systems in a changing environment [Usage] - Outline the process of regression testing and its role in release management [Usage] - Discuss the advantages and disadvantages of different types of software reuse [Usage]
Readings: 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)	

UNIT 4: Software Project Management (12)	
Competences:	
Content	**Generales Goals**

Content	Generales Goals
- Team participation - Team processes including responsabilities for task, meeting structure, and work schedule - Roles and responsabilities in a software team - Team conflict resolution - Risks associated with virtual teams (communication, perception, structure) - Effort estimation (at the personal level) - Risk - The role of risk in the lifecycle - Risk categories including security, safety, market, financial, technology, people, quality, structure and process - Team management - Team organization and decision-making - Role identification and assigment - Individual and team performance assessment - Project management - Scheduling and tracking - Project management tools - Cost/benefit analysis - Software measurement and estimation techniques - Software quality assurance and the role of measurements - Risk - Risk identification and management - Risk analysis and evaluation - Risk tolerance (e.g., risk-adverse, risk-neutral, risk-seeking) - Risk planning - System-wide approach to risk including hazards associated with tools	- Discuss common behaviors that contribute to the effective functioning of a team [Usage] - Create and follow an agenda for a team meeting [Usage] - Identify and justify necessary roles in a software development team [Usage] - Understand the sources, hazards, and potential benefits of team conflict [Usage] - Apply a conflict resolution strategy in a team setting [Usage] - Use an ad hoc method to estimate software development effort (eg, time) and compare to actual effort required [Usage] - List several examples of software risks [Usage] - Describe the impact of risk in a software development lifecycle [Usage] - Describe different categories of risk in software systems [Usage] - Demonstrate through involvement in a team project the central elements of team building and team management [Usage]

Readings: 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. Methodology

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. Assessment

Continuous Assessment 1 : 20 %

Partial Exam : 30 %

Continuous Assessment 2 : 20 %

Final exam : 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
Undergraduate Program in Information Systems
SILABO

CS2S1. Operating systems (Mandatory)

2022-II

1. General information		
1.1 School	:	Sistemas de Información
1.2 Course	:	CS2S1. Operating systems
1.3 Semester	:	6to Semestre.
1.4 Prerrequisites	:	CS221. Computer Architecture. (3rd Sem)
1.5 Type of course	:	Mandatory
1.6 Learning modality	:	Face to face
1.7 Horas	:	2 HT; 2 HP; 2 HL;
1.8 Credits	:	4

2. Professors

3. Course foundation
An Operating System (OS) manages the computing resources to complete the execution of multiple applications and their associated processes. This course teaches the design of modern operating systems; and introduces their fundamental concepts covering multiple-program execution, scheduling, memory management, file systems, and security. Also, the course includes programming activities on a minimal operating system to solve problems and extend its functionality. Notice that these activities require much time to complete. However, working on them provides valuable insight into operating systems.

4. Summary
1. Overview of Operating Systems 2. Operating System Principles 3. Concurrency 4. Scheduling and Dispatch 5. Memory Management 6. Security and Protection 7. Virtual Machines 8. Device Management 9. File Systems 10. Real Time and Embedded Systems 11. Fault Tolerance 12. System Performance Evaluation

5. Generales Goals
- Study the design of modern operating systems.   - Provide a practical experience by designing and implementing a minimal operating system.

6. Contribution to Outcomes

This discipline contributes to the achievement of the following outcomes:

1) Analyze a complex computing problem and to apply principles of computing and other relevant disciplines to identify solutions. (**Assessment**)

4) Recognize professional responsabilities and make informed judgments in computing practice based on legal and ethical principles. (**Familiarity**)

5) Function effectively as a member or leader of a team engaged in activities appropriate to the program's discipline. (**Usage**)

6) Apply computer science theory and software development fundamentals to produce computing-based solutions. (**Usage**)

7) Develop computational technology for the well-being of all, contributing with human formation, scientific, technological and professional skills to solve social problems of our community. (**Assessment**)

7. Content

UNIT 1: Overview of Operating Systems (3)	
Competences:	
Content	**Generales Goals**
- Role and purpose of the operating system - Functionality of a typical operating system - Mechanisms to support client-server models. - Design issues (efficiency, robustness, flexibility, portability, security, compatibility) - Influences of security, networking, multimedia, windowing systems	- Explain the objectives and functions of modern operating systems [Familiarity] - Analyze the tradeoffs inherent in operating system design [Assessment] - Describe the functions of a contemporary operating system with respect to convenience, efficiency, and the ability to evolve [Familiarity] - Discuss networked, client-server, distributed operating systems and how they differ from single user operating systems [Familiarity] - Identify potential threats to operating systems and the security features design to guard against them [Familiarity]
Readings: Avi Silberschatz (2012), Stallings (2005), Tanenbaum (2006), Tanenbaum (2001), Anderson and Dahlin (2014)	

UNIT 2: Operating System Principles (6)	
Competences:	
Content	**Generales Goals**
• Operating Sistems Structure (monolithic, layered, modular, micro-kernel models) • Abstractions, processes, and resources • Concepts of application program interfaces (APIs) • The evolution of hardware/software techniques and application needs • Device organization • Interrupts: methods and implementations • Concept of user/system state and protection, transition to kernel mode	• Explain the concept of a logical layer [Familiarity] • Explain the benefits of building abstract layers in hierarchical fashion [Familiarity] • Describe the value of APIs and middleware [Familiarity] • Describe how computing resources are used by application software and managed by system software [Familiarity] • Contrast kernel and user mode in an operating system [Assessment] • Discuss the advantages and disadvantages of using interrupt processing [Familiarity] • Explain the use of a device list and driver I/O queue [Familiarity]
Readings: Avi Silberschatz (2012), Stallings (2005), Tanenbaum (2006), Tanenbaum (2001), Anderson and Dahlin (2014)	

UNIT 3: Concurrency (9)	
Competences:	
Content	**Generales Goals**
• States diagrams • Structures (ready list, process control blocks, and so forth) • Dispatching and context switching • The role of interrupts • Managing atomic access to OS objects • Implementing synchronization primitives • Multiprocessor issues (spin-locks, reentrancy)	• Describe the need for concurrency within the framework of an operating system [Familiarity] • Demonstrate the potential run-time problems arising from the concurrent operation of many separate tasks [Usage] • Summarize the range of mechanisms that can be employed at the operating system level to realize concurrent systems and describe the benefits of each [Familiarity] • Explain the different states that a task may pass through and the data structures needed to support the management of many tasks [Familiarity] • Summarize techniques for achieving synchronization in an operating system (eg, describe how to implement a semaphore using OS primitives) [Familiarity] • Describe reasons for using interrupts, dispatching, and context switching to support concurrency in an operating system [Familiarity] • Create state and transition diagrams for simple problem domains [Usage]
Readings: Avi Silberschatz (2012), Stallings (2005), Tanenbaum (2006), Tanenbaum (2001), Anderson and Dahlin (2014)	

UNIT 4: Scheduling and Dispatch (6)	
Competences:	
Content	**Generales Goals**
- Preemptive and non-preemptive scheduling - Schedulers and policies - Processes and threads - Deadlines and real-time issues	- Compare and contrast the common algorithms used for both preemptive and non-preemptive scheduling of tasks in operating systems, such as priority, performance comparison, and fair-share schemes [Assessment] - Describe relationships between scheduling algorithms and application domains [Familiarity] - Discuss the types of processor scheduling such as short-term, medium-term, long-term, and I/O [Familiarity] - Describe the difference between processes and threads [Familiarity] - Compare and contrast static and dynamic approaches to real-time scheduling [Assessment] - Discuss the need for preemption and deadline scheduling [Familiarity] - Identify ways that the logic embodied in scheduling algorithms are applicable to other domains, such as disk I/O, network scheduling, project scheduling, and problems beyond computing [Familiarity]
Readings: Avi Silberschatz (2012), Stallings (2005), Tanenbaum (2006), Tanenbaum (2001), Anderson and Dahlin (2014)	

UNIT 5: Memory Management (6)	
Competences:	
Content	**Generales Goals**
• Review of physical memory and memory management hardware • Working sets and thrashing • Caching	• Explain memory hierarchy and cost-performance trade-offs [Familiarity] • Summarize the principles of virtual memory as applied to caching and paging [Familiarity] • Evaluate the trade-offs in terms of memory size (main memory, cache memory, auxiliary memory) and processor speed [Assessment] • Defend the different ways of allocating memory to tasks, citing the relative merits of each [Familiarity] • Describe the reason for and use of cache memory (performance and proximity, different dimension of how caches complicate isolation and VM abstraction) [Familiarity] • Discuss the concept of thrashing, both in terms of the reasons it occurs and the techniques used to recognize and manage the problem [Familiarity]
Readings: Avi Silberschatz (2012), Stallings (2005), Tanenbaum (2006), Tanenbaum (2001), Anderson and Dahlin (2014)	

UNIT 6: Security and Protection (6)	
Competences:	
Content	**Generales Goals**
• Overview of system security • Policy/mechanism separation • Security methods and devices • Protection, access control, and authentication • Backups	• Articulate the need for protection and security in an OS [Familiarity] • Summarize the features and limitations of an operating system used to provide protection and security [Familiarity] • Explain the mechanisms available in an OS to control access to resources (cross reference IAS/Security Architecture and Systems Administration/Access Control/Configuring systems to operate securely as an IT system) [Familiarity] • Carry out simple system administration tasks according to a security policy, for example creating accounts, setting permissions, applying patches, and arranging for regular backups (cross reference IAS/Security Architecture and Systems Administration) [Familiarity]
Readings: Avi Silberschatz (2012), Stallings (2005), Tanenbaum (2006), Tanenbaum (2001), Anderson and Dahlin (2014)	

UNIT 7: Virtual Machines (6)	
Competences:	
Content	**Generales Goals**
• Types of virtualization (including Hardware/Software, OS, Server, Service, Network) • Paging and virtual memory • Virtual file systems • Hypervisors • Portable virtualization; emulation vs. isolation • Cost of virtualization	• Explain the concept of virtual memory and how it is realized in hardware and software [Familiarity] • Differentiate emulation and isolation [Familiarity] • Evaluate virtualization trade-offs [Assessment] • Discuss hypervisors and the need for them in conjunction with different types of hypervisors [Familiarity]
Readings: Avi Silberschatz (2012), Stallings (2005), Tanenbaum (2006), Tanenbaum (2001), Anderson and Dahlin (2014)	

UNIT 8: Device Management (6)	
Competences:	
Content	**Generales Goals**
• Characteristics of serial and parallel devices • Abstracting device differences • Buffering strategies • Direct memory access • Recovery from failures	• Explain the key difference between serial and parallel devices and identify the conditions in which each is appropriate [Familiarity] • Identify the relationship between the physical hardware and the virtual devices maintained by the operating system [Familiarity] • Explain buffering and describe strategies for implementing it [Familiarity] • Differentiate the mechanisms used in interfacing a range of devices (including hand-held devices, networks, multimedia) to a computer and explain the implications of these for the design of an operating system [Familiarity] • Describe the advantages and disadvantages of direct memory access and discuss the circumstances in which its use is warranted [Familiarity] • Identify the requirements for failure recovery [Familiarity] • Implement a simple device driver for a range of possible devices [Usage]
Readings: Avi Silberschatz (2012), Stallings (2005), Tanenbaum (2006), Tanenbaum (2001), Anderson and Dahlin (2014)	

UNIT 9: File Systems (6)	
Competences:	
Content	**Generales Goals**
- Files: data, metadata, operations, organization, buffering, sequential, nonsequential. - Directories: contents and structure. - File systems: partitioning, mount/unmount, virtual file systems. - Standard implementation techniques - Memory-mapped files - Special-purpose file systems. - Naming, searching, access, backups - Journaling and log-structured file systems	- Describe the choices to be made in designing file systems [Familiarity] - Compare and contrast different approaches to file organization, recognizing the strengths and weaknesses of each [Assessment] - Summarize how hardware developments have led to changes in the priorities for the design and the management of file systems [Familiarity] - Summarize the use of journaling and how log-structured file systems enhance fault tolerance [Familiarity]
Readings: Avi Silberschatz (2012), Stallings (2005), Tanenbaum (2006), Tanenbaum (2001), Anderson and Dahlin (2014)	

UNIT 10: Real Time and Embedded Systems (6)	
Competences:	
Content	**Generales Goals**
- Process and task scheduling - Memory/disk management requirements in a real-time environment - Failures, risks, and recovery. - Special concerns in real-time systems	- Describe what makes a system a real-time system [Familiarity] - Explain the presence of and describe the characteristics of latency in real-time systems [Familiarity] - Summarize special concerns that real-time systems present, including risk, and how these concerns are addressed [Familiarity]
Readings: Avi Silberschatz (2012), Stallings (2005), Tanenbaum (2006), Tanenbaum (2001), Anderson and Dahlin (2014)	

UNIT 11: Fault Tolerance (3)	
Competences:	
Content	**Generales Goals**
- Fundamental concepts: reliable and available systems - Spatial and temporal redundancy - Methods used to implement fault tolerance - Examples of OS mechanisms for detection, recovery, restart to implement fault tolerance, use of these techniques for the OS's own services.	- Explain the relevance of the terms fault tolerance, reliability, and availability [Familiarity] - Outline the range of methods for implementing fault tolerance in an operating system [Familiarity] - Explain how an operating system can continue functioning after a fault occurs [Familiarity]
Readings: Avi Silberschatz (2012), Stallings (2005), Tanenbaum (2006), Tanenbaum (2001), Anderson and Dahlin (2014)	

UNIT 12: System Performance Evaluation (3)	
Competences:	
Content	**Generales Goals**
• Why system performance needs to be evaluated? • What is to be evaluated? • Systems performance policies, e.g., caching, paging, scheduling, memory management, and security • Evaluation models: deterministic, analytic, simulation, or implementation-specific • How to collect evaluation data (profiling and tracing mechanisms)	• Describe the performance measurements used to determine how a system performs [Familiarity] • Explain the main evaluation models used to evaluate a system [Familiarity]
Readings: Avi Silberschatz (2012), Stallings (2005), Tanenbaum (2006), Tanenbaum (2001), Anderson and Dahlin (2014)	

8. Methodology
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. Assessment
Continuous Assessment 1 : 20 % **Partial Exam** : 30 % **Continuous Assessment 2** : 20 % **Final exam** : 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
Undergraduate Program in Information Systems
SILABO

CS311. Competitive Programming (Mandatory)

2022-II

1. General information		
1.1 School	:	Sistemas de Información
1.2 Course	:	CS311. Competitive Programming
1.3 Semester	:	6to Semestre.
1.4 Prerrequisites	:	CS212. Algorithm Analysis and Design. (5th Sem)
1.5 Type of course	:	Mandatory
1.6 Learning modality	:	Face to face
1.7 Horas	:	2 HT; 2 HP; 2 HL;
1.8 Credits	:	4

2. Professors

3. Course foundation
Competitive Programming combines problem-solving challenges with the fun of competing with others. It teaches participants to think faster and develop problem-solving skills that are in high demand in the industry. This course will teach you to solve algorithmic problems quickly by combining theory of algorithms and data structures with practice solving problems.

4. Summary
1. Introduction 2. Data structure 3. Algorithmic Design Paradigms 4. Graphs 5. Advanced topics 6. Domain specific problems

5. Generales Goals

- That the student uses techniques of data structures and complex algorithms..

- That the student apply the concepts learned for the application on a real problem.

- That the student investigate the possibility of creating a new algorithm and / or new technique to solve a real problem.

6. Contribution to Outcomes

This discipline contributes to the achievement of the following outcomes:

1) Analyze a complex computing problem and to apply principles of computing and other relevant disciplines to identify solutions. (**Usage**)

6) Apply computer science theory and software development fundamentals to produce computing-based solutions. (**Usage**)

7. Content

UNIT 1: Introduction (20)	
Competences:	
Content	**Generales Goals**
- Introduction to Competetive Programming - Computacional model - Runtime and space complexity - Recurrence and recursion - Divide and conquer	- Identify and learn how to use the resources in the Random Access Machine (RAM) computacional model. [Usage] - Compute the runtime and space complexity for written algorithms. [Usage] - Compute the recurrence relations for recursive algorithms. [Usage] - Solve problems related to searching and sorting. [Usage] - Learning to select the right algorithms for divide-and-conquer problems. [Usage] - Design new algorithms for real-world problem solving.[Usage]
Readings: Cormen et al. (2009), Halim (2013), Kulikov (2019), Miguel A. Revilla (2003), Laaksonen (2017), Aziz, Lee, and Prakash (2012)	

UNIT 2: Data structure (20)	
Competences:	
Content	**Generales Goals**
- Arrays and strings problems - Linked lists problems - Stacks and queues problems - Trees problems - Hash tables problems - Heaps problems	- Recognize different data structures, their complexities, uses and restrictions.[Usage] - Identify the type of data structure appropriate to the resolution of the problem. [Usage] - Recognize types of problems associated with operations on data structures such as searching, inserting, deleting and updating.[Usage]
Readings: Cormen et al. (2009), Halim (2013), Kulikov (2019), Miguel A. Revilla (2003), Laaksonen (2017), Aziz, Lee, and Prakash (2012)	

UNIT 3: Algorithmic Design Paradigms (20)	
Competences:	
Content	**Generales Goals**
- Brute force - Divide and conquer - Backtracking - Greedy - Dynamic Programming	- Learning the different algorithhmic design paradigms.[Usage] - Learning to select the right algorithms for different problems applying different algorithhmic design paradigms.[Usage]
Readings: Cormen et al. (2009), Halim (2013), Kulikov (2019), Miguel A. Revilla (2003), Laaksonen (2017), Aziz, Lee, and Prakash (2012)	

UNIT 4: Graphs (20)	
Competences:	
Content	**Generales Goals**
- Graphs transversal - Graphs aplications - Shortest path - Networks and flows	- Identify problems classified as graph problems. [Usage] - Learn how to select the right algorithms for network problems (transversal, MST, shortest-path, network and flows). [Usage]
Readings: Cormen et al. (2009), Halim (2013), Kulikov (2019), Miguel A. Revilla (2003), Laaksonen (2017), Aziz, Lee, and Prakash (2012)	

UNIT 5: Advanced topics (20)	
Competences:	
Content	**Generales Goals**
- Number theory - Probabilities and combinations - String algorithms (tries, string hashing, z-algorithm) - Geometric algorithms	- Learning to select the right algorithms for problems in number theory and mathematics as they are important in competitive programming. [Usage] - Learning to select the right algorithms for problems about probabilities and combinations, strings and computational geometry. [Usage]
Readings: Cormen et al. (2009), Halim (2013), Kulikov (2019), Miguel A. Revilla (2003), Laaksonen (2017), Aziz, Lee, and Prakash (2012)	

UNIT 6: Domain specific problems (20)	
Competences:	
Content	**Generales Goals**
- Latency and throughput - Parallelism - Networks - Storage - High availability - Caching - Proxies - Load balancers - Key-value stores - Replicating and sharing - Leader election - Rate limiting - Logging and monitoring	Learning to design systems for different domain-specific problems by applying knowledge about networks, distributed computing, high availability, storage and system architecture.[Usage]
Readings: Cormen et al. (2009), Halim (2013), Kulikov (2019), Miguel A. Revilla (2003), Laaksonen (2017), Aziz, Lee, and Prakash (2012)	

8. Methodology

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. Assessment

Continuous Assessment 1 : 20 %

Partial Exam : 30 %

Continuous Assessment 2 : 20 %

Final exam : 30 %

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
Undergraduate Program in Information Systems
SILABO

CS312. Advanced Data Structures (Mandatory)

2022-II

1. General information		
1.1 School	:	Sistemas de Información
1.2 Course	:	CS312. Advanced Data Structures
1.3 Semester	:	6to Semestre.
1.4 Prerrequisites	:	CS212. Algorithm Analysis and Design. (5th Sem)
1.5 Type of course	:	Mandatory
1.6 Learning modality	:	Face to face
1.7 Horas	:	2 HT; 2 HP; 2 HL;
1.8 Credits	:	4

2. Professors

3. Course foundation
Algorithms and data structures are a fundamental part of computer science that allow us to organize information more efficiently, so it is important for every professional in the area to have a solid background in this regard.   In the course of advanced data structures our goal is for the student to know and analyze complex structures, such as Multidimensional Access Methods, Spatio-Temporal Access Methods and Metric Access Methods, Compact Data Structures, etc.

4. Summary
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. Generales Goals
That the student understands, designs, implements, applies and Propose innovative data structures to solve problems related to the handling of multidimensional data, retrieval of information by similarity, search engines and other computational problems.

6. Contribution to Outcomes
This discipline contributes to the achievement of the following outcomes: **1)** Analyze a complex computing problem and to apply principles of computing and other relevant disciplines to identify solutions. (**Usage**)   **2)** Design, implement and evaluate a computing-based solution to meet a given set of computing requirements in the context of the program's discipline. (**Familiarity**)   **6)** Apply computer science theory and software development fundamentals to produce computing-based solutions. (**Familiarity**)

7. Content

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

Competences:

Content	Generales Goals
- Structured Programming - Object-oriented programming - Abstract Data Types - Independence of the user programming language of the structure - Platform Independence - Concurrency control - Data Protection - Encapsulation levels (struct, class, namespace, etc)	- That the student understands the basic differences that involve the different techniques of implementation of data structures[Usage] - That the student analyze the advantages and disadvantages of each of the existing techniques[Usage]

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

UNIT 2: Métodos de Acceso Multidimensionales (16)

Competences:

Content	Generales Goals
- Access Methods for Point Data - Access Methods for non-point data - Problems with dimension enhancement	- That the student understands to know and implement some Access Methods for multidimensional data and temporal space[Usage] - That the student understands the potential of these Access Methods in the future of commercial databases[Usage]

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

UNIT 3: Métodos de Acceso Métrico (20)

Competences:

Content	Generales Goals
- Metric Access Methods for discrete distances - Metric Access Methods for Continuous Distances	- That the student understands to know and implement some methods of metric access[Usage] - That the student understands the importance of these Access Methods for Information Retrieval by similarity[Usage]

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

UNIT 4: Métodos de Acceso Aproximados (20)	
Competences:	
Content	**Generales Goals**
- Space Filling Curves - Locality Sensitive Hashing	- That the student understands to know and implement some approximate access methods[Usage] - That the student understands the importance of these Access Methods for Information Retrieval by Similarity in environments where Scalability is a very important factor [Usage]
Readings: Samet (2006), PGregory Shakhnarovich and Indyk (2006), Zezula et al. (2007)	

UNIT 5: Seminarios (8)	
Competences:	
Content	**Generales Goals**
- Access Methods Temporary Space - Generic Data Structures	That the student can discuss the latest advances in access methods for different domains of knowledge [Usage]
Readings: Samet (2006), Navarro (2016), Chávez et al. (2001)	

8. Methodology
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. Assessment
Continuous Assessment 1 : 20 %
Partial Exam : 30 %
Continuous Assessment 2 : 20 %
Final exam : 30 %

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.

Navarro, Gonzalo (2016). *Compact Data Structures*. Cambridge University Press. ISBN: 978-1107152380.

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
Undergraduate Program in Information Systems
SILABO

CS393. Information systems (Mandatory)

2022-II

1. General information		
1.1 School	:	Sistemas de Información
1.2 Course	:	CS393. Information systems
1.3 Semester	:	6to Semestre.
1.4 Prerrequisites	:	CS291. Software Engineering I. (5th Sem)
1.5 Type of course	:	Mandatory
1.6 Learning modality	:	Face to face
1.7 Horas	:	2 HT; 2 HP; 2 HL;
1.8 Credits	:	4

2. Professors

3. Course foundation
Analyze techniques for the correct implementation of scalable, robust, reliable and efficient information systems in organizations.

4. Summary
1. Introduction 2. Strategy 3. Implementation

5. Generales Goals
• Implement correctly (scalable, robust, reliable and efficient) Information Systems in organizations.

6. Contribution to Outcomes
This discipline contributes to the achievement of the following outcomes:

2) Design, implement and evaluate a computing-based solution to meet a given set of computing requirements in the context of the program's discipline. (**Usage**)

6) Apply computer science theory and software development fundamentals to produce computing-based solutions. (**Assessment**)

7. Content

UNIT 1: Introduction (15)	
Competences:	
Content	Generales Goals
• Introduction to information management. • Software for information management. • Technology for information management.	• Correctly apply technology for information management [Assessment]
Readings: Sommerville (2017), Pressman and Maxim (2015), K. C. Laudon and J. P. Laudon (2017)	

UNIT 2: Strategy (15)	
Competences:	
Content	Generales Goals
• Strategy for information management. • Strategy for knowledge management • Strategy for information system.	• Apply and evaluate correctly management strategies [Assessment]
Readings: Sommerville (2017), Pressman and Maxim (2015)	

UNIT 3: Implementation (15)	
Competences:	
Content	Generales Goals
• Management Information Systems Development. • Change management • Information Architecture	• Implement and correctly evaluate implementation strategies [Assessment]
Readings: Sommerville (2017), Pressman and Maxim (2015)	

---

8. Methodology

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. Assessment

Continuous Assessment 1 : 20 %

Partial Exam : 30 %

Continuous Assessment 2 : 20 %

Final exam : 30 %

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
Undergraduate Program in Information Systems
SILABO

MA307. Mathematics applied to computing (Mandatory)

2022-II

1. General information		
1.1 School	:	Sistemas de Información
1.2 Course	:	MA307. Mathematics applied to computing
1.3 Semester	:	6to Semestre.
1.4 Prerrequisites	:	MA201. Calculus II. (4th Sem)
1.5 Type of course	:	Mandatory
1.6 Learning modality	:	Face to face
1.7 Horas	:	2 HT; 2 HP; 2 HL;
1.8 Credits	:	4

2. Professors

3. Course foundation
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. Summary
1. 2. 3. 4. 5. 6.

5. Generales Goals
• 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. Contribution to Outcomes
This discipline contributes to the achievement of the following outcomes:

1) Analyze a complex computing problem and to apply principles of computing and other relevant disciplines to identify solutions. (**Usage**)

6) Apply computer science theory and software development fundamentals to produce computing-based solutions. (**Usage**)

7. Content

UNIT 1: (0)
Competences:

Content	Generales Goals
- Espacios vectoriales. - Independencia, base y dimensión. - Dimensiones y ortogonalidad de los cuatro subespacios. - Aproximaciones por mínimos cuadrados. - Proyecciones - Bases ortogonales y Gram-Schmidt	- Identificar espacios generados por vectores linealmente independientes[Usage] - Construir conjuntos de vectores ortogonales[Usage] - Aproximar funciones por polinomios trigonométricos[Usage]

Readings: Strang (2003), Apóstol (1973)

UNIT 2: (0)
Competences:

Content	Generales Goals
- Concepto de transformación lineal. - Matriz de una transformación lineal. - Cambio de base. - Diagonalización y pseudoinversa	- Determinar el núcleo y la imagen de una transformación[Usage] - Construir la matriz de una transformación[Usage] - Determinar la matriz de cambio de base[Usage]

Readings: Strang (2003), Apóstol (1973)

UNIT 3: (0)
Competences:

Content	Generales Goals
- Diagonalización de una matriz - Matrices simétricas - Matrices definidas positivas - Matrices similares - La descomposición de valor singular	- Encontrar la representación diagonal de una matriz[Usage] - Determinar la similaridad entre matrices[Usage] - Reducir una forma cuadrática real a diagonal[Usage]

Readings: Strang (2003), Apóstol (1973)

UNIT 4: (0)
Competences:

Content	Generales Goals
- Exponencial de una matriz - Teoremas de existencia y unicidad para sistemas lineales homogéneos con coeficientes constantes - Sistemas lineales no homogéneas con coeficientes constantes.	- Hallar la solución general de un sistema lineal no homogéneo[Usage] - Resolver problemas donde intervengan sistemas de ecuaciones diferenciales[Usage]

Readings: Zill (2002), Apóstol (1973)

UNIT 5: (0)	
Competences:	
Content	**Generales Goals**
- Sistemas dinámicos - El teorema fundamental - Existencia y unicidad - El flujo de una ecuación diferencial	- Discutir la existencia y la unicidad de una ecuación diferencial[Usage] - Analizar la continuidad de las soluciones[Usage] - Estudiar la prolongación de una solución[Usage]
Readings: Hirsh and Smale (1974)	

UNIT 6: (0)	
Competences:	
Content	**Generales Goals**
- Estabilidad - Funciones de Liapunov - Sistemas gradientes	- Analizar la estabilidad de una solución[Usage] - Hallar la función de Liapunov para puntos de equilibrio[Usage] - Trazar el retrato de fase un flujo gradiente[Usage]
Readings: Zill (2002), Hirsh and Smale (1974)	

8. Methodology
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. Assessment
Continuous Assessment 1 : 20 %
Partial Exam : 30 %
Continuous Assessment 2 : 20 %
Final exam : 30 %

References

Apóstol, Tom M (1973). *Calculus Vol II*. Editorial Reverté.

Hirsh, Morris W. and Stephen Smale (1974). *Differential Equatons, Dynamical Systems, and Linear Álgebra*. 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
Undergraduate Program in Information Systems
SILABO

FG203. Public Speaking (Mandatory)

2022-II

1. General information		
1.1 School	:	Sistemas de Información
1.2 Course	:	FG203. Public Speaking
1.3 Semester	:	6to Semestre.
1.4 Prerrequisites	:	FG106. Theater. (5th Sem)
1.5 Type of course	:	Mandatory
1.6 Learning modality	:	Face to face
1.7 Horas	:	1 HT; 2 HP;
1.8 Credits	:	2

2. Professors

3. Course foundation
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. Summary
1. 2.

5. Generales Goals
• 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. Contribution to Outcomes
This discipline contributes to the achievement of the following outcomes:

3) Communicate effectively in a variety of professional contexts. (**Usage**)

4) Recognize professional responsabilities and make informed judgments in computing practice based on legal and ethical principles. (**Usage**)

7) Develop computational technology for the well-being of all, contributing with human formation, scientific, technological and professional skills to solve social problems of our community. (**Usage**)

7. Content

UNIT 1: (3)
Competences:

Content	Generales Goals
• La Oratoria • La función de la palabra. • El proceso de la comunicación. • Bases racionales y emocionales de la oratoria – La expresión oral en la participación. • Fuentes de conocimiento para la oratoria: niveles de cultura general.	• Comprensión: interpretar, ejemplificar y generalizar las bases de la oratoria como fundamento teórico y práctico. [Usage].

Readings: Monroe and Ehninger (1976), Rodríguez (n.d.)

UNIT 2: (4)
Competences:

Content	Generales Goals

• Cualidades de un buen orador.
 	
• Normas para primeros discursos.
 	
• El cuerpo humano como instrumento de comunicación:
 	
    – La expresión corporal en el discurso
 	
    – La voz en el discurso.
 	
• Oradores con historia y su ejemplo.
	• Comprensión: Interpretar, ejemplificar y generalizar conocimientos y habilidades de la comunicación oral mediante la experiencia de grandes oradores y la suya propia. [Usage].
	
	• Aplicación: Implementar, usar, elegir y desempeñar los conocimientos adquiridos para expresarse en público en forma eficiente, inteligente y agradable. [Usage].

Readings: Rodríguez (n.d.)

8. Methodology
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. Assessment
Continuous Assessment 1 : 20 % **Partial Exam** : 30 % **Continuous Assessment 2** : 20 % **Final exam** : 30 %

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
Undergraduate Program in Information Systems
SILABO

CS231. Networking and Communication (Mandatory)

2022-II

1. General information		
1.1 School	:	Sistemas de Información
1.2 Course	:	CS231. Networking and Communication
1.3 Semester	:	7mo Semestre.
1.4 Prerrequisites	:	CS2S1. Operating systems . (6th Sem)
1.5 Type of course	:	Mandatory
1.6 Learning modality	:	Face to face
1.7 Horas	:	1 HT; 2 HP; 2 HL;
1.8 Credits	:	3

2. Professors

3. Course foundation
The ever-growing development of communication and information technologies means that there is a marked tendency to establish more computer networks that allow better information management..   In this second course, participants will be introduced to the problems of communication between computers, through the study and implementation of communication protocols such as TCP / IP and the implementation of software on these protocols

4. Summary
1. Introduction 2. Networked Applications 3. Reliable Data Delivery 4. Routing and Forwarding 5. Local Area Networks 6. Resource Allocation 7. Mobility 8. Social Networking

5. Generales Goals
- That the student implements and / or modifies a data communication protocols.   - That the student master the data transmission techniques used by the existing network protocols.   - That the student knows the latest trends in networks that are being applied on the Internet.

6. Contribution to Outcomes

This discipline contributes to the achievement of the following outcomes:

1) Analyze a complex computing problem and to apply principles of computing and other relevant disciplines to identify solutions. (**Usage**)

2) Design, implement and evaluate a computing-based solution to meet a given set of computing requirements in the context of the program's discipline. (**Usage**)

4) Recognize professional responsabilities and make informed judgments in computing practice based on legal and ethical principles. (**Familiarity**)

6) Apply computer science theory and software development fundamentals to produce computing-based solutions. (**Usage**)

7) Develop computational technology for the well-being of all, contributing with human formation, scientific, technological and professional skills to solve social problems of our community. (**Assessment**)

---

7. Content

UNIT 1: Introduction (5)

Competences:

Content	Generales Goals
- Organization of the Internet (Internet Service Providers, Content Providers, etc.) - Switching techniques (e.g., circuit, packet) - Physical pieces of a network, including hosts, routers, switches, ISPs, wireless, LAN, access point, and firewalls - Layering principles (encapsulation, multiplexing) - Roles of the different layers (application, transport, network, datalink, physical)	- Articulate the organization of the Internet [Familiarity] - List and define the appropriate network terminology [Familiarity] - Describe the layered structure of a typical networked architecture [Familiarity] - Identify the different types of complexity in a network (edges, core, etc) [Familiarity]

Readings: Kurose and Ross (2013)

UNIT 2: Networked Applications (5)

Competences:

Content	Generales Goals
- Naming and address schemes (DNS, IP addresses, Uniform Resource Identifiers, etc.) - Distributed applications (client/server, peer-to-peer, cloud, etc.) - HTTP as an application layer protocol - Multiplexing with TCP and UDP - Socket APIs	- List the differences and the relations between names and addresses in a network [Familiarity] - Define the principles behind naming schemes and resource location [Familiarity] - Implement a simple client-server socket-based application [Usage]

Readings: Kurose and Ross (2013)

UNIT 3: Reliable Data Delivery (10)	
Competences:	
Content	**Generales Goals**
- Error control (retransmission techniques, timers) - Flow control (acknowledgements, sliding window) - Performance issues (pipelining) - TCP	- Describe the operation of reliable delivery protocols [Familiarity] - List the factors that affect the performance of reliable delivery protocols [Familiarity] - Design and implement a simple reliable protocol [Usage]
Readings: Kurose and Ross (2013)	

UNIT 4: Routing and Forwarding (12)	
Competences:	
Content	**Generales Goals**
- Routing versus forwarding - Static routing - Internet Protocol (IP) - Scalability issues (hierarchical addressing)	- Describe the organization of the network layer [Familiarity] - Describe how packets are forwarded in an IP network [Familiarity] - List the scalability benefits of hierarchical addressing [Familiarity]
Readings: Kurose and Ross (2013)	

UNIT 5: Local Area Networks (10)	
Competences:	
Content	**Generales Goals**
- Multiple Access Problem - Common approaches to multiple access (exponential-backoff, time division multiplexing, etc) - Local Area Networks - Ethernet - Switching	- Describe how frames are forwarded in an Ethernet network [Familiarity] - Describe the interrelations between IP and Ethernet [Familiarity] - Describe the steps used in one common approach to the multiple access problem [Familiarity]
Readings: Kurose and Ross (2013)	

UNIT 6: Resource Allocation (12)	
Competences:	
Content	**Generales Goals**
- Need for resource allocation - Fixed allocation (TDM, FDM, WDM) versus dynamic allocation - End-to-end versus network assisted approaches - Fairness - Principles of congestion control - Approaches to Congestion (e.g., Content Distribution Networks)	- Describe how resources can be allocated in a network [Familiarity] - Describe the congestion problem in a large network [Familiarity] - Compare and contrast fixed and dynamic allocation techniques [Familiarity] - Compare and contrast current approaches to congestion [Familiarity]
Readings: Kurose and Ross (2013)	

UNIT 7: Mobility (5)	
Competences:	
Content	**Generales Goals**
- Principles of cellular networks 802.11 networks - Issues in supporting mobile nodes (home agents)	- Describe the organization of a wireless network [Familiarity] - Describe how wireless networks support mobile users [Familiarity]
Readings: Kurose and Ross (2013), Chayapathi (2016)	

UNIT 8: Social Networking (5)	
Competences:	
Content	**Generales Goals**
- Social networks overview - Example social network platforms - Structure of social network graphs - Social network analysis	- Discuss the key principles (such as membership, trust) of social networking [Familiarity] - Describe how existing social networks operate [Familiarity] - Construct a social network graph from network data [Usage] - Analyze a social network to determine who the key people are [Usage] - Evaluate a given interpretation of a social network question with associated data [Familiarity]
Readings: Kurose and Ross (2013), Kadushin (2011)	

8. Methodology
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. Assessment
Continuous Assessment 1 : 20 %   **Partial Exam** : 30 %   **Continuous Assessment 2** : 20 %   **Final exam** : 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
Undergraduate Program in Information Systems
SILABO

CS261. Artificial intelligence (Mandatory)

2022-II

1. General information		
1.1 School	:	Sistemas de Información
1.2 Course	:	CS261. Artificial intelligence
1.3 Semester	:	7mo Semestre.
1.4 Prerrequisites	:	MA203. Statistics and Probability. (4th Sem)
1.5 Type of course	:	Mandatory
1.6 Learning modality	:	Face to face
1.7 Horas	:	2 HT; 2 HP; 2 HL;
1.8 Credits	:	4

2. Professors

3. Course foundation
Research in Artificial Intelligence has led to the development of numerous relevant tonic, aimed at the automation of human intelligence, giving a panoramic view of different algorithms that simulate the different aspects of the behavior and the intelligence of the human being.

4. Summary
1. Fundamental Issues 2. Agents 3. Basic Search Strategies 4. Advanced Search 5. Reasoning Under Uncertainty 6. Basic Machine Learning 7. Advanced Machine Learning 8. Natural Language Processing 9. Perception and Computer Vision

5. Generales Goals
- Evaluate the possibilities of simulation of intelligence, for which the techniques of knowledge modeling will be studied.   - Build a notion of intelligence that later supports the tasks of your simulation.

6. Contribution to Outcomes
This discipline contributes to the achievement of the following outcomes:

1) Analyze a complex computing problem and to apply principles of computing and other relevant disciplines to identify solutions. (**Usage**)

5) Function effectively as a member or leader of a team engaged in activities appropriate to the program's discipline. (**Familiarity**)

6) Apply computer science theory and software development fundamentals to produce computing-based solutions. (**Familiarity**)

7. Content

UNIT 1: Fundamental Issues (2)	
Competences:	
Content	**Generales Goals**

• Overview of AI problems, examples of successful recent AI applications
 	
• What is intelligent behavior?
 	
    – The Turing test
    – Rational versus non-rational reasoning
 	
• Problem characteristics
 	
    – Fully versus partially observable
    – Single versus multi-agent
    – Deterministic versus stochastic
    – Static versus dynamic
    – Discrete versus continuous
 	
• Nature of agents
 	
    – Autonomous versus semi-autonomous
    – Reflexive, goal-based, and utility-based
    – The importance of perception and environmental interactions
 	
• Philosophical and ethical issues.
	• Describe Turing test and the "Chinese Room" thought experiment [Usage]
	
	• Determing the characteristics of a given problem that an intelligent systems must solve [Usage]

Readings: De Castro (2006), Ponce-Gallegos et al. (2014)	

UNIT 2: Agents (2)	
Competences:	
Content	**Generales Goals**
- Definitions of agents - Agent architectures (e.g., reactive, layered, cognitive) - Agent theory - Rationality, game theory - Decision-theoretic agents - Markov decision processes (MDP) - Software agents, personal assistants, and information access - Collaborative agents - Information-gathering agents - Believable agents (synthetic characters, modeling emotions in agents) - Learning agents - Multi-agent systems - Collaborating agents - Agent teams - Competitive agents (e.g., auctions, voting) - Swarm systems and biologically inspired models	- List the defining characteristics of an intelligent agent [Usage] - Characterize and contrast the standard agent architectures [Usage] - Describe the applications of agent theory to domains such as software agents, personal assistants, and believable agents [Usage] - Describe the primary paradigms used by learning agents [Usage] - Demonstrate using appropriate examples how multi-agent systems support agent interaction [Usage]
Readings: Nilsson (2001), Russell and Norvig (2003), Ponce-Gallegos et al. (2014)	

UNIT 3: Basic Search Strategies (2)	
Competences:	
Content	**Generales Goals**
- Problem spaces (states, goals and operators), problem solving by search - Factored representation (factoring state into variables) - Uninformed search (breadth-first, depth-first, depth-first with iterative deepening) - Heuristics and informed search (hill-climbing, generic best-first, A*) - Space and time efficiency of search - Two-player games (introduction to minimax search) - Constraint satisfaction (backtracking and local search methods)	- Formulate an efficient problem space for a problem expressed in natural language (eg, English) in terms of initial and goal states, and operators [Usage] - Describe the role of heuristics and describe the trade-offs among completeness, optimality, time complexity, and space complexity [Usage] - Describe the problem of combinatorial explosion of search space and its consequences [Usage] - Compare and contrast basic search issues with game playing issues [Usage]
Readings: Nilsson (2001), Ponce-Gallegos et al. (2014)	

UNIT 4: Advanced Search (18)	
Competences:	
Content	**Generales Goals**
- Stochastic search - Simulated annealing - Genetic algorithms - Monte-Carlo tree search - Constructing search trees, dynamic search space, combinatorial explosion of search space - Implementation of A* search, beam search - Minimax search, alpha-beta pruning - Expectimax search (MDP-solving) and chance nodes	- Design and implement a genetic algorithm solution to a problem [Usage] - Design and implement a simulated annealing schedule to avoid local minima in a problem [Usage] - Design and implement A*,beam search to solve a problem [Usage] - Apply minimax search with alpha-beta pruning to prune search space in a two-player game [Usage] - Compare and contrast genetic algorithms with classic search techniques [Usage] - Compare and contrast various heuristic searches vis-a-vis applicability to a given problem [Usage]
Readings: Goldberg (1989), Nilsson (2001), Russell and Norvig (2003), Ponce-Gallegos et al. (2014)	

UNIT 5: Reasoning Under Uncertainty (18)	
Competences:	
Content	**Generales Goals**
- Review of basic probability - Random variables and probability distributions - Axioms of probability - Probabilistic inference - Bayes' Rule - Conditional Independence - Knowledge representations - Bayesian Networks - Exact inference and its complexity - Randomized sampling (Monte Carlo) methods (e.g. Gibbs sampling) - Markov Networks - Relational probability models - Hidden Markov Models	- Apply Bayes' rule to determine the probability of a hypothesis given evidence [Usage] - Explain how conditional independence assertions allow for greater efficiency of probabilistic systems [Usage] - Identify examples of knowledge representations for reasoning under uncertainty [Usage] - State the complexity of exact inference Identify methods for approximate inference [Usage]
Readings: Koller and Friedman (2009), Russell and Norvig (2003)	

UNIT 6: Basic Machine Learning (4)	
Competences:	
Content	**Generales Goals**
• Definition and examples of broad variety of machine learning tasks, including classification • Inductive learning • Simple statistical-based learning, such as Naive Bayesian Classifier, decision trees • The over-fitting problem • Measuring classifier accuracy	• List the differences among the three main styles of learning: supervised, reinforcement, and unsupervised [Usage] • Identify examples of classification tasks, including the available input features and output to be predicted [Usage] • Explain the difference between inductive and deductive learning [Usage] • Describe over-fitting in the context of a problem [Usage] • Apply the simple statistical learning algorithm such as Naive Bayesian Classifier to a classification task and measure the classifier's accuracy [Usage]
Readings: Mitchell (1998), Russell and Norvig (2003), Ponce-Gallegos et al. (2014)	

UNIT 7: Advanced Machine Learning (20)	
Competences:	
Content	**Generales Goals**

• Definition and examples of broad variety of machine learning tasks
 	
• General statistical-based learning, parameter estimation (maximum likelihood)
 	
• Inductive logic programming (ILP)
 	
• Supervised learning
  – Learning decision trees
  – Learning neural networks
  – Support vector machines (SVMs)
 	
• Unsupervised Learning and clustering
  – EM
  – K-means
  – Self-organizing maps
 	
• Semi-supervised learning
 	
• Learning graphical models
 	
• Performance evaluation (such as cross-validation, area under ROC curve)
 	
• Application of Machine Learning algorithms to Data Mining (cross-reference IM/Data Mining)
	• Explain the differences among the three main styles of learning: supervised, reinforcement, and unsupervised [Usage]
	
	• Implement simple algorithms for supervised learning, reinforcement learning, and unsupervised learning [Usage]
	
	• Determine which of the three learning styles is appropriate to a particular problem domain [Usage]
	
	• Compare and contrast each of the following techniques, providing examples of when each strategy is superior: decision trees, neural networks, and belief networks [Usage]
	
	• Evaluate the performance of a simple learning system on a real-world dataset [Usage]
	
	• Characterize the state of the art in learning theory, including its achievements and its shortcomings [Usage]
	
	• Explain the problem of overfitting, along with techniques for detecting and managing the problem [Usage]

Readings: Russell and Norvig (2003), Koller and Friedman (2009), Murphy (2012)	

UNIT 8: Natural Language Processing (12)	
Competences:	
Content	**Generales Goals**

• Deterministic and stochastic grammars
 	
• Parsing algorithms
 	
   – CFGs and chart parsers (e.g. CYK)
   – Probabilistic CFGs and weighted CYK
 	
• Representing meaning / Semantics
 	
   – Logic-based knowledge representations
   – Semantic roles
   – Temporal representations
   – Beliefs, desires, and intentions
 	
• Corpus-based methods
 	
• N-grams and HMMs
 	
• Smoothing and backoff
 	
• Examples of use: POS tagging and morphology
 	
• Information retrieval
 	
   – Vector space model
      ∗ TF & IDF
   – Precision and recall
 	
• Information extraction
 	
• Language translation
 	
• Text classification, categorization
 	
   – Bag of words model
	• Define and contrast deterministic and stochastic grammars, providing examples to show the adequacy of each [Usage]
	
	• Simulate, apply, or implement classic and stochastic algorithms for parsing natural language [Usage]
	
	• Identify the challenges of representing meaning [Usage]
	
	• List the advantages of using standard corpora Identify examples of current corpora for a variety of NLP tasks [Usage]
	
	• Identify techniques for information retrieval, language translation, and text classification [Usage]

Readings: Nilsson (2001), Russell and Norvig (2003), Ponce-Gallegos et al. (2014)	

UNIT 9: Perception and Computer Vision (12)	
Competences:	
Content	**Generales Goals**

• Computer vision
    – Image acquisition, representation, processing and properties
    – Shape representation, object recognition and segmentation
    – Motion analysis
• Modularity in recognition
• Approaches to pattern recognition
    – Classification algorithms and measures of classification quality
    – Statistical techniques
	• Summarize the importance of image and object recognition in AI and indicate several significant applications of this technology [Usage]
	• List at least three image-segmentation approaches, such as thresholding, edge-based and region-based algorithms, along with their defining characteristics, strengths, and weaknesses [Usage]
	• Implement 2d object recognition based on contour- and/or region-based shape representations [Usage]
	• Provide at least two examples of a transformation of a data source from one sensory domain to another, eg, tactile data interpreted as single-band 2d images [Usage]
	• Implement a feature-extraction algorithm on real data, eg, an edge or corner detector for images or vectors of Fourier coefficients describing a short slice of audio signal [Usage]
	• Implement a classification algorithm that segments input percepts into output categories and quantitatively evaluates the resulting classification [Usage]
	• Evaluate the performance of the underlying feature-extraction, relative to at least one alternative possible approach (whether implemented or not) in its contribution to the classification task (8), above [Usage]

Readings: Nilsson (2001), Russell and Norvig (2003), Ponce-Gallegos et al. (2014)	

---

8. Methodology

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. Assessment

Continuous Assessment 1 : 20 %

Partial Exam : 30 %

Continuous Assessment 2 : 20 %

Final exam : 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 Artifical: Un enfoque moderno*. Prentice Hall.

Universidad Nacional de Colombia (UNAL) Sede Manizales
Undergraduate Program in Information Systems
SILABO

CS2H1. Computer Human Interaction (Mandatory)

2022-II

1. General information		
1.1 School	:	Sistemas de Información
1.2 Course	:	CS2H1. Computer Human Interaction
1.3 Semester	:	7mo Semestre.
1.4 Prerrequisites	:	CS393. Information systems. (6th Sem)
1.5 Type of course	:	Mandatory
1.6 Learning modality	:	Face to face
1.7 Horas	:	1 HT; 4 HL;
1.8 Credits	:	3

2. Professors

3. Course foundation
Language has been one of the most significant creations of humanity. From body language and gesture, through verbal and written communication, to iconic symbolic codes and others, it has made possible complex interactions Among humans and facilitated considerably the communication of information. With the invention of automatic and semi-automatic devices, including computers, The need for languages or interfaces to be able to interact with them, has gained great importance. The utility of the software, coupled with user satisfaction and increased productivity, depends on the effectiveness of the User-Computer Interface. So much so, that often the interface is the most important factor in the success and failure of any computer system. The design and implementation of appropriate Human-Computer Interfaces, which in addition to complying with the technical requirements and the transactional logic of the application, consider the subtle psychological implications, sciences and user facilities, It consumes a good part of the life cycle of a software project, and requires specialized skills, both for the construction of the same, and for the performance of usability tests.

4. Summary
1. Foundations 2. Factores Humanos 3. User-centered design and testing 4. Designing Interaction 5. New Interactive Technologies 6. Collaboration and communication

5. Generales Goals
- Know and apply criteria of usability and accessibility to the design and construction of human-computer interfaces, always looking for technology to adapt to people and not people to technology.   - That the student has a vision focused on the user experience by applying appropriate conceptual and technological approaches.   - Understand how emerging technology makes possible new styles of interaction.   - Determine the basic requirements at the interface level, hardware and software for the construction of immersive environments.

6. Contribution to Outcomes

This discipline contributes to the achievement of the following outcomes:

1) Analyze a complex computing problem and to apply principles of computing and other relevant disciplines to identify solutions. (**Familiarity**)

2) Design, implement and evaluate a computing-based solution to meet a given set of computing requirements in the context of the program's discipline. (**Assessment**)

3) Communicate effectively in a variety of professional contexts. (**Usage**)

4) Recognize professional responsabilities and make informed judgments in computing practice based on legal and ethical principles. (**Familiarity**)

5) Function effectively as a member or leader of a team engaged in activities appropriate to the program's discipline. (**Usage**)

7) Develop computational technology for the well-being of all, contributing with human formation, scientific, technological and professional skills to solve social problems of our community. (**Familiarity**)

7. Content

UNIT 1: Foundations (8)

Competences:

Content	Generales Goals
- Contexts for HCI (anything with a user interface, e.g., webpage, business applications, mobile applications, and games) - Usability heuristics and the principles of usability testing - Processes for user-centered development, e.g., early focus on users, empirical testing, iterative design - Principles of good design and good designers; engineering tradeoffs - Different measures for evaluation, e.g., utility, efficiency, learnability, user satisfaction	- Discuss why human-centered software development is important [Familiarity] - Define a user-centered design process that explicitly takes account of the fact that the user is not like the developer or their acquaintances [Familiarity] - Summarize the basic precepts of psychological and social interaction [Familiarity] - Develop and use a conceptual vocabulary for analyzing human interaction with software: affordance, conceptual model, feedback, and so forth [Familiarity]

Readings: Dix et al. (2004), Stone et al. (2005), Rogers and Sharp (2011)

UNIT 2: Factores Humanos (8)	
Competences:	
Content	**Generales Goals**
- Cognitive models that inform interaction design, e.g., attention, perception and recognition, movement, and memory; gulfs of expectation and execution - Physical capabilities that inform interaction design, e.g., color perception, ergonomics - Accessibility, e.g., interfaces for differently-abled populations (e.g., blind, motion-impaired) - Interfaces for differently-aged population groups (e.g., children, 80+)	Create and conduct a simple usability test for an existing software application [Familiarity]
Readings: Dix et al. (2004), Stone et al. (2005), Rogers and Sharp (2011), Mathis (2011), Norman (2004)	

UNIT 3: User-centered design and testing (16)	
Competences:	
Content	**Generales Goals**

Content	Generales Goals

• Approaches to, and characteristics of, the design process
 	
• Functionality and usability requirements
 	
• Techniques for gathering requirements, e.g., interviews, surveys, ethnographic and contextual enquiry
 	
• Techniques and tools for the analysis and presentation of requirements, e.g., reports, personas
 	
• Task analysis, including qualitative aspects of generating task analytic models
 	
• Consideration of HCI as a design discipline
 	
    – Sketching
    – Participatory design
 	
    – Sketching
    – Diseño participativo
 	
• Prototyping techniques and tools, e.g., sketching, storyboards, low-fidelity prototyping, wireframes
 	
• Low-fidelity (paper) prototyping
 	
• Quantitative evaluation techniques, e.g., keystroke-level evaluation
 	
• Evaluation without users, using both qualitative and quantitative techniques, e.g., walkthroughs, GOMS, expert-based analysis, heuristics, guidelines, and standard
 	
• Evaluation with users, e.g., observation, think-aloud, interview, survey, experiment
 	
• Challenges to effective evaluation, e.g., sampling, generalization
 	
• Reporting the results of evaluations
 	
• Internationalization, designing for users from other cultures, cross-cultural
	• Conduct a quantitative evaluation and discuss/report the results [Familiarity]
	
	• For an identified user group, undertake and document an analysis of their needs [Familiarity]
	
	• Discuss at least one national or international user interface design standard [Familiarity]
	
	• Explain how user-centred design complements other software process models [Familiarity]
	
	• Use lo-fi (low fidelity) prototyping techniques to gather, and report, user responses [Usage]
	
	• Choose appropriate methods to support the development of a specific UI [Assessment]
	
	• Use a variety of techniques to evaluate a given UI [Assessment]
	
	• Compare the constraints and benefits of different evaluative methods [Assessment]

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

UNIT 4: Designing Interaction (8)	
Competences:	
Content	**Generales Goals**
- Principles of graphical user interfaces (GUIs) - Elements of visual design (layout, color, fonts, labeling) - Handling human/system failure - User interface standards - Presenting information: navigation, representation, manipulation - Interface animation techniques (e.g., scene graphs) - Widget classes and libraries - Internationalization, designing for users from other cultures, cross-cultural - Choosing interaction styles and interaction techniques	Create a simple application, together with help and documentation, that supports a graphical user interface [Usage]
Readings: Dix et al. (2004), Stone et al. (2005), Rogers and Sharp (2011), Johnson (2010), Mathis (2011), Leavitt and Shneiderman (2006)	

UNIT 5: New Interactive Technologies (8)	
Competences:	
Content	**Generales Goals**
- Choosing interaction styles and interaction techniques - Approaches to design, implementation and evaluation of non-mouse interaction - Touch and multi-touch interfaces - Shared, embodied, and large interfaces - New input modalities (such as sensor and location data) - New Windows, e.g., iPhone, Android - Speech recognition and natural language processing - Wearable and tangible interfaces - Persuasive interaction and emotion - Ubiquitous and context-aware interaction technologies (Ubicomp) - Bayesian inference (e.g. predictive text, guided pointing) - Ambient/peripheral display and interaction - Output - Sound - Stereoscopic display - Force feedback simulation, haptic devices - System architectures - Game engines - Mobile augmented reality - Flight simulators - CAVEs - Medical imaging	- Describe when non-mouse interfaces are appropriate [Familiarity] - Understand the interaction possibilities beyond mouse-and-pointer interfaces [Familiarity] - Discuss the advantages (and disadvantages) of non-mouse interfaces [Usage] - Describe the optical model realized by a computer graphics system to synthesize stereoscopic view [Familiarity] - Describe the principles of different viewer tracking technologies [Familiarity] - Determine the basic requirements on interface, hardware, and software configurations of a VR system for a specified application [Assessment]
Readings: Dix et al. (2004), Stone et al. (2005), Rogers and Sharp (2011), Wigdor and Wixon (2011), Mathis (2011)	

UNIT 6: Collaboration and communication (8)	
Competences:	
Content	**Generales Goals**
• Asynchronous group communication, e.g., e-mail, forums, social networks • Social media, social computing, and social network analysis • Online collaboration, 'smart' spaces, and social co-ordination aspects of workflow technologies • Online communities • Software characters and intelligent agents, virtual worlds and avatars • Social psychology	• Describe the difference between synchronous and asynchronous communication [Familiarity] • Compare the HCI issues in individual interaction with group interaction [Familiarity] • Discuss several issues of social concern raised by collaborative software [Usage] • Discuss the HCI issues in software that embodies human intention [Assessment]
Readings: Dix et al. (2004), Stone et al. (2005), Rogers and Sharp (2011)	

8. Methodology
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. Assessment
Continuous Assessment 1 : 20 %   **Partial Exam** : 30 %   **Continuous Assessment 2** : 20 %   **Final exam** : 30 %

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
Undergraduate Program in Information Systems
SILABO

CS391. Software Engineering III (Mandatory)

2022-II

1. General information		
1.1 School	:	Sistemas de Información
1.2 Course	:	CS391. Software Engineering III
1.3 Semester	:	7mo Semestre.
1.4 Prerrequisites	:	CS292. Software Engineering II. (6th Sem)
1.5 Type of course	:	Mandatory
1.6 Learning modality	:	Face to face
1.7 Horas	:	2 HT; 2 HP;
1.8 Credits	:	3

2. Professors

3. Course foundation
Software development requires the use of best development practices, IT project management, equipment management And efficient and rational use of quality assurance frameworks, these elements are key and transversal during the whole productive process. The construction of software contemplates the implementation and use of processes, methods, models and tools that allow to achieve the realization of the quality attributes of a product.

4. Summary
1. Software Evolution 2. Software Project Management 3. Software Project Management 4. Software Processes 5. Estándares ISO/IEC

5. Generales Goals

- Understand and implement the fundamental concepts of project management and software equipment management.

- Understand the fundamentals of project management, including its definition, scope, and need for project management in the modern organization.

- Students have to understand the fundamental concepts of CMMI, PSP, TSP to be adopted in software projects.

- Describe and understand quality assurance models as a key framework for the success of IT projects.

6. Contribution to Outcomes

This discipline contributes to the achievement of the following outcomes:

1) Analyze a complex computing problem and to apply principles of computing and other relevant disciplines to identify solutions. (**Assessment**)

2) Design, implement and evaluate a computing-based solution to meet a given set of computing requirements in the context of the program's discipline. (**Assessment**)

3) Communicate effectively in a variety of professional contexts. (**Usage**)

5) Function effectively as a member or leader of a team engaged in activities appropriate to the program's discipline. (**Usage**)

6) Apply computer science theory and software development fundamentals to produce computing-based solutions. (**Assessment**)

7) Develop computational technology for the well-being of all, contributing with human formation, scientific, technological and professional skills to solve social problems of our community. (**Assessment**)

7. Content

UNIT 1: Software Evolution (12)

Competences:

Content	Generales Goals
- Software development in the context of large, pre-existing code bases - Software change - Concerns and concernlocation - Refactoring - Software evolution - Characteristics of maintainable software - Reengineering systems - Software reuse - Code segments - Libraries and frameworks - Components - Product lines	- Identify the principal issues associated with software evolution and explain their impact on the software lifecycle [Familiarity] - Estimate the impact of a change request to an existing product of medium size [Usage] - Use refactoring in the process of modifying a software component [Usage] - Discuss the challenges of evolving systems in a changing environment [Familiarity] - Outline the process of regression testing and its role in release management [Familiarity] - Discuss the advantages and disadvantages of different types of software reuse [Familiarity]

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

UNIT 2: Software Project Management (10)	
Competences:	
Content	**Generales Goals**

• Team participation
 	
   – Team processes including responsabilities for task, meeting structure, and work schedule
   – Roles and responsabilities in a software team
   – Team conflict resolution
   – Risks associated with virtual teams (communication, perception, structure)
 	
• Effort estimation (at the personal level)
 	
• Risk
 	
   – The role of risk in the lifecycle
   – Risk categories including security, safety, market, financial, technology, people, quality, structure and process
 	
• Team management
 	
   – Team organization and decision-making
   – Role identification and assigment
   – Individual and team performance assessment
 	
• Project management
 	
   – Scheduling and tracking
   – Project management tools
   – Cost/benefit analysis
	• Discuss common behaviors that contribute to the effective functioning of a team [Familiarity]
	
	• Create and follow an agenda for a team meeting [Usage]
	
	• Identify and justify necessary roles in a software development team [Usage]
	
	• Understand the sources, hazards, and potential benefits of team conflict [Usage]
	
	• Apply a conflict resolution strategy in a team setting [Usage]
	
	• Use an ad hoc method to estimate software development effort (eg, time) and compare to actual effort required [Usage]
	
	• List several examples of software risks [Familiarity]
	
	• Describe the impact of risk in a software development lifecycle [Familiarity]
	
	• Describe different categories of risk in software systems [Familiarity]
	
	• Demonstrate through involvement in a team project the central elements of team building and team management [Usage]
	
	• Describe how the choice of process model affects team organizational structures and decision-making processes [Familiarity]
	
	• Create a team by identifying appropriate roles and assigning roles to team members [Usage]
	
	• Assess and provide feedback to teams and individuals on their performance in a team setting [Usage]
	
	• Using a particular software process, describe the aspects of a project that need to be planned and monitored, (eg, estimates of size and effort, a schedule, resource allocation, configuration control, change management, and project risk identification and management) [Familiarity]

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

UNIT 3: Software Project Management (8)	
Competences:	
Content	**Generales Goals**

• Software measurement and estimation techniques
 	
• Software quality assurance and the role of measurements
 	
• Risk
 	
    – Risk identification and management
 	
    – Risk analysis and evaluation
 	
    – Risk tolerance (e.g., risk-adverse, risk-neutral, risk-seeking)
 	
    – Risk planning
 	
• System-wide approach to risk including hazards associated with tools
	• Track the progress of some stage in a project using appropriate project metrics [Usage]
	
	• Compare simple software size and cost estimation techniques [Usage]
	
	• Use a project management tool to assist in the assignment and tracking of tasks in a software development project [Usage]
	
	• Describe the impact of risk tolerance on the software development process [Assessment]
	
	• Identify risks and describe approaches to managing risk (avoidance, acceptance, transference, mitigation), and characterize the strengths and shortcomings of each [Familiarity]
	
	• Explain how risk affects decisions in the software development process [Usage]
	
	• Identify security risks for a software system [Usage]
	
	• Demonstrate a systematic approach to the task of identifying hazards and risks in a particular situation [Usage]
	
	• Apply the basic principles of risk management in a variety of simple scenarios including a security situation [Usage]
	
	• Conduct a cost/benefit analysis for a risk mitigation approach [Usage]
	
	• Identify and analyze some of the risks for an entire system that arise from aspects other than the software [Usage]

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

UNIT 4: Software Processes (12)	
Competences:	
Content	**Generales Goals**
- System level considerations, i.e., the iteration of software with its intended enviroment - Introduction to software process models (e.g., waterfall, incremental, agile) - Activities with software lifecycles - Programming in the large vs. individual programming - Evaluation of software process models - Software quality concepts - Process improvement - Software process capability maturity models - Software process measurements	- Describe how software can interact with and participate in various systems including information management, embedded, process control, and communications systems [Usage] - Describe the relative advantages and disadvantages among several major process models (eg, waterfall, iterative, and agile) [Usage] - Describe the different practices that are key components of various process models [Usage] - Differentiate among the phases of software development [Usage] - Describe how programming in the large differs from individual efforts with respect to understanding a large code base, code reading, understanding builds, and understanding context of changes [Usage] - Explain the concept of a software lifecycle and provide an example, illustrating its phases including the deliverables that are produced [Usage] - Compare several common process models with respect to their value for development of particular classes of software systems taking into account issues such as requirement stability, size, and non-functional characteristics [Usage] - Define software quality and describe the role of quality assurance activities in the software process [Usage] - Describe the intent and fundamental similarities among process improvement approaches [Usage] - Compare several process improvement models such as CMM, CMMI, CQI, Plan-Do-Check-Act, or ISO9000 [Usage] - Assess a development effort and recommend potential changes by participating in process improvement (using a model such as PSP) or engaging in a project retrospective [Usage] - Explain the role of process maturity models in process improvement [Usage] - Describe several process metrics for assessing and controlling a project [Usage] - Use project metrics to describe the current state of a project [Usage]
Readings: Pressman and Maxim (2015), Sommerville (2017)	

UNIT 5: Estándares ISO/IEC (6)	
Competences:	
Content	**Generales Goals**
- 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.	Learn and apply correctly standards and international standards . [Usage]
Readings: Sommerville (2017), Pressman and Maxim (2015)	

8. Methodology
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. Assessment
Continuous Assessment 1 : 20 % **Partial Exam** : 30 % **Continuous Assessment 2** : 20 % **Final exam** : 30 %

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
Undergraduate Program in Information Systems
SILABO

CS401. Methodology of Computation Research (Mandatory)

2022-II

1. General information		
1.1 School	:	Sistemas de Información
1.2 Course	:	CS401. Methodology of Computation Research
1.3 Semester	:	7mo Semestre.
1.4 Prerrequisites	:	CS212. Algorithm Analysis and Design. (5th Sem)
1.5 Type of course	:	Mandatory
1.6 Learning modality	:	Face to face
1.7 Horas	:	2 HT; 2 HP;
1.8 Credits	:	3

2. Professors

3. Course foundation
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. Summary
1.

5. Generales Goals

- 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 computacion: Researchindex, IEEE-CS[1], ACM[2].

- 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 LATEX.

- Que el alumno sea capaz de reproducir 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 LATEX.

  Final: Entendimiento del artículo del tipo survey, documento concluído 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. Contribution to Outcomes

This discipline contributes to the achievement of the following outcomes:

Nooutcomes

7. Content

UNIT 1: (60)

Competences:

Content	Generales Goals
- Búsqueda bibliográfica en computación. - Redacción de artículos técnicos en computación.	- Aprender a hacer una investigación correcta en el área de computación[Usage] - Conocer las fuentes de bibliografía adecuada para esta área[Usage] - Saber redactar un documento de acorde con las carácteristicas que las conferencias de esta área exigen[Usage]

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

8. Methodology

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. Assessment

Continuous Assessment 1 : 20 %

Partial Exam : 30 %

Continuous Assessment 2 : 20 %

Final exam : 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
Undergraduate Program in Information Systems
SILABO

CS251. Computer graphics (Elective)

2022-II

1. General information		
1.1 School	:	Sistemas de Información
1.2 Course	:	CS251. Computer graphics
1.3 Semester	:	7mo Semestre.
1.4 Prerrequisites	:	

- CS312. Advanced Data Structures . (6th Sem)

- MA307. Mathematics applied to computing . (6th Sem)

1.5 Type of course	:	Elective
1.6 Learning modality	:	Face to face
1.7 Horas	:	2 HT; 2 HP; 2 HL;
1.8 Credits	:	4

2. Professors

3. Course foundation

It offers an introduction to the area of Computer Graphics, which is an important part of Computer Science. The purpose of this course is to investigate the fundamental principles, techniques and tools for this area.

4. Summary

1. Fundamental Concepts 2. Basic Rendering 3. Programming Interactive Systems 4. Geometric Modeling 5. Advanced Rendering 6. Computer Animation

5. Generales Goals

- Bring students to concepts and techniques used in complex 3-D graphics applications.

- Give the student the necessary tools to determine which graphics software and which platform are best suited to develop a specific application.

6. Contribution to Outcomes

This discipline contributes to the achievement of the following outcomes:

1) Analyze a complex computing problem and to apply principles of computing and other relevant disciplines to identify solutions. (**Usage**)

6) Apply computer science theory and software development fundamentals to produce computing-based solutions. (**Usage**)

7. Content

UNIT 1: Fundamental Concepts (6)	
Competences:	
Content	**Generales Goals**
- Media applications including user interfaces, audio and video editing, game engines, cad, visualization, virtual reality - Tradeoffs between storing data and re-computing data as embodied by vector and raster representations of images - Additive and subtractive color models (CMYK and RGB) and why these provide a range of colors - Animation as a sequence of still images	- Explain in general terms how analog signals can be reasonably represented by discrete samples, for example, how images can be represented by pixels [Familiarity] - Describe color models and their use in graphics display devices [Familiarity] - Describe the tradeoffs between storing information vs storing enough information to reproduce the information, as in the difference between vector and raster rendering [Familiarity] - Describe the basic process of producing continuous motion from a sequence of discrete frames (sometimes called "flicker fusion") [Familiarity]
Readings: Hearn and Baker (1990)	

UNIT 2: Basic Rendering (12)	
Competences:	
Content	**Generales Goals**
- Rendering in nature, e.g., the emission and scattering of light and its relation to numerical integration - Forward and backward rendering (i.e., ray-casting and rasterization) - Basic radiometry, similar triangles, and projection model - Affine and coordinate system transformations - Ray tracing - Visibility and occlusion, including solutions to this problem such as depth buffering, Painter's algorithm, and ray tracing - Simple triangle rasterization - Rendering with a shader-based API - Application of spatial data structures to rendering - Sampling and anti-aliasing - Forward and backward rendering (i.e., ray-casting and rasterization)	- Discuss the light transport problem and its relation to numerical integration ie, light is emitted, scatters around the scene, and is measured by the eye [Familiarity] - Describe the basic graphics pipeline and how forward and backward rendering factor in this [Familiarity] - Create a program to display 3D models of simple graphics images [Usage] - Obtain 2-dimensional and 3-dimensional points by applying affine transformations [Usage] - Apply 3-dimensional coordinate system and the changes required to extend 2D transformation operations to handle transformations in 3D [Usage] - Contrast forward and backward rendering [Assessment] - Explain the concept and applications of texture mapping, sampling, and anti-aliasing [Familiarity] - Explain the ray tracing/rasterization duality for the visibility problem [Familiarity] - Implement a simple real-time renderer using a rasterization API (eg, OpenGL) using vertex buffers and shaders [Usage] - Compute space requirements based on resolution and color coding [Assessment] - Compute time requirements based on refresh rates, rasterization techniques [Assessment]
Readings: Hearn and Baker (1990), Hughes et al. (2013), Wolff (2011), Shreiner et al. (2013)	

UNIT 3: Programming Interactive Systems (2)	
Competences:	
Content	**Generales Goals**
- Event management and user interaction - Approaches to design, implementation and evaluation of non-mouse interaction - Touch and multi-touch interfaces - Shared, embodied, and large interfaces - New input modalities (such as sensor and location data) - New Windows, e.g., iPhone, Android - Speech recognition and natural language processing - Wearable and tangible interfaces - Persuasive interaction and emotion - Ubiquitous and context-aware interaction technologies (Ubicomp) - Bayesian inference (e.g. predictive text, guided pointing) - Ambient/peripheral display and interaction	Discuss the advantages (and disadvantages) of non-mouse interfaces [Assessment]
Readings: Hearn and Baker (1990)	

UNIT 4: Geometric Modeling (15)	
Competences:	
Content	**Generales Goals**
- Basic geometric operations such as intersection calculation and proximity tests - Volumes, voxels, and point-based representations - Parametric polynomial curves and surfaces - Implicit representation of curves and surfaces - Approximation techniques such as polynomial curves, Bezier curves, spline curves and surfaces, and nonuniform rational basis (NURB) spines, and level set method - Surface representation techniques including tessellation, mesh representation, mesh fairing, and mesh generation techniques such as Delaunay triangulation, marching cubes - Spatial subdivision techniques - Procedural models such as fractals, generative modeling, and L-systems - Elastically deformable and freeform deformable models - Subdivision surfaces - Multiresolution modeling - Reconstruction - Constructive Solid Geometry (CSG) representation	- Represent curves and surfaces using both implicit and parametric forms [Usage] - Create simple polyhedral models by surface tessellation [Usage] - Generate a mesh representation from an implicit surface [Usage] - Generate a mesh from data points acquired with a laser scanner [Usage] - Construct CSG models from simple primitives, such as cubes and quadric surfaces [Usage] - Contrast modeling approaches with respect to space and time complexity and quality of image [Assessment]
Readings: Hearn and Baker (1990), Shreiner et al. (2013)	

UNIT 5: Advanced Rendering (6)	
Competences:	
Content	**Generales Goals**
- Time (motion blur), lens position (focus), and continuous frequency (color) and their impact on rendering - Shadow mapping - Occlusion culling - Subsurface scattering - Non-photorealistic rendering - GPU architecture - Human visual systems including adaptation to light, sensitivity to noise, and flicker fusion	- Demonstrate how an algorithm estimates a solution to the rendering equation [Assessment] - Prove the properties of a rendering algorithm, eg, complete, consistent, and unbiased [Assessment] - Implement a non-trivial shading algorithm (eg, toon shading, cascaded shadow maps) under a rasterization API [Usage] - Discuss how a particular artistic technique might be implemented in a renderer [Familiarity] - Explain how to recognize the graphics techniques used to create a particular image [Familiarity]
Readings: Hearn and Baker (1990), Hughes et al. (2013), Wolff (2011), Shreiner et al. (2013)	

5

UNIT 6: Computer Animation (4)	
Competences:	
Content	**Generales Goals**
• Forward and inverse kinematics • Collision detection and response • Procedural animation using noise, rules (boids/crowds), and particle systems • Skinning algorithms • Physics based motions including rigid body dynamics, physical particle systems, mass-spring networks for cloth and flesh and hair • Key-frame animation • Splines • Data structures for rotations, such as quaternions • Camera animation • Motion capture	• Compute the location and orientation of model parts using an forward kinematic approach [Usage] • Implement the spline interpolation method for producing in-between positions and orientations [Usage] • Implement algorithms for physical modeling of particle dynamics using simple Newtonian mechanics, for example Witkin & Kass, snakes and worms, symplectic Euler, Stormer/Verlet, or midpoint Euler methods [Usage] • Discuss the basic ideas behind some methods for fluid dynamics for modeling ballistic trajectories, for example for splashes, dust, fire, or smoke [Familiarity] • Use common animation software to construct simple organic forms using metaball and skeleton [Usage]
Readings: Hearn and Baker (1990), Shreiner et al. (2013)	

8. Methodology
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. Assessment
Continuous Assessment 1 : 20 % **Partial Exam** : 30 % **Continuous Assessment 2** : 20 % **Final exam** : 30 %

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.

Shreiner, Dave et al. (2013). *OpenGL, Programming Guide, Eighth Edition*. Addison-Wesley.

Wolff, David (2011). *OpenGL 4.0 Shading Language Cookbook*. Packt Publishing.

Universidad Nacional de Colombia (UNAL) Sede Manizales
Undergraduate Program in Information Systems
SILABO

FG350. Leadership (Mandatory)

2022-II

1. General information		
1.1 School	:	Sistemas de Información
1.2 Course	:	FG350. Leadership
1.3 Semester	:	7mo Semestre.
1.4 Prerrequisites	:	FG203. Public Speaking. (6th Sem)
1.5 Type of course	:	Mandatory
1.6 Learning modality	:	Face to face
1.7 Horas	:	1 HT; 2 HP;
1.8 Credits	:	2

2. Professors

3. Course foundation
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. Summary
1. 2. 3.

5. Generales Goals
• 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. Contribution to Outcomes
This discipline contributes to the achievement of the following outcomes:

3) Communicate effectively in a variety of professional contexts. (**Usage**)

4) Recognize professional responsabilities and make informed judgments in computing practice based on legal and ethical principles. (**Usage**)

5) Function effectively as a member or leader of a team engaged in activities appropriate to the program's discipline. (**Usage**)

7) Develop computational technology for the well-being of all, contributing with human formation, scientific, technological and professional skills to solve social problems of our community. (**Usage**)

7. Content

UNIT 1: (15)
Competences:

Content	Generales Goals
- 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.	- Analizar y comprender las bases teóricas del ejercicio de Liderazgo.[Familiarity] - En base a lo comprendido, asumir la actitud correcta para llevarlo a la práctica.[Familiarity] - Iniciar un proceso de autoconocimiento orientado a descubrir rasgos de liderazgo en sí mismo.[Familiarity]

Readings: Pilar (2002), Manuel. (2009), Alexandre. (2009), D' Souza (n.d.), Alfred. (2010)

UNIT 2: (15)
Competences:

Content	Generales Goals
- Teoría de las Competencias - Reconocimiento de Competencias - Plan de Desarrollo - Modelos Mentales - Necesidades Emocionales - Perfiles Emocionales - Vicios Motivacionales	- Conocer y Desarrollar competencias de Liderazgo, centradas en lograr la eficacia, sin dejar de lado el deber de servicio con los demás.[Familiarity] - Reconocer las tendencias personales y grupales necesarias para el ejercicio de Liderazgo.[Familiarity]

Readings: Wilkinson. (2009), Luis. (2008), Pilar (2002), Maruja. (2007)

UNIT 3: (18)
Competences:

Content	Generales Goals
- La relación personal con el equipo - Liderazgo integral - Acompañamiento y discipulado - Fundamentos de unidad	Desarrollar habilidades para el trabajo en equipo[Familiarity]

Readings: Goleman (2012), **CardonaP**, **Hersey**, Hunsaker (2010), Hawkins (2012), **Ginebra**

8. Methodology

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. Assessment

Continuous Assessment 1 : 20 %

Partial Exam : 30 %

Continuous Assessment 2 : 20 %

Final exam : 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
Undergraduate Program in Information Systems
SILABO

CS281. Computing in Society (Mandatory)

2022-II

1. General information		
1.1 School	:	Sistemas de Información
1.2 Course	:	CS281. Computing in Society
1.3 Semester	:	8vo Semestre.
1.4 Prerrequisites	:	None
1.5 Type of course	:	Mandatory
1.6 Learning modality	:	Face to face
1.7 Horas	:	2 HT;
1.8 Credits	:	2

2. Professors

3. Course foundation
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. Summary
1. History 2. Social Context 3. Analytical Tools 4. Professional Ethics 5. Intellectual Property 6. Privacy and Civil Liberties 7. Security Policies, Laws and Computer Crimes 8. Economies of Computing

5. Generales Goals
• 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. Contribution to Outcomes
This discipline contributes to the achievement of the following outcomes:

3) Communicate effectively in a variety of professional contexts. (**Familiarity**)

4) Recognize professional responsabilities and make informed judgments in computing practice based on legal and ethical principles. (**Usage**)

6) Apply computer science theory and software development fundamentals to produce computing-based solutions. (**Usage**)

7) Develop computational technology for the well-being of all, contributing with human formation, scientific, technological and professional skills to solve social problems of our community. (**Usage**)

7. Content

UNIT 1: History (2)

Competences:

Content	Generales Goals
- Prehistory, the world before 1946 - History of computer hardware, software, networking - Pioneers of computing - History of the Internet	- Identify significant continuing trends in the history of the computing field [Familiarity] - Identify the contributions of several pioneers in the computing field [Familiarity] - Discuss the historical context for several programming language paradigms [Familiarity] - Compare daily life before and after the advent of personal computers and the Internet [Familiarity]

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

UNIT 2: Social Context (4)	
Competences:	
Content	**Generales Goals**
• Social implications of computing in a networked world • Impact of social media on individualism, collectivism and culture • Growth and control of the Internet • Often referred to as the digital divide, differences in access to digital technology resources and its resulting ramifications for gender, class, ethnicity, geography, and/or underdeveloped countries • Accessibility issues, including legal requirements • Context-aware computing	• Describe positive and negative ways in which computer technology (networks, mobile computing, cloud computing) alters modes of social interaction at the personal level [Familiarity] • Identify developers' assumptions and values embedded in hardware and software design, especially as they pertain to usability for diverse populations including under-represented populations and the disabled [Usage] • Interpret the social context of a given design and its implementation [Assessment] • Evaluate the efficacy of a given design and implementation using empirical data [Familiarity] • Summarize the implications of social media on individualism versus collectivism and culture [Familiarity] • Discuss how Internet access serves as a liberating force for people living under oppressive forms of government; explain how limits on Internet access are used as tools of political and social repression [Familiarity] • Analyze the pros and cons of reliance on computing in the implementation of democracy (eg delivery of social services, electronic voting) [Familiarity] • Describe the impact of the under-representation of diverse populations in the computing profession (eg, industry culture, product diversity) [Usage] • Explain the implications of context awareness in ubiquitous computing systems [Familiarity]
Readings: K. C. Laudon and J. P. Laudon (2004), McLeod Jr (2000)	

UNIT 3: Analytical Tools (2)	
Competences:	
Content	**Generales Goals**
• Ethical argumentation • Ethical theories and decision-making • Moral assumptions and values	• Evaluate stakeholder positions in a given situation [Familiarity] • Analyze basic logical fallacies in an argument [Usage] • Analyze an argument to identify premises and conclusion [Familiarity] • Illustrate the use of example and analogy in ethical argument [Familiarity] • Evaluate ethical/social tradeoffs in technical decisions [Familiarity]
Readings: K. C. Laudon and J. P. Laudon (2004), McLeod Jr (2000)	

UNIT 4: Professional Ethics (4)	
Competences:	
Content	**Generales Goals**
- Community values and the laws by which we live - The nature of professionalism including care, attention and discipline, fiduciary responsibility, andmentoring - Keeping up-to-date as a computing professional in terms of familiarity, tools, skills, legal and professional framework as well as the ability to self-assess and progress in the computing field - Professional certification, codes of ethics, conduct, and practice, such as the ACM/IEEE-CS, SE, AITP, IFIP and international societies - Accountability, responsibility and liability (e.g. software correctness, reliability and safety, as well as ethical confidentiality of cybersecurity professionals) - The role of the computing professional in public policy - Maintaining awareness of consequences - Ethical dissent and whistle-blowing - The relationship between regional culture and ethical dilemmas - Dealing with harassment and discrimination - Forms of professional credentialing - Acceptable use policies for computing in the workplace - Ergonomics and healthy computing environments - Time to market and cost considerations versus quality professional standards	- Identify ethical issues that arise in software development and determine how to address them technically and ethically [Usage] - Explain the ethical responsibility of ensuring software correctness, reliability and safety. [Assessment] - Describe the mechanisms that typically exist for a professional to keep up-to-date [Familiarity] - Describe the strengths and weaknesses of relevant professional codes as expressions of professionalism and guides to decision-making [Familiarity] - Analyze a global computing issue, observing the role of professionals and government officials in managing this problem [Familiarity] - Evaluate the professional codes of ethics from the ACM, the IEEE Computer Society, and other organizations [Familiarity] - Describe ways in which professionals may contribute to public policy [Familiarity] - Describe the consequences of inappropriate professional behavior [Usage] - Identify progressive stages in a whistle-blowing incident [Usage] - Identify examples of how regional culture interplays with ethical dilemmas [Familiarity] - Investigate forms of harassment and discrimination and avenues of assistance [Usage] - Examine various forms of professional credentialing [Usage] - Explain the relationship between ergonomics in computing environments and people's health [Usage] - Develop a computer usage/acceptable use policy with enforcement measures [Familiarity] - Describe issues associated with industries' push to focus on time to market versus enforcing quality professional standards [Usage]
Readings: K. C. Laudon and J. P. Laudon (2004), McLeod Jr (2000), Ediciones (2009a), Ediciones (2009b), Ediciones (2010)	

UNIT 5: Intellectual Property (4)	
Competences:	
Content	**Generales Goals**
• Philosophical foundations of intellectual property • Intellectual property rights (cross-reference IM/Information Storage and Retrieval/intellectual property and protection) • Intangible digital intellectual property (IDIP) • Legal foundations for intellectual property protection • Digital rights management • Copyrights, patents, trade secrets, trademarks • Plagiarism • Foundations of the open source movement • Software piracy	• Discuss the philosophical bases of intellectual property [Assessment] • Discuss the rationale for the legal protection of intellectual property [Familiarity] • Describe legislation aimed at digital copyright infringements [Assessment] • Critique legislation aimed at digital copyright infringements [Familiarity] • Identify contemporary examples of intangible digital intellectual property [Assessment] • Justify uses of copyrighted materials [Assessment] [Familiarity] • Evaluate the ethical issues inherent in various plagiarism detection mechanisms [Familiarity] • Interpret the intent and implementation of software licensing [Familiarity] • Discuss the issues involved in securing software patents [Familiarity] • Characterize and contrast the concepts of copyright, patenting and trademarks [Familiarity] • Identify the goals of the open source movement [Assessment] • Identify the global nature of software piracy [Familiarity]
Readings: K. C. Laudon and J. P. Laudon (2004), McLeod Jr (2000), Ediciones (2009a), Ediciones (2009b), Ediciones (2010)	

UNIT 6: Privacy and Civil Liberties (4)	
Competences:	
Content	**Generales Goals**
- Philosophical foundations of privacy rights - Legal foundations of privacy protection - Privacy implications of widespread data collection for transactional databases, data warehouses, surveillance systems, and cloud computing - Ramifications of differential privacy - Technology-based solutions for privacy protection - Privacy legislation in areas of practice - Civil liberties and cultural differences - Freedom of expression and its limitations	- Discuss the philosophical basis for the legal protection of personal privacy [Familiarity] - Evaluate solutions to privacy threats in transactional databases and data warehouses [Familiarity] - Describe the role of data collection in the implementation of pervasive surveillance systems (e.g., RFID, face recognition, toll collection, mobile computing). [Familiarity] - Describe the ramifications of differential privacy. [Familiarity] - Investigate the impact of technological solutions to privacy problems [Familiarity] - Critique the intent, potential value and implementation of various forms of privacy legislation [Familiarity] - Identify strategies to enable appropriate freedom of expression [Familiarity]
Readings: K. C. Laudon and J. P. Laudon (2004), McLeod Jr (2000), Ediciones (2009a), Ediciones (2009b), Ediciones (2010)	

UNIT 7: Security Policies, Laws and Computer Crimes (2)	
Competences:	
Content	**Generales Goals**
• Examples of computer crimes and legal redress for computer criminals • Social engineering, identity theft and recovery • Issues surrounding the misuse of access and breaches in security • Motivations and ramifications of cyber terrorism and criminal hacking, "cracking" • Effects of malware, such as viruses, worms and Trojan horses • Crime prevention strategies • Security policies	• List classic examples of computer crimes and social engineering incidents with societal impact [Familiarity] • Identify laws that apply to computer crimes [Familiarity] • Describe the motivation and ramifications of cyber terrorism and criminal hacking [Familiarity] • Examine the ethical and legal issues surrounding the misuse of access and various breaches in security [Familiarity] • Discuss the professional's role in security and the trade-offs involved [Familiarity] • Investigate measures that can be taken by both individuals and organizations including governments to prevent or mitigate the undesirable effects of computer crimes and identity theft [Familiarity] • Write a company-wide security policy, which includes procedures for managing passwords and employee monitoring [Familiarity]
Readings: K. C. Laudon and J. P. Laudon (2004), McLeod Jr (2000), Ediciones (2009a), Ediciones (2009b), Ediciones (2010)	

UNIT 8: Economies of Computing (2)	
Competences:	
Content	**Generales Goals**
- Monopolies and their economic implications - Effect of skilled labor supply and demand on the quality of computing products - Pricing strategies in the computing domain - The phenomenon of outsourcing and off-shoring software development; impacts on employment and on economics - Consequences of globalization for the computer science profession - Differences in access to computing resources and the possible effects thereof - Cost/benefit analysis of jobs with considerations to manufacturing, hardware, software, and engineering implications - Cost estimates versus actual costs in relation to total costs - Entrepreneurship: prospects and pitfalls - Network effect or demand-side economies of scale - Use of engineering economics in dealing with finances	- Summarize the rationale for antimonopoly efforts [Familiarity] - Identify several ways in which the information technology industry is affected by shortages in the labor supply [Familiarity] - Identify the evolution of pricing strategies for computing goods and services [Familiarity] - Discuss the benefits, the drawbacks and the implications of off-shoring and outsourcing [Familiarity] - Investigate and defend ways to address limitations on access to computing [Usage] - Describe the economic benefits of network effects [Usage]
Readings: K. C. Laudon and J. P. Laudon (2004), McLeod Jr (2000), Ediciones (2009a), Ediciones (2009b), Ediciones (2010)	

8. Methodology
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. Assessment
Continuous Assessment 1 : 20 %   **Partial Exam** : 30 %   **Continuous Assessment 2** : 20 %   **Final exam** : 30 %

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
Undergraduate Program in Information Systems
SILABO

CS3I1. Computer Security (Mandatory)

2022-II

1. General information		
1.1 School	:	Sistemas de Información
1.2 Course	:	CS3I1. Computer Security
1.3 Semester	:	8vo Semestre.
1.4 Prerrequisites	:	CS231. Networking and Communication. (7th Sem)
1.5 Type of course	:	Mandatory
1.6 Learning modality	:	Face to face
1.7 Horas	:	1 HT; 2 HP; 2 HL;
1.8 Credits	:	3

2. Professors

3. Course foundation
Nowadays, information is one of the most valuable assets in any organization. This course is oriented to be able to provide the student with the security elements oriented to protect the Information of the organization and mainly to be able to foresee the possible problems related to this heading. This subject involves the development of a preventive attitude on the part of the student in all areas related to software development.

4. Summary
1. Foundational Concepts in Security 2. Principles of Secure Design 3. Defensive Programming 4. Threats and Attacks 5. Network Security 6. Cryptography 7. Web Security 8. Platform Security 9. Digital Forensics 10. Secure Software Engineering

5. Generales Goals
- Discuss at an intermediate intermediate level the fundamentals of Computer Security.   - Provide different aspects of the malicious code.   - That the student knows the concepts of cryptography and security in computer networks.   - Discuss and analyze together with the student the aspects of Internet Security.

6. Contribution to Outcomes

This discipline contributes to the achievement of the following outcomes:

1) Analyze a complex computing problem and to apply principles of computing and other relevant disciplines to identify solutions. (**Assessment**)

2) Design, implement and evaluate a computing-based solution to meet a given set of computing requirements in the context of the program's discipline. (**Assessment**)

5) Function effectively as a member or leader of a team engaged in activities appropriate to the program's discipline. (**Usage**)

6) Apply computer science theory and software development fundamentals to produce computing-based solutions. (**Assessment**)

7) Develop computational technology for the well-being of all, contributing with human formation, scientific, technological and professional skills to solve social problems of our community. (**Assessment**)

7. Content

UNIT 1: Foundational Concepts in Security (25)	
Competences:	
Content	**Generales Goals**
- CIA (Confidentiality, Integrity, Availability) - Concepts of risk, threats, vulnerabilities, and attack vectors - Authentication and authorization, access control (mandatory vs. discretionary) - Concept of trust and trustworthiness - Ethics (responsible disclosure)	- Analyze the tradeoffs of balancing key security properties (Confidentiality, Integrity, Availability) [Familiarity] - Describe the concepts of risk, threats, vulnerabilities and attack vectors (including the fact that there is no such thing as perfect security) [Familiarity] - Explain the concepts of authentication, authorization, access control [Familiarity] - Explain the concept of trust and trustworthiness [Familiarity] - Recognize that there are important ethical issues to consider in computer security, including ethical issues associated with fixing or not fixing vulnerabilities and disclosing or not disclosing vulnerabilities [Familiarity]
Readings: W and L (2014)	

UNIT 2: Principles of Secure Design (25)	
Competences:	
Content	**Generales Goals**
- Least privilege and isolation - Fail-safe defaults - Open design - End-to-end security - Defense in depth (e.g., defensive programming, layered defense) - Security by design - Tensions between security and other design goals - Complete mediation - Use of vetted security components - Economy of mechanism (reducing trusted computing base, minimize attack surface) - Usable security - Security composability - Prevention, detection, and deterrence	- Describe the principle of least privilege and isolation as applied to system design [Familiarity] - Summarize the principle of fail-safe and deny-by-default [Familiarity] - Discuss the implications of relying on open design or the secrecy of design for security. [Familiarity] - Explain the goals of end-to-end data security [Familiarity] - Discuss the benefits of having multiple layers of defenses [Familiarity] - For each stage in the lifecycle of a product, describe what security considerations should be evaluated. [Familiarity] - Describe the cost and tradeoffs associated with designing security into a product [Familiarity] - Describe the concept of mediation and the principle of complete mediation [Familiarity] - Be aware of standard components for security operations, instead of re-inventing fundamentals operations [Familiarity] - Explain the concept of trusted computing including trusted computing base and attack surface and the principle of minimizing trusted computing base [Familiarity] - Discuss the importance of usability in security mechanism design [Familiarity] - Describe security issues that arise at boundaries between multiple components. [Familiarity] - Identify the different roles of prevention mechanisms and detection/deterrence mechanisms [Familiarity]
Readings: W and L (2014)	

UNIT 3: Defensive Programming (25)	
Competences:	
Content	**Generales Goals**
- Input validation and data sanitization - Choice of programming language and type-safe languages - Examples of input validation and data sanitization errors - Buffer overflows - Integer errors - SQL injection - XSS vulnerability - Race conditions - Correct handling of exceptions and unexpected behaviors - Correct usage of third-party components - Effectively deploying security updates - Information flow control - Correctly generating randomness for security purposes - Mechanisms for detecting and mitigating input and data sanitization errors - Fuzzing - Static analysis and dynamic analysis - Program verification - Operating system support (e.g., address space randomization, canaries) - Hardware support (e.g, DEP, TPM)	- Explain why input validation and data sanitization is necessary in the face of adversarial control of the input channel. [Usage] - Explain why you might choose to develop a program in a type-safe language like Java, in contrast to an unsafe programming language like C/C++ [Usage] - Classify common input validation errors, and write correct input validation code [Usage] - Demonstrate using a high-level programming language how to prevent a race condition from occurring and how to handle an exception [Usage] - Demonstrate the identification and graceful handling of error conditions [Familiarity] - Explain the risks with misusing interfaces with third-party code and how to correctly use third-party code [Familiarity] - Discuss the need to update software to fix security vulnerabilities and the lifecycle management of the fix [Familiarity]
Readings: W and L (2014)	

UNIT 4: Threats and Attacks (25)	
Competences:	
Content	**Generales Goals**
- Attacker goals, capabilities, and motivations (such as underground economy, digital espionage, cyberwarfare, insider threats, hacktivism, advanced persistent threats) - Examples of malware (e.g., viruses, worms, spyware, botnets, Trojan horses or rootkits) - Denial of Service (DoS) and Distributed Denial of Service (DDoS) - Social engineering (e.g., phishing) - Attacks on privacy and anonymity - Malware/unwanted communication such as covert channels and steganography	- Describe likely attacker types against a particular system [Familiarity] - Discuss the limitations of malware countermeasures (eg, signature-based detection, behavioral detection) [Familiarity] - Identify instances of social engineering attacks and Denial of Service attacks [Familiarity] - Discuss how Denial of Service attacks can be identified and mitigated [Familiarity] - Describe risks to privacy and anonymity in commonly used applications [Familiarity] - Discuss the concepts of covert channels and other data leakage procedures [Familiarity]
Readings: W and L (2014)	

UNIT 5: Network Security (25)	
Competences:	
Content	**Generales Goals**
- Network specific threats and attack types (e.g., denial of service, spoofing, sniffing and traffic redirection, man-in-the-middle, message integrity attacks, routing attacks, and traffic analysis) - Use of cryptography for data and network security - Architectures for secure networks (e.g., secure channels, secure routing protocols, secure DNS, VPNs, anonymous communication protocols, isolation) - Defense mechanisms and countermeasures (e.g., network monitoring, intrusion detection, firewalls, spoofing and DoS protection, honeypots, tracebacks) - Security for wireless, cellular networks - Other non-wired networks (e.g., ad hoc, sensor, and vehicular networks) - Censorship resistance - Operational network security management (e.g., configure network access control)	- Describe the different categories of network threats and attacks [Familiarity] - Describe the architecture for public and private key cryptography and how PKI supports network security [Familiarity] - Describe virtues and limitations of security technologies at each layer of the network stack [Familiarity] - Identify the appropriate defense mechanism(s) and its limitations given a network threat [Usage]
Readings: W and L (2014)	

UNIT 6: Cryptography (25)	
Competences:	
Content	**Generales Goals**

Content

- Basic Cryptography Terminology covering notions pertaining to the different (communication) partners, secure/unsecure channel, attackers and their capabilities, encryption, decryption, keys and their characteristics, signatures

- Cipher types (e.g., Caesar cipher, affine cipher) together with typical attack methods such as frequency analysis

- Public Key Infrastructure support for digital signature and encryption and its challenges

- Symmetric key cryptography

  - Perfect secrecy and the one time pad

  - Modes of operation for semantic security and authenticated encryption (e.g., encrypt-then-MAC, OCB, GCM)

  - Message integrity (e.g., CMAC, HMAC)

- Public key cryptography:

  - Trapdoor permutation, e.g., RSA

  - Public key encryption, e.g., RSA encryption, El Gamal encryption

  - Digital signatures

  - Public-key infrastructure (PKI) and certificates

  - Hardness assumptions, e.g., Diffie-Hellman, integer factoring

- Authenticated key exchange protocols, e.g., TLS

- Cryptographic primitives:

  - pseudo-random generators and stream ciphers

  - block ciphers (pseudo-random permutations), e.g., AES

  - pseudo-random functions

  - hash functions, e.g., SHA2, collision resistance

  - message authentication codes

  - key derivations functions

Generales Goals

- Describe the purpose of Cryptography and list ways it is used in data communications [Familiarity]

- Define the following terms: Cipher, Cryptanalysis, Cryptographic Algorithm, and Cryptology and describe the two basic methods (ciphers) for transforming plain text in cipher text [Familiarity]

- Discuss the importance of prime numbers in cryptography and explain their use in cryptographic algorithms [Familiarity]

- Illustrate how to measure entropy and how to generate cryptographic randomness [Usage]

- Use public-key primitives and their applications [Usage]

- Explain how key exchange protocols work and how they fail [Familiarity]

- Discuss cryptographic protocols and their properties [Familiarity]

Readings: W and L (2014)

UNIT 7: Web Security (25)	
Competences:	
Content	**Generales Goals**

• Web security model
  – Browser security model including same-origin policy
  – Client-server trust boundaries, e.g., cannot rely on secure execution in the client
 	
• Session management, authentication
  – Single sign-on
  – HTTPS and certificates
 	
• Application vulnerabilities and defenses
  – SQL injection
  – XSS
  – CSRF
 	
• Client-side security
  – Cookies security policy
  – HTTP security extensions, e.g. HSTS
  – Plugins, extensions, and web apps
  – Web user tracking
 	
• Server-side security tools, e.g. Web Application Firewalls (WAFs) and fuzzers
	
	• Describe the browser security model including same-origin policy and threat models in web security [Familiarity]
	
	• Discuss the concept of web sessions, secure communication channels such as TLS and importance of secure certificates, authentication including single sign-on such as OAuth and SAML [Familiarity]
	
	• Investigate common types of vulnerabilities and attacks in web applications, and defenses against them [Familiarity]
	
	• Use client-side security capabilities [Usage]

Readings: W and L (2014)	

UNIT 8: Platform Security (25)	
Competences:	
Content	**Generales Goals**
- Code integrity and code signing - Secure boot, measured boot, and root of trust - Attestation - TPM and secure co-processors - Security threats from peripherals, e.g., DMA, IOMMU - Physical attacks: hardware Trojans, memory probes, cold boot attacks - Security of embedded devices, e.g., medical devices, cars - Trusted path	- Explain the concept of code integrity and code signing and the scope it applies to [Familiarity] - Discuss the concept of root of trust and the process of secure boot and secure loading [Familiarity] - Describe the mechanism of remote attestation of system integrity [Familiarity] - Summarize the goals and key primitives of TPM [Familiarity] - Identify the threats of plugging peripherals into a device [Familiarity] - Identify physical attacks and countermeasures [Familiarity] - Identify attacks on non-PC hardware platforms [Familiarity] - Discuss the concept and importance of trusted path [Familiarity]
Readings: W and L (2014)	

UNIT 9: Digital Forensics (25)	
Competences:	
Content	**Generales Goals**
- Basic Principles and methodologies for digital forensics - Design systems with forensic needs in mind - Rules of Evidence - general concepts and differences between jurisdictions and Chain of Custody - Search and Seizure of evidence: legal and procedural requirements - Digital Evidence methods and standards - Techniques and standards for Preservation of Data - Legal and Reporting Issues including working as an expert witness - OS/File System Forensics - Application Forensics - Web Forensics - Network Forensics - Mobile Device Forensics - Computer/network/system attacks - Attack detection and investigation - Anti-forensics	- Describe what is a Digital Investigation is, the sources of digital evidence, and the limitations of forensics [Familiarity] - Explain how to design software to support forensics [Familiarity] - Describe the legal requirements for use of seized data [Familiarity] - Describe the process of evidence seizure from the time when the requirement was identified to the disposition of the data [Familiarity] - Describe how data collection is accomplished and the proper storage of the original and forensics copy [Familiarity] - Conduct data collection on a hard drive [Usage] - Describe a person's responsibility and liability while testifying as a forensics examiner [Familiarity] - Recover data based on a given search term from an imaged system [Usage] - Reconstruct application history from application artifacts [Familiarity] - Reconstruct web browsing history from web artifacts [Familiarity] - Capture and interpret network traffic [Familiarity] - Discuss the challenges associated with mobile device forensics [Familiarity]
Readings: W and L (2014)	

UNIT 10: Secure Software Engineering (25)	
Competences:	
Content	**Generales Goals**
- Building security into the software development life-cycle - Secure design principles and patterns - Secure software specifications and requirements - Secure software development practices - Secure testing- the process of testing that security requirements are met (including static and dynamic analysis).	- Describe the requirements for integrating security into the SDL [Familiarity] - Apply the concepts of the Design Principles for Protection Mechanisms, the Principles for Software Security (Viega and McGraw), and the Principles for Secure Design (Morrie Gasser) on a software development project [Familiarity] - Develop specifications for a software development effort that fully specify functional requirements and identifies the expected execution paths [Familiarity]
Readings: W and L (2014)	

8. Methodology

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. Assessment

Continuous Assessment 1 : 20 %

Partial Exam : 30 %

Continuous Assessment 2 : 20 %

Final exam : 30 %

References

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

Universidad Nacional de Colombia (UNAL) Sede Manizales
Undergraduate Program in Information Systems
SILABO

CS3P1. Parallel and Distributed Computing (Mandatory)

2022-II

1. General information		
1.1 School	:	Sistemas de Información
1.2 Course	:	CS3P1. Parallel and Distributed Computing
1.3 Semester	:	8vo Semestre.
1.4 Prerrequisites	:	

- CS212. Algorithm Analysis and Design. (5th Sem)
- CS231. Networking and Communication. (7th Sem)

1.5 Type of course	:	Mandatory
1.6 Learning modality	:	Face to face
1.7 Horas	:	2 HT; 2 HP; 2 HL;
1.8 Credits	:	4

2. Professors

3. Course foundation

The last decade has brought explosive growth in computing with multiprocessors, including Multi-core processors and distributed data centers. As a result, computing parallel and distributed has become a widely elective subject to be one of the main components in the mesh studies in computer science undergraduate. Both parallel and distributed computing the simultaneous execution of multiple processes, whose operations have the potential to intercalar in a complex way. Parallel and distributed computing builds on foundations in many areas, including understanding the fundamental concepts of systems, such as: concurrency and parallel execution, consistency in state / memory manipulation, and latency. The communication and coordination between processes has its foundations in the passage of messages and models of shared memory of computing and algorithmic concepts like atomicity, consensus and conditional waiting.Achieving acceleration in practice requires an understanding of parallel algorithms, strategies for decomposition problem, systems architecture, implementation strategies and analysis of performance. Distributed systems highlight the problems of security and tolerance to Failures, emphasize the maintenance of the replicated state and introduce additional problems in the field of computer networks.

4. Summary

1. Parallelism Fundamentals 2. Parallel Architecture 3. Parallel Decomposition 4. Communication and Coordination 5. Parallel Algorithms, Analysis, and Programming 6. Parallel Performance

5. Generales Goals

- That the student is able to create parallel applications of medium complexity by efficiently leveraging machines with multiple cores.

- That the student is able to compare sequential and parallel applications.

- That the student is able to convert, when the situation warrants, sequential applications to parallel efficiently

6. Contribution to Outcomes

This discipline contributes to the achievement of the following outcomes:

1) Analyze a complex computing problem and to apply principles of computing and other relevant disciplines to identify solutions. (**Usage**)

6) Apply computer science theory and software development fundamentals to produce computing-based solutions. (**Usage**)

7. Content

UNIT 1: Parallelism Fundamentals (18)	
Competences:	
Content	**Generales Goals**
- Multiple simultaneous computations - Goals of parallelism (e.g., throughput) versus concurrency (e.g., controlling access to shared resources) - Parallelism, communication, and coordination - Parallelism, communication, and coordination - Need for synchronization - Programming errors not found in sequential programming - Data races (simultaneous read/write or write/write of shared state) - Higher-level races (interleavings violating program intention, undesired non-determinism) - Lack of liveness/progress (deadlock, starvation)	- Distinguish using computational resources for a faster answer from managing efficient access to a shared resource [Familiarity] - Distinguish multiple sufficient programming constructs for synchronization that may be inter-implementable but have complementary advantages [Familiarity] - Distinguish data races from higher level races [Familiarity]
Readings: Pacheco (2011), Matloff (2014), **quinnz**, Georg Hager (2010)	

UNIT 2: Parallel Architecture (12)	
Competences:	
Content	**Generales Goals**
- Multicore processors - Shared vs distributed memory - Symmetric multiprocessing (SMP) - SIMD, vector processing - GPU, co-processing - Flynn's taxonomy - Instruction level support for parallel programming - Atomic instructions such as Compare and Set - Memory issues - Multiprocessor caches and cache coherence - Non-uniform memory access (NUMA) - Topologies - Interconnects - Clusters - Resource sharing (e.g., buses and interconnects)	- Explain the differences between shared and distributed memory [Assessment] - Describe the SMP architecture and note its key features [Assessment] - Characterize the kinds of tasks that are a natural match for SIMD machines [Usage] - Describe the advantages and limitations of GPUs vs CPUs [Usage] - Explain the features of each classification in Flynn's taxonomy [Usage] - Describe the challenges in maintaining cache coherence [Familiarity] - Describe the key performance challenges in different memory and distributed system topologies [Familiarity]
Readings: Pacheco (2011), Kirk and Hwu (2013), Sanders and Kandrot (2010), Georg Hager (2010)	

UNIT 3: Parallel Decomposition (18)	
Competences:	
Content	**Generales Goals**
- Need for communication and coordination/synchronization - Independence and partitioning - Basic knowledge of parallel decomposition concept - Task-based decomposition - Implementation strategies such as threads - Data-parallel decomposition - Strategies such as SIMD and MapReduce - Actors and reactive processes (e.g., request handlers)	- Explain why synchronization is necessary in a specific parallel program [Usage] - Identify opportunities to partition a serial program into independent parallel modules [Familiarity] - Write a correct and scalable parallel algorithm [Usage] - Parallelize an algorithm by applying task-based decomposition [Usage] - Parallelize an algorithm by applying data-parallel decomposition [Usage] - Write a program using actors and/or reactive processes [Usage]
Readings: Pacheco (2011), Matloff (2014), Quinn (2003), Georg Hager (2010)	

UNIT 4: Communication and Coordination (18)	
Competences:	
Content	**Generales Goals**
- Shared Memory - Consistency, and its role in programming language guarantees for data-race-free programs - Message passing - Point-to-point versus multicast (or event-based) messages - Blocking versus non-blocking styles for sending and receiving messages - Message buffering (cross-reference PF/Fundamental Data Structures/Queues) - Atomicity - Specifying and testing atomicity and safety requirements - Granularity of atomic accesses and updates, and the use of constructs such as critical sections or transactions to describe them - Mutual Exclusion using locks, semaphores, monitors, or related constructs * Potential for liveness failures and deadlock (causes, conditions, prevention) - Composition * Composing larger granularity atomic actions using synchronization * Transactions, including optimistic and conservative approaches - Consensus (Cyclic) barriers, counters, or related constructs - Conditional actions - Conditional waiting (e.g., using condition variables)	- Use mutual exclusion to avoid a given race condition [Usage] - Give an example of an ordering of accesses among concurrent activities (eg, program with a data race) that is not sequentially consistent [Familiarity] - Give an example of a scenario in which blocking message sends can deadlock [Usage] - Explain when and why multicast or event-based messaging can be preferable to alternatives [Familiarity] - Write a program that correctly terminates when all of a set of concurrent tasks have completed [Usage] - Give an example of a scenario in which an attempted optimistic update may never complete [Familiarity] - Use semaphores or condition variables to block threads until a necessary precondition holds [Usage]
Readings: Pacheco (2011), Matloff (2014), Quinn (2003), Georg Hager (2010)	

UNIT 5: Parallel Algorithms, Analysis, and Programming (18)	
Competences:	
Content	**Generales Goals**
- Critical paths, work and span, and the relation to Amdahl's law - Speed-up and scalability - Naturally (embarrassingly) parallel algorithms - Parallel algorithmic patterns (divide-and-conquer, map and reduce, master-workers, others) - Specific algorithms (e.g., parallel MergeSort) - Parallel graph algorithms (e.g., parallel shortest path, parallel spanning tree) (cross-reference AL/Algorithmic Strategies/Divide-and-conquer) - Parallel matrix computations - Producer-consumer and pipelined algorithms - Examples of non-scalable parallel algorithms	- Define "critical path", "work", and "span" [Familiarity] - Compute the work and span, and determine the critical path with respect to a parallel execution diagram [Usage] - Define "speed-up" and explain the notion of an algorithm's scalability in this regard [Familiarity] - Identify independent tasks in a program that may be parallelized [Usage] - Characterize features of a workload that allow or prevent it from being naturally parallelized [Familiarity] - Implement a parallel divide-and-conquer (and/or graph algorithm) and empirically measure its performance relative to its sequential analog [Usage] - Decompose a problem (eg, counting the number of occurrences of some word in a document) via map and reduce operations [Usage] - Provide an example of a problem that fits the producer-consumer paradigm [Usage] - Give examples of problems where pipelining would be an effective means of parallelization [Usage] - Implement a parallel matrix algorithm [Usage] - Identify issues that arise in producer-consumer algorithms and mechanisms that may be used for addressing them [Usage]
Readings: Matloff (2014), Quinn (2003), Georg Hager (2010)	

UNIT 6: Parallel Performance (18)	
Competences:	
Content	Generales Goals
- Load balancing - Performance measurement - Scheduling and contention (cross-reference OS/Scheduling and Dispatch) - Evaluating communication overhead - Data management - Non-uniform communication costs due to proximity (cross-reference SF/Proximity) - Cache effects (e.g., false sharing) - Maintaining spatial locality - Power usage and management	- Detect and correct a load imbalance [Usage] - Calculate the implications of Amdahl's law for a particular parallel algorithm (cross-reference SF/Evaluation for Amdahl's Law) [Usage] - Describe how data distribution/layout can affect an algorithm's communication costs [Familiarity] - Detect and correct an instance of false sharing [Usage] - Explain the impact of scheduling on parallel performance [Familiarity] - Explain performance impacts of data locality [Familiarity] - Explain the impact and trade-off related to power usage on parallel performance [Familiarity]
Readings: Pacheco (2011), Matloff (2014), Kirk and Hwu (2013), Sanders and Kandrot (2010), Georg Hager (2010)	

8. Methodology
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. Assessment
Continuous Assessment 1 : 20 % **Partial Exam** : 30 % **Continuous Assessment 2** : 20 % **Final exam** : 30 %

References

Georg Hager, Gerhard Wellein (2010). *Introduction to High Performance Computing for Scientists and Engineers (Chapman & HallCRC 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
Undergraduate Program in Information Systems
SILABO

CS402. Capstone Project I (Mandatory)

2022-II

1. General information		
1.1 School	:	Sistemas de Información
1.2 Course	:	CS402. Capstone Project I
1.3 Semester	:	8vo Semestre.
1.4 Prerrequisites	:	CS401. Methodology of Computation Research . (7th Sem)
1.5 Type of course	:	Mandatory
1.6 Learning modality	:	Face to face
1.7 Horas	:	2 HT; 2 HP;
1.8 Credits	:	3

2. Professors

3. Course foundation
This course aims to allow the student to carry out a study of the state of the art of a topic chosen by the student for his thesis.

4. Summary
1. Lifting the state of the art

5. Generales Goals

- That the student carries out an initial investigation in a specific subject realizing the study of the state of the art of the chosen subject.

- That the student shows mastery in the subject of the line of investigation chosen

- That the student choose a teacher who dominates the research chosen as an advisor.

- The deliverables of this course are:

  Avance parcial: Solid bibliography and progress of a Technical Reporto.

  Final: Technical Report with preliminary comparative experiments that demonstrate that the student already knows the existing techniques in the area of his project and choose a teacher who dominates the area of his project as an adviser of his project.

6. Contribution to Outcomes

This discipline contributes to the achievement of the following outcomes:

1) Analyze a complex computing problem and to apply principles of computing and other relevant disciplines to identify solutions. (**Assessment**)

2) Design, implement and evaluate a computing-based solution to meet a given set of computing requirements in the context of the program's discipline. (**Usage**)

3) Communicate effectively in a variety of professional contexts. (**Usage**)

4) Recognize professional responsabilities and make informed judgments in computing practice based on legal and ethical principles. (**Assessment**)

5) Function effectively as a member or leader of a team engaged in activities appropriate to the program's discipline. (**Usage**)

6) Apply computer science theory and software development fundamentals to produce computing-based solutions. (**Assessment**)

7) Develop computational technology for the well-being of all, contributing with human formation, scientific, technological and professional skills to solve social problems of our community. (**Usage**)

7. Content

UNIT 1: Lifting the state of the art (60)
Competences:

Content	Generales Goals
• Perform an in-depth study of the state of the art in a certain topic in the area of Computation. • Writing technical articles in computing.	• Make a bibliographical survey of the state of the art of the chosen subject (this probably means 1 or 2 chapters of theoretical framework in addition to the introduction that is chapter I of the thesis) [Usage] • Writing a latex document in paper format with higher quality than Project I (master tables, figures, equations, indices, bibtex, cross references, citations, pstricks) [Usage] • Try to make presentations using prosper [Usage] • Show basic experiments [Usage] • Choose an advisor who dominates the research area [Usage]

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

8. Methodology

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. Assessment
Continuous Assessment 1 : 20 %   **Partial Exam** : 30 %   **Continuous Assessment 2** : 20 %   **Final exam** : 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
Undergraduate Program in Information Systems
SILABO

ET201. Entrerpreneurship I (Mandatory)

2022-II

1. General information		
1.1 School	:	Sistemas de Información
1.2 Course	:	ET201. Entrerpreneurship I
1.3 Semester	:	8vo Semestre.
1.4 Prerrequisites	:	None
1.5 Type of course	:	Mandatory
1.6 Learning modality	:	Face to face
1.7 Horas	:	2 HT; 2 HP;
1.8 Credits	:	3

2. Professors

3. Course foundation
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. Summary
1. 2. 3. 4. 5. 6. 7. 8.

5. Generales Goals
- 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. Contribution to Outcomes

This discipline contributes to the achievement of the following outcomes:

1) Analyze a complex computing problem and to apply principles of computing and other relevant disciplines to identify solutions. (**Usage**)

2) Design, implement and evaluate a computing-based solution to meet a given set of computing requirements in the context of the program's discipline. (**Assessment**)

3) Communicate effectively in a variety of professional contexts. (**Usage**)

5) Function effectively as a member or leader of a team engaged in activities appropriate to the program's discipline. (**Usage**)

6) Apply computer science theory and software development fundamentals to produce computing-based solutions. (**Usage**)

7) Develop computational technology for the well-being of all, contributing with human formation, scientific, technological and professional skills to solve social problems of our community. (**Assessment**)

7. Content

UNIT 1: (5)

Competences:

Content	Generales Goals
- Emprendedor, emprendedurismo e innovación tecnológica - Modelos de negocio - Formación de equipos	- Identificar características de los emprendedores [Familiarity] - Introducir modelos de negocio [Familiarity]

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

UNIT 2: (5)

Competences:

Content	Generales Goals
- Visión - Misión - La Propuesta de valor - Creatividad e invención - Tipos y fuentes de innovación - Estrategia y Tecnología - Escala y ámbito	- Plantear correctamente la vision y misión de empresa [Usage] - Caracterizar una propuesta de valor innovadora [Assessment] - Identificar los diversos tipos y fuentes de innovación [Familiarity]

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

UNIT 3: (5)
Competences:

Content	Generales Goals
- Estrategia de la Empresa - Barreras - Ventaja competitiva sostenible - Alianzas - Aprendizaje organizacional - Desarrollo y diseño de productos	- Conocer estrategias empresariales [Familiarity] - Caracterizar barreras y ventajas competitivas [Familiarity]

Readings: Byers, R. Dorf, and Nelson (2010), Osterwalder and Pigneur (2010), Ries (2011), Garzozi-Pincay et al. (2014)

UNIT 4: (20)
Competences:

Content	Generales Goals
- Creación de un nuevo negocio - El plan de negocio - Canvas - Elementos del Canvas	- Conocer los elementos del modelo Canvas [Usage] - Elaborar un plan de negocio basado en el modelo Canvas [Usage]

Readings: Osterwalder and Pigneur (2010), Blank and B. Dorf (2012), Garzozi-Pincay et al. (2014)

UNIT 5: (20)
Competences:

Content	Generales Goals
- Aceleración versus incubación - Customer Development - Lean Startup	- Conocer y aplicar el modelo Customer Development [Usage] - Conocer y aplicar el modelo Lean Startup [Usage]

Readings: Blank and B. Dorf (2012), Ries (2011), Garzozi-Pincay et al. (2014)

UNIT 6: (5)
Competences:

Content	Generales Goals
- 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	- Conocer los aspectos legales necesarios para la formación de una empresa tecnológica [Familiarity] - Identificar segmentos de mercado y objetivos de marketing [Familiarity]

Readings: Byers, R. Dorf, and Nelson (2010), Ries (2011), Congreso de la Republica del Perú (1996), Republica del Peru (1997), Garzozi-Pincay et al. (2014)

UNIT 7: (5)
Competences:

Content	Generales Goals
- Modelo de costos - Modelo de utilidades - Precio - Plan financiero - Formas de financiamiento - Fuentes de capital - Capital de riesgo	- Definir um modelo de costos y utilidades [Assessment] - Conocer las diversas fuentes de financiamento [Familiarity]

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

UNIT 8: (5)
Competences:

Content	Generales Goals
- The Elevator Pitch - Presentación - Negociación	- Conocer las diversas formas de presentar propuestas de negocio [Familiarity] - Realizar la presentación de una propuesta de negocio [Usage]

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

8. Methodology

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. Assessment

Continuous Assessment 1 : 20 %

Partial Exam : 30 %

Continuous Assessment 2 : 20 %

Final exam : 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 Nº823. 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 Nº26887. 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
Undergraduate Program in Information Systems
SILABO

CS370. Big Data (Mandatory)

2022-II

1. General information		
1.1 School	:	Sistemas de Información
1.2 Course	:	CS370. Big Data
1.3 Semester	:	9no Semestre.
1.4 Prerrequisites	:	

- CS272. Databases II. (5th Sem)

- CS3P1. Parallel and Distributed Computing . (8th Sem)

1.5 Type of course	:	Mandatory
1.6 Learning modality	:	Face to face
1.7 Horas	:	1 HT; 2 HP; 2 HL;
1.8 Credits	:	3

2. Professors

3. Course foundation

Nowadays, knowing scalable approaches to processing and storing large volumes of information (terabytes, petabytes and even exabytes) is fundamental in computer science courses. Every day, every hour, every minute generates a large amount of information which needs to be processed, stored, analyzed.

4. Summary

1. Introducción a Big Data 2. Hadoop 3. Procesamiento de Grafos en larga escala

5. Generales Goals

- That the student is able to create parallel applications to process large volumes of information

- That the student is able to compare the alternatives for the processing of big data

- That the student is able to propose architectures for a scalable application

6. Contribution to Outcomes

This discipline contributes to the achievement of the following outcomes:

1) Analyze a complex computing problem and to apply principles of computing and other relevant disciplines to identify solutions. (**Usage**)

6) Apply computer science theory and software development fundamentals to produce computing-based solutions. (**Usage**)

7. Content

UNIT 1: Introducción a Big Data (15)	
Competences:	
Content	**Generales Goals**
• Overview on Cloud Computing • Distributed File System Overview • Overview of the MapReduce programming model	• Explain the concept of Cloud Computing from the point of view of Big Data[Familiarity] • Explain the concept of Distributed File System [Familiarity] • Explain the concept of the MapReduce programming model[Familiarity]
Readings: Coulouris et al. (2011)	

UNIT 2: Hadoop (15)	
Competences:	
Content	**Generales Goals**
• Hadoop overview. • History. • Hadoop Structure. • HDFS, Hadoop Distributed File System. • Programming Model MapReduce	• Understand and explain the Hadoop suite [Familiarity] • Implement solutions using the MapReduce programming model. [Usage] • Understand how data is saved in the HDFS. [Familiarity]
Readings: Hwang, Dongarra, and Fox (2011), Buyya, Vecchiola, and Selvi (2013)	

UNIT 3: Procesamiento de Grafos en larga escala (10)	
Competences:	
Content	**Generales Goals**
• 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.	• Understand and explain the architecture of the Pregel project. [Familiarity] • Understand the GraphLab project architecture. [Familiarity] • Understand the architecture of the Giraph project. [Familiarity] • Implement solutions using Pregel, GraphLab or Giraph. [Usage]
Readings: Low et al. (2012), Malewicz et al. (2010), Baluja et al. (2008)	

8. Methodology

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. Assessment

Continuous Assessment 1 : 20 %

Partial Exam : 30 %

Continuous Assessment 2 : 20 %

Final exam : 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
Undergraduate Program in Information Systems
SILABO

CS403. Capstone Project II (Mandatory)

2022-II

1. General information		
1.1 School	:	Sistemas de Información
1.2 Course	:	CS403. Capstone Project II
1.3 Semester	:	9no Semestre.
1.4 Prerrequisites	:	CS402. Capstone Project I. (8th Sem)
1.5 Type of course	:	Mandatory
1.6 Learning modality	:	Face to face
1.7 Horas	:	2 HT; 2 HP;
1.8 Credits	:	3

2. Professors

3. Course foundation
This course aims at the student to conclude his thesis project.

4. Summary
1. Thesis project 2. Thesis progress

5. Generales Goals

- That the student is in the capacity to formally present his thesis project with the theoretical framework and complete bibliographic survey.

- That the student master the state of the art of his area of research.

- The deliverables of this course are:

  Avance parcial: Thesis plan progress including motivation and context, problem definition, objectives, schedule of activities up to the final thesis project and the state of the art of the topic addressed.

  Final: Complete thesis plan and advancement of Thesis including theoretical framework chapters, related works and preliminary (formal or statistical) results oriented to your thesis topic.

6. Contribution to Outcomes

This discipline contributes to the achievement of the following outcomes:

1) Analyze a complex computing problem and to apply principles of computing and other relevant disciplines to identify solutions. (**Assessment**)

2) Design, implement and evaluate a computing-based solution to meet a given set of computing requirements in the context of the program's discipline. (**Assessment**)

3) Communicate effectively in a variety of professional contexts. (**Assessment**)

4) Recognize professional responsabilities and make informed judgments in computing practice based on legal and ethical principles. (**Assessment**)

5) Function effectively as a member or leader of a team engaged in activities appropriate to the program's discipline. (**Assessment**)

6) Apply computer science theory and software development fundamentals to produce computing-based solutions. (**Assessment**)

7) Develop computational technology for the well-being of all, contributing with human formation, scientific, technological and professional skills to solve social problems of our community. (**Assessment**)

7. Content

UNIT 1: Thesis project (30)	
Competences:	
Content	**Generales Goals**
• Thesis project.	• Description of the format used by the University for the thesis[Assessment] • Conclude the thesis project plan[Assessment] • Present the state of the art thesis topic(50%)[Assessment]
Readings: IEEE-Computer Society (2008), Association for Computing Machinery (2008), CiteSeer.IST (2008)	

UNIT 2: Thesis progress (30)	
Competences:	
Content	**Generales Goals**
• Thesis Progress.	• Description of the format used by the University for the thesis[Assessment] • Conclude the chapter of the theoretical framework of the Thesis[Assessment] • Complete the chapter on related works(35%)[Assessment] • Plan, develop and present results (formal or statistical) of experiments oriented to your thesis topic (35%)[Assessment]
Readings: IEEE-Computer Society (2008), Association for Computing Machinery (2008), CiteSeer.IST (2008)	

8. Methodology
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. Assessment
Continuous Assessment 1 : 20 % **Partial Exam** : 30 % **Continuous Assessment 2** : 20 % **Final exam** : 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
Undergraduate Program in
Information Systems
SILABO

CS351. Topics in Computer Graphics (Elective)

2022-II

1. General information		
1.1 School	:	Sistemas de Información
1.2 Course	:	CS351. Topics in Computer Graphics
1.3 Semester	:	9no Semestre.
1.4 Prerrequisites	:	CS251. Computer graphics . (7th Sem)
1.5 Type of course	:	Elective
1.6 Learning modality	:	Face to face
1.7 Horas	:	2 HT; 2 HP; 2 HL;
1.8 Credits	:	4

2. Professors

3. Course foundation
In this course you can delve into any of the topics Mentioned in the area of Graphics Computing (Graphics and Visual Computing - GV).
This course is designed to perform some advanced course suggested by the ACM / IEEE curriculum. Hughes et al. (2013); Hearn and Baker (1990)

4. Summary
1. Advanced Topics on Computer Graphics

5. Generales Goals
• That the student uses computer techniques Graphs that involve complex data structures and algorithms.
• That the student apply the concepts learned to create an application about a real problem.
• That the student investigate the possibility of creating a new algorithm and / or new technique to solve a real problem

6. Contribution to Outcomes
This discipline contributes to the achievement of the following outcomes:
1) Analyze a complex computing problem and to apply principles of computing and other relevant disciplines to identify solutions. (**Usage**)
6) Apply computer science theory and software development fundamentals to produce computing-based solutions. (**Usage**)

7. Content

UNIT 1: Advanced Topics on Computer Graphics (0)	
Competences:	
Content	**Generales Goals**
- CS355. Advanced Computer Graphics - CS356. Computer animation - CS313. Geometric Algorithms - CS357. visualization - CS358. Virtual reality - CS359. Genetic algorithms	Advanced Topics on Computer Graphics
Readings: Soars022S, Soars022W, Soars022T, Cambridge06, MacGrew99	

---

8. Methodology

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. Assessment

Continuous Assessment 1 : 20 %

Partial Exam : 30 %

Continuous Assessment 2 : 20 %

Final exam : 30 %

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.

Universidad Nacional de Colombia (UNAL) Sede Manizales
Undergraduate Program in Information Systems
SILABO

CS361. Topics in Artificial Intelligence (Elective)

2022-II

1. General information		
1.1 School	:	Sistemas de Información
1.2 Course	:	CS361. Topics in Artificial Intelligence
1.3 Semester	:	9no Semestre.
1.4 Prerrequisites	:	CS261. Artificial intelligence . (7th Sem)
1.5 Type of course	:	Elective
1.6 Learning modality	:	Face to face
1.7 Horas	:	2 HT; 2 HP; 2 HL;
1.8 Credits	:	4

2. Professors

3. Course foundation
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: asi como el aprendizaje de lenguaje natural, visión artificial y robótica entre otros.

4. Summary
1.

5. Generales Goals
• Realizar algún curso avanzado de Inteligencia Artificial sugerido por el curriculo de la ACM/IEEE.

6. Contribution to Outcomes
This discipline contributes to the achievement of the following outcomes:

1) Analyze a complex computing problem and to apply principles of computing and other relevant disciplines to identify solutions. (**Assessment**)

5) Function effectively as a member or leader of a team engaged in activities appropriate to the program's discipline. (**Usage**)

6) Apply computer science theory and software development fundamentals to produce computing-based solutions. (**Assessment**)

7. Content

UNIT 1: (60)	
Competences:	
Content	**Generales Goals**
- 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)	Profundizar en diversas técnicas relacionadas a la Inteligencia Artificial [Usage]
Readings: Russell and Norvig (2003), Haykin (1999), Goldberg (1989)	

8. Methodology
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. Assessment
Continuous Assessment 1 : 20 %   **Partial Exam** : 30 %   **Continuous Assessment 2** : 20 %   **Final exam** : 30 %

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 Artifical: Un enfoque moderno.* Prentice Hall.

Universidad Nacional de Colombia (UNAL) Sede Manizales
Undergraduate Program in Information Systems
SILABO

CS392. Advanced Topics in Software Engineering (Elective)

2022-II

1. General information		
1.1 School	:	Sistemas de Información
1.2 Course	:	CS392. Advanced Topics in Software Engineering
1.3 Semester	:	9no Semestre.
1.4 Prerrequisites	:	CS391. Software Engineering III. (7th Sem)
1.5 Type of course	:	Elective
1.6 Learning modality	:	Face to face
1.7 Horas	:	2 HT; 2 HP; 2 HL;
1.8 Credits	:	4

2. Professors

3. Course foundation
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. Summary
1. Software Design 2. Software Project Management 3. 4.

5. Generales Goals
- 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 portfolio 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 portfolio de aplicaciones Empresariales

6. Contribution to Outcomes

This discipline contributes to the achievement of the following outcomes:

1) Analyze a complex computing problem and to apply principles of computing and other relevant disciplines to identify solutions. (**Assessment**)

2) Design, implement and evaluate a computing-based solution to meet a given set of computing requirements in the context of the program's discipline. (**Assessment**)

3) Communicate effectively in a variety of professional contexts. (**Usage**)

5) Function effectively as a member or leader of a team engaged in activities appropriate to the program's discipline. (**Usage**)

6) Apply computer science theory and software development fundamentals to produce computing-based solutions. (**Assessment**)

7) Develop computational technology for the well-being of all, contributing with human formation, scientific, technological and professional skills to solve social problems of our community. (**Assessment**)

7. Content

UNIT 1: Software Design (18)	
Competences:	
Content	**Generales Goals**

Content	Generales Goals

• System design principles: levels of abstraction (architectural design and detailed design), separation of concerns, information hiding, coupling and cohesion , re-use of standard structures
 	
• Design Paradigms such as structured design (top-down functional decomposition), object-oriented analysis and design, event driven design, component-level design, data-structured centered, aspect oriented, function oriented, service oriented
 	
• Structural and behavioral models of software designs
 	
• Design patterns
 	
• Relationships between requirements and designs: transformation of models, design of contracts, invariants
 	
• Software architecture concepts and standard architectures (e.g. client-server, n-layer, transform centered, pipes-and-filters)
 	
• The use of component desing: component selection, design, adaptation and assembly of components, component and patterns, components and objects (for example, building a GUI using a standar widget set)
 	
• Refactoring designs using design patterns
 	
• Internal design qualities, and models for them: efficiency and performance, redundacy and fault tolerance, traceability of requeriments
 	
• Measurement and analysis of design quality
 	
• Tradeoffs between different aspects of quality
 	
• Application frameworks
 	
• Middleware: the object-oriented paradigm within middleware, object request brokers and marshalling, transaction processing monitors, workflow systems
 	
• Principles of secure design and coding
   – Principle of least privilege
   – Principle of fail-safe defaults
   – Principle of psychological acceptability
	• Articulate design principles including separation of concerns, information hiding, coupling and cohesion, and encapsulation [Usage]
	
	• Use a design paradigm to design a simple software system, and explain how system design principles have been applied in this design [Usage]
	
	• Construct models of the design of a simple software system that are appropriate for the paradigm used to design it [Usage]
	
	• Within the context of a single design paradigm, describe one or more design patterns that could be applicable to the design of a simple software system [Usage]
	
	• For a simple system suitable for a given scenario, discuss and select an appropriate design paradigm [Usage]
	
	• Create appropriate models for the structure and behavior of software products from their requirements specifications [Usage]
	
	• Explain the relationships between the requirements for a software product and its design, using appropriate models [Usage]
	
	• For the design of a simple software system within the context of a single design paradigm, describe the software architecture of that system [Usage]
	
	• Given a high-level design, identify the software architecture by differentiating among common software architectures such as 3-tier, pipe-and-filter, and client-server [Usage]
	
	• Investigate the impact of software architectures selection on the design of a simple system [Usage]
	
	• Apply simple examples of patterns in a software design [Usage]
	
	• Describe a form of refactoring and discuss when it may be applicable [Usage]
	
	• Select suitable components for use in the design of a software product [Usage]
	
	• Explain how suitable components might need to be adapted for use in the design of a software product [Usage]
	
	• Design a contract for a typical small software component for use in a given system [Usage]
	
	• Discuss and select appropriate software architecture for a simple system suitable for a given scenario [Usage]
	
	• Apply models for internal and external qualities in designing software components to achieve an accept-
	able tradeoff between conflicting quality aspects [Us-

UNIT 2: Software Project Management (14)	
Competences:	
Content	**Generales Goals**

Content	Generales Goals
- Team participation - Team processes including responsabilities for task, meeting structure, and work schedule - Roles and responsabilities in a software team - Team conflict resolution - Risks associated with virtual teams (communication, perception, structure) - Effort estimation (at the personal level) - Risk - The role of risk in the lifecycle - Risk categories including security, safety, market, financial, technology, people, quality, structure and process - Team management - Team organization and decision-making - Role identification and assigment - Individual and team performance assessment - Project management - Scheduling and tracking - Project management tools - Cost/benefit analysis - Software measurement and estimation techniques - Software quality assurance and the role of measurements - Risk - The role of risk in the lifecycle - Risk categories including security, safety, market, financial, technology, people, quality, structure and process - System-wide approach to risk including hazards associated with tools	- Discuss common behaviors that contribute to the effective functioning of a team [Usage] - Create and follow an agenda for a team meeting [Usage] - Identify and justify necessary roles in a software development team [Usage] - Understand the sources, hazards, and potential benefits of team conflict [Usage] - Apply a conflict resolution strategy in a team setting [Usage] - Use an ad hoc method to estimate software development effort (eg, time) and compare to actual effort required [Usage] - List several examples of software risks [Usage] - Describe the impact of risk in a software development lifecycle [Usage] - Describe different categories of risk in software systems [Usage] - Demonstrate through involvement in a team project the central elements of team building and team management [Usage] - Describe how the choice of process model affects team organizational structures and decision-making processes [Usage] - Create a team by identifying appropriate roles and assigning roles to team members [Usage] - Assess and provide feedback to teams and individuals on their performance in a team setting [Usage] - Using a particular software process, describe the aspects of a project that need to be planned and monitored, (eg, estimates of size and effort, a schedule, resource allocation, configuration control, change management, and project risk identification and management) [Usage] - Track the progress of some stage in a project using appropriate project metrics [Usage] - Compare simple software size and cost estimation techniques [Usage] - Use a project management tool to assist in the assignment and tracking of tasks in a software development project [Usage] - Describe the impact of risk tolerance on the software development process [Usage] - Identify risks and describe approaches to managing risk (avoidance, acceptance, transference, mitigation), and characterize the strengths and short-

UNIT 3: (14)	
Competences:	
Content	**Generales Goals**
• 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.	• Utilizar y aplicar correctamente ITIL en el proceso de software. [Usage]
Readings: Sommerville (2017), Pressman and Maxim (2015)	

UNIT 4: (14)	
Competences:	
Content	**Generales Goals**
• Fundamentos e Introducción. • Frameworks de Control y IT Governance.	• Utilizar y aplicar correctamente COBIT en el proceso de software. [Usage]
Readings: Sommerville (2017), Pressman and Maxim (2015)	

8. Methodology
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. Assessment
Continuous Assessment 1 : 20 % **Partial Exam** : 30 % **Continuous Assessment 2** : 20 % **Final exam** : 30 %

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
Undergraduate Program in Information Systems
SILABO

CB309. Bioinformatics (Mandatory)

2022-II

1. General information		
1.1 School	:	Sistemas de Información
1.2 Course	:	CB309. Bioinformatics
1.3 Semester	:	9no Semestre.
1.4 Prerrequisites	:	

- CS212. Algorithm Analysis and Design. (5th Sem)

- MA307. Mathematics applied to computing . (6th Sem)

1.5 Type of course	:	Mandatory
1.6 Learning modality	:	Face to face
1.7 Horas	:	1 HT; 2 HL;
1.8 Credits	:	2

2. Professors

3. Course foundation
The use of computational methods in the biological sciences has become one of the key tools for the field of molecular biology, being a fundamental part of research in this area. In Molecular Biology, there are several applications that involve both DNA, protein analysis or sequencing of the human genome, which depend on computational methods. Many of these problems are really complex and deal with large data sets. This course can be used to see concrete use cases of several areas of knowledge of Computer Science such as Programming Languages (PL), Algorithms and Complexity (AL), Probabilities and Statistics, Information Management (IM), Intelligent Systems (IS).

4. Summary
1. Introduction to Molecular Biology 2. Sequence Comparison 3. Phylogenetic Trees 4. DNA Sequence Assembling 5. Secondary and tertiary structures 6. Probabilistic Models in Molecular Biology

5. Generales Goals

- That the student has a solid knowledge of molecular biological problems that challenge computing.

- That the student is able to abstract the essence of the various biological problems to pose solutions using their knowledge of Computer Science

6. Contribution to Outcomes

This discipline contributes to the achievement of the following outcomes:

1) Analyze a complex computing problem and to apply principles of computing and other relevant disciplines to identify solutions. (**Assessment**)

2) Design, implement and evaluate a computing-based solution to meet a given set of computing requirements in the context of the program's discipline. (**Usage**)

6) Apply computer science theory and software development fundamentals to produce computing-based solutions. (**Usage**)

7) Develop computational technology for the well-being of all, contributing with human formation, scientific, technological and professional skills to solve social problems of our community. (**Usage**)

7. Content

UNIT 1: Introduction to Molecular Biology (4)	
Competences:	
Content	**Generales Goals**
• Review of organic chemistry: molecules and macro-molecules, sugars, nucleic acids, nucleotides, RNA, DNA, proteins, amino acids and levels of structure in proteins. • The Dogma of Life: From DNA to Proteins, Transcription, Translation, Protein Synthesis. • Genome study: Maps and sequences, specific techniques	• Achive a general knowledge of the most important topics in Molecular Biology. [Familiarity] • Understand that biological problems are a challenge to the computational world. [Assessment]
Readings: Clote and Backofen (2000), Setubal and Meidanis (1997)	

UNIT 2: Sequence Comparison (4)	
Competences:	
Content	**Generales Goals**
• Sequences of nucleotides and amino acid sequences. • Sequence alignment, paired alignment problem, exhaustive search, Dynamic programming, global alignment, local alignment, gaps penalty • Comparison of multiple sequences: sum of pairs, complexity analysis by dynamic programming, alignment heuristics, star algorithm, progressive alignment algorithms.	• Understand and solve the problem of aligning a pair of sequences. [Usage] • Understand and solve the problem of multiple sequence alignment. [Usage] • Know the various algorithms for aligning existing sequences in the literature . [Familiarity]
Readings: Clote and Backofen (2000), Setubal and Meidanis (1997), Pevzner (2000)	

UNIT 3: Phylogenetic Trees (4)	
Competences:	
Content	**Generales Goals**
• Phylogeny: Introduction and phylogenetic relations • Phylogenetic trees: definition, type of trees, problem of search and reconstruction of trees • Reconstruction methods: parsimony methods, distance methods, maximum likelihood methods, confidence of reconstructed trees	• Understand the concept of phylogeny, phylogenetic trees and the methodological difference between biology and molecular biology. [Familiarity] • Understand the problem of the reconstruction of phylogenetic trees, to know and apply the main algorithms for the reconstruction of phylogenetic trees. [Assessment]
Readings: Clote and Backofen (2000), Setubal and Meidanis (1997), Pevzner (2000)	

UNIT 4: DNA Sequence Assembling (4)	
Competences:	
Content	**Generales Goals**
• Biological basis: ideal case, difficulties, alternative methods for DNA sequencing • Formal Assembly Models: Shortest Common Superstring, Reconstruction, Multicontig • Algorithms for sequence assembly: representation of overlaps, paths to create superstrings, voracious algorithm, acyclic graphs. • Assembly heuristics: search for overlays, ordering fragments, alignments and consensus.	• Understand the computational challenge of the Sequence Assembly problem. [Familiarity] • Understand the principle of formal model for assembly. [Assessment] • Know the main heuristics for the problem of assembjale of DNA sequences[Usage]
Readings: Setubal and Meidanis (1997), Aluru (2006)	

UNIT 5: Secondary and tertiary structures (4)	
Competences:	
Content	**Generales Goals**
• Molecular structures: primary, secondary, tertiary, quaternary. • Prediction of secondary structures of RNA: formal model, pair energy, structures with independent bases, solution with Dynamic Programming, structures with loops. • *Protein folding*: Estructuras en proteinas, problema de protein folding. • *Protein Threading*: Definitions, Branch Bound Algorithm, Branch Bound for protein threading. • *Structural Alignment*: Definitions, DALI algorithm	• Know the protein structures and the necessity of computational methods for the prediction of the geometry. [Familiarity] • Know the algorithms for solving prediction problems of secondary structures RNA, and structures in proteins. [Assessment]
Readings: Setubal and Meidanis (1997), Clote and Backofen (2000), Aluru (2006)	

UNIT 6: Probabilistic Models in Molecular Biology (4)	
Competences:	
Content	**Generales Goals**
- Probability: Random Variables, Markov Chains, Metropoli-Hasting Algorithm, Markov Random Fields, and Gibbs Sampler, Maximum Likelihood. - Hidden Markov Models (HMM), parameter estimation, Viterbi algorithm and Baul-Welch method, Application in paired and multiple alignments, Motifs detection in proteins, in eukaryotic DNA, in sequences families. - Probabilistic phylogeny: probabilistic models of evolution, likelihood of alignments, likelihood for inference, comparison of probailistic and non-probabilistic methods	- Review concepts of Probabilistic Models and understand their importance in Computational Molecular Biology. [Assessment] - Know and apply Hidden Markov Models for various analyzes in Molecular Biology.. [Usage] - Know the application of probabilistic models in Phylogeny and to compare them with non-probabilistic models[Assessment]
Readings: Durbin et al. (1998), Clote and Backofen (2000), Aluru (2006), Krogh et al. (1994)	

8. Methodology
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. Assessment
Continuous Assessment 1 : 20 %   **Partial Exam** : 30 %   **Continuous Assessment 2** : 20 %   **Final exam** : 30 %

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
Undergraduate Program in Information Systems
SILABO

ET301. Entrerpreneurship II (Mandatory)

2022-II

1. General information		
1.1 School	:	Sistemas de Información
1.2 Course	:	ET301. Entrerpreneurship II
1.3 Semester	:	9no Semestre.
1.4 Prerrequisites	:	ET201. Entrerpreneurship I. (8th Sem)
1.5 Type of course	:	Mandatory
1.6 Learning modality	:	Face to face
1.7 Horas	:	2 HT; 2 HP;
1.8 Credits	:	3

2. Professors

3. Course foundation
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. Summary
1. 2. 3.

5. Generales Goals
- 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. Contribution to Outcomes

This discipline contributes to the achievement of the following outcomes:

7) Develop computational technology for the well-being of all, contributing with human formation, scientific, technological and professional skills to solve social problems of our community. (**Familiarity**)

7) Develop computational technology for the well-being of all, contributing with human formation, scientific, technological and professional skills to solve social problems of our community. (**Usage**)

3) Communicate effectively in a variety of professional contexts. (**Usage**)

5) Function effectively as a member or leader of a team engaged in activities appropriate to the program's discipline. (**Usage**)

7) Develop computational technology for the well-being of all, contributing with human formation, scientific, technological and professional skills to solve social problems of our community. (**Assessment**)

7. Content

UNIT 1: (20)

Competences:

Content	Generales Goals
- 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	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. [Assessment]

Readings: Blank and Tarkin (2006)

UNIT 2: (30)
Competences:

Content	Generales Goals
- 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	- Brindar las herramientas al alumno para que pueda identificar, analizar y aprovechar las oportunidades de marketing que generan valor en un emprendimiento. [Usage] - 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. [Usage]

Readings: Kotler and Keller (2006), Lovelock and Wirtz (2009)

UNIT 3: (10)
Competences:

Content	Generales Goals
- 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	- Conocer los puntos clave en el proceso de negociación. [Usage] - Establecer una metodología de negociación eficaz. [Usage] - Desarrollar destrezas y habilidades que permitan llevar a cabo una negociación exitosa. [Usage]

Readings: Fisher, Ury, and Patton (1996), Manuel Dasí and Martínez (2006)

8. Methodology

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.

3

9. Assessment
Continuous Assessment 1 : 20 %   **Partial Exam** : 30 %   **Continuous Assessment 2** : 20 %   **Final exam** : 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 estratégia*. 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
Undergraduate Program in Information Systems
SILABO

CS3P2. Cloud Computing (Mandatory)

2022-II

1. General information		
1.1 School	:	Sistemas de Información
1.2 Course	:	CS3P2. Cloud Computing
1.3 Semester	:	10mo Semestre.
1.4 Prerrequisites	:	CS370. Big Data. (9th Sem)
1.5 Type of course	:	Mandatory
1.6 Learning modality	:	Face to face
1.7 Horas	:	1 HT; 2 HP; 2 HL;
1.8 Credits	:	3

2. Professors

3. Course foundation
In order to understand the advanced computational techniques, the students must have a strong knowledge of the various discrete structures, structures that will be implemented and used in the laboratory in the programming language.

4. Summary
1. Distributed Systems 2. Cloud Computing 3. Centros de Procesamiento de Datos 4. Cloud Computing 5. Cloud Computing 6. Modelos de Programación

5. Generales Goals
• That the student is able to model computer science problems using graphs and trees related to data structures.   • That the student apply efficient travel strategies to be able to search data in an optimal way.

6. Contribution to Outcomes
This discipline contributes to the achievement of the following outcomes:

1) Analyze a complex computing problem and to apply principles of computing and other relevant disciplines to identify solutions. (**Usage**)

6) Apply computer science theory and software development fundamentals to produce computing-based solutions. (**Usage**)

7. Content

UNIT 1: Distributed Systems (15)	
Competences:	
Content	**Generales Goals**
- Faults (cross-reference OS/Fault Tolerance) - Network-based (including partitions) and node-based failures - Impact on system-wide guarantees (e.g., availability) - Distributed message sending - Data conversion and transmission - Sockets - Message sequencing - Buffering, retrying, and dropping messages - Distributed system design tradeoffs - Latency versus throughput - Consistency, availability, partition tolerance - Distributed service design - Stateful versus stateless protocols and services - Session (connection-based) designs - Reactive (IO-triggered) and multithreaded designs - Core distributed algorithms - Election, discovery	- Distinguish network faults from other kinds of failures [Familiarity] - Explain why synchronization constructs such as simple locks are not useful in the presence of distributed faults [Familiarity] - Write a program that performs any required marshalling and conversion into message units, such as packets, to communicate interesting data between two hosts [Usage] - Measure the observed throughput and response latency across hosts in a given network [Usage] - Explain why no distributed system can be simultaneously consistent, available, and partition tolerant [Familiarity] - Implement a simple server – for example, a spell checking service [Usage] - Explain the tradeoffs among overhead, scalability, and fault tolerance when choosing a stateful v stateless design for a given service [Familiarity] - Describe the scalability challenges associated with a service growing to accommodate many clients, as well as those associated with a service only transiently having many clients [Familiarity] - Give examples of problems for which consensus algorithms such as leader election are required [Usage]
Readings: Coulouris et al. (2011)	

UNIT 2: Cloud Computing (15)	
Competences:	
Content	**Generales Goals**
- Visión global de *Cloud Computing.* - Historia. - Visión global de las tecnologias que envuelve. - Beneficios, riesgos y aspectos económicos. - Cloud services – Infrastructure as a service * Elasticity of resources * Platform APIs – Software as a service – Security – Cost management - Internet-Scale computing – Task partitioning – Data access – Clusters, grids, and meshes	- Explicar el concepto de Cloud Computing. [Familiarity] - Listar algunas tecnologias relacionadas con Cloud Computing. [Familiarity] - Explain strategies to synchronize a common view of shared data across a collection of devices [Familiarity] - Discutir las ventajas y desventajas del paradigma de Cloud Computing. [Familiarity] - Expresar los beneficios económicos así como las carácteristicas y riesgos del paradigma de Cloud para negocios y proveedores de cloud. [Familiarity] - Diferenciar entre los modelos de servicio. [Usage]
Readings: Hwang, Dongarra, and Fox (2011), Buyya, Vecchiola, and Selvi (2013)	

UNIT 3: Centros de Procesamiento de Datos (10)	
Competences:	
Content	**Generales Goals**
- Visión global de un centro de procesamiento de datos. - Consideraciones en el diseño. - Comparación de actuales grandes centros de procesamiento de datos.	- Describir la evolución de los Data Centers. [Familiarity] - Esbozar la arquitectura de un data center en detalle. [Familiarity] - Indicar consideraciones de diseño y discutir su impacto. [Familiarity]
Readings: Hwang, Dongarra, and Fox (2011), Buyya, Vecchiola, and Selvi (2013)	

UNIT 4: Cloud Computing (20)	
Competences:	
Content	**Generales Goals**

• Virtualization
 	
    – Shared resource management
 	
    – Migration of processes
 	
• Seguridad, recursos y isolamiento de fallas.
 	
• Almacenamiento como servicio.
 	
• Elasticidad.
 	
• Xen y WMware.
 	
• Amazon EC2.
	• Virtualization
	
	    – Shared resource management
	
	    – Migration of processes
	
	  . [Familiarity]
	
	• Explain the advantages and disadvantages of using virtualized infrastructure. [Familiarity]
	
	• Identificar las razones por qué la virtualización está llegando a ser enormente útil, especialmente en la cloud. [Familiarity]
	
	• Explicar diferentes tipos de isolamiento como falla, recursos y seguridad proporcionados por la virtualización y utilizado por la cloud. [Familiarity]
	
	• 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. [Familiarity]
	
	• Definir virtualización y identificar diferentes tipos de máquinas virtuales. [Familiarity]
	
	• Identificar condiciones de virtualización de CPU, reconocer la diferencia entre *full virtualization* y *paravirtualization*, explicar emulación como mayor técnica para virtualización del CPU y examinar planificación virtual del CPU en Xen. [Familiarity]
	
	• Esbozar la diferencia entre la clásica memoria virtual del SO y la virtualización de memoria. Explicar los múltiplos niveles de mapeamiento de páginas en oposición a la virtualización de la memoria. Definir memoria *over-commitment* e ilustrar sobre WMware *memory ballooning* como técnica de reclamo para sistemas virtualizados con memoria *over-committed*. [Familiarity]

Readings: Hwang, Dongarra, and Fox (2011), Buyya, Vecchiola, and Selvi (2013)	

UNIT 5: Cloud Computing (12)
Competences:

Content	Generales Goals
- Cloud-based data storage - Shared access to weakly consistent data stores - Data synchronization - Data partitioning - Distributed file systems - Replication - 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.	- Describir la organización general de datos y almacenamiento. [Familiarity] - Identificar los problemas de escalabilidad y administración de la big data. Discutir varias abstracciones en almacenamiento. [Familiarity] - 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). [Usage] - Comparar y contrastar diferentes tipos de bases de datos. Discutir las ventajas y desventajas sobre las bases de datos NoSQL. [Usage] - Discutir los conceptos de almacenamiento en la cloud. [Familiarity]

Readings: Hwang, Dongarra, and Fox (2011), Buyya, Vecchiola, and Selvi (2013)

UNIT 6: Modelos de Programación (12)
Competences:

Content	Generales Goals
- Visión global de los modelso de programación basados en cloud computing. - Modelo de Programación MapReduce. - Modelo de programación para aplicaciones basadas en Grafos.	- Explicar los aspectos fundamentales de los modelos de programación paralela y distribuida. [Familiarity] - Diferencias entre los modelos de programación: MapReduce, Pregel, GraphLab y Giraph. [Usage] - Explicar los principales conceptos en el modelo de programación MapReduce. [Usage]

Readings: Hwang, Dongarra, and Fox (2011), Buyya, Vecchiola, and Selvi (2013), Low et al. (2012), Malewicz et al. (2010), Baluja et al. (2008)

8. Methodology
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. Assessment

Continuous Assessment 1 : 20 %

Partial Exam : 30 %

Continuous Assessment 2 : 20 %

Final exam : 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
Undergraduate Program in Information Systems
SILABO

CS3P3. Internet of Things (Mandatory)

2022-II

1. General information		
1.1 School	:	Sistemas de Información
1.2 Course	:	CS3P3. Internet of Things
1.3 Semester	:	10mo Semestre.
1.4 Prerrequisites	:	CS3P1. Parallel and Distributed Computing . (8th Sem)
1.5 Type of course	:	Mandatory
1.6 Learning modality	:	Face to face
1.7 Horas	:	1 HT; 2 HP; 2 HL;
1.8 Credits	:	3

2. Professors

3. Course foundation
The last decade has an explosive growth in multiprocessor computing, including multi-core processors and distributed data centers. As a result, parallel and distributed computing has evolved from a broadly elective subject to be one of the major components in mesh studies in undergraduate computer science. Both parallel computing and distribution involve the simultaneous execution of multiple processes on different devices that change position.

4. Summary
1. Parallelism Fundamentals 2. Parallel Architecture 3. Parallel Decomposition 4. Communication and Coordination 5. Parallel Algorithms, Analysis, and Programming 6. Parallel Performance

5. Generales Goals
• That the student is able to create parallel applications of medium complexity by efficiently taking advantage of different mobile devices.

6. Contribution to Outcomes
This discipline contributes to the achievement of the following outcomes:
Nooutcomes

7. Content

UNIT 1: Parallelism Fundamentals (18)	
Competences:	
Content	**Generales Goals**
- Multiple simultaneous computations - Goals of parallelism (e.g., throughput) versus concurrency (e.g., controlling access to shared resources) - Parallelism, communication, and coordination - Parallelism, communication, and coordination - Need for synchronization - Programming errors not found in sequential programming - Data races (simultaneous read/write or write/write of shared state) - Higher-level races (interleavings violating program intention, undesired non-determinism) - Lack of liveness/progress (deadlock, starvation)	- Distinguish using computational resources for a faster answer from managing efficient access to a shared resource [Familiarity] - Distinguish multiple sufficient programming constructs for synchronization that may be inter-implementable but have complementary advantages [Familiarity] - Distinguish data races from higher level races [Familiarity]
Readings: Pacheco (2011), Matloff (2014), Quinn (2003)	

UNIT 2: Parallel Architecture (12)	
Competences:	
Content	**Generales Goals**
- Multicore processors - Shared vs distributed memory - Symmetric multiprocessing (SMP) - SIMD, vector processing - GPU, co-processing - Flynn's taxonomy - Instruction level support for parallel programming - Atomic instructions such as Compare and Set - Memory issues - Multiprocessor caches and cache coherence - Non-uniform memory access (NUMA) - Topologies - Interconnects - Clusters - Resource sharing (e.g., buses and interconnects)	- Explain the differences between shared and distributed memory [Assessment] - Describe the SMP architecture and note its key features [Assessment] - Characterize the kinds of tasks that are a natural match for SIMD machines [Usage] - Describe the advantages and limitations of GPUs vs CPUs [Usage] - Explain the features of each classification in Flynn's taxonomy [Usage] - Describe the challenges in maintaining cache coherence [Familiarity] - Describe the key performance challenges in different memory and distributed system topologies [Familiarity]
Readings: Pacheco (2011), Kirk and Hwu (2013), Sanders and Kandrot (2010)	

UNIT 3: Parallel Decomposition (18)	
Competences:	
Content	**Generales Goals**

• Need for communication and coordination/synchronization
 	
• Independence and partitioning
 	
• Basic knowledge of parallel decomposition concept
 	
• Task-based decomposition
 	
   – Implementation strategies such as threads
 	
• Data-parallel decomposition
 	
   – Strategies such as SIMD and MapReduce
 	
• Actors and reactive processes (e.g., request handlers)
	• Explain why synchronization is necessary in a specific parallel program [Usage]
	
	• Identify opportunities to partition a serial program into independent parallel modules [Familiarity]
	
	• Write a correct and scalable parallel algorithm [Usage]
	
	• Parallelize an algorithm by applying task-based decomposition [Usage]
	
	• Parallelize an algorithm by applying data-parallel decomposition [Usage]
	
	• Write a program using actors and/or reactive processes [Usage]

Readings: Pacheco (2011), Matloff (2014), Quinn (2003)	

UNIT 4: Communication and Coordination (18)	
Competences:	
Content	**Generales Goals**
- Shared Memory - Consistency, and its role in programming language guarantees for data-race-free programs - Message passing - Point-to-point versus multicast (or event-based) messages - Blocking versus non-blocking styles for sending and receiving messages - Message buffering (cross-reference PF/Fundamental Data Structures/Queues) - Atomicity - Specifying and testing atomicity and safety requirements - Granularity of atomic accesses and updates, and the use of constructs such as critical sections or transactions to describe them - Mutual Exclusion using locks, semaphores, monitors, or related constructs * Potential for liveness failures and deadlock (causes, conditions, prevention) - Composition * Composing larger granularity atomic actions using synchronization * Transactions, including optimistic and conservative approaches - Consensus (Cyclic) barriers, counters, or related constructs - Conditional actions - Conditional waiting (e.g., using condition variables)	- Use mutual exclusion to avoid a given race condition [Usage] - Give an example of an ordering of accesses among concurrent activities (eg, program with a data race) that is not sequentially consistent [Familiarity] - Give an example of a scenario in which blocking message sends can deadlock [Usage] - Explain when and why multicast or event-based messaging can be preferable to alternatives [Familiarity] - Write a program that correctly terminates when all of a set of concurrent tasks have completed [Usage] - Give an example of a scenario in which an attempted optimistic update may never complete [Familiarity] - Use semaphores or condition variables to block threads until a necessary precondition holds [Usage]
Readings: Pacheco (2011), Matloff (2014), Quinn (2003)	

UNIT 5: Parallel Algorithms, Analysis, and Programming (18)	
Competences:	
Content	**Generales Goals**
- Critical paths, work and span, and the relation to Amdahl's law - Speed-up and scalability - Naturally (embarrassingly) parallel algorithms - Parallel algorithmic patterns (divide-and-conquer, map and reduce, master-workers, others) - Specific algorithms (e.g., parallel MergeSort) - Parallel graph algorithms (e.g., parallel shortest path, parallel spanning tree) (cross-reference AL/Algorithmic Strategies/Divide-and-conquer) - Parallel matrix computations - Producer-consumer and pipelined algorithms - Examples of non-scalable parallel algorithms	- Define "critical path", "work", and "span" [Familiarity] - Compute the work and span, and determine the critical path with respect to a parallel execution diagram [Usage] - Define "speed-up" and explain the notion of an algorithm's scalability in this regard [Familiarity] - Identify independent tasks in a program that may be parallelized [Usage] - Characterize features of a workload that allow or prevent it from being naturally parallelized [Familiarity] - Implement a parallel divide-and-conquer (and/or graph algorithm) and empirically measure its performance relative to its sequential analog [Usage] - Decompose a problem (eg, counting the number of occurrences of some word in a document) via map and reduce operations [Usage] - Provide an example of a problem that fits the producer-consumer paradigm [Usage] - Give examples of problems where pipelining would be an effective means of parallelization [Usage] - Implement a parallel matrix algorithm [Usage] - Identify issues that arise in producer-consumer algorithms and mechanisms that may be used for addressing them [Usage]
Readings: Matloff (2014), Quinn (2003)	

UNIT 6: Parallel Performance (18)	
Competences:	
Content	**Generales Goals**
- Load balancing - Performance measurement - Scheduling and contention (cross-reference OS/Scheduling and Dispatch) - Evaluating communication overhead - Data management - Non-uniform communication costs due to proximity (cross-reference SF/Proximity) - Cache effects (e.g., false sharing) - Maintaining spatial locality - Power usage and management	- Detect and correct a load imbalance [Usage] - Calculate the implications of Amdahl's law for a particular parallel algorithm (cross-reference SF/Evaluation for Amdahl's Law) [Usage] - Describe how data distribution/layout can affect an algorithm's communication costs [Familiarity] - Detect and correct an instance of false sharing [Usage] - Explain the impact of scheduling on parallel performance [Familiarity] - Explain performance impacts of data locality [Familiarity] - Explain the impact and trade-off related to power usage on parallel performance [Familiarity]
Readings: Pacheco (2011), Matloff (2014), Kirk and Hwu (2013), Sanders and Kandrot (2010)	

8. Methodology
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. Assessment
Continuous Assessment 1 : 20 % **Partial Exam** : 30 % **Continuous Assessment 2** : 20 % **Final exam** : 30 %

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
Undergraduate Program in Information Systems
SILABO

CS404. Final Project III (Mandatory)

2022-II

1. General information		
1.1 School	:	Sistemas de Información
1.2 Course	:	CS404. Final Project III
1.3 Semester	:	10mo Semestre.
1.4 Prerrequisites	:	CS403. Capstone Project II. (9th Sem)
1.5 Type of course	:	Mandatory
1.6 Learning modality	:	Face to face
1.7 Horas	:	2 HT; 8 HP;
1.8 Credits	:	6

2. Professors

3. Course foundation
This course aims to enable students to complete properly their draft of thesis.

4. Summary
1. Escritura del Borrador del trabajo de final de carrera (tesis)

5. Generales Goals

- That the student completes this course with his thesis elaborated in sufficient quality as for an immediate support.

- That the student formally present the draft dissertation before the authorities of the faculty

- The deliverables of this course are:

  Parcial: Advancement of the thesis project including in the document: introduction, theoretical framework, state of the art, proposal, analysis and / or experiments and solid bibliography.

  Final: Full thesis document and ready to support in a period of no more than fifteen days.

6. Contribution to Outcomes

This discipline contributes to the achievement of the following outcomes:

1) Analyze a complex computing problem and to apply principles of computing and other relevant disciplines to identify solutions. (**Assessment**)

2) Design, implement and evaluate a computing-based solution to meet a given set of computing requirements in the context of the program's discipline. (**Assessment**)

3) Communicate effectively in a variety of professional contexts. (**Assessment**)

4) Recognize professional responsabilities and make informed judgments in computing practice based on legal and ethical principles. (**Assessment**)

5) Function effectively as a member or leader of a team engaged in activities appropriate to the program's discipline. (**Assessment**)

6) Apply computer science theory and software development fundamentals to produce computing-based solutions. (**Assessment**)

7) Develop computational technology for the well-being of all, contributing with human formation, scientific, technological and professional skills to solve social problems of our community. (**Assessment**)

7. Content

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

Content	Generales Goals
• Writing and correction of the work of end of career	• Experimental part completed (if appropriate to the project) [Assessment] • Verify that the document complies with the thesis format of the course [Assessment] • Delivery of the completed thesis draft and considered ready for public support (approval requirement)[Assessment]

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

8. Methodology

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. Assessment

Continuous Assessment 1 : 20 %

Partial Exam : 30 %

Continuous Assessment 2 : 20 %

Final exam : 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
Undergraduate Program in Information Systems
SILABO

CS362. Robotics (Elective)

2022-II

1. General information		
1.1 School	:	Sistemas de Información
1.2 Course	:	CS362. Robotics
1.3 Semester	:	10mo Semestre.
1.4 Prerrequisites	:	CS361. Topics in Artificial Intelligence. (9th Sem)
1.5 Type of course	:	Elective
1.6 Learning modality	:	Face to face
1.7 Horas	:	2 HT; 4 HL;
1.8 Credits	:	4

2. Professors

3. Course foundation
That the student knows and understands the concepts and fundamental principles of control, road planning and the definition of strategies in robotics as well as concepts of robotic perception in a way that understands the potential of robotic systems

4. Summary
1. Robotics 2. Robotics 3. Robotics 4. Perception and Computer Vision 5. Robotics

5. Generales Goals
- Synthesize the potential and limitations of the state-of-the-art of today's robotic systems.   - Implement Simple Motion Planning Algorithms.   - Explain the uncertainties associated with sensors and how to treat them.   - Designing a Simple Control Architecture.   - Describes several navigation strategies   - Describe the importance of recognizing images and objects in intelligent systems   - Outline the main techniques of object recognition   - Describe the different characteristics of the technologies used in perception

6. Contribution to Outcomes

This discipline contributes to the achievement of the following outcomes:

1) Analyze a complex computing problem and to apply principles of computing and other relevant disciplines to identify solutions. (**Usage**)

5) Function effectively as a member or leader of a team engaged in activities appropriate to the program's discipline. (**Usage**)

6) Apply computer science theory and software development fundamentals to produce computing-based solutions. (**Usage**)

7. Content

UNIT 1: Robotics (5)
Competences:

Content	Generales Goals
- Overview: problems and progress - State-of-the-art robot systems, including their sensors and an overview of their sensor processing - Robot control architectures, e.g., deliberative vs. reactive control and Braitenberg vehicles - World modeling and world models - Inherent uncertainty in sensing and in control - Configuration space and environmental maps	- List capabilities and limitations of today's state-of-the-art robot systems, including their sensors and the crucial sensor processing that informs those systems [Familiarity] - Integrate sensors, actuators, and software into a robot designed to undertake some task [Usage]

Readings: Siegwart and Nourbakhsh (2004), S, W, and D (2005), Stone (2000)

UNIT 2: Robotics (15)
Competences:

Content	Generales Goals
- Interpreting uncertain sensor data - Localizing and mapping	- Program a robot to accomplish simple tasks using deliberative, reactive, and/or hybrid control architectures [Usage] - Implement fundamental motion planning algorithms within a robot configuration space [Usage]

Readings: Siegwart and Nourbakhsh (2004), S, W, and D (2005)

UNIT 3: Robotics (20)
Competences:

Content	Generales Goals
- Navigation and control - Motion planning	- Characterize the uncertainties associated with common robot sensors and actuators; articulate strategies for mitigating these uncertainties [Usage] - List the differences among robots' representations of their external environment, including their strengths and shortcomings [Usage]

Readings: Siegwart and Nourbakhsh (2004)

UNIT 4: Perception and Computer Vision (10)	
Competences:	

Content	Generales Goals

• Computer vision
    – Image acquisition, representation, processing and properties
    – Shape representation, object recognition and segmentation
    – Motion analysis
 • Modularity in recognition
	• Summarize the importance of image and object recognition in AI and indicate several significant applications of this technology [Usage]
	
	 • Implement 2d object recognition based on contour- and/or region-based shape representations [Usage]

Readings: M, V, and B. R (2007), G. R C. and W. R E. (2007)

UNIT 5: Robotics (10)	
Competences:	

Content	Generales Goals
• Multiple-robot coordination	• Compare and contrast at least three strategies for robot navigation within known and/or unknown environments, including their strengths and shortcomings [Familiarity] • Describe at least one approach for coordinating the actions and sensing of several robots to accomplish a single task [Familiarity]

Readings: Stone (2000)

8. Methodology
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. Assessment
Continuous Assessment 1 : 20 % **Partial Exam** : 30 % **Continuous Assessment 2** : 20 % **Final exam** : 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
Undergraduate Program in Information Systems
SILABO

FG211. Professional Ethics (Mandatory)

2022-II

1. General information		
1.1 School	:	Sistemas de Información
1.2 Course	:	FG211. Professional Ethics
1.3 Semester	:	10mo Semestre.
1.4 Prerrequisites	:	None
1.5 Type of course	:	Mandatory
1.6 Learning modality	:	Face to face
1.7 Horas	:	2 HT; 2 HP;
1.8 Credits	:	3

2. Professors

3. Course foundation
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. Summary
1. 2. 3. 4.

5. Generales Goals
• 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.[Usage]

6. Contribution to Outcomes
This discipline contributes to the achievement of the following outcomes:

4) Recognize professional responsabilities and make informed judgments in computing practice based on legal and ethical principles. (**Usage**)

6) Apply computer science theory and software development fundamentals to produce computing-based solutions. (**Usage**)

7) Develop computational technology for the well-being of all, contributing with human formation, scientific, technological and professional skills to solve social problems of our community. (**Usage**)

7. Content

UNIT 1: (12)

Competences:

Content	Generales Goals
- 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.	- Presentar al alumno la importancia de tener principios y valores en la sociedad actual.[Usage] - Presentar algunos de los principios de podrían contribuir en la sociedad de ser aplicados y vividos día a día. [Usage] - Presentar a los alumnos el aporte de la Doctrina Social de la Iglesia en el quehacer profesional. [Usage]

Readings: Computing Machinery (ACM) (1992), Schmidt (1995), Loza (2000), Argandoña (2006)

UNIT 2: (12)

Competences:

Content	Generales Goals
- 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?	- Presentar al alumno el rol de la responsabilidad social individual y del liderazgo en la empresa. [Familiarity] - Conocer el juicio de la ética frente a la corrupción y sobornos como forma de relación laboral. [Familiarity] - Presentar la profesión como una forma de realización personal, y como consecuencia. []

Readings: Computing Machinery (ACM) (1992), Manzone (2007), Schmidt (1995), Pérez López (1998), Nieburh (2003)

UNIT 3: (12)

Competences:

Content	Generales Goals
- 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.	Presentar al alumno las interrelaciones entre ética y las disciplinas de la última era tecnológica.[Familiarity]

Readings: Computing Machinery (ACM) (1992), IEEE (2004), Hernández (2006)

UNIT 4: (12)	
Competences:	
Content	**Generales Goals**

• Ética informática.
   – Ética y software.
   – El software libre.
• Regulación y ética de telecomunicaciones.
   – 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.
	• Presentar al alumno algunos aspectos que confrontan la ética con el quehacer de las disciplinas emergentes en la sociedad de la información.[Familiarity]

Readings: Comunicaciones Sociales (2002), Hernández (2006), Computing Machinery (ACM) (1992)	

8. Methodology
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. Assessment
Continuous Assessment 1 : 20 %   **Partial Exam** : 30 %   **Continuous Assessment 2** : 20 %   **Final exam** : 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.