UNA

UNIVERSIDAD NACIONAL

COSTA RICA

Book of Syllabi

School of Informatics

– 2024-I–

: January 16, 2024

Task Force

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)
Mdmber of the Board of Governors of the IEEE Computer Society (2020-2023)
email: *ecuadros@spc.org.pe*
http://socios.spc.org.pe/ecuadros

Contents

First Semester **5**
1.1 CS100. Computing Foundations 5
1.2 CS111. Computing Foundations 7
1.3 CS1D1. Discrete Structures I 14
1.4 MA100. Mathematics Foundations 19
1.5 EG001. General Studies I 22
1.6 EG002. General Studies II 24

Second Semester **26**
2.1 CS112. Programming I 26
2.2 SE1R1. Requirements and interface design 34
2.3 MA101. Calculus 36
2.4 EG003. General Studies III 40
2.5 EG004. General Studies IV 42

Third Semester **44**
3.1 CS113. Programming II 44
3.2 CS221. Computer Systems Architecture 55
3.3 CS2B1. Platform Based Development I 62
3.4 SE1A1. Software Architecture and Design 66
3.5 MA102. Linear Algebra 68

Fourth Semester **72**
4.1 CS210. Algorithms and Data Structures 72
4.2 CS2S1. Operating systems 75
4.3 SE2C1. Software development 83
4.4 MA203. Statistics and Probabilities 85

Fifth Semester **87**
5.1 CS212. Analysis and Design of Algorithms 87
5.2 CS231. Networking and Communication 91
5.3 CS271. Databases I 95
5.4 CS3I2. Computer Security 101
5.5 SE201. Elective I 103

Sixth Semester **105**
6.1 CS3I1. Computer Security 105
6.2 CS3P1. Parallel and Distributed Computing 115
6.3 ID105. Technical and professional English V 121
6.4 DS371. Topics in Data Science 124
6.5 SE302. Elective II 126

Seventh Semester **128**
7.1 CS341. Programming languages 128
7.2 CS3P2. Cloud Computing 135
7.3 ET201. Entrerpreneurship I 141
7.4 SE403. Elective III 146

Eighth Semester **148**
8.1 CS261. Intelligent Systems 148
8.2 CS2H1. User Experience (UX) 157
8.3 SE3E1. General and Professional Ethics 163
8.4 SE3E2. Supervised professional practice 165
8.5 SE404. Elective IV 167

National University of Copsta Rica (UNA)
School of Informatics
Sillabus 2024-I

1. COURSE

CS100. Computing Foundations (Mandatory)

2. GENERAL INFORMATION

2.1 Course	:	CS100. Computing Foundations
2.2 Semester	:	1^{er} Semestre.
2.3 Credits	:	2
2.4 Horas	:	3 HT;
2.5 Duration of the period	:	16 weeks
2.6 Type of course	:	Mandatory
2.7 Learning modality	:	Face to face
2.8 Prerrequisites	:	None None

3. PROFESSORS

Meetings after coordination with the professor

4. INTRODUCTION TO THE COURSE

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.

5. GOALS

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

6. COMPETENCES

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. TOPICS

Unit 1: (2)	
Competences Expected:	
Topics	**Learning Outcomes**
• Introducción a la computación. • Historia de la computación.	• Encourage students to study Computer Science. [Familiarity]
Readings : [Bro15]	

Unit 2: Lógica básica (2)	
Competences Expected:	
Topics	**Learning Outcomes**
• Lógica proposicional. • Conectores lógicos. • Tablas de verdad. • Forma normal (conjuntiva y disyuntiva)	• Convertir declaraciones lógicas desde el lenguaje informal a expresiones de lógica proposicional y de predicados [Familiarity] • Aplicar métodos formales de simbolismo proposicional y lógica de predicados, como el cálculo de la validez de formulas y cálculo de formas normales [Familiarity]
Readings : [Bro15]	

8. WORKPLAN

8.1 Methodology

Individual and team participation is encouraged to present their ideas, motivating them with additional points in the different stages of the course evaluation.

8.2 Theory Sessions

The theory sessions are held in master classes with activities including active learning and roleplay to allow students to internalize the concepts.

8.3 Practical Sessions

The practical sessions are held in class where a series of exercises and/or practical concepts are developed through problem solving, problem solving, specific exercises and/or in application contexts.

9. EVALUATION SYSTEM

********* EVALUATION MISSING ********

10. BASIC BIBLIOGRAPHY

[Bro15] J. G. Brookshear. *Computer Science: An Overview*. 12th. Addison-Wesley, 2015.

National University of Copsta Rica (UNA)
School of Informatics
Sillabus 2024-I

1. COURSE

CS111. Computing Foundations (Mandatory)

2. GENERAL INFORMATION

2.1 Course	:	CS111. Computing Foundations
2.2 Semester	:	1^{er} Semestre.
2.3 Credits	:	3
2.4 Horas	:	3 HT; 3 HP;
2.5 Duration of the period	:	16 weeks
2.6 Type of course	:	Mandatory
2.7 Learning modality	:	Face to face
2.8 Prerrequisites	:	None None

3. PROFESSORS

Meetings after coordination with the professor

4. INTRODUCTION TO THE COURSE

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.

5. GOALS

- Introduce the fundamental concepts of programming.
- Develop the ability of abstraction using programming language

6. COMPETENCES

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. TOPICS

Unit 1: Historia (5)	
Competences Expected:	
Topics	**Learning Outcomes**
• Pre-historia – El mundo antes de 1946. • Historia del hardware, software, redes. • Pioneros de la Computación. • Historia de Internet.	• Identificar importantes tendencias en la historia del campo de la computación [Familiarity] • Identificar las contribuciones de varios pioneros en el campo de la computación [Familiarity] • Discutir el contexto histórico de los paradigmas de diversos lenguajes de programación [Familiarity] • Comparar la vida diaria antes y después de la llegada de los ordenadores personales y el Internet [Assessment]
Readings : [BB19], [Gut13], [Zel10]	

Unit 2: Sistemas de tipos básicos (2)	
Competences Expected:	
Topics	**Learning Outcomes**
• Tipos como conjunto de valores junto con un conjunto de operaciones. – Tipos primitivos (p.e. números, booleanos) – Composición de tipos construídos de otros tipos (p.e., registros, uniones, arreglos, listas, funciones, referencias) • Asociación de tipos de variables, argumentos, resultados y campos. • Tipo de seguridad y los errores causados por el uso de valores de manera incompatible dadas sus tipos previstos.	• Tanto para tipo primitivo y un tipo compuesto, describir de manera informal los valores que tiene dicho tipo [Familiarity] • Para un lenguaje con sistema de tipos estático, describir las operaciones que están prohibidas de forma estática, como pasar el tipo incorrecto de valor a una función o método [Familiarity] • Describir ejemplos de errores de programa detectadas por un sistema de tipos [Familiarity] • Para múltiples lenguajes de programación, identificar propiedades de un programa con verificación estática y propiedades de un programa con verificación dinámica [Usage] • Usar tipos y mensajes de error de tipos para escribir y depurar programas [Usage] • Definir y usar piezas de programas (tales como, funciones, clases, métodos) que usan tipos genéricos, incluyendo para colecciones [Usage]
Readings : [Gut13], [Zel10]	

Unit 3: Conceptos Fundamentales de Programación (9)	
Competences Expected:	
Topics	**Learning Outcomes**
• Sintaxis y semántica básica de un lenguaje de alto nivel. • Variables y tipos de datos primitivos (ej., numeros, caracteres, booleanos) • Expresiones y asignaciones. • Operaciones básicas I/O incluyendo archivos I/O. • Estructuras de control condicional e iterativas. • Paso de funciones y parámetros. • Concepto de recursividad.	• Analiza y explica el comportamiento de programas simples que involucran estructuras fundamentales de programación variables, expresiones, asignaciones, E/S, estructuras de control, funciones, paso de parámetros, y recursividad [Assessment] • Identifica y describe el uso de tipos de datos primitivos [Familiarity] • Escribe programas que usan tipos de datos primitivos [Usage] • Modifica y expande programas cortos que usen estructuras de control condicionales e iterativas así como funciones [Usage] • Diseña, implementa, prueba, y depura un programa que usa cada una de las siguientes estructuras de datos fundamentales: cálculos básicos, E/S simple, condicional estándar y estructuras iterativas, definición de funciones, y paso de parámetros [Usage] • Escribe un programa que usa E/S de archivos para brindar persistencia a través de ejecuciones múltiples [Usage] • Escoje estructuras de condición y repetición adecuadas para una tarea de programación dada [Familiarity] • Describe el concepto de recursividad y da ejemplos de su uso [Assessment] • Identifica el caso base y el caso general de un problema basado en recursividad [Familiarity]
Readings : [Gut13], [Zel10]	

Unit 4: Análisis Básico (2)	
Competences Expected:	
Topics	**Learning Outcomes**
• Diferencias entre el mejor, el esperado y el peor caso de un algoritmo. • Definición formal de la Notación Big O. • Clases de complejidad como constante, logarítmica, lineal, cuadrática y exponencial. • Uso de la notación Big O. • Análisis de algoritmos iterativos y recursivos.	• Explique a que se refiere con "mejor", "esperado" y "peor" caso de comportamiento de un algoritmo [Familiarity] • En el contexto de a algoritmos específicos, identifique las características de data y/o otras condiciones o suposiciones que lleven a diferentes comportamientos [Familiarity] • Indique la definición formal de Big O [Familiarity] • Use la notación formal de la Big O para dar límites superiores asintóticos en la complejidad de tiempo y espacio de los algoritmos [Usage] • Usar la notación formal Big O para dar límites de casos esperados en el tiempo de complejidad de los algoritmos [Usage]
Readings : [Gut13], [Zel10]	

Unit 5: Algoritmos y Estructuras de Datos fundamentales (8)

Competences Expected:

Topics	Learning Outcomes

• Algoritmos numéricos simples, tales como el cálculo de la media de una lista de números, encontrar el mínimo y máximo.
• Algoritmos de búsqueda secuencial y binaria.
• Algoritmos de ordenamiento de peor caso cuadrático (selección, inserción)
• Algoritmos de ordenamiento con peor caso o caso promedio en O(N lg N) (Quicksort, Heapsort, Mergesort)
• Tablas Hash, incluyendo estratégias para evitar y resolver colisiones.
• Árboles de búsqueda binaria:
  – Operaciones comunes en árboles de búsqueda binaria como seleccionar el mínimo, máximo, insertar, eliminar, recorrido en árboles.
• Grafos y algoritmos en grafos:
  – Representación de grafos (ej., lista de adyacencia, matriz de adyacencia)
  – Recorrido en profundidad y amplitud
• Montículos (Heaps)
• Grafos y algoritmos en grafos:
  – Problema de corte máximo y mínimo
  – Busqueda local
• Búsqueda de patrones y algoritmos de cadenas/texto (ej. búsqueda de subcadena, búsqueda de expresiones regulares, algoritmos de subsecuencia común más larga)
	• Implementar algoritmos numéricos básicos [Usage]
	• Implementar algoritmos de busqueda simple y explicar las diferencias en sus tiempos de complejidad [Assessment]
	• Ser capaz de implementar algoritmos de ordenamiento comunes cuádraticos y O(N log N) [Usage]
	• Describir la implementación de tablas hash, incluyendo resolución y el evitamiento de colisiones [Familiarity]
	• Discutir el tiempo de ejecución y eficiencia de memoria de los principales algoritmos de ordenamiento, busqueda y hashing [Familiarity]
	• Discutir factores otros que no sean eficiencia computacional que influyan en la elección de algoritmos, tales como tiempo de programación, mantenibilidad, y el uso de patrones específicos de la aplicación en los datos de entrada [Familiarity]
	• Explicar como el balanceamiento del arbol afecta la eficiencia de varias operaciones de un arbol de búsqueda binaria [Familiarity]
	• Resolver problemas usando algoritmos básicos de grafos, incluyendo busqueda por profundidad y busqueda por amplitud [Usage]
	• Demostrar habilidad para evaluar algoritmos, para seleccionar de un rango de posibles opciones, para proveer una justificación por esa selección,y para implementar el algoritmo en un contexto en específico [Assessment]
	• Describir la propiedad del heap y el uso de heaps como una implementación de colas de prioridad [Familiarity]
	• Resolver problemas usando algoritmos de grafos, incluyendo camino más corto de una sola fuente y camino más corto de todos los pares, y como mínimo un algoritmo de arbol de expansion minima [Usage]
	• Trazar y/o implementar un algoritmo de comparación de string [Usage]

Readings : [Gut13], [Zel10]

Unit 6: Algoritmos y Diseño (9)	
Competences Expected:	
Topics	**Learning Outcomes**
• Conceptos y propiedades de los algoritmos – Comparación informal de la eficiencia de los algoritmos (ej., conteo de operaciones) • Rol de los algoritmos en el proceso de solución de problemas • Estrategias de solución de problemas – Funciones matemáticas iterativas y recursivas – Recorrido iterativo y recursivo en estructura de datos – Estrategias Divide y Conquistar • Conceptos y principios fundamentales de diseño – Abstracción – Descomposición de Program – Encapsulamiento y camuflaje de información – Separación de comportamiento y aplicación	• Discute la importancia de los algoritmos en el proceso de solución de un problema [Familiarity] • Discute como un problema puede ser resuelto por múltiples algoritmos, cada uno con propiedades diferentes [Familiarity] • Crea algoritmos para resolver problemas simples [Usage] • Usa un lenguaje de programación para implementar, probar, y depurar algoritmos para resolver problemas simples [Usage] • Implementa, prueba, y depura funciones recursivas simples y sus procedimientos [Usage] • Determina si una solución iterativa o recursiva es la más apropiada para un problema [Assessment] • Implementa un algoritmo de divide y vencerás para resolver un problema [Usage] • Aplica técnicas de descomposición para dividir un programa en partes más pequeñas [Usage] • Identifica los componentes de datos y el comportamiento de mútiples tipos de datos abstractos [Usage] • Implementa un tipo de dato abstracto coherente, con la menor pérdida de acoplamiento entre componentes y comportamientos [Usage] • Identifica las fortalezas y las debilidades relativas entre múltiples diseños e implementaciones de un problema [Assessment]
Readings : [Gut13], [Zel10]	

Unit 7: Métodos de Desarrollo (1)	
Competences Expected:	
Topics	**Learning Outcomes**
• Entornos modernos de programación: – Búsqueda de código. – Programación usando libreria de componentes y sus APIs.	• Construir y depurar programas que utilizan las bibliotecas estándar disponibles con un lenguaje de programación elegido [Familiarity]
Readings : [Gut13], [Zel10]	

8. WORKPLAN

8.1 Methodology

Individual and team participation is encouraged to present their ideas, motivating them with additional points in the different stages of the course evaluation.

8.2 Theory Sessions

The theory sessions are held in master classes with activities including active learning and roleplay to allow students to internalize the concepts.

8.3 Practical Sessions

The practical sessions are held in class where a series of exercises and/or practical concepts are developed through problem solving, problem solving, specific exercises and/or in application contexts.

9. EVALUATION SYSTEM

********* EVALUATION MISSING ********

10. BASIC BIBLIOGRAPHY

[BB19] J. Glenn Brookshear and Dennis Brylow. *Computer Science: An Overview*. Ed. by PEARSON. Global Edition. Pearson, 2019. ISBN: 1292263423. URL: `http://www.pearsonhighered.com/brookshear`.

[Gut13] John V Guttag. . *Introduction To Computation And Programming Using Python*. MIT Press, 2013.

[Zel10] John Zelle. *Python Programming: An Introduction to Computer Science*. Franklin, Beedle & Associates Inc, 2010.

National University of Copsta Rica (UNA)
School of Informatics
Sillabus 2024-I

1. COURSE

CS1D1. Discrete Structures I (Mandatory)

2. GENERAL INFORMATION

2.1 Course	:	CS1D1. Discrete Structures I
2.2 Semester	:	1^{er} Semestre.
2.3 Credits	:	3
2.4 Horas	:	2 HT; 2 HP;
2.5 Duration of the period	:	16 weeks
2.6 Type of course	:	Mandatory
2.7 Learning modality	:	Face to face
2.8 Prerrequisites	:	None None

3. PROFESSORS

Meetings after coordination with the professor

4. INTRODUCTION TO THE COURSE

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.

5. 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. COMPETENCES

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. TOPICS

Unit 1: Funciones, relaciones y conjuntos (22)	
Competences Expected:	
Topics	**Learning Outcomes**
• Conjuntos: – Diagramas de Venn – Unión, intersección, complemento – Producto Cartesiano – Potencia de conjuntos – Cardinalidad de Conjuntos finitos • Relations: – Reflexivity, simmetry, transitivity – Equivalence relations – Partial order relations and sets – Extremal elements of a partially ordered sets • Funciones: – Suryecciones, inyecciones, biyecciones – Inversas – Composición	• Explicar con ejemplos la terminología básica de funciones, relaciones y conjuntos [Assessment] • Realizar las operaciones asociadas con conjuntos, funciones y relaciones [Assessment] • Relacionar ejemplos prácticos para conjuntos funciones o modelos de relación apropiados e interpretar la asociación de operaciones y terminología en contexto [Assessment]
Readings : [Gri03], [Ros07], [Vel06]	

Unit 2: Lógica básica (14)	
Competences Expected:	
Topics	**Learning Outcomes**
• Lógica proposicional. • Conectores lógicos. • Tablas de verdad. • Forma normal (conjuntiva y disyuntiva) • Validación de fórmula bien formada. • Reglas de inferencia proposicional (conceptos de modus ponens y modus tollens) • Logica de predicados: – Cuantificación universal y existencial • Limitaciones de la lógica proposicional y de predicados (ej. problemas de expresividad)	• Convertir declaraciones lógicas desde el lenguaje informal a expresiones de lógica proposicional y de predicados [Usage] • Aplicar métodos formales de simbolismo proposicional y lógica de predicados, como el cálculo de la validez de formulas y cálculo de formas normales [Usage] • Usar reglas de inferencia para construir demostraciones en lógica proposicional y de predicados [Usage] • Describir como la lógica simbólica puede ser usada para modelar situaciones o aplicaciones de la vida real, incluidos aquellos planteados en el contexto computacional como análisis de software (ejm. programas correctores), consulta de base de datos y algoritmos [Familiarity] • Aplicar demostraciones de lógica formal y/o informal, pero rigurosa, razonamiento lógico para problemas reales, como la predicción del comportamiento de software o solución de problemas tales como rompecabezas [Usage] • Describir las fortalezas y limitaciones de la lógica proposicional y de predicados [Usage]
Readings : [Ros07], [Gri03], [Vel06]	

Unit 3: Técnicas de demostración (14)	
Competences Expected:	
Topics	**Learning Outcomes**
• Nociones de implicancia, equivalencia, conversión, inversa, contrapositivo, negación, y contradicción • Estructura de pruebas matemáticas. • Demostración directa. • Refutar por contraejemplo. • Demostracción por contradicción. • Inducción sobre números naturales. • Inducción estructural. • Inducción leve y fuerte (Ej. Primer y Segundo principio de la inducción) • Definiciones matemáticas recursivas. • Conjuntos bien ordenados.	• Identificar la técnica de demostración utilizada en una demostración dada [Assessment] • Describir la estructura básica de cada técnica de demostración (demostración directa, demostración por contradicción e inducción) descritas en esta unidad [Usage] • Aplicar las técnicas de demostración (demostración directa, demostración por contradicción e inducción) correctamente en la construcción de un argumento solido [Usage] • Determine que tipo de demostración es la mejor para un problema dado [Assessment] • Explicar el paralelismo entre ideas matemáticas y/o inducción estructural para la recursión y definir estructuras recursivamente [Familiarity] • Explicar la relación entre inducción fuerte y débil y dar ejemplos del apropiado uso de cada uno [Assessment] • Enunciar el principio del buen-orden y su relación con la inducción matemática [Familiarity]
Readings : [Ros07], [**Vel06**], [Sch12], [Vel06]	

Unit 4: Data Representation (10)	
Competences Expected:	
Topics	**Learning Outcomes**
• 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 : [Ros07], [Gri03], [Vel06]	

8. WORKPLAN

8.1 Methodology

Individual and team participation is encouraged to present their ideas, motivating them with additional points in the different stages of the course evaluation.

8.2 Theory Sessions

The theory sessions are held in master classes with activities including active learning and roleplay to allow students to internalize the concepts.

8.3 Practical Sessions

The practical sessions are held in class where a series of exercises and/or practical concepts are developed through problem solving, problem solving, specific exercises and/or in application contexts.

9. EVALUATION SYSTEM

********* EVALUATION MISSING ********

10. BASIC BIBLIOGRAPHY

[Gri03] R. Grimaldi. *Discrete and Combinatorial Mathematics: An Applied Introduction*. 5 ed. Pearson, 2003.

[Ros07] Kenneth H. Rosen. *Discrete Mathematics and Its Applications*. 7 ed. 2007.

[Sch12] Edward R. Scheinerman. *Mathematics: A Discrete Introduction*. 3 ed. 2012.

[Vel06] Daniel J. Velleman. *How to Prove It: A Structured Approach*. Ed. by Cambridge University Pres. 2nd. 2006. ISBN: 978-0521675994.

National University of Copsta Rica (UNA)
School of Informatics
Sillabus 2024-I

1. COURSE

MA100. Mathematics Foundations (Mandatory)

2. GENERAL INFORMATION

2.1 Course	:	MA100. Mathematics Foundations
2.2 Semester	:	1^{er} Semestre.
2.3 Credits	:	4
2.4 Horas	:	4 HT;
2.5 Duration of the period	:	16 weeks
2.6 Type of course	:	Mandatory
2.7 Learning modality	:	Face to face
2.8 Prerrequisites	:	None None

3. PROFESSORS

Meetings after coordination with the professor

4. INTRODUCTION TO THE COURSE

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.

5. GOALS

- .
- .
- .

6. COMPETENCES

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. TOPICS

Unit 1: (20)	
Competences Expected:	
Topics	**Learning Outcomes**
• . • .	• . • .
Readings : [Ste12], [ión14]	

Unit 2: (10)	
Competences Expected:	
Topics	**Learning Outcomes**
• . • . • . • . • . • .	• . • . • . • . • . • .
Readings : [Ste12], [ión14]	

Unit 3: (20)	
Competences Expected:	
Topics	**Learning Outcomes**
• . • . • . • . • .	• . • . • . • . • . • . • . • . • . • . • .
Readings : [Ste12], [ión14]	

Unit 4: (22)	
Competences Expected:	
Topics	**Learning Outcomes**
• . • . • . • .	• . • . • . • . • . • . • . • . • . • . • .
Readings : [Ste12], [ión14]	

8. WORKPLAN

8.1 Methodology

Individual and team participation is encouraged to present their ideas, motivating them with additional points in the different stages of the course evaluation.

8.2 Theory Sessions

The theory sessions are held in master classes with activities including active learning and roleplay to allow students to internalize the concepts.

8.3 Practical Sessions

The practical sessions are held in class where a series of exercises and/or practical concepts are developed through problem solving, problem solving, specific exercises and/or in application contexts.

9. EVALUATION SYSTEM

********* EVALUATION MISSING ********

10. BASIC BIBLIOGRAPHY

[ión14] ROn Larson ión. *Calculus*. 10th. 2014.

[Ste12] James Stewart. *Calculus*. 7th. 2012.

National University of Copsta Rica (UNA)
School of Informatics
Sillabus 2024-I

1. COURSE

EG001. General Studies I (Mandatory)

2. GENERAL INFORMATION

2.1 Course	:	EG001. General Studies I
2.2 Semester	:	1^{er} Semestre.
2.3 Credits	:	3
2.4 Horas	:	4 HT;
2.5 Duration of the period	:	16 weeks
2.6 Type of course	:	Mandatory
2.7 Learning modality	:	Face to face
2.8 Prerrequisites	:	None None

3. PROFESSORS

Meetings after coordination with the professor

4. INTRODUCTION TO THE COURSE

Write justification for this course here ...

5. GOALS

- Write your first goal here.
- Write your second goal here.
- Just in case you need more goals write them here

6. COMPETENCES

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

7. TOPICS

Unit 1: title for the unit goes here (5)	
Competences Expected:	
Topics	**Learning Outcomes**
• Topic1 • Topic2 • Topic3	• Learning outcome1 [Levelforthislearningoutcome]. • Apply computing in complex problems [Usage]. • Create a search engine [Assessment]. • Study data structures [Familiarity].
Readings : [**Bibitem1**], [**Bibitem2**]	

Unit 2: another unit goes here (1)	
Competences Expected:	
Topics	**Learning Outcomes**
• Topic1	• Learning outcome xyz [Levelforthislearningoutcome].
Readings : [**Bibitem3**], [**Bibitem1**]	

8. WORKPLAN

8.1 Methodology

Individual and team participation is encouraged to present their ideas, motivating them with additional points in the different stages of the course evaluation.

8.2 Theory Sessions

The theory sessions are held in master classes with activities including active learning and roleplay to allow students to internalize the concepts.

8.3 Practical Sessions

The practical sessions are held in class where a series of exercises and/or practical concepts are developed through problem solving, problem solving, specific exercises and/or in application contexts.

9. EVALUATION SYSTEM

********* EVALUATION MISSING ********

10. BASIC BIBLIOGRAPHY

National University of Copsta Rica (UNA)
School of Informatics
Sillabus 2024-I

1. COURSE

EG002. General Studies II (Mandatory)

2. GENERAL INFORMATION

2.1 Course	:	EG002. General Studies II
2.2 Semester	:	1^{er} Semestre.
2.3 Credits	:	3
2.4 Horas	:	4 HT;
2.5 Duration of the period	:	16 weeks
2.6 Type of course	:	Mandatory
2.7 Learning modality	:	Face to face
2.8 Prerrequisites	:	None None

3. PROFESSORS

Meetings after coordination with the professor

4. INTRODUCTION TO THE COURSE

Write justification for this course here ...

5. GOALS

- Write your first goal here.
- Write your second goal here.
- Just in case you need more goals write them here

6. COMPETENCES

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

7. TOPICS

Unit 1: title for the unit goes here (5)	
Competences Expected:	
Topics	**Learning Outcomes**
• Topic1 • Topic2 • Topic3	• Learning outcome1 [Levelforthislearningoutcome]. • Apply computing in complex problems [Usage]. • Create a search engine [Assessment]. • Study data structures [Familiarity].
Readings : [**Bibitem1**], [**Bibitem2**]	

Unit 2: another unit goes here (1)	
Competences Expected:	
Topics	**Learning Outcomes**
• Topic1	• Learning outcome xyz [Levelforthislearningoutcome].
Readings : [**Bibitem3**], [**Bibitem1**]	

8. WORKPLAN

8.1 Methodology

Individual and team participation is encouraged to present their ideas, motivating them with additional points in the different stages of the course evaluation.

8.2 Theory Sessions

The theory sessions are held in master classes with activities including active learning and roleplay to allow students to internalize the concepts.

8.3 Practical Sessions

The practical sessions are held in class where a series of exercises and/or practical concepts are developed through problem solving, problem solving, specific exercises and/or in application contexts.

9. EVALUATION SYSTEM

********* EVALUATION MISSING ********

10. BASIC BIBLIOGRAPHY

National University of Copsta Rica (UNA)
School of Informatics
Sillabus 2024-I

1. COURSE

CS112. Programming I (Mandatory)

2. GENERAL INFORMATION

2.1 Course	:	CS112. Programming I
2.2 Semester	:	2^{do} Semestre.
2.3 Credits	:	4
2.4 Horas	:	3 HT; 3 HP;
2.5 Duration of the period	:	16 weeks
2.6 Type of course	:	Mandatory
2.7 Learning modality	:	Face to face
2.8 Prerrequisites	:	CS100. Computing Foundations. (1^{st} Sem) CS100. Computing Foundations. (1^{st} Sem)

3. PROFESSORS

Meetings after coordination with the professor

4. INTRODUCTION TO THE COURSE

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.

5. GOALS

- Introduce the student to the foundations of the object orientation paradigm, allowing the assimilation of concepts necessary to develop information systems.

6. COMPETENCES

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. TOPICS

Unit 1: General overwiew of Programming Languages (1)	
Competences Expected:	
Topics	**Learning Outcomes**
• Brief review of programming paradigms. • Comparison between functional programming and imperative programming. • History of programming languages.	• Discutir el contexto histórico de los paradigmas de diversos lenguajes de programación [Familiarity]
Readings : [**Stroustrup2013**], [**Deitel17**]	

Unit 2: Máquinas virtuales (1)	
Competences Expected:	
Topics	**Learning Outcomes**
• The virtual machine concept. • Tipos de virtualización (incluyendo Hardware / Software, OS, Servidor, Servicio, Red) . • Intermediate languages.	• Explicar el concepto de memoria virtual y la forma cómo se realiza en hadware y software [Familiarity] • Diferenciar emulacion y el aislamiento [Familiarity] • Evaluar virtualización de compensaciones [Assessment]
Readings : [**Stroustrup2013**], [**Deitel17**]	

Unit 3: Sistemas de tipos básicos (2)	
Competences Expected:	
Topics	**Learning Outcomes**

• Tipos como conjunto de valores junto con un conjunto de operaciones.
  – Tipos primitivos (p.e. números, booleanos)
  – Composición de tipos construídos de otros tipos (p.e., registros, uniones, arreglos, listas, funciones, referencias)
• Model statement (link, visibility, scope and life time).
• General view of type checking.
	• Tanto para tipo primitivo y un tipo compuesto, describir de manera informal los valores que tiene dicho tipo [Familiarity]
	• Para un lenguaje con sistema de tipos estático, describir las operaciones que están prohibidas de forma estática, como pasar el tipo incorrecto de valor a una función o método [Familiarity]
	• Describir ejemplos de errores de programa detectadas por un sistema de tipos [Familiarity]
	• Para múltiples lenguajes de programación, identificar propiedades de un programa con verificación estática y propiedades de un programa con verificación dinámica [Usage]
	• Dar un ejemplo de un programa que no verifique tipos en un lenguaje particular y sin embargo no tenga error cuando es ejecutado [Familiarity]
	• Usar tipos y mensajes de error de tipos para escribir y depurar programas [Usage]
	• Explicar como las reglas de tipificación definen el conjunto de operaciones que legales para un tipo [Familiarity]
	• Escribir las reglas de tipo que rigen el uso de un particular tipo compuesto [Usage]
	• Explicar por qué indecidibilidad requiere sistemas de tipo para conservadoramente aproximar el comportamiento de un programa [Familiarity]
	• Definir y usar piezas de programas (tales como, funciones, clases, métodos) que usan tipos genéricos, incluyendo para colecciones [Usage]
	• Discutir las diferencias entre, genéricos (*generics*), subtipo y sobrecarga [Familiarity]
	• Explicar múltiples beneficios y limitaciones de tipificación estática en escritura, mantenimiento y depuración de un software [Familiarity]

Readings : [**Stroustrup2013**], [**Deitel17**]	

Unit 4: Conceptos Fundamentales de Programación (6)	
Competences Expected:	
Topics	**Learning Outcomes**
• Sintaxis y semántica básica de un lenguaje de alto nivel. • Variables y tipos de datos primitivos (ej., numeros, caracteres, booleanos) • Expresiones y asignaciones. • Operaciones básicas I/O incluyendo archivos I/O. • Estructuras de control condicional e iterativas. • Paso de funciones y parámetros.	• Analiza y explica el comportamiento de programas simples que involucran estructuras fundamentales de programación variables, expresiones, asignaciones, E/S, estructuras de control, funciones, paso de parámetros, y recursividad [Assessment] • Identifica y describe el uso de tipos de datos primitivos [Familiarity] • Escribe programas que usan tipos de datos primitivos [Usage] • Modifica y expande programas cortos que usen estructuras de control condicionales e iterativas así como funciones [Usage] • Diseña, implementa, prueba, y depura un programa que usa cada una de las siguientes estructuras de datos fundamentales: cálculos básicos, E/S simple, condicional estándar y estructuras iterativas, definición de funciones, y paso de parámetros [Usage] • Escribe un programa que usa E/S de archivos para brindar persistencia a través de ejecuciones múltiples [Usage] • Escoje estructuras de condición y repetición adecuadas para una tarea de programación dada [Assessment] • Describe el concepto de recursividad y da ejemplos de su uso [Familiarity] • Identifica el caso base y el caso general de un problema basado en recursividad [Assessment]
Readings : [**Stroustrup2013**], [**Deitel17**]	

Unit 5: Programación orientada a objetos (10)

Competences Expected:

Topics	**Learning Outcomes**
• Diseño orientado a objetos: – Descomposicion en objetos que almacenan estados y poseen comportamiento – Diseño basado en jerarquia de clases para modelamiento • Lenguajes orientados a objetos para la encapsulación: – privacidad y la visibilidad de miembros de la clase – Interfaces revelan único método de firmas – clases base abstractas • Definición de las categorías, campos, métodos y constructores. • Las subclases, herencia y método de alteración temporal. • Subtipificación: – Polimorfismo artículo Subtipo; upcasts implícitos en lenguajes con tipos. – Noción de reemplazo de comportamiento: los subtipos de actuar como supertipos. – Relación entre subtipos y la herencia. • Uso de coleccion de clases, iteradores, y otros componentes de la libreria estandar. • Asignación dinámica: definición de método de llamada.	• Diseñar e implementar una clase [Usage] • Usar subclase para diseñar una jerarquía simple de clases que permita al código ser reusable por diferentes subclases [Usage] • Razonar correctamente sobre el flujo de control en un programa mediante el envío dinámico [Usage] • Comparar y contrastar (1) el enfoque procedurar/funcional- definiendo una función por cada operación con el cuerdo de la función proporcionando un caso por cada variación de dato - y (2) el enfoque orientado a objetos - definiendo una clase por cada variación de dato con la definición de la clase proporcionando un método por cada operación. Entender ambos enfoques como una definición de variaciones y operaciones de una matriz [Assessment] • Explicar la relación entre la herencia orientada a objetos (codigo compartido y *overriding*) y subtipificación (la idea de un subtipo es ser utilizable en un contexto en el que espera al supertipo) [Familiarity] • Usar mecanismos de encapsulación orientada a objetos, tal como interfaces y miembros privados [Usage] • Definir y usar iteradores y otras operaciones sobre agregaciones, incluyendo operaciones que tienen funciones como argumentos, en múltiples lenguajes de programación, selecionar la forma mas natural por cada lenguaje [Usage]
Readings : [**Stroustrup2013**], [**Deitel17**]	

Unit 6: Algoritmos y Diseño (3)	
Competences Expected:	
Topics	**Learning Outcomes**
• Estrategias de solución de problemas – Funciones matemáticas iterativas y recursivas – Recorrido iterativo y recursivo en estructura de datos – Estrategias Divide y Conquistar • Rol de los algoritmos en el proceso de solución de problemas • Estrategias de solución de problemas – Funciones matemáticas iterativas y recursivas – Recorrido iterativo y recursivo en estructura de datos – Estrategias Divide y Conquistar • Conceptos y principios fundamentales de diseño – Abstracción – Descomposición de Program – Encapsulamiento y camuflaje de información – Separación de comportamiento y aplicación	• Discute la importancia de los algoritmos en el proceso de solución de un problema [Familiarity] • Discute como un problema puede ser resuelto por múltiples algoritmos, cada uno con propiedades diferentes [Familiarity] • Crea algoritmos para resolver problemas simples [Usage] • Usa un lenguaje de programación para implementar, probar, y depurar algoritmos para resolver problemas simples [Usage] • Implementa, prueba, y depura funciones recursivas simples y sus procedimientos [Usage] • Determina si una solución iterativa o recursiva es la más apropiada para un problema [Assessment] • Implementa un algoritmo de divide y vencerás para resolver un problema [Usage] • Aplica técnicas de descomposición para dividir un programa en partes más pequeñas [Usage] • Identifica los componentes de datos y el comportamiento de mútiples tipos de datos abstractos [Usage] • Implementa un tipo de dato abstracto coherente, con la menor pérdida de acoplamiento entre componentes y comportamientos [Usage] • Identifica las fortalezas y las debilidades relativas entre múltiples diseños e implementaciones de un problema [Assessment]
Readings : [**Stroustrup2013**], [**Deitel17**]	

Unit 7: Estrategias Algorítmicas (3)	
Competences Expected:	
Topics	**Learning Outcomes**
• Algoritmos de fuerza bruta. • Algoritmos voraces. • Divide y vencerás. • Bactraking recursivo. • Programación Dinámica.	• Para cada una de las estrategias (fuerza bruta, algoritmo goloso, divide y vencerás, recursividad en reversa y programación dinámica), identifica un ejemplo práctico en el cual se pueda aplicar [Familiarity] • Utiliza un enfoque voraz para resolver un problema específico y determina si la regla escogida lo guía a una solución óptima [Assessment] • Usa un algoritmo de divide-y-vencerás para resolver un determinado problema [Usage] • Usa recursividad en reversa a fin de resover un problema como en el caso de recorrer un laberinto [Usage] • Usa programación dinámica para resolver un problema determinado [Usage] • Determina el enfoque algorítmico adecuado para un problema [Assessment] • Describe varios métodos basados en heurísticas para resolver problemas [Familiarity]
Readings : [**Stroustrup2013**], [**Deitel17**]	

Unit 8: Análisis Básico (2)	
Competences Expected:	
Topics	**Learning Outcomes**
• Diferencias entre el mejor, el esperado y el peor caso de un algoritmo.	• Explique a que se refiere con “mejor”, “esperado” y “peor” caso de comportamiento de un algoritmo [Familiarity]
Readings : [**Stroustrup2013**], [**Deitel17**]	

Unit 9: Algoritmos y Estructuras de Datos fundamentales (6)	
Competences Expected:	
Topics	**Learning Outcomes**
• Algoritmos numéricos simples, tales como el cálculo de la media de una lista de números, encontrar el mínimo y máximo. • Algoritmos de búsqueda secuencial y binaria. • Algoritmos de ordenamiento de peor caso cuadrático (selección, inserción) • Algoritmos de ordenamiento con peor caso o caso promedio en O(N lg N) (Quicksort, Heapsort, Mergesort)	• Implementar algoritmos numéricos básicos [Usage] • Implementar algoritmos de busqueda simple y explicar las diferencias en sus tiempos de complejidad [Assessment] • Ser capaz de implementar algoritmos de ordenamiento comunes cuádraticos y O(N log N) [Usage] • Discutir el tiempo de ejecución y eficiencia de memoria de los principales algoritmos de ordenamiento, busqueda y hashing [Familiarity] • Discutir factores otros que no sean eficiencia computacional que influyan en la elección de algoritmos, tales como tiempo de programación, mantenibilidad, y el uso de patrones específicos de la aplicación en los datos de entrada [Familiarity] • Explicar como el balanceamiento del arbol afecta la eficiencia de varias operaciones de un arbol de búsqueda binaria [Familiarity] • Demostrar habilidad para evaluar algoritmos, para seleccionar de un rango de posibles opciones, para proveer una justificación por esa selección,y para implementar el algoritmo en un contexto en específico [Assessment] • Trazar y/o implementar un algoritmo de comparación de string [Usage]
Readings : [**Stroustrup2013**], [**Deitel17**]	

8. WORKPLAN

8.1 Methodology

Individual and team participation is encouraged to present their ideas, motivating them with additional points in the different stages of the course evaluation.

8.2 Theory Sessions

The theory sessions are held in master classes with activities including active learning and roleplay to allow students to internalize the concepts.

8.3 Practical Sessions

The practical sessions are held in class where a series of exercises and/or practical concepts are developed through problem solving, problem solving, specific exercises and/or in application contexts.

9. EVALUATION SYSTEM

********* EVALUATION MISSING ********

10. BASIC BIBLIOGRAPHY

National University of Copsta Rica (UNA)
School of Informatics
Sillabus 2024-I

1. COURSE

SE1R1. Requirements and interface design (Mandatory)

2. GENERAL INFORMATION

2.1 Course	:	SE1R1. Requirements and interface design
2.2 Semester	:	2^{do} Semestre.
2.3 Credits	:	4
2.4 Horas	:	3 HT; 3 HP;
2.5 Duration of the period	:	16 weeks
2.6 Type of course	:	Mandatory
2.7 Learning modality	:	Face to face
2.8 Prerrequisites	:	CS111. Computing Foundations. (1^{st} Sem) CS111. Computing Foundations. (1^{st} Sem)

3. PROFESSORS

Meetings after coordination with the professor

4. INTRODUCTION TO THE COURSE

Write justification for this course here ...

5. GOALS

- Write your first goal here.
- Write your second goal here.
- Just in case you need more goals write them here

6. COMPETENCES

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

7. TOPICS

Unit 1: title for the unit goes here (5)	
Competences Expected:	
Topics	**Learning Outcomes**
• Topic1 • Topic2 • Topic3	• Learning outcome1 [Levelforthislearningoutcome]. • Apply computing in complex problems [Usage]. • Create a search engine [Assessment]. • Study data structures [Familiarity].
Readings : [**Bibitem1**], [**Bibitem2**]	

Unit 2: another unit goes here (1)	
Competences Expected:	
Topics	**Learning Outcomes**
• Topic1	• Learning outcome xyz [Levelforthislearningoutcome].
Readings : [**Bibitem3**], [**Bibitem1**]	

8. WORKPLAN

8.1 Methodology

Individual and team participation is encouraged to present their ideas, motivating them with additional points in the different stages of the course evaluation.

8.2 Theory Sessions

The theory sessions are held in master classes with activities including active learning and roleplay to allow students to internalize the concepts.

8.3 Practical Sessions

The practical sessions are held in class where a series of exercises and/or practical concepts are developed through problem solving, problem solving, specific exercises and/or in application contexts.

9. EVALUATION SYSTEM

********* EVALUATION MISSING ********

10. BASIC BIBLIOGRAPHY

National University of Copsta Rica (UNA)
School of Informatics
Sillabus 2024-I

1. COURSE

MA101. Calculus (Mandatory)

2. GENERAL INFORMATION

2.1 Course	:	MA101. Calculus
2.2 Semester	:	2^{do} Semestre.
2.3 Credits	:	4
2.4 Horas	:	4 HT;
2.5 Duration of the period	:	16 weeks
2.6 Type of course	:	Mandatory
2.7 Learning modality	:	Face to face
2.8 Prerrequisites	:	MA100. Mathematics Foundations. (1^{st} Sem)

MA100. Mathematics Foundations. (1^{st} Sem)

3. PROFESSORS

Meetings after coordination with the professor

4. INTRODUCTION TO THE COURSE

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.

5. 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. COMPETENCES

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. TOPICS

Unit 1: Multi-Variable Function Differential (24)	
Competences Expected:	
Topics	**Learning Outcomes**
• 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 : [Ste12], [Zil13]	

Unit 2: Multi-Variable function Integral (12)	
Competences Expected:	
Topics	**Learning Outcomes**
• 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 : [Ste12], [Zil13]	

Unit 3: Series (24)	
Competences Expected:	
Topics	**Learning Outcomes**
• 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 : [Ste12], [Zil13]	

Unit 4: Ordinary Differential Equations (30)	
Competences Expected:	
Topics	**Learning Outcomes**
• 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 : [Ste12], [Zil13]	

8. WORKPLAN

8.1 Methodology

Individual and team participation is encouraged to present their ideas, motivating them with additional points in the different stages of the course evaluation.

8.2 Theory Sessions

The theory sessions are held in master classes with activities including active learning and roleplay to allow students to internalize the concepts.

8.3 Practical Sessions

The practical sessions are held in class where a series of exercises and/or practical concepts are developed through problem solving, problem solving, specific exercises and/or in application contexts.

9. EVALUATION SYSTEM

********* EVALUATION MISSING ********

10. BASIC BIBLIOGRAPHY

[Ste12] James Stewart. *Calculus*. 7th. CENGAGE Learning, 2012.

[Zil13] Dennis G. Zill. *Differential equations with Boundary value problems*. 8th. CENGAGE Learning, 2013.

National University of Copsta Rica (UNA)
School of Informatics
Sillabus 2024-I

1. COURSE

EG003. General Studies III (Mandatory)

2. GENERAL INFORMATION

2.1 Course	:	EG003. General Studies III
2.2 Semester	:	2^{do} Semestre.
2.3 Credits	:	3
2.4 Horas	:	4 HT;
2.5 Duration of the period	:	16 weeks
2.6 Type of course	:	Mandatory
2.7 Learning modality	:	Face to face
2.8 Prerrequisites	:	None None

3. PROFESSORS

Meetings after coordination with the professor

4. INTRODUCTION TO THE COURSE

Write justification for this course here ...

5. GOALS

- Write your first goal here.
- Write your second goal here.
- Just in case you need more goals write them here

6. COMPETENCES

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

7. TOPICS

Unit 1: title for the unit goes here (5)	
Competences Expected:	
Topics	**Learning Outcomes**
• Topic1 • Topic2 • Topic3	• Learning outcome1 [Levelforthislearningoutcome]. • Apply computing in complex problems [Usage]. • Create a search engine [Assessment]. • Study data structures [Familiarity].
Readings : [**Bibitem1**], [**Bibitem2**]	

Unit 2: another unit goes here (1)	
Competences Expected:	
Topics	**Learning Outcomes**
• Topic1	• Learning outcome xyz [Levelforthislearningoutcome].
Readings : [**Bibitem3**], [**Bibitem1**]	

8. WORKPLAN

8.1 Methodology

Individual and team participation is encouraged to present their ideas, motivating them with additional points in the different stages of the course evaluation.

8.2 Theory Sessions

The theory sessions are held in master classes with activities including active learning and roleplay to allow students to internalize the concepts.

8.3 Practical Sessions

The practical sessions are held in class where a series of exercises and/or practical concepts are developed through problem solving, problem solving, specific exercises and/or in application contexts.

9. EVALUATION SYSTEM

********* EVALUATION MISSING ********

10. BASIC BIBLIOGRAPHY

National University of Copsta Rica (UNA)
School of Informatics
Sillabus 2024-I

1. COURSE

EG004. General Studies IV (Mandatory)

2. GENERAL INFORMATION

2.1 Course	:	EG004. General Studies IV
2.2 Semester	:	2^{do} Semestre.
2.3 Credits	:	3
2.4 Horas	:	4 HT;
2.5 Duration of the period	:	16 weeks
2.6 Type of course	:	Mandatory
2.7 Learning modality	:	Face to face
2.8 Prerrequisites	:	None None

3. PROFESSORS

Meetings after coordination with the professor

4. INTRODUCTION TO THE COURSE

Write justification for this course here ...

5. GOALS

- Write your first goal here.
- Write your second goal here.
- Just in case you need more goals write them here

6. COMPETENCES

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

7. TOPICS

Unit 1: title for the unit goes here (5)	
Competences Expected:	
Topics	**Learning Outcomes**
• Topic1 • Topic2 • Topic3	• Learning outcome1 [Levelforthislearningoutcome]. • Apply computing in complex problems [Usage]. • Create a search engine [Assessment]. • Study data structures [Familiarity].
Readings : [**Bibitem1**], [**Bibitem2**]	

Unit 2: another unit goes here (1)	
Competences Expected:	
Topics	**Learning Outcomes**
• Topic1	• Learning outcome xyz [Levelforthislearningoutcome].
Readings : [**Bibitem3**], [**Bibitem1**]	

8. WORKPLAN

8.1 Methodology

Individual and team participation is encouraged to present their ideas, motivating them with additional points in the different stages of the course evaluation.

8.2 Theory Sessions

The theory sessions are held in master classes with activities including active learning and roleplay to allow students to internalize the concepts.

8.3 Practical Sessions

The practical sessions are held in class where a series of exercises and/or practical concepts are developed through problem solving, problem solving, specific exercises and/or in application contexts.

9. EVALUATION SYSTEM

********* EVALUATION MISSING ********

10. BASIC BIBLIOGRAPHY

National University of Copsta Rica (UNA)
School of Informatics
Sillabus 2024-I

1. COURSE

CS113. Programming II (Mandatory)

2. GENERAL INFORMATION

2.1 Course	:	CS113. Programming II
2.2 Semester	:	3^{er} Semestre.
2.3 Credits	:	4
2.4 Horas	:	3 HT; 3 HP;
2.5 Duration of the period	:	16 weeks
2.6 Type of course	:	Mandatory
2.7 Learning modality	:	Face to face
2.8 Prerrequisites	:	CS112. Programming I. (2^{nd} Sem) CS112. Programming I. (2^{nd} Sem)

3. PROFESSORS

Meetings after coordination with the professor

4. INTRODUCTION TO THE COURSE

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.

5. 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. COMPETENCES

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. TOPICS

Unit 1: Conceptos Fundamentales de Programación (5)	
Competences Expected:	
Topics	**Learning Outcomes**
• Sintaxis y semántica básica de un lenguaje de alto nivel. • Variables y tipos de datos primitivos (ej., numeros, caracteres, booleanos) • Expresiones y asignaciones. • Operaciones básicas I/O incluyendo archivos I/O. • Estructuras de control condicional e iterativas. • Paso de funciones y parámetros. • Concepto de recursividad.	• Analiza y explica el comportamiento de programas simples que involucran estructuras fundamentales de programación variables, expresiones, asignaciones, E/S, estructuras de control, funciones, paso de parámetros, y recursividad [Usage] • Identifica y describe el uso de tipos de datos primitivos [Usage] • Escribe programas que usan tipos de datos primitivos [Usage] • Modifica y expande programas cortos que usen estructuras de control condicionales e iterativas así como funciones [Usage] • Diseña, implementa, prueba, y depura un programa que usa cada una de las siguientes estructuras de datos fundamentales: cálculos básicos, E/S simple, condicional estándar y estructuras iterativas, definición de funciones, y paso de parámetros [Usage] • Escribe un programa que usa E/S de archivos para brindar persistencia a través de ejecuciones múltiples [Usage] • Escoje estructuras de condición y repetición adecuadas para una tarea de programación dada [Usage] • Describe el concepto de recursividad y da ejemplos de su uso [Usage] • Identifica el caso base y el caso general de un problema basado en recursividad [Usage]
Readings : [**stroustrup2013**], [Van02], [LE13]	

Unit 2: Programación orientada a objetos (7)

Competences Expected:

Topics	**Learning Outcomes**
• Diseño orientado a objetos: – Descomposicion en objetos que almacenan estados y poseen comportamiento – Diseño basado en jerarquia de clases para modelamiento • Definición de las categorías, campos, métodos y constructores. • Las subclases, herencia y método de alteración temporal. • Asignación dinámica: definición de método de llamada. • Subtipificación: – Polimorfismo artículo Subtipo; upcasts implícitos en lenguajes con tipos. – Noción de reemplazo de comportamiento: los subtipos de actuar como supertipos. – Relación entre subtipos y la herencia. • Lenguajes orientados a objetos para la encapsulación: – privacidad y la visibilidad de miembros de la clase – Interfaces revelan único método de firmas – clases base abstractas • Uso de coleccion de clases, iteradores, y otros componentes de la libreria estandar.	• Diseñar e implementar una clase [Usage] • Usar subclase para diseñar una jerarquía simple de clases que permita al código ser reusable por diferentes subclases [Usage] • Razonar correctamente sobre el flujo de control en un programa mediante el envío dinámico [Usage] • Comparar y contrastar (1) el enfoque procedurar/funcional- definiendo una función por cada operación con el cuerdo de la función proporcionando un caso por cada variación de dato - y (2) el enfoque orientado a objetos - definiendo una clase por cada variación de dato con la definición de la clase proporcionando un método por cada operación. Entender ambos enfoques como una definición de variaciones y operaciones de una matriz [Usage] • Explicar la relación entre la herencia orientada a objetos (codigo compartido y *overriding*) y subtipificación (la idea de un subtipo es ser utilizable en un contexto en el que espera al supertipo) [Usage] • Usar mecanismos de encapsulación orientada a objetos, tal como interfaces y miembros privados [Usage] • Definir y usar iteradores y otras operaciones sobre agregaciones, incluyendo operaciones que tienen funciones como argumentos, en múltiples lenguajes de programación, selecionar la forma mas natural por cada lenguaje [Usage]
Readings : [**stroustrup2013**]	

Unit 3: Algoritmos y Diseño (5)	
Competences Expected:	
Topics	**Learning Outcomes**

• Conceptos y propiedades de los algoritmos
  – Comparación informal de la eficiencia de los algoritmos (ej., conteo de operaciones)
• Rol de los algoritmos en el proceso de solución de problemas
• Estrategias de solución de problemas
  – Funciones matemáticas iterativas y recursivas
  – Recorrido iterativo y recursivo en estructura de datos
  – Estrategias Divide y Conquistar
• Conceptos y principios fundamentales de diseño
  – Abstracción
  – Descomposición de Program
  – Encapsulamiento y camuflaje de información
  – Separación de comportamiento y aplicación
	• Discute la importancia de los algoritmos en el proceso de solución de un problema [Usage]
	• Discute como un problema puede ser resuelto por múltiples algoritmos, cada uno con propiedades diferentes [Usage]
	• Crea algoritmos para resolver problemas simples [Usage]
	• Usa un lenguaje de programación para implementar, probar, y depurar algoritmos para resolver problemas simples [Usage]
	• Implementa, prueba, y depura funciones recursivas simples y sus procedimientos [Usage]
	• Determina si una solución iterativa o recursiva es la más apropiada para un problema [Usage]
	• Implementa un algoritmo de divide y vencerás para resolver un problema [Usage]
	• Aplica técnicas de descomposición para dividir un programa en partes más pequeñas [Usage]
	• Identifica los componentes de datos y el comportamiento de mútiples tipos de datos abstractos [Usage]
	• Implementa un tipo de dato abstracto coherente, con la menor pérdida de acoplamiento entre componentes y comportamientos [Usage]
	• Identifica las fortalezas y las debilidades relativas entre múltiples diseños e implementaciones de un problema [Usage]

Readings : [**stroustrup2013**], [**Weert16**], [LE13]	

Unit 4: Análisis Básico (3)	
Competences Expected:	
Topics	**Learning Outcomes**
• Diferencias entre el mejor, el esperado y el peor caso de un algoritmo. • Análisis asintótico de complejidad de cotas superior y esperada. • Definición formal de la Notación Big O. • Clases de complejidad como constante, logarítmica, lineal, cuadrática y exponencial. • Medidas empíricas de desempeño. • Compensación entre espacio y tiempo en los algoritmos. • Uso de la notación Big O. • Notación Little o, Big omega y Big theta. • Relaciones recurrentes. • Análisis de algoritmos iterativos y recursivos. • Teorema Maestro y Árboles Recursivos.	• Explique a que se refiere con "mejor", "esperado" y "peor" caso de comportamiento de un algoritmo [Usage] • En el contexto de a algoritmos específicos, identifique las características de data y/o otras condiciones o suposiciones que lleven a diferentes comportamientos [Usage] • Determine informalmente el tiempo y el espacio de complejidad de diferentes algoritmos [Usage] • Indique la definición formal de Big O [Usage] • Lista y contraste de clases estándares de complejidad [Usage] • Realizar estúdios empíricos para validar una hipótesis sobre runtime stemming desde un análisis matemático Ejecute algoritmos con entrada de varios tamaños y compare el desempeño [Usage] • Da ejemplos que ilustran las compensaciones entre espacio y tiempo que se dan en los algoritmos [Usage] • Use la notación formal de la Big O para dar límites superiores asintóticos en la complejidad de tiempo y espacio de los algoritmos [Usage] • Usar la notación formal Big O para dar límites de casos esperados en el tiempo de complejidad de los algoritmos [Usage] • Explicar el uso de la notación theta grande, omega grande y o pequeña para describir la cantidad de trabajo hecho por un algoritmo [Usage] • Usar relaciones recurrentes para determinar el tiempo de complejidad de algoritmos recursivamente definidos [Usage] • Resuelve relaciones de recurrencia básicas, por ejemplo. usando alguna forma del Teorema Maestro [Usage]
Readings : [**stroustrup2013**]	

Unit 5: Sistemas de tipos básicos (5)

Competences Expected:

Topics	**Learning Outcomes**

• Tipos como conjunto de valores junto con un conjunto de operaciones.
  – Tipos primitivos (p.e. números, booleanos)
  – Composición de tipos construídos de otros tipos (p.e., registros, uniones, arreglos, listas, funciones, referencias)
• Asociación de tipos de variables, argumentos, resultados y campos.
• Tipo de seguridad y los errores causados por el uso de valores de manera incompatible dadas sus tipos previstos.
• Metas y limitaciones de tipos estáticos
  – Eliminación de algunas clases de errores sin ejecutar el programa
  – Indecisión significa que un análisis estatico puede aproximar el comportamiento de un programa
• Tipos genéricos (polimorfismo paramétrico)
  – Definición
  – Uso de librerías genéricas tales como colecciones.
  – Comparación con polimorfismo ad-hoc y polimorfismo de subtipos
• Beneficios complementarios de tipos estáticos y dinámicos:
  – Errores tempranos vs. errores tardíos/evitados.
  – Refuerzo invariante durante el desarrollo y mantenimiento del código vs. decisiones pospuestas de tipos durante la la creación de prototipos y permitir convenientemente la codificación flexible de patrones tales como colecciones heterogéneas.
  – Evitar el mal uso del código vs. permitir más reuso de código.
  – Detectar programas incompletos vs. permitir que programas incompletos se ejecuten
	• Tanto para tipo primitivo y un tipo compuesto, describir de manera informal los valores que tiene dicho tipo [Usage]
	• Para un lenguaje con sistema de tipos estático, describir las operaciones que están prohibidas de forma estática, como pasar el tipo incorrecto de valor a una función o método [Usage]
	• Describir ejemplos de errores de programa detectadas por un sistema de tipos [Usage]
	• Para múltiples lenguajes de programación, identificar propiedades de un programa con verificación estática y propiedades de un programa con verificación dinámica [Usage]
	• Dar un ejemplo de un programa que no verifique tipos en un lenguaje particular y sin embargo no tenga error cuando es ejecutado [Usage]
	• Usar tipos y mensajes de error de tipos para escribir y depurar programas [Usage]
	• Explicar como las reglas de tipificación definen el conjunto de operaciones que legales para un tipo [Usage]
	• Escribir las reglas de tipo que rigen el uso de un particular tipo compuesto [Usage]
	• Explicar por qué indecidibilidad requiere sistemas de tipo para conservadoramente aproximar el comportamiento de un programa [Usage]
	• Definir y usar piezas de programas (tales como, funciones, clases, métodos) que usan tipos genéricos, incluyendo para colecciones [Usage]
	• Discutir las diferencias entre, genéricos (*generics*), subtipo y sobrecarga [Usage]
	• Explicar múltiples beneficios y limitaciones de tipificación estática en escritura, mantenimiento y depuración de un software [Usage]

Readings : [**stroustrup2013**]	

Unit 6: Algoritmos y Estructuras de Datos fundamentales (3)

Competences Expected:

Topics	**Learning Outcomes**

• Algoritmos numéricos simples, tales como el cálculo de la media de una lista de números, encontrar el mínimo y máximo.
• Algoritmos de búsqueda secuencial y binaria.
• Algoritmos de ordenamiento de peor caso cuadrático (selección, inserción)
• Algoritmos de ordenamiento con peor caso o caso promedio en O(N lg N) (Quicksort, Heapsort, Mergesort)
• Tablas Hash, incluyendo estratégias para evitar y resolver colisiones.
• Árboles de búsqueda binaria:
  – Operaciones comunes en árboles de búsqueda binaria como seleccionar el mínimo, máximo, insertar, eliminar, recorrido en árboles.
• Grafos y algoritmos en grafos:
  – Representación de grafos (ej., lista de adyacencia, matriz de adyacencia)
  – Recorrido en profundidad y amplitud
• Montículos (Heaps)
• Grafos y algoritmos en grafos:
  – Problema de corte máximo y mínimo
  – Busqueda local
• Búsqueda de patrones y algoritmos de cadenas/texto (ej. búsqueda de subcadena, búsqueda de expresiones regulares, algoritmos de subsecuencia común más larga)
	• Implementar algoritmos numéricos básicos [Usage]
	• Implementar algoritmos de busqueda simple y explicar las diferencias en sus tiempos de complejidad [Usage]
	• Ser capaz de implementar algoritmos de ordenamiento comunes cuádraticos y O(N log N) [Usage]
	• Describir la implementación de tablas hash, incluyendo resolución y el evitamiento de colisiones [Usage]
	• Discutir el tiempo de ejecución y eficiencia de memoria de los principales algoritmos de ordenamiento, busqueda y hashing [Usage]
	• Discutir factores otros que no sean eficiencia computacional que influyan en la elección de algoritmos, tales como tiempo de programación, mantenibilidad, y el uso de patrones específicos de la aplicación en los datos de entrada [Usage]
	• Explicar como el balanceamiento del arbol afecta la eficiencia de varias operaciones de un arbol de búsqueda binaria [Usage]
	• Resolver problemas usando algoritmos básicos de grafos, incluyendo busqueda por profundidad y busqueda por amplitud [Usage]
	• Demostrar habilidad para evaluar algoritmos, para seleccionar de un rango de posibles opciones, para proveer una justificación por esa selección,y para implementar el algoritmo en un contexto en específico [Usage]
	• Describir la propiedad del heap y el uso de heaps como una implementación de colas de prioridad [Usage]
	• Resolver problemas usando algoritmos de grafos, incluyendo camino más corto de una sola fuente y camino más corto de todos los pares, y como mínimo un algoritmo de arbol de expansion minima [Usage]
	• Trazar y/o implementar un algoritmo de comparación de string [Usage]

Readings : [**stroustrup2013**], [PA18]

Unit 7: Programación reactiva y dirigida por eventos (2)	
Competences Expected:	
Topics	**Learning Outcomes**

• Eventos y controladores de eventos.
• Usos canónicos como interfaces gráficas de usuario, dispositivos móviles, robots, servidores.
• Uso de frameworks reactivos.
  – Definición de controladores/oyentes (handles/listeners) de eventos.
  – Bucle principal de enventos no controlado po el escritor controlador de eventos (event-handler-writer)
• Eventos y eventos del programa generados externamente generada.
• La separación de modelo, vista y controlador.
	• Escribir manejadores de eventos para su uso en sistemas reactivos tales como GUIs [Usage]
	• Explicar porque el estilo de programación manejada por eventos es natural en dominios donde el programa reacciona a eventos externos [Usage]
	• Describir un sistema interactivo en términos de un modelo, una vista y un controlador [Usage]

Readings : [**stroustrup2013**], [Wil11]	

Unit 8: Árboles y Grafos (7)	
Competences Expected:	
Topics	**Learning Outcomes**

• Árboles.
  – Propiedades
  – Estrategias de recorrido
• Grafos no dirigidos
• Grafos dirigidos
• Grafos ponderados
• Arboles de expansion/bosques.
• Isomorfismo en grafos.
	• Ilustrar mediante ejemplos la terminología básica de teoría de grafos, y de alguna de las propiedades y casos especiales de cada tipo de grafos/árboles [Usage]
	• Demostrar diversos métodos de recorrer árboles y grafos, incluyendo recorridos pre, post e inorden de árboles [Usage]
	• Modelar una variedad de problemas del mundo real en ciencia de la computación usando formas adecuadas de grafos y árboles, como son la representación de una topología de red o la organización jerárquica de un sistema de archivos [Usage]
	• Demuestrar como los conceptos de grafos y árboles aparecen en estructuras de datos, algoritmos, técnicas de prueba (inducción estructurada), y conteos [Usage]
	• Explicar como construir un árbol de expansión de un grafo [Usage]
	• Determinar si dos grafos son isomorfos [Usage]

Readings : [Nak13]	

Unit 9: Diseño de Software (6)

Competences Expected:

Topics	Learning Outcomes
• Principios de diseño del sistema: niveles de abstracción (diseño arquitectónico y el diseño detallado), separación de intereses, ocultamiento de información, de acoplamiento y de cohesión, de reutilización de estructuras estándar. • Diseño de paradigmas tales como diseño estructurado (descomposición funcional de arriba hacia abajo), el análisis orientado a objetos y diseño, orientado a eventos de diseño, diseño de nivel de componente, centrado datos estructurada, orientada a aspectos, orientado a la función, orientado al servicio. • Modelos estructurales y de comportamiento de los diseños de software. • Diseño de patrones. • Relaciones entre los requisitos y diseños: La transformación de modelos, el diseño de los contratos, invariantes. • Conceptos de arquitectura de software y arquitecturas estándar (por ejemplo, cliente-servidor, n-capas, transforman centrados, tubos y filtros). • El uso de componentes de diseño: seleccion de componentes,diseño,adaptacion y componentes de ensamblaje, componentes y patrones, componentes y objetos(por ejemplo,construir una GUI usando un standar widget set) • Diseños de refactorización utilizando patrones de diseño • Calidad del diseño interno, y modelos para: eficiencia y desempeño, redundancia y tolerancia a fallos, trazavilidad de los requerimientos. • Medición y análisis de la calidad de un diseño. • Compensasiones entre diferentes aspectos de la calidad. • Aaplicaciones en frameworks. • Middleware: El paradigma de la orientacion a objetos con middleware, requerimientos para correr y clasificar objetos, monitores de procesamiento de transacciones y el sistema de flujo de trabajo. • Principales diseños de seguridad y codificación(cross-reference IAS/Principles of securre design). – Principio de privilegios mínimos – Principio de falla segura por defecto – Principio de aceptabilidad psicológica	• Formular los principios de diseño, incluyendo la separación de problemas, ocultación de información, acoplamiento y cohesión, y la encapsulación [Usage] • Usar un paradigma de diseño para diseñar un sistema de software básico y explicar cómo los principios de diseño del sistema se han aplicado en este diseño [Usage] • Construir modelos del diseño de un sistema de software simple los cuales son apropiado para el paradigma utilizado para diseñarlo [Usage] • En el contexto de un paradigma de diseño simple, describir uno o más patrones de diseño que podrían ser aplicables al diseño de un sistema de software simple [Usage] • Para un sistema simple adecuado para una situación dada, discutir y seleccionar un paradigma de diseño apropiado [Usage] • Crear modelos apropiados para la estructura y el comportamiento de los productos de software desde la especificaciones de requisitos [Usage] • Explicar las relaciones entre los requisitos para un producto de software y su diseño, utilizando los modelos apropiados [Usage] • Para el diseño de un sistema de software simple dentro del contexto de un único paradigma de diseño, describir la arquitectura de software de ese sistema [Usage] • Dado un diseño de alto nivel, identificar la arquitectura de software mediante la diferenciación entre las arquitecturas comunes de software, tales como 3 capas (*3-tier*), *pipe-and-filter*, y cliente-servidor [Usage] • Investigar el impacto de la selección arquitecturas de software en el diseño de un sistema simple [Usage] • Aplicar ejemplos simples de patrones en un diseño de software [Usage] • Describir una manera de refactorar y discutir cuando esto debe ser aplicado [Usage] • Seleccionar componentes adecuados para el uso en un diseño de un producto de software [Usage] • Explicar cómo los componentes deben ser adaptados para ser usados en el diseño de un producto de software [Usage] • Diseñar un contrato para un típico componente de software pequeño para el uso de un dado sistema [Usage] • Discutir y seleccionar la arquitectura de software adecuada para un sistema de software simple para

Unit 10: Ingeniería de Requisitos (1)	
Competences Expected:	
Topics	**Learning Outcomes**
• Al describir los requisitos funcionales utilizando, por ejemplo, los casos de uso o historias de los usuarios. • Propiedades de requisitos, incluyendo la consistencia, validez, integridad y viabilidad. • Requisitos de software elicitatión. • Descripción de datos del sistema utilizando, por ejemplo, los diagramas de clases o diagramas entidad-relación. • Requisitos no funcionales y su relación con la calidad del software. • Evaluación y uso de especificaciones de requisitos. • Requisitos de las técnicas de modelado de análisis. • La aceptabilidad de las consideraciones de certeza/incertidumbre sobre el comportamiento del software/sistema. • Prototipos. • Conceptos básicos de la especificación formal de requisitos. • Especificación de requisitos. • Validación de requisitos. • Rastreo de requisitos.	• Enumerar los componentes clave de un caso de uso o una descripción similar de algún comportamiento que es requerido para un sistema [Usage] • Describir cómo el proceso de ingeniería de requisitos apoya la obtención y validación de los requisitos de comportamiento [Usage] • Interpretar un modelo de requisitos dada por un sistema de software simple [Usage] • Describir los retos fundamentales y técnicas comunes que se utilizan para la obtención de requisitos [Usage] • Enumerar los componentes clave de un modelo de datos (por ejemplo, diagramas de clases o diagramas ER) [Usage] • Identificar los requisitos funcionales y no funcionales en una especificación de requisitos dada por un sistema de software [Usage] • Realizar una revisión de un conjunto de requisitos de software para determinar la calidad de los requisitos con respecto a las características de los buenos requisitos [Usage] • Aplicar elementos clave y métodos comunes para la obtención y el análisis para producir un conjunto de requisitos de software para un sistema de software de tamaño medio [Usage] • Comparar los métodos ágiles y el dirigido por planes para la especificación y validación de requisitos y describir los beneficios y riesgos asociados con cada uno [Usage] • Usar un método común, no formal para modelar y especificar los requisitos para un sistema de software de tamaño medio [Usage] • Traducir al lenguaje natural una especificación de requisitos de software (por ejemplo, un contrato de componentes de software) escrito en un lenguaje de especificación formal [Usage] • Crear un prototipo de un sistema de software para reducir el riesgo en los requisitos [Usage] • Diferenciar entre el rastreo (*tracing*) hacia adelante y hacia atrás y explicar su papel en el proceso de validación de requisitos [Usage]
Readings : [**stroustrup2013**]	

8. WORKPLAN

8.1 Methodology

Individual and team participation is encouraged to present their ideas, motivating them with additional points in the different stages of the course evaluation.

8.2 Theory Sessions

The theory sessions are held in master classes with activities including active learning and roleplay to allow students to internalize the concepts.

8.3 Practical Sessions

The practical sessions are held in class where a series of exercises and/or practical concepts are developed through problem solving, problem solving, specific exercises and/or in application contexts.

9. EVALUATION SYSTEM

********* EVALUATION MISSING ********

10. BASIC BIBLIOGRAPHY

[LE13] Stanley B. Lippman and Barbara E.Moo. *C++ Primer*. 5th. O'Reilly, 2013. ISBN: 9780133053043.

[Nak13] S. Nakariakov. *The Boost C++ Libraries: Generic Programming*. CreateSpace Independent Publishing Platforml, 2013.

[PA18] Praseed Pai and Peter Abraham. *C++ Reactive Programming*. 1st. Packt, 2018.

[Van02] David Vandervoorde. *C++ Templates:The Complete Guide*. 1st. Addison-Wesley, 2002. ISBN: 978-0134448237.

[Wil11] Anthony Williams. *C++ Concurrency in Action*. 1st. Manning, 2011.

National University of Copsta Rica (UNA)
School of Informatics
Sillabus 2024-I

1. COURSE

CS221. Computer Systems Architecture (Mandatory)

2. GENERAL INFORMATION

2.1 Course	:	CS221. Computer Systems Architecture
2.2 Semester	:	3^{er} Semestre.
2.3 Credits	:	3
2.4 Horas	:	2 HT; 2 HP;
2.5 Duration of the period	:	16 weeks
2.6 Type of course	:	Mandatory
2.7 Learning modality	:	Face to face
2.8 Prerrequisites	:	CS112. Programming I. (2^{nd} Sem) CS112. Programming I. (2^{nd} Sem)

3. PROFESSORS

Meetings after coordination with the professor

4. INTRODUCTION TO THE COURSE

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 addreses 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.

5. 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. COMPETENCES

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. TOPICS

Unit 1: Lógica digital y sistemas digitales (18)	
Competences Expected:	
Topics	**Learning Outcomes**
• Revisión e historia de la Arquitectura de Computadores. • Lógica combinacional y secuencial/*field programmable gate arrays* como bloque fundamental de construcción lógico combinacional secuencial. • Modelos de representación(abstracción) • Herramientas de diseño asistidas por computadora que procesan hardware y representaciones arquitecturales. • Registrar transferencia notación / Hardware lenguage descriptivo (Verilog/VHDL) • Restriccion física (Retrasos de Entrada, fan-in, fan-out, energia/potencia)	• Describir el avance de la tecnología de dispositivos, desde los tubos de vacío hasta VLSI, desde las arquitecturas mainframe a las arquitecturas en escala warehouse [Familiarity] • Comprender que la tendencia de las arquitecturas modernas de computadores es hacia núcleos múltiples y que el paraleliso es inherente en todos los sistemas de hardware [Usage] • Explicar las implicancias de los límites de potencia para mejoras adicionales en el rendimiento de los procesadores y también en el aprovechamiento del paralelismo [Usage] • Relacionar las varias representaciones equivalentes de la funcionalidad de un computador, incluyendo expresiones y puertas lógicas, y ser capces de utilizar expresiones matemáticas para describir las funciones de circuitos combinacionales y secuenciales sencillos [Familiarity] • Diseñar los componentes básicos de construcción de un computador: unidad aritmético lógica (a nivel de puertas lógicas), unidad central de procesamiento (a nivel de registros de transferencia), memoria (a nivel de registros de transferencia) [Usage] • Usar herramientas CAD para capturar, sistetizar, y simular bloques de construcción (como ALUs, registros, movimiento entre registros) de un computador simple [Familiarity] • Evaluar el comportamiento de un diagrama de tiempos y funcional de un procesador simple implementado a nivel de circuitos lógicos [Assessment]
Readings : [HH12], [PP05], [PH04], [JAs07], [HP06], [Par05], [Sta10], [PCh06]	

Unit 2: Representación de datos a nivel máquina (8)	
Competences Expected:	
Topics	**Learning Outcomes**
• Bits, Bytes y Words. • Representacion de datos numérica y bases numéricas. • Sistemas de punto flotante y punto fijo. • Representaciones con signo y complemento a 2. • Representación de información no numérica (códigos de caracteres, información gráfica) • Representación de registros y arreglos.	• Explicar porqué en computación todo es datos, inclusive las instrucciones [Assessment] • Explicar las razones de usar formatos alternativos para representar datos numéricos [Familiarity] • Describir cómo los enteros negativos se almacenan con representaciones de bit de signo y complemento a 2 [Usage] • Explicar cómo las representaciones de tamaño fijo afectan en la exactitud y la precisión [Usage] • Describir la representación interna de datos no numéricos como caracteres, cadenas, registros y arreglos [Usage] • Convertir datos numéricos de un formato a otro [Usage]
Readings : [HH12], [PP05], [PH04], [JAs07], [HP06], [Par05], [Sta10], [PCh06]	

Unit 3: Organización de la Máquina a Nivel Ensamblador (8)	
Competences Expected:	
Topics	**Learning Outcomes**
• Organización Básica de la Máquina de Von Neumann. • Unidad de Control. • *Instruction sets* y tipos (manipulación de información, control, I/O) • Assembler y Programación en Lenguaje de Máquina. • Formato de instrucciones. • Modos de direccionamiento. • Llamada a subrutinas y mecanismos de retorno. • I/O e Interrupciones. • Montículo (Heap) vs. Estático vs. Pila vs. Segmentos de código.	• Explicar la organización de la maquina clásica de von Neumann y sus principales unidades funcionales [Familiarity] • Describir cómo se ejecuta una instrucción en una máquina de von Neumann con extensión para hebras, sincronización multiproceso y ejecucion SIMD (máquina vectorial) [Familiarity] • Describir el paralelismo a nivel de instrucciones y sus peligros, y cómo es esto tratado en pipelines de proceso típicos [Familiarity] • Resumir cómo se representan las instrucciones, tanto a nivel de máquina bajo el contexto de un ensamblador simbólico [Familiarity] • Demostrar cómo se mapean los patrones de lenguajes de alto nivel en notaciones en lenguaje ensamblador o en código máquina [Usage] • Explicar los diferentes formatos de instrucciones, así como el direccionamiento por instrucción, y comparar formatos de tamaño fijo y variable [Usage] • Explicar como las llamadas a subrutinas son manejadas a nivel de ensamblador [Usage] • Explicar los conceptos básicos de interrupciones y operaciones de entrada y salida (I/O) [Familiarity] • Escribir segmentos de programa simples en lenguaje ensamblador [Usage] • Ilustrar cómo los bloques constructores fundamentales en lenguajes de alto nivel son implementados a nivel de lenguaje máquina [Usage]
Readings : [HH12], [PP05], [PH04], [JAs07], [HP06], [Par05], [Sta10], [PCh06]	

Unit 4: Organización funcional (8)	
Competences Expected:	
Topics	**Learning Outcomes**
• Implementación de *datapath*, incluyendo un *pipeline* de instrucciones, detección de *hazards* y la resolución. • Control de unidades: Microprogramada. • Instruccion (Pipelining) • Introducción al paralelismo al nivel de instrucción (PNI)	• Comparar implementaciones alternativas de ruta de datos [Assessment] • Discutir el concepto de puntos de control y la generación de señales de control usando implementaciones a nivel de circuito o microprogramadas [Familiarity] • Explicar el paralelismo a nivel de instrucciones básicas usando pipelining y los mayores riesgos que pueden ocurrir [Usage] • Diseñar e implementar un procesador completo, incluyendo ruta de datos y control [Usage] • Calcular la cantidad promedio de ciclos por instrucción de una implementación con procesador y sistema de memoria determinados [Assessment]
Readings : [HH12], [PP05], [PH04], [JAs07], [HP06], [Par05], [Sta10], [PCh06]	

Unit 5: Organización y Arquitectura del Sistema de Memoria (8)	
Competences Expected:	
Topics	**Learning Outcomes**
• Sistemas de Almacenamiento y su Tecnología. • Jerarquía de Memoria: importancia de la localización temporal y espacial. • Organización y Operaciones de la Memoria Principal. • Latencia, ciclos de tiempo, ancho de banda e *interleading*. • Memorias caché (Mapeo de direcciones, Tamaño de bloques, Reemplazo y Politicas de almacenamiento) • Multiprocesador coherencia cache / Usando el sistema de memoria para las operaciones de sincronización de memoria / atómica inter-core. • Memoria virtual (tabla de página, TLB) • Manejo de Errores y confiabilidad. • Error de codificación, compresión de datos y la integridad de datos.	• Identificar las principales tecnologías de memoria (Por ejemplo: SRAM, DRAM, Flash,Disco Magnetico) y su relación costo beneficio [Familiarity] • Explique el efecto de latencia de memoria en tiempo de ejecución [Familiarity] • Describir como el uso de jerarquía de memoria (caché, memoria virtual) es aplicado para reducir el atraso efectivo en la memoria [Usage] • Describir los principios de la administración de memoria [Usage] • Explique el funcionamiento de un sistema con gestión de memoria virtual [Usage] • Calcule el tiempo de acceso promedio a memoria bajo varias configuraciones de caché y memoria y para diversas combinaciones de instrucciones y referencias a datos [Assessment]
Readings : [HH12], [PP05], [PH04], [JAs07], [HP06], [Par05], [Sta10], [PCh06]	

Unit 6: Interfaz y comunicación (8)	
Competences Expected:	
Topics	**Learning Outcomes**
• Fundamentos de I/O: Handshaking, Bbuffering, I/O programadas, interrupciones dirigidas de I/O. • Interrumpir estructuras: interrumpir reconocimiento, vectorizado y priorizado. • Almacenamiento externo, organización fisica y discos. • Buses: Protocoles de bus, arbitraje, acceso directo a memoria (DMA). • Introducción a Redes: comunicación de redes como otra capa de acceso remoto. • Soporte Multimedia. • Arquitecturas RAID.	• Explicar como las interrupciones son aplicadas para implementar control de entrada-salida y transferencia de datos [Familiarity] • Identificar diversos tipos de buses en un sistema computacional [Familiarity] • Describir el acceso a datos desde una unidad de disco magnético [Usage] • Comparar organizaciones de red conocidas como organizaciones en bus/Ethernet, en anillo y organizaciones conmutadas versus ruteadas [Assessment] • Identificar las interfaces entre capas necesarios para el acceso y presentación multimedia, desde la captura de la imagen en almacenamiento remoto, a través del transporte por una red de comunicaciones, hasta la puesta en la memoria local y la presentación final en una pantalla gráfica [Familiarity] • Describir las ventajas y limitaciones de las arquitecturas RAID [Familiarity]
Readings : [HH12], [PP05], [PH04], [JAs07], [HP06], [Par05], [Sta10], [PCh06]	

Unit 7: Multiprocesamiento y arquitecturas alternativas (8)	
Competences Expected:	
Topics	**Learning Outcomes**
• *Power Law.* • Ejemplos de *sets* de instrucciones y arquitecturas SIMD y MIMD. • Redes de interconexión (Hypercube, Shuffle-exchange, Mesh, Crossbar) • Sistemas de memoria de multiprocesador compartido y consistencia de memoria. • Coherencia de cache multiprocesador.	• Discutir el concepto de procesamiento paralelo mas allá del clásico modelo de von Neumann [Assessment] • Describir diferentes arquitecturas paralelas como SIMD y MIMD [Familiarity] • Explicar el concepto de redes de interconexión y mostrar diferentes enfoques [Usage] • Discutir los principales cuidados en los sistemas de multiprocesamiento presentes con respecto a la gestión de memoria y describir como son tratados [Familiarity] • Describir las diferencias entre conectores electricos en paralelo backplane, interconexión memoria procesador y memoria remota via red, sus implicaciones para la latencia de acceso y el impacto en el rendimiento de un programa [Assessment]
Readings : [HH12], [PP05], [PH04], [JAs07], [HP06], [Par05], [Sta10], [PCh06]	

Unit 8: Mejoras de rendimiento (8)	
Competences Expected:	
Topics	**Learning Outcomes**
• Arquitectura superescalar. • Predicción de ramificación, Ejecución especulativa, Ejecución fuera de orden. • Prefetching. • Procesadores vectoriales y GPU's • Soporte de hardware para multiprocesamiento. • Escalabilidad. • Arquitecturas alternativas, como VLIW / EPIC y aceleradores y otros tipos de procesadores de propósito especial.	• Describir las arquitecturas superescalares y sus ventajas [Familiarity] • Explicar el concepto de predicción de bifurcaciones y su utilidad [Usage] • Caracterizar los costos y beneficios de la precarga prefetching [Assessment] • Explicar la ejecución especulativa e identifique las condiciones que la justifican [Assessment] • Discutir las ventajas de rendimiento ofrecida en una arquitectura de multihebras junto con los factores que hacen dificil dar el maximo beneficio de estas [Assessment] • Describir la importancia de la escalabilidad en el rendimiento [Assessment]
Readings : [HH12], [PP05], [PH04], [JAs07], [HP06], [Par05], [Sta10], [PCh06]	

8. WORKPLAN

8.1 Methodology

Individual and team participation is encouraged to present their ideas, motivating them with additional points in the different stages of the course evaluation.

8.2 Theory Sessions

The theory sessions are held in master classes with activities including active learning and roleplay to allow students to internalize the concepts.

8.3 Practical Sessions

The practical sessions are held in class where a series of exercises and/or practical concepts are developed through problem solving, problem solving, specific exercises and/or in application contexts.

9. EVALUATION SYSTEM

********* EVALUATION MISSING ********

10. BASIC BIBLIOGRAPHY

[HH12] David Harris and Sarah Harris. *Digital Design and Computer Architecture*. 2nd. Morgan Kaufmann, 2012. ISBN: 978-0123944245.

[HP06] J. L. Hennessy and D. A. Patterson. *Computer Architecture: A Quantitative Approach*. 4th. San Mateo, CA: Morgan Kaufman, 2006.

[JAs07] Peter J.Ashenden. *Digital Design (Verilog): An Embedded Systems Approach Using Verilog*. Morgan Kaufmann, 2007. ISBN: 978-0123695277.

[Par05] Behrooz Parhami. *Computer Architecture: From Microprocessors to Supercomputers*. New York: Oxford Univ. Press, 2005. ISBN: ISBN 0-19-515455-X.

[PCh06] Pong P.Chu. *RTL Hardware Design Using VHDL*. 1st. Wiley-Interscience, 2006.

[PH04] D. A. Patterson and J. L. Hennessy. *Computer Organization and Design: The Hardware/Software Interface*. 3rd ed. San Mateo, CA: Morgan Kaufman, 2004.

[PP05] Yale N Patt and Sanjay J Patel. *Introduction to Computing Systems*. 2nd. McGraw Hill, 2005.

[Sta10] William Stalings. *Computer Organization and Architecture: Designing for Performance*. 8th. Upper Saddle River, NJ: Prentice Hall, 2010.

National University of Copsta Rica (UNA)
School of Informatics
Sillabus 2024-I

1. COURSE

CS2B1. Platform Based Development I (Mandatory)

2. GENERAL INFORMATION

2.1 Course	:	CS2B1. Platform Based Development I
2.2 Semester	:	3^{er} Semestre.
2.3 Credits	:	3
2.4 Horas	:	2 HT; 2 HP;
2.5 Duration of the period	:	16 weeks
2.6 Type of course	:	Mandatory
2.7 Learning modality	:	Face to face
2.8 Prerrequisites	:	CS112. Programming I. (2^{nd} Sem) CS112. Programming I. (2^{nd} Sem)

3. PROFESSORS

Meetings after coordination with the professor

4. INTRODUCTION TO THE COURSE

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.

5. 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. COMPETENCES

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. TOPICS

Unit 1: Introducción (5)	
Competences Expected:	
Topics	**Learning Outcomes**
• Visión general de plataformas (ejemplo, Web, Mobil, Juegos, Industrial) • Programacíon a través de APIs específicos. • Visión general de lenguajes de plataforma (ejemplo, Objective C, HTML5) • Pogramacíón bajo restricciónes de plataforma.	• Describir cómo el desarrollo basado en plataforma difiere de la programación de proposito general [Familiarity] • Listar las características de lenguajes de plataforma [Familiarity] • Escribir y ejecutar un programa simple basado en plataforma [Familiarity] • Listar las ventajas y desventajas de la programación con restricciones de plataforma [Familiarity]
Readings : [**fielding2000fielding**], [**grove2009web**], [**annuzzi2013introduction**], [**Cornez2015**]	

Unit 2: Plataformas web (5)	
Competences Expected:	
Topics	**Learning Outcomes**
• Lenguajes de programación web (e.g., HTML5, Javascript, PHP, CSS) • • Web Platform constraints: Client-Server, Stateless-Stateful, Cache, Uniform Interface, Layered System, Code on Demand, ReST. • Restricción de plataformas web. • Software como servicio. • Estándares web.	• Diseñar e implementar una aplicación web sencilla [Familiarity] • Describir las limitaciones que la web pone a los desarrolladores [Familiarity] • Comparar y contrastar la programación web con la programación de proposito general [Familiarity] • Describir las diferencias entre software como un servicio y productos de software tradicionales [Familiarity] • Discutir cómo los estándares de web impactan el desarrollo de software [Familiarity] • Revise una aplicación web existente con un estándar web actual [Familiarity]
Readings : [**fielding2000fielding**]	

Unit 3: Desarrollo de servicios y aplicaciones web (25)	
Competences Expected:	
Topics	**Learning Outcomes**
• 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: Plataformas móviles (5)	
Competences Expected:	
Topics	**Learning Outcomes**
• Lenguajes de Programación para Móviles. • Design Principles: Segregation of Interfaces, Single Responsability, Separation of concerns, Dependency Inversion. • Desafíos con mobilidad y comunicación inalámbrica. • Aplicaciones Location-aware. • Rendimiento / Compensación de Potencia. • Restricciones de las Plataformas Móviles. • Tecnologías Emergentes.	• Diseñar e implementar una aplicación móvil para una plataforma móvil dada [Familiarity] • Discutir las limitaciones que las plataformas móviles ponen a los desarrolladores [Familiarity] • Discutir el rendimiento vs perdida de potencia [Familiarity] • Compare y contraste la programación móvil con la programación de proposito general [Familiarity]
Readings : [**martin2017clean**], [**annuzzi2013introduction**]	

Unit 5: Mobile Applications for Android Handheld Systems (25)	
Competences Expected:	
Topics	**Learning Outcomes**
• 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. WORKPLAN

8.1 Methodology

Individual and team participation is encouraged to present their ideas, motivating them with additional points in the different stages of the course evaluation.

8.2 Theory Sessions

The theory sessions are held in master classes with activities including active learning and roleplay to allow students to internalize the concepts.

8.3 Practical Sessions

The practical sessions are held in class where a series of exercises and/or practical concepts are developed through problem solving, problem solving, specific exercises and/or in application contexts.

9. EVALUATION SYSTEM

********* EVALUATION MISSING ********

10. BASIC BIBLIOGRAPHY

National University of Copsta Rica (UNA)
School of Informatics
Sillabus 2024-I

1. COURSE

SE1A1. Software Architecture and Design (Mandatory)

2. GENERAL INFORMATION

2.1 Course	:	SE1A1. Software Architecture and Design
2.2 Semester	:	3^{er} Semestre.
2.3 Credits	:	4
2.4 Horas	:	3 HT; 3 HP;
2.5 Duration of the period	:	16 weeks
2.6 Type of course	:	Mandatory
2.7 Learning modality	:	Face to face
2.8 Prerrequisites	:	SE1R1. Requirements and interface design. (2^{nd} Sem) SE1R1. Requirements and interface design. (2^{nd} Sem)

3. PROFESSORS

Meetings after coordination with the professor

4. INTRODUCTION TO THE COURSE

Write justification for this course here ...

5. GOALS

- Write your first goal here.
- Write your second goal here.
- Just in case you need more goals write them here

6. COMPETENCES

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

7. TOPICS

Unit 1: title for the unit goes here (5)	
Competences Expected:	
Topics	**Learning Outcomes**
• Topic1 • Topic2 • Topic3	• Learning outcome1 [Levelforthislearningoutcome]. • Apply computing in complex problems [Usage]. • Create a search engine [Assessment]. • Study data structures [Familiarity].
Readings : [**Bibitem1**], [**Bibitem2**]	

Unit 2: another unit goes here (1)	
Competences Expected:	
Topics	**Learning Outcomes**
• Topic1	• Learning outcome xyz [Levelforthislearningoutcome].
Readings : [**Bibitem3**], [**Bibitem1**]	

8. WORKPLAN

8.1 Methodology

Individual and team participation is encouraged to present their ideas, motivating them with additional points in the different stages of the course evaluation.

8.2 Theory Sessions

The theory sessions are held in master classes with activities including active learning and roleplay to allow students to internalize the concepts.

8.3 Practical Sessions

The practical sessions are held in class where a series of exercises and/or practical concepts are developed through problem solving, problem solving, specific exercises and/or in application contexts.

9. EVALUATION SYSTEM

********* EVALUATION MISSING ********

10. BASIC BIBLIOGRAPHY

National University of Copsta Rica (UNA)
School of Informatics
Sillabus 2024-I

1. COURSE

MA102. Linear Algebra (Mandatory)

2. GENERAL INFORMATION

2.1 Course	:	MA102. Linear Algebra
2.2 Semester	:	3^{er} Semestre.
2.3 Credits	:	4
2.4 Horas	:	4 HT;
2.5 Duration of the period	:	16 weeks
2.6 Type of course	:	Mandatory
2.7 Learning modality	:	Face to face
2.8 Prerrequisites	:	MA101. Calculus. (2^{nd} Sem) MA101. Calculus. (2^{nd} Sem)

3. PROFESSORS

Meetings after coordination with the professor

4. INTRODUCTION TO THE COURSE

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.

5. 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. COMPETENCES

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. TOPICS

Unit 1: Introduction (18)	
Competences Expected:	
Topics	**Learning Outcomes**
• Importance of linear algebra and numerical methods. Examples.	• Be able to understand the basic concepts and importance of Linear Algebra and Numerical Methods.
Readings : [AR14], [CC15]	

Unit 2: Linear Algebra (14)	
Competences Expected:	
Topics	**Learning Outcomes**
• 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 : [AR14], [CC15]	

Unit 3: Numerical methods (22)	
Competences Expected:	
Topics	**Learning Outcomes**
• 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 : [AR14], [CC15]	

8. WORKPLAN

8.1 Methodology

Individual and team participation is encouraged to present their ideas, motivating them with additional points in the different stages of the course evaluation.

8.2 Theory Sessions

The theory sessions are held in master classes with activities including active learning and roleplay to allow students to internalize the concepts.

8.3 Practical Sessions

The practical sessions are held in class where a series of exercises and/or practical concepts are developed through problem solving, problem solving, specific exercises and/or in application contexts.

9. EVALUATION SYSTEM

********* EVALUATION MISSING ********

10. BASIC BIBLIOGRAPHY

[AR14] H. Anton and C. Rorres. *Elementary Linear Algebra, Applications Version*. 11th. Wiley, 2014.

[CC15] S.C. Chapra and R.P. Canale. *Numerical Methods for Engineers,* 7th. Vol. 1. McGraw-Hill, 12015.

National University of Copsta Rica (UNA)
School of Informatics
Sillabus 2024-I

1. COURSE

CS210. Algorithms and Data Structures (Mandatory)

2. GENERAL INFORMATION

2.1 Course	:	CS210. Algorithms and Data Structures
2.2 Semester	:	4^{to} Semestre.
2.3 Credits	:	4
2.4 Horas	:	3 HT; 3 HP;
2.5 Duration of the period	:	16 weeks
2.6 Type of course	:	Mandatory
2.7 Learning modality	:	Face to face
2.8 Prerrequisites	:	CS113. Programming II. (3^{rd} Sem) CS113. Programming II. (3^{rd} Sem)

3. PROFESSORS

Meetings after coordination with the professor

4. INTRODUCTION TO THE COURSE

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.

5. GOALS

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

6. COMPETENCES

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. TOPICS

Unit 1: Graphs (12)	
Competences Expected:	
Topics	**Learning Outcomes**
• 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 : [Cor+09], [Fag+14], [Knu97], [Knu98]	

Unit 2: Scatter Matrices (8)	
Competences Expected:	
Topics	**Learning Outcomes**
• 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 : [Cor+09], [Fag+14], [Knu97], [Knu98]	

Unit 3: Balanced Trees (16)	
Competences Expected:	
Topics	**Learning Outcomes**
• 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 : [Cor+09], [Fag+14], [Knu97], [Knu98]	

8. WORKPLAN

8.1 Methodology

Individual and team participation is encouraged to present their ideas, motivating them with additional points in the different stages of the course evaluation.

8.2 Theory Sessions

The theory sessions are held in master classes with activities including active learning and roleplay to allow students to internalize the concepts.

8.3 Practical Sessions

The practical sessions are held in class where a series of exercises and/or practical concepts are developed through problem solving, problem solving, specific exercises and/or in application contexts.

9. EVALUATION SYSTEM

********* EVALUATION MISSING ********

10. BASIC BIBLIOGRAPHY

[Cor+09] Thomas H. Cormen et al. *Introduction to Algorithms*. Third Edition. ISBN: 978-0-262-53305-8. MIT Press, 2009.

[Fag+14] José Fager et al. *Estructura de datos*. First Edition. Iniciativa Latinoamericana de Libros de Texto Abiertos (LATIN), 2014.

[Knu97] Donald E. Knuth. *The Art of Computer Programming, Vol. 1: Fundamental Algorithms*. 3rd. Addison-Wesley Professional, 1997.

[Knu98] Donald E. Knuth. *The art of computer programming, volume 3:Sorting and searching*. 2nd. Addison-Wesley Professional, 1998.

National University of Copsta Rica (UNA)
School of Informatics
Sillabus 2024-I

1. COURSE

CS2S1. Operating systems (Mandatory)

2. GENERAL INFORMATION

2.1 Course	:	CS2S1. Operating systems
2.2 Semester	:	4^{to} Semestre.
2.3 Credits	:	3
2.4 Horas	:	2 HT; 2 HP;
2.5 Duration of the period	:	16 weeks
2.6 Type of course	:	Mandatory
2.7 Learning modality	:	Face to face
2.8 Prerrequisites	:	CS221. Computer Systems Architecture. (3^{rd} Sem) CS221. Computer Systems Architecture. (3^{rd} Sem)

3. PROFESSORS

Meetings after coordination with the professor

4. INTRODUCTION TO THE COURSE

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.

5. GOALS

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

6. COMPETENCES

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. TOPICS

Unit 1: Visión general de Sistemas Operativos (3)	
Competences Expected:	
Topics	**Learning Outcomes**
• Papel y el propósito del sistema operativo. • Funcionalidad de un sistema operativo típico. • Los mecanismos de apoyo modelos cliente-servidor. • Cuestiones de diseño (eficiencia, robustez, flexibilidad, portabilidad, seguridad, compatibilidad) • Influencias de seguridad, creación de redes, multimedia, sistemas de ventanas.	• Explicar los objetivos y funciones de un sistema operativo moderno [Familiarity] • Analizar las ventajas y desventajas inherentes en el diseño de un sistema operativo [Assessment] • Describir las funciones de un sistema operativo contemporaneo respecto a conveniencia, eficiencia, y su habilidad para evolucionar [Familiarity] • Discutir acerca de sistemas operativos cliente-servidor, en red, distribuidos y cómo se diferencian de los sistemas operativos de un solo usuario [Familiarity] • Identificar amenazas potenciales a sistemas operativos y las características del diseño de seguridad para protegerse de ellos [Familiarity]
Readings : [Avi12], [Sta05], [Tan06], [Tan01], [AD14]	

Unit 2: Principios de Sistemas Operativos (6)	
Competences Expected:	
Topics	**Learning Outcomes**
• Estructuración de Sistemas Operativos (monolítico, capas, modular, los modelos micro-kernel) • Abstracciones, procesos y recursos. • Los conceptos de interfaces de programa de aplicación (API) • La evolución de las técnicas de hardware / software y las necesidades de aplicación • Organización de dispositivos. • Interrupciones: métodos e implementaciones. • Concepto de estado de usuario / sistema y la protección, la transición al modo kernel.	• Explicar el concepto de una capa lógica [Familiarity] • Explicar los beneficios de construir capas abstractas en forma jerárquica [Familiarity] • Describir el valor de la API y *middleware* [Familiarity] • Describir como los recursos computacionales son usados por aplicaciones de software y administradas por el software del sistema [Familiarity] • Contrastar el modo *kernel* y modo usuario en un sistema operativo [Assessment] • Discutir las ventajas y desventajas del uso de procesamiento con interrupciones [Familiarity] • Explicar el uso de una lista de dispositivos y el controlador de colas de entrada y salida [Familiarity]
Readings : [Avi12], [Sta05], [Tan06], [Tan01], [AD14]	

Unit 3: Concurrencia (9)	
Competences Expected:	
Topics	**Learning Outcomes**
• Diagramas de estado. • Estructuras (lista preparada, bloques de control de procesos, y así sucesivamente) • *Dispatching* y cambio de contexto. • El papel de las interrupciones. • Gestionar el acceso a los objetos del sistema operativo de forma atómica. • La implementación de primitivas de sincronización. • Problemas de multiprocesador (spin-locks, reentrada)	• Describir la necesidad de concurrencia en el marco de un sistema operativo [Familiarity] • Demostrar los potenciales problemas de tiempo de ejecución derivados de la operación simultánea de muchas tareas diferentes [Usage] • Resumir el rango de mecanismos que pueden ser usados a nivel del sistema operativo para realizar sistemas concurrentes y describir los beneficios de cada uno [Familiarity] • Explicar los diferentes estados por los que una tarea debe pasar y las estructuras de datos necesarias para el manejo de varias tareas [Familiarity] • Resumir las técnicas para lograr sicronización en un sistema operativo(por ejemplo, describir como implementar semáforos usando primitivas del sistema operativo.) [Familiarity] • Describir las razones para usar interrupciones, *dispatching*, y cambio de contexto para soportar concurrencia en un sistema operativo [Familiarity] • Crear diagramas de estado y transición para los problemas de dominios simples [Usage]
Readings : [Avi12], [Sta05], [Tan06], [Tan01], [AD14]	

Unit 4: Planificación y despacho (6)	
Competences Expected:	
Topics	**Learning Outcomes**
• *Scheduling preemptive* y *non-preemptive.* • *Scheduling* y políticas. • Procesos y subprocesos. • Plazos y cuestiones en tiempo real.	• Comparar y contrastar los algoritmos comunes que se utilizan tanto para *scheduling preemptive* y *preemptive* de tareas en los sistemas operativos, como la comparación de prioridad, el rendimiento, y los esquemas de distribución equitativa [Assessment] • Describir las relaciones entre los algoritmos de *scheduling* y dominios de aplicación [Familiarity] • Discutir los tipos de *scheduling* en procesadores en de corto, mediano, largo plazo y I/O [Familiarity] • Describir las diferencias entre procesos y *threads* [Familiarity] • Comparar y contrastar enfoques estáticos y dinámicos para *scheduling* en tiempo real [Assessment] • Discutir sobre la necesidad de *preemption* y *deadline scheduling* [Familiarity] • Identificar formas en que la lógica expresada en algoritmos de planificación son de aplicación a otros ámbitos, tales como I/O del disco, la programación de disco de red, programación de proyectos y problemas más allá de la computación [Familiarity]
Readings : [Avi12], [Sta05], [Tan06], [Tan01], [AD14]	

Unit 5: Manejo de memoria (6)	
Competences Expected:	
Topics	**Learning Outcomes**
• Revisión de la memoria física y hardware de gestión de memoria. • Conjuntos de trabajo y thrashing. • El almacenamiento en caché	• Explicar la jerarquía de la memoria y *tradeoffs* de costo-rendimiento [Familiarity] • Resumir los principios de memoria virtual tal como se aplica para el almacenamiento en cache y paginación [Familiarity] • Evaluar las ventajas y desventajas en términos del tamaño de memoria (memoria principal, memoria caché, memoria axiliar) y la velocidad del procesador [Assessment] • Describir las diferentes formas de asignar memoria a las tareas, citando las ventajas relativas de cada uno [Familiarity] • Describir el motivo y el uso de memoria caché (rendimiento y proximidad, dimensión diferente de como los caches complican el aislamiento y abstracción en VM) [Familiarity] • Estudiar los conceptos de *thrashing*, tanto en términos de las razones por las que se produce y las técnicas usadas para el reconocimiento y manejo del problema [Familiarity]
Readings : [Avi12], [Sta05], [Tan06], [Tan01], [AD14]	

Unit 6: Seguridad y protección (6)	
Competences Expected:	
Topics	**Learning Outcomes**
• Visión general de la seguridad del sistema . • Política / mecanismo de separación. • Métodos de seguridad y dispositivos. • Protección, control de acceso y autenticación. • Las copias de seguridad.	• Explicar la necesidad para la protección y seguridad en un sistema operativo [Familiarity] • Resumir las caracteristicas y limitaciones de un sistema operativo usado para proporcionar protección y seguridad [Familiarity] • Explicar el mecanismo disponible en un OS para controlar los accesos a los recursos [Familiarity] • Realizar tareas de administración de sistemas sencillas de acuerdo a una política de seguridad, por ejemplo la creación de cuentas, el establecimiento de permisos, aplicación de parches y organización de backups regulares [Familiarity]
Readings : [Avi12], [Sta05], [Tan06], [Tan01], [AD14]	

Unit 7: Máquinas virtuales (6)	
Competences Expected:	
Topics	**Learning Outcomes**
• Tipos de virtualización (incluyendo Hardware / Software, OS, Servidor, Servicio, Red) • Paginación y la memoria virtual. • Sistemas de archivos virtuales. • Los Hypervisores. • Virtualización portátil; emulación vs aislamiento. • Costo de la virtualización.	• Explicar el concepto de memoria virtual y la forma cómo se realiza en hadware y software [Familiarity] • Diferenciar emulacion y el aislamiento [Familiarity] • Evaluar virtualización de compensaciones [Assessment] • Discutir sobre hipervisores y la necesidad para ellos en conjunto con diferentes tipos de hipervisores [Familiarity]
Readings : [Avi12], [Sta05], [Tan06], [Tan01], [AD14]	

Unit 8: Manejo de dispositivos (6)	
Competences Expected:	
Topics	**Learning Outcomes**
• Características de los dispositivos serie y paralelo. • Haciendo de abstracción de dispositivos. • Estrategias de buffering. • Acceso directo a memoria. • La recuperación de fallos.	• Explique la diferencia clave entre dispositivos seriales y paralelos e identificar las condiciones en las cuales cada uno es apropiado [Familiarity] • Identificar la relación entre el hardware físico y los dispositivos virtuales mantenidos por el sistema operativo [Familiarity] • Explicar *buffering* y describir las estrategias para su aplicación [Familiarity] • Diferenciar los mecanismos utilizados en la interconexión de un rango de dispositivos (incluyendo dispositivos portátiles, redes, multimedia) a un ordenador y explicar las implicaciones de éstas para el diseño de un sistema operativo [Familiarity] • Describir las ventajas y desventajas de acceso directo a memoria y discutir las cirscunstancias en cuales se justifica su uso [Familiarity] • Identificar los requerimientos para recuperación de errores [Familiarity] • Implementar un controlador de dispositivo simple para una gama de posibles equipos [Usage]
Readings : [Avi12], [Sta05], [Tan06], [Tan01], [AD14]	

Unit 9: Sistema de archivos (6)	
Competences Expected:	
Topics	**Learning Outcomes**
• Archivos: los datos, metadatos, operaciones, organización, amortiguadores, secuenciales, no secuencial. • Directorios: contenido y estructura. • Los sistemas de archivos: partición, montar / desmontar sistemas de archivos virtuales. • Técnicas estándar de implementación . • Archivos asignados en memoria. • Sistemas de archivos de propósito especial. • Naming, búsqueda, acceso, copias de seguridad. • La bitacora y los sistemas de archivos estructurados (log)	• Describir las decisiones que deben tomarse en el diseño de sistemas de archivos [Familiarity] • Comparar y contrastar los diferentes enfoques para la organización de archivos, el reconocimiento de las fortalezas y debilidades de cada uno. [Assessment] • Resumir cómo el desarrollo de hadware ha dado lugar a cambios en las prioridades para el diseño y la gestión de sistemas de archivos [Familiarity] • Resumir el uso de diarios y como los sistemas de archivos de registro estructurado mejora la tolerancia a fallos [Familiarity]
Readings : [Avi12], [Sta05], [Tan06], [Tan01], [AD14]	

Unit 10: Sistemas *embedded* y de tiempo real (6)	
Competences Expected:	
Topics	**Learning Outcomes**
• Proceso y programación de tareas. • Los requisitos de gestión de memoria / disco en un entorno en tiempo real. • Los fracasos, los riesgos y la recuperación. • Preocupaciones especiales en sistemas de tiempo real.	• Describir que hace a un sistema un sistema en tiempo real [Familiarity] • Explicar la presencia y describir las características de latencia en sistemas de tiempo real [Familiarity] • Resumir los problemas especiales que los sistemas en tiempo real presentan, incluyendo el riesgo, y cómo se tratan estos problemas [Familiarity]
Readings : [Avi12], [Sta05], [Tan06], [Tan01], [AD14]	

Unit 11: Tolerancia a fallas (3)	
Competences Expected:	
Topics	**Learning Outcomes**
• Conceptos fundamentales: sistemas fiables y disponibles. • Redundancia espacial y temporal. • Los métodos utilizados para implementar la tolerancia a fallos. • Los ejemplos de los mecanismos del sistema operativo para la detección, recuperación, reinicio para implementar la tolerancia a fallos, el uso de estas técnicas para los servicios propios del sistema operativo.	• Explicar la importancia de los términos tolerancia a fallos, fiabilidad y disponibilidad [Familiarity] • Explicar en términos generales la gama de métodos para implementar la tolerancia a fallos en un sistema operativo [Familiarity] • Explicar cómo un sistema operativo puede continar funcionando después de que ocurra una falla [Familiarity]
Readings : [Avi12], [Sta05], [Tan06], [Tan01], [AD14]	

Unit 12: Evaluación del desempeño de sistemas (3)	
Competences Expected:	
Topics	**Learning Outcomes**
• ¿Por qué el rendimiento del sistema debe ser evaluado? • ¿Qué se va a evaluar? • Sistemas de políticas de rendimiento, por ejemplo, el almacenamiento en caché, de paginación, la programación, la gestión de memoria, y la seguridad. • Modelos de evaluación: analítica, simulación, o de implementación específico determinista. • Cómo recoger los datos de evaluación (perfiles y mecanismos de localización)	• Describir las medidas de rendimiento utilizados para determinar cómo el sistema funciona [Familiarity] • Explicar los principales modelos de evaluación utilizados para evaluar un sistema [Familiarity]
Readings : [Avi12], [Sta05], [Tan06], [Tan01], [AD14]	

8. WORKPLAN

8.1 Methodology

Individual and team participation is encouraged to present their ideas, motivating them with additional points in the different stages of the course evaluation.

8.2 Theory Sessions

The theory sessions are held in master classes with activities including active learning and roleplay to allow students to internalize the concepts.

8.3 Practical Sessions

The practical sessions are held in class where a series of exercises and/or practical concepts are developed through problem solving, problem solving, specific exercises and/or in application contexts.

9. EVALUATION SYSTEM

********* EVALUATION MISSING ********

10. BASIC BIBLIOGRAPHY

[AD14] Thomas Anderson and Michael Dahlin. *Operating Systems: Principles and Practice*. 2nd. Recursive Books, 2014. ISBN: 978-0985673529.

[Avi12] Greg Gagne Avi Silberschatz Peter Baer Galvin. *Operating System Concepts, 9/E*. John Wiley & Sons, Inc., 2012. ISBN: 978-1-118-06333-0.

[Sta05] William Stallings. *Operating Systems: Internals and Design Principles, 5/E*. Prentice Hall, 2005. ISBN: 0-13-147954-7.

[Tan01] Andrew S. Tanenbaum. *Modern Operating Systems, 4/E*. Prentice Hall, 2001. ISBN: 0-13-031358-0.

[Tan06] Andrew S. Tanenbaum. *Operating Systems Design and Implementation, 3/E*. Prentice Hall, 2006. ISBN: 0-13-142938-8.

National University of Copsta Rica (UNA)
School of Informatics
Sillabus 2024-I

1. COURSE

SE2C1. Software development (Mandatory)

2. GENERAL INFORMATION

2.1 Course	:	SE2C1. Software development
2.2 Semester	:	4^{to} Semestre.
2.3 Credits	:	4
2.4 Horas	:	3 HT; 3 HP;
2.5 Duration of the period	:	16 weeks
2.6 Type of course	:	Mandatory
2.7 Learning modality	:	Face to face
2.8 Prerrequisites	:	

- SE1A1. Software Architecture and Design. (3^{rd} Sem)
- CS112. Programming I. (2^{nd} Sem)
- SE1A1. Software Architecture and Design. (3^{rd} Sem)
- CS112. Programming I. (2^{nd} Sem)

3. PROFESSORS

Meetings after coordination with the professor

4. INTRODUCTION TO THE COURSE

Write justification for this course here ...

5. GOALS

- Write your first goal here.
- Write your second goal here.
- Just in case you need more goals write them here

6. COMPETENCES

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

7. TOPICS

Unit 1: title for the unit goes here (5)	
Competences Expected:	
Topics	**Learning Outcomes**
• Topic1 • Topic2 • Topic3	• Learning outcome1 [Levelforthislearningoutcome]. • Apply computing in complex problems [Usage]. • Create a search engine [Assessment]. • Study data structures [Familiarity].
Readings : [**Bibitem1**], [**Bibitem2**]	

Unit 2: another unit goes here (1)	
Competences Expected:	
Topics	**Learning Outcomes**
• Topic1	• Learning outcome xyz [Levelforthislearningoutcome].
Readings : [**Bibitem3**], [**Bibitem1**]	

8. WORKPLAN

8.1 Methodology

Individual and team participation is encouraged to present their ideas, motivating them with additional points in the different stages of the course evaluation.

8.2 Theory Sessions

The theory sessions are held in master classes with activities including active learning and roleplay to allow students to internalize the concepts.

8.3 Practical Sessions

The practical sessions are held in class where a series of exercises and/or practical concepts are developed through problem solving, problem solving, specific exercises and/or in application contexts.

9. EVALUATION SYSTEM

********* EVALUATION MISSING ********

10. BASIC BIBLIOGRAPHY

National University of Copsta Rica (UNA)
School of Informatics
Sillabus 2024-I

1. COURSE

MA203. Statistics and Probabilities (Mandatory)

2. GENERAL INFORMATION

2.1 Course	:	MA203. Statistics and Probabilities
2.2 Semester	:	4^{to} Semestre.
2.3 Credits	:	3
2.4 Horas	:	2 HT; 2 HP;
2.5 Duration of the period	:	16 weeks
2.6 Type of course	:	Mandatory
2.7 Learning modality	:	Face to face
2.8 Prerrequisites	:	MA101. Calculus. (2^{nd} Sem) MA101. Calculus. (2^{nd} Sem)

3. PROFESSORS

Meetings after coordination with the professor

4. INTRODUCTION TO THE COURSE

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

5. 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. COMPETENCES

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. TOPICS

Unit 1: Variable Type (6)	
Competences Expected:	
Topics	**Learning Outcomes**
• 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 : [MRo14], [Men14]	

Unit 2: Descriptive Statistics (6)	
Competences Expected:	
Topics	**Learning Outcomes**
• 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 : [MRo14], [Men14]	

Unit 3: Inferential Statistics (6)	
Competences Expected:	
Topics	**Learning Outcomes**
• 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 : [MRo14], [Men14]	

8. WORKPLAN

8.1 Methodology

Individual and team participation is encouraged to present their ideas, motivating them with additional points in the different stages of the course evaluation.

8.2 Theory Sessions

The theory sessions are held in master classes with activities including active learning and roleplay to allow students to internalize the concepts.

8.3 Practical Sessions

The practical sessions are held in class where a series of exercises and/or practical concepts are developed through problem solving, problem solving, specific exercises and/or in application contexts.

9. EVALUATION SYSTEM

********* EVALUATION MISSING ********

10. BASIC BIBLIOGRAPHY

[Men14] Beaver Mendenhall. *Introducción a la probabilidad y estadística*. 13th. 2014.

[MRo14] Sheldon M.Ross. *Introduction to Probability and Statistics for Engineers and Scientists*. 5th. 2014.

National University of Copsta Rica (UNA)
School of Informatics
Sillabus 2024-I

1. COURSE

CS212. Analysis and Design of Algorithms (Mandatory)

2. GENERAL INFORMATION

2.1 Course	:	CS212. Analysis and Design of Algorithms
2.2 Semester	:	5^{to} Semestre.
2.3 Credits	:	3
2.4 Horas	:	2 HT; 2 HP;
2.5 Duration of the period	:	16 weeks
2.6 Type of course	:	Mandatory
2.7 Learning modality	:	Face to face
2.8 Prerrequisites	:	CS210. Algorithms and Data Structures. (4^{th} Sem)
		CS210. Algorithms and Data Structures. (4^{th} Sem)

3. PROFESSORS

Meetings after coordination with the professor

4. INTRODUCTION TO THE COURSE

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

5. 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. COMPETENCES

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. TOPICS

Unit 1: Análisis Básico (10)

Competences Expected:

Topics	Learning Outcomes
• Diferencias entre el mejor, el esperado y el peor caso de un algoritmo. • Análisis asintótico de complejidad de cotas superior y esperada. • Clases de complejidad como constante, logarítmica, lineal, cuadrática y exponencial. • Asymptotic Notation • Análisis de algoritmos iterativos y recursivos. • Inductive proofs and correctness of algorithms • Teorema Maestro y Árboles Recursivos.	• Explique a que se refiere con "mejor", "esperado" y "peor" caso de comportamiento de un algoritmo [Assessment] • Determine informalmente el tiempo y el espacio de complejidad de diferentes algoritmos [Assessment] • Lista y contraste de clases estándares de complejidad [Assessment] • Explicar el uso de la notación theta grande, omega grande y o pequeña para describir la cantidad de trabajo hecho por un algoritmo [Assessment] • Analyze worst-case running times of algorithms using asymptotic analysis [Assessment] • Usar relaciones recurrentes para determinar el tiempo de complejidad de algoritmos recursivamente definidos [Assessment] • Resuelve relaciones de recurrencia básicas, por ejemplo. usando alguna forma del Teorema Maestro [Assessment] • Argue the correctness of algorithms using inductive proofs [Assessment]

Readings : [KT05], [DPV06], [Cor+09], [SF13], [Knu97]

Unit 2: Estrategias Algorítmicas (30)

Competences Expected:

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

Readings : [KT05], [DPV06], [Cor+09], [Als99]

Unit 3: Algoritmos y Estructuras de Datos fundamentales (6)	
Competences Expected:	
Topics	**Learning Outcomes**

• Grafos y algoritmos en grafos:
  – Problema de corte máximo y mínimo
  – Busqueda local
• Cache oblivious algorithms
• Number theory and cryptography
	• Discutir factores otros que no sean eficiencia computacional que influyan en la elección de algoritmos, tales como tiempo de programación, mantenibilidad, y el uso de patrones específicos de la aplicación en los datos de entrada [Familiarity]
	• Resolver problemas usando algoritmos básicos de grafos, incluyendo busqueda por profundidad y busqueda por amplitud [Assessment]
	• Demostrar habilidad para evaluar algoritmos, para seleccionar de un rango de posibles opciones, para proveer una justificación por esa selección,y para implementar el algoritmo en un contexto en específico [Assessment]
	• Resolver problemas usando algoritmos de grafos, incluyendo camino más corto de una sola fuente y camino más corto de todos los pares, y como mínimo un algoritmo de arbol de expansion minima [Assessment]

Readings : [KT05], [DPV06], [Cor+09], [SW11], [GT09]	

Unit 4: Computabilidad y complejidad básica de autómatas (2)	
Competences Expected:	
Topics	**Learning Outcomes**
• Introducción a las clases P y NP y al problema P vs. NP. • Introducción y ejemplos de problemas NP- Completos y a clases NP-Completos. • Reductions	• Define las clases P y NP [Familiarity] • Explique el significado de NP-Completitud [Familiarity]
Readings : [KT05], [DPV06], [Cor+09]	

Unit 5: Estructuras de Datos Avanzadas y Análisis de Algoritmos (12)	
Competences Expected:	
Topics	**Learning Outcomes**
• Grafos (ej. Ordenamiento Topológico, encontrando componentes puertemente conectados) • Algoritmos aleatorios. • Análisis amortizado. • Análisis Probabilístico. • Approximation Algorithms • Linear Programming	• Entender el mapeamento de problemas del mundo real a soluciones algorítmicas (ejemplo, problemas de grafos, programas lineares,etc) [Familiarity] • Seleccionar y aplicar técnicas avanzadas de análisis (ejemplo, amortizado, probabilistico,etc) para algoritmos [Usage]
Readings : [KT05], [DPV06], [Cor+09], [Tar83], [Raw92]	

8. WORKPLAN

8.1 Methodology

Individual and team participation is encouraged to present their ideas, motivating them with additional points in the different stages of the course evaluation.

8.2 Theory Sessions

The theory sessions are held in master classes with activities including active learning and roleplay to allow students to internalize the concepts.

8.3 Practical Sessions

The practical sessions are held in class where a series of exercises and/or practical concepts are developed through problem solving, problem solving, specific exercises and/or in application contexts.

9. EVALUATION SYSTEM

********* EVALUATION MISSING ********

10. BASIC BIBLIOGRAPHY

[Als99] H. Alsuwaiyel. *Algorithms: Design Techniques and Analysis*. World Scientific, 1999. ISBN: 9789810237400.

[Cor+09] Thomas H. Cormen et al. *Introduction to Algorithms, Third Edition*. 3rd. The MIT Press, 2009. ISBN: 0262033844.

[DPV06] S. Dasgupta, C. Papadimitriou, and U. Vazirani. *Algorithms*. McGraw-Hill Education, 2006. ISBN: 9780073523408.

[GT09] Michael T. Goodrich and Roberto Tamassia. *Algorithm Design: Foundations, Analysis and Internet Examples*. 2nd. John Wiley & Sons, Inc., 2009. ISBN: 0470088540, 9780470088548.

[Knu97] D.E. Knuth. *The Art of Computer Programming: Fundamental algorithms*. v. 1. Addison-Wesley, 1997. ISBN: 9780201896831.

[KT05] Jon Kleinberg and Eva Tardos. *Algorithm Design*. Addison-Wesley Longman Publishing Co., Inc., 2005. ISBN: 0321295358.

[Raw92] G.J.E. Rawlins. *Compared to What?: An Introduction to the Analysis of Algorithms*. Computer Science Press, 1992. ISBN: 9780716782438.

[SF13] R. Sedgewick and P. Flajolet. *An Introduction to the Analysis of Algorithms*. Pearson Education, 2013. ISBN: 9780133373486.

[SW11] R. Sedgewick and K. Wayne. *Algorithms*. Pearson Education, 2011. ISBN: 9780132762564.

[Tar83] Robert Endre Tarjan. *Data Structures and Network Algorithms*. Society for Industrial and Applied Mathematics, 1983. ISBN: 0-89871-187-8.

National University of Copsta Rica (UNA)
School of Informatics
Sillabus 2024-I

1. COURSE

CS231. Networking and Communication (Mandatory)

2. GENERAL INFORMATION

2.1 Course	:	CS231. Networking and Communication
2.2 Semester	:	5^{to} Semestre.
2.3 Credits	:	3
2.4 Horas	:	2 HT; 2 HP;
2.5 Duration of the period	:	16 weeks
2.6 Type of course	:	Mandatory
2.7 Learning modality	:	Face to face
2.8 Prerrequisites	:	CS2S1. Operating systems . (4^{th} Sem) CS2S1. Operating systems . (4^{th} Sem)

3. PROFESSORS

Meetings after coordination with the professor

4. INTRODUCTION TO THE COURSE

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

5. 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. COMPETENCES

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. TOPICS

Unit 1: Introducción (5)	
Competences Expected:	
Topics	**Learning Outcomes**
• Organización de la Internet (proveedores de servicios de Internet, proveedores de contenido, etc) • Técnicas de Switching (por ejemplo, de circuitos, de paquetes) • Piezas físicas de una red, incluidos hosts, routers, switches, ISPs, inalámbrico, LAN, punto de acceso y firewalls. • Principios de capas (encapsulación, multiplexación) • Roles de las diferentes capas (aplicación, transporte, red, enlace de datos, física)	• Articular la organización de la Internet [Familiarity] • Listar y definir la terminología de red apropiada [Familiarity] • Describir la estructura en capas de una arquitectura típica en red [Familiarity] • Identificar los diferentes tipos de complejidad en una red (bordes, núcleo, etc.) [Familiarity]
Readings : [KR13]	

Unit 2: Aplicaciones en red (5)	
Competences Expected:	
Topics	**Learning Outcomes**
• Esquemas de denominación y dirección (DNS, direcciones IP, identificadores de recursos uniformes, etc) • Las aplicaciones distribuidas (cliente / servidor, peer-to-peer, nube, etc) • HTTP como protocolo de capa de aplicación . • Multiplexación con TCP y UDP • API de Socket	• Listar las diferencias y las relaciones entre los nombres y direcciones en una red [Familiarity] • Definir los principios detrás de esquemas de denominación y ubicación del recurso [Familiarity] • Implementar una aplicación simple cliente-servidor basada en *sockets* [Usage]
Readings : [KR13]	

Unit 3: Entrega confiable de datos (10)	
Competences Expected:	
Topics	**Learning Outcomes**
• Control de errores (técnicas de retransmisión, temporizadores) • El control de flujo (agradecimientos, ventana deslizante) • Problemas de rendimiento (pipelining) • TCP	• Describir el funcionamiento de los protocolos de entrega fiables [Familiarity] • Listar los factores que afectan al rendimiento de los protocolos de entrega fiables [Familiarity] • Diseñar e implementar un protocolo confiable simple [Usage]
Readings : [KR13]	

Unit 4: Ruteo y reenvío (12)	
Competences Expected:	
Topics	**Learning Outcomes**
• Enrutamiento vs reenvío . • Enrutamiento estático . • Protocolo de Internet (IP) • Problemas de escalabilidad (direccionamiento jerárquico)	• Describir la organización de la capa de red [Familiarity] • Describir cómo los paquetes se envían en una red IP [Familiarity] • Listar las ventajas de escalabilidad de direccionamiento jerárquico [Familiarity]
Readings : [KR13]	

Unit 5: Redes de área local (10)	
Competences Expected:	
Topics	**Learning Outcomes**
• Problemas de Acceso Múltiple. • Enfoques comunes a Acceso múltiple (exponencial backoff, multiplexación por división de tiempo, etc) • Redes de área local . • Ethernet . • Switching .	• Describir como los paquetes son enviados en una red Ethernet [Familiarity] • Describir las relaciones entre IP y Ethernet [Familiarity] • Describir las etapas usadas en un enfoque común para el problema de múltiples accesos [Familiarity]
Readings : [KR13]	

Unit 6: Asignación de recursos (12)	
Competences Expected:	
Topics	**Learning Outcomes**
• Necesidad de asignación de recursos . • Asignación fija (TDM, FDM, WDM) versus la asignación dinámica . • De extremo a extremo frente a las red de enfoque asistida . • Justicia. • Principios del control de congestión. • Enfoques para la congestión (por ejemplo, redes de distribución de contenidos)	• Describir como los recursos pueden ser almacenados en la red [Familiarity] • Describir los problemas de congestión en una red grande [Familiarity] • Comparar y contrastar las técnicas de almacenamiento estático y dinámico [Familiarity] • Comparar y contrastar los enfoques actuales de la congestión [Familiarity]
Readings : [KR13]	

Unit 7: Celulares (5)	
Competences Expected:	
Topics	**Learning Outcomes**
• Principios de redes celulares. • Redes 802.11 • Problemas en el apoyo a los nodos móviles (agente local)	• Describir la organización de una red inalambrica [Familiarity] • Describir como las redes inalámbricas soportan usuarios móviles [Familiarity]
Readings : [KR13], [Cha16]	

Unit 8: Redes sociales (5)	
Competences Expected:	
Topics	**Learning Outcomes**
• Panorama de las redes sociales. • Ejemplo plataformas de redes sociales. • Estructura de los grafos de redes sociales. • Análisis de redes sociales.	• Discutir los principios fundamentales(como pertenencia, confianza) de una red social [Familiarity] • Describir como redes sociales existentes operan [Familiarity] • Construir un grafo de una red social a partir de datos de la red [Usage] • Analizar una red social para determinar quienes son las personas importantes [Usage] • Evaluar una determinada interpretación de una pregunta de red social con los datos asociados [Familiarity]
Readings : [KR13], [Kad11]	

8. WORKPLAN

8.1 Methodology

Individual and team participation is encouraged to present their ideas, motivating them with additional points in the different stages of the course evaluation.

8.2 Theory Sessions

The theory sessions are held in master classes with activities including active learning and roleplay to allow students to internalize the concepts.

8.3 Practical Sessions

The practical sessions are held in class where a series of exercises and/or practical concepts are developed through problem solving, problem solving, specific exercises and/or in application contexts.

9. EVALUATION SYSTEM

********* EVALUATION MISSING ********

10. BASIC BIBLIOGRAPHY

[Cha16] Paresh Chayapathi Rajendra; Syed F. Hassan; Shah. *Network Functions Virtualization (NFV) with a Touch of SDN*. Addison-Wesley Professional; 1 edition, 2016. ISBN: 978-0134463056.

[Kad11] Charles Kadushin. *Understanding Social Networks: Theories, Concepts, And Findings*. Oxford University Press, Usa; 1 edition, 2011. ISBN: 978-0195379471.

[KR13] J.F. Kurose and K.W. Ross. *Computer Networking: A Top-down Approach*. 7th. Always learning. Pearson, 2013. ISBN: 978-0133594140.

National University of Copsta Rica (UNA)
School of Informatics
Sillabus 2024-I

1. COURSE

CS271. Databases I (Mandatory)

2. GENERAL INFORMATION

2.1 Course	:	CS271. Databases I
2.2 Semester	:	5^{to} Semestre.
2.3 Credits	:	3
2.4 Horas	:	2 HT; 2 HP;
2.5 Duration of the period	:	16 weeks
2.6 Type of course	:	Mandatory
2.7 Learning modality	:	Face to face
2.8 Prerrequisites	:	CS210. Algorithms and Data Structures. (4^{th} Sem) CS210. Algorithms and Data Structures. (4^{th} Sem)

3. PROFESSORS

Meetings after coordination with the professor

4. INTRODUCTION TO THE COURSE

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.

5. 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. COMPETENCES

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. TOPICS

Unit 1: Sistemas de Bases de Datos (14)	
Competences Expected:	
Topics	**Learning Outcomes**
• Enfoque y Evolución de Sistemas de Bases de Datos. • Componentes del Sistema de Bases de Datos. • Diseño de las funciones principales de un DBMS. • Arquitectura de base de datos e independencia de datos. • Uso de un lenguaje de consulta declarativa. • Sistemas de apoyo a contenido estructurado y / o corriente. • Enfoques para la gestión de grandes volúmenes de datos (por ejemplo, sistemas de bases de datos NoSQL, uso de MapReduce).	• Explica las características que distinguen un esquema de base de datos de aquellos basados en la programación de archivos de datos [Usage] • Describe los diseños más comunes para los componentes base de sistemas de bases de datos incluyendo el optimizador de consultas, ejecutor de consultas, administrador de almacenamiento, métodos de acceso y procesador de transacciones [Usage] • Cita las metas básicas, funciones y modelos de un sistema de bases de datos [Usage] • Describe los componentes de un sistema de bases datos y da ejemplos de su uso [Usage] • Identifica las funciones principales de un SGBD y describe sus roles en un sistema de bases de datos [Usage] • Explica los conceptos de independencia de datos y su importancia en un sistema de bases de datos [Usage] • Usa un lenguaje de consulta declarativo para recoger información de una base de datos [Usage] • Describe las capacidades que las bases de datos brindan al apoyar estructuras y/o la secuencia de flujo de datos, ejm. texto [Usage] • Describe los enfoques principales para almacenar y procesar larges volúmenes de datos [Usage]
Readings : [RC04], [EN04], [RG03], [ER15], [CJ11], [KS02]	

Unit 2: Modelado de datos (14)	
Competences Expected:	
Topics	**Learning Outcomes**
• Modelado de datos • Modelos conceptuales (e.g., entidad-relación, diagramas UML) • Modelos de hoja de cálculo • Modelos Relacionales. • Modelos orientados a objetos. • Modelos de datos semi-estructurados (expresados usando DTD o XML Schema, por ejemplo)	• Compare y contrasta modelos apropiados de datos, incluyendo estructuras sus estructuras internas, para diversos tipos de datos [Usage] • Describe los conceptos en notación de modelos (ejm. Diagramas Entidad-Relación o UML) y cómo deben de ser usados [Usage] • Define la terminología fundamental a ser usada en un modelo relacional de datos [Usage] • Describe los principios básicos del modelo relacional de datos [Usage] • Aplica los conceptos de modelado y la notación de un modelo relacional de datos [Usage] • Describe los conceptos principales del modelado OO como son identidad de objetos, constructores de tipos, encapsulación, herencia, polimorfismo, y versiones [Usage] • Describe las diferencias entre modelos de datos relacionales y semi-estructurados [Usage] • Da una semi estructura equivalente (ejm. en DTD o Esquema XML) para un esquema relacional dado [Usage]
Readings : [SW04], [EN04], [KS02]	

Unit 3: Indexación (4)	
Competences Expected:	
Topics	**Learning Outcomes**
• El impacto de indices en el rendimiento de consultas. • La estructura basica de un indice. • Mantener un buffer de datos en memoria. • Creando indices con SQL. • Indexando texto. • Indexando la web (e.g., web crawling)	• Generar un archivo índice para una colección de recursos [Usage] • Explicar la función de un índice invertido en la localización de un documento en una colección [Usage] • Explicar cómo rechazar y detener palabras que afectan a la indexación [Usage] • Identificar los índices adecuados para determinado el esquema relacional y el conjunto de consultas [Usage] • Estimar el tiempo para recuperar información, cuando son usados los índices comparado con cuando no son usados [Usage] • Describir los desafíos claves en el rastreo web, por ejemplo, la detección de documentos duplicados, la determinación de la frontera de rastreo [Usage]
Readings : [WM01], [RG03], [ER15], [CJ11], [KS02]	

Unit 4: Bases de Datos Relacionales (14)

Competences Expected:

Topics	Learning Outcomes
• Mapeo de esquemas conceptuales a esquemas relacionales. • Entidad y integridad referencial. • Algebra relacional y calculo relacional. • Diseño de bases de datos relacionales. • Dependencia funcional. • Descomposición de un esquema. • Llaves candidatas, SuperLlaves y cierre de un conjunto de atributos. • Formas Normales (BCNF) • Dependencias multi-valoradas (4NF) • Uniendo dependencias (PJNF, 5NF) • Teoría de la representación.	• Prepara un esquema relacional de un modelo conceptual desarrollado usando el modelo entidad-relación [Usage] • Explica y demuestra los conceptos de restricciones de integridad de la entidad e integridad referencial (incluyendo la definición del concepto de clave foránea) [Usage] • Demuestra el uso de las operaciones de álgebra relacional de la teoría matemática de conjuntos (unión, intersección, diferencia, y producto Cartesiano) y de las operaciones de álgebra relacional desarrolladas específicamente para las bases de datos relacionales (selección (restringida), proyección, unión y división) [Usage] • Escribe consultas en álgebra relacional [Usage] • Escribe consultas en cálculo relacional de tuplas [Usage] • Determina la dependencia funcional entre dos o más atributos que son subconjunto de una relación [Usage] • Conecta restricciones expresadas como clave primaria y foránea, con dependencias funcionales [Usage] • Calcula la cerradura de un conjunto de atributos dado dependencias funcionales [Usage] • Determina si un conjunto de atributos forma una superclave y/o una clave candidata de una relación dada dependencias funcionales [Usage] • Evalua una descomposición propuesta, a fin de determinar si tiene una unión sin pérdidas o preservación de dependencias [Usage] • Describe las propiedades de la FNBC, FNUP (forma normal unión de proyecto), 5FN [Usage] • Explica el impacto de la normalización en la eficacia de las operaciones de una base de datos especialmente en la optimización de consultas [Usage] • Describe que es una dependencia de multi valor y cual es el tipo de restricciones que especifica [Usage]

Readings : [WM01], [RG03], [ER15], [CJ11], [KS02]

Unit 5: Lenguajes de Consulta (12)	
Competences Expected:	
Topics	**Learning Outcomes**
• Visión general de lenguajes de base de datos. • SQL (definición de datos, formulacion de consultas, sublenguaje update, restricciones, integridad) • Selecciones • Proyecciones • Select-project-join • Agregaciones y agrupaciones. • Subconsultas. • Entornos QBE de cuarta generación. • Diferentes maneras de invocar las consultas no procedimentales en lenguajes convencionales. • Introducción a otros lenguajes importantes de consulta (por ejemplo, XPATH, SPARQL) • Procedimientos almacenados.	• Crear un esquema relacional de bases de datos en SQL que incorpora restricciones clave y restricciones de integridad de entidad e integridad referencial [Usage] • Usar SQL para crear tablas y devuelve (SELECT) la información de una base de datos [Usage] • Evaluar un conjunto de estrategias de procesamiento de consultas y selecciona la estrategia óptima [Usage] • Crear una consulta no-procedimental al llenar plantillas de relacines para construir un ejemplo del resultado de una consulta requerida [Usage] • Adicionar consultas orientadas a objetos en un lenguaje stand-alone como C++ o Java (ejm. SELECT ColMethod() FROM Objeto) [Usage] • Escribe un procedimiento almacenado que trata con parámetros y con algo de flujo de control de tal forma que tenga funcionalidad [Usage]
Readings : [Die01], [EN04], [Cel05], [KS02]	

8. WORKPLAN

8.1 Methodology

Individual and team participation is encouraged to present their ideas, motivating them with additional points in the different stages of the course evaluation.

8.2 Theory Sessions

The theory sessions are held in master classes with activities including active learning and roleplay to allow students to internalize the concepts.

8.3 Practical Sessions

The practical sessions are held in class where a series of exercises and/or practical concepts are developed through problem solving, problem solving, specific exercises and/or in application contexts.

9. EVALUATION SYSTEM

********* EVALUATION MISSING ********

10. BASIC BIBLIOGRAPHY

[Cel05] Joe Celko. *Joe Celko's SQL Programming Style*. Elsevier, 2005.

[CJ11] Date C.J. *SQL and Relational Theory: How to Write Accurate SQL Code*. O'Reilly Media, 2011.

[Die01] Suzanne W Dietrich. *Understanding Relational Database Query Languages, First Edition*. Prentice Hall, 2001.

[EN04] Ramez Elmasri and Shamkant B. Navathe. *Fundamentals of Database Systems, Fourth Edition*. Addison Wesley, 2004.

[ER15] Jim Webber Emil Eifrem and Ian Robinson. *Graph Databases*. 2nd. O'Reilly Media, 2015.

[KS02] Henry F. Korth and Abraham Silberschatz. *Fundamentos de Base de Datos*. McGraw-Hill, 2002.

[RC04] Peter Rob and Carlos Coronel. *Database Systems: Design, Implementation and Management, Sixth Edition*. Morgan Kaufmann, 2004.

[RG03] Raghu Ramakrishnan and Johannes Gehrke. *Database Management Systems*. 3rd. McGraw-Hill, 2003.

[SW04] Graeme Simsion and Graham Witt. *Data Modeling Essentials, Third Edition*. Morgan Kaufmann, 2004.

[WM01] Mark Whitehorn and Bill Marklyn. *Inside Relational Databases, Second Edition*. Springer, 2001.

National University of Copsta Rica (UNA)
School of Informatics
Sillabus 2024-I

1. COURSE

CS3I2. Computer Security (Mandatory)

2. GENERAL INFORMATION

2.1 Course : CS3I2. Computer Security
2.2 Semester : 5^{to} Semestre.
2.3 Credits : 3
2.4 Horas : 2 HT; 2 HP;

2.5 Duration of the period : 16 weeks
2.6 Type of course : Mandatory
2.7 Learning modality : Face to face
2.8 Prerrequisites :

- CS113. Programming II. (3^{rd} Sem)
- CS221. Computer Systems Architecture. (3^{rd} Sem)
- CS113. Programming II. (3^{rd} Sem)
- CS221. Computer Systems Architecture. (3^{rd} Sem)

3. PROFESSORS

Meetings after coordination with the professor

4. INTRODUCTION TO THE COURSE

Write justification for this course here ...

5. GOALS

- Write your first goal here.
- Write your second goal here.
- Just in case you need more goals write them here

6. COMPETENCES

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

7. TOPICS

Unit 1: title for the unit goes here (5)	
Competences Expected:	
Topics	**Learning Outcomes**
• Topic1 • Topic2 • Topic3	• Learning outcome1 [Levelforthislearningoutcome]. • Apply computing in complex problems [Usage]. • Create a search engine [Assessment]. • Study data structures [Familiarity].
Readings : [**Bibitem1**], [**Bibitem2**]	

Unit 2: another unit goes here (1)	
Competences Expected:	
Topics	**Learning Outcomes**
• Topic1	• Learning outcome xyz [Levelforthislearningoutcome].
Readings : [**Bibitem3**], [**Bibitem1**]	

8. WORKPLAN

8.1 Methodology

Individual and team participation is encouraged to present their ideas, motivating them with additional points in the different stages of the course evaluation.

8.2 Theory Sessions

The theory sessions are held in master classes with activities including active learning and roleplay to allow students to internalize the concepts.

8.3 Practical Sessions

The practical sessions are held in class where a series of exercises and/or practical concepts are developed through problem solving, problem solving, specific exercises and/or in application contexts.

9. EVALUATION SYSTEM

********* EVALUATION MISSING ********

10. BASIC BIBLIOGRAPHY

National University of Copsta Rica (UNA)
School of Informatics
Sillabus 2024-I

1. COURSE

SE201. Elective I (Elective)

2. GENERAL INFORMATION

2.1 Course	:	SE201. Elective I
2.2 Semester	:	5^{to} Semestre.
2.3 Credits	:	3
2.4 Horas	:	4 HT;
2.5 Duration of the period	:	16 weeks
2.6 Type of course	:	Elective
2.7 Learning modality	:	Face to face
2.8 Prerrequisites	:	None None

3. PROFESSORS

Meetings after coordination with the professor

4. INTRODUCTION TO THE COURSE

Write justification for this course here ...

5. GOALS

- Write your first goal here.
- Write your second goal here.
- Just in case you need more goals write them here

6. COMPETENCES

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

7. TOPICS

Unit 1: title for the unit goes here (5)	
Competences Expected:	
Topics	**Learning Outcomes**
• Topic1 • Topic2 • Topic3	• Learning outcome1 [Levelforthislearningoutcome]. • Apply computing in complex problems [Usage]. • Create a search engine [Assessment]. • Study data structures [Familiarity].
Readings : [**Bibitem1**], [**Bibitem2**]	

Unit 2: another unit goes here (1)	
Competences Expected:	
Topics	**Learning Outcomes**
• Topic1	• Learning outcome xyz [Levelforthislearningoutcome].
Readings : [**Bibitem3**], [**Bibitem1**]	

8. WORKPLAN

8.1 Methodology

Individual and team participation is encouraged to present their ideas, motivating them with additional points in the different stages of the course evaluation.

8.2 Theory Sessions

The theory sessions are held in master classes with activities including active learning and roleplay to allow students to internalize the concepts.

8.3 Practical Sessions

The practical sessions are held in class where a series of exercises and/or practical concepts are developed through problem solving, problem solving, specific exercises and/or in application contexts.

9. EVALUATION SYSTEM

********* EVALUATION MISSING ********

10. BASIC BIBLIOGRAPHY

National University of Copsta Rica (UNA)
School of Informatics
Sillabus 2024-I

1. COURSE

CS3I1. Computer Security (Mandatory)

2. GENERAL INFORMATION

2.1 Course	:	CS3I1. Computer Security
2.2 Semester	:	6^{to} Semestre.
2.3 Credits	:	3
2.4 Horas	:	2 HT; 2 HP;
2.5 Duration of the period	:	16 weeks
2.6 Type of course	:	Mandatory
2.7 Learning modality	:	Face to face
2.8 Prerrequisites	:	CS231. Networking and Communication. (5^{th} Sem) CS231. Networking and Communication. (5^{th} Sem)

3. PROFESSORS

Meetings after coordination with the professor

4. INTRODUCTION TO THE COURSE

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.

5. 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. COMPETENCES

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. TOPICS

Unit 1: Fundamentos y Conceptos en Seguridad (25)	
Competences Expected:	
Topics	**Learning Outcomes**
• CIA (Confidencialidad, Integridad, Disponibilidad) • Conceptos de riesgo, amenazas, vulnerabilidades, y los tipos de ataque . • Autenticación y autorización, control de acceso (vs. obligatoria discrecional) • Concepto de la confianza y la honradez . • Ética (revelación responsable)	• Analizar las ventajas y desventajas de equilibrar las propiedades clave de seguridad(Confidenciabilidad, Integridad, Disponibilidad) [Familiarity] • Describir los conceptos de riesgo, amenazas, vulnerabilidades y vectores de ataque(incluyendo el hecho de que no existe tal cosa como la seguridad perfecta) [Familiarity] • Explicar los conceptos de autentificación, autorización, control de acceso [Familiarity] • Explicar el concepto de confianza y confiabilidad [Familiarity] • Reconocer de que hay problemas éticos más importantes que considerar en seguridad computacional, incluyendo problemas éticos asociados a arreglar o no arreglar vulnerabilidades y revelar o no revelar vulnerabilidades [Familiarity]
Readings : [WL14]	

Unit 2: Principios de Diseño Seguro (25)	
Competences Expected:	
Topics	**Learning Outcomes**
• Menor privilegio y aislamiento. • Valores predeterminados a prueba de fallos. • Diseño abierto. • La seguridad de extremo a extremo. • La defensa en profundidad (por ejemplo, la programación defensiva, defensa en capas) • Diseño de seguridad. • Las tensiones entre la seguridad y otros objetivos de diseño. • Mediación completa. • El uso de componentes de seguridad vetados. • Economía del mecanismo (la reducción de la base informática de confianza, minimizar la superficie de ataque) • Seguridad utilizable. • Componibilidad de seguridad. • Prevención, detección y disuasión.	• Describir el principio de privilegios mínimos y el aislamiento que se aplican al diseño del sistema [Familiarity] • Resumir el principio de prueba de fallos y negar por defecto [Familiarity] • Discutir las implicaciones de depender de diseño abierto o secreto de diseño para la seguridad [Familiarity] • Explicar los objetivos de seguridad de datos de extremo a extremo [Familiarity] • Discutir los beneficios de tener múltiples capas de defensas [Familiarity] • Por cada etapa en el ciclo de vida de un producto, describir que consideraciones de seguridad deberian ser evaluadas [Familiarity] • Describir el costo y ventajas y desventajas asociadas con el diseño de seguridad de un producto. [Familiarity] • Describir el concepto de mediación y el principio de mediación completa [Familiarity] • Conocer los componentes estándar para las operaciones de seguridad, en lugar de reinventar las operaciones fundamentales [Familiarity] • Explicar el concepto de computación confiable incluyendo base informática confiable y de la superficie de ataque y el principio de minimización de base informática confiable [Familiarity] • Discutir la importancia de la usabilidad en el diseño de mecanismos de seguridad [Familiarity] • Describir problemas de seguridad que surgen en los límites entre varios componentes [Familiarity] • Identificar los diferentes roles de mecanismos de prevención y mecanismos de eliminación/disuación [Familiarity]
Readings : [WL14]	

Unit 3: Programación Defensiva (25)	
Competences Expected:	
Topics	**Learning Outcomes**

• Validación de datos de entrada y sanitización
• Elección del lenguaje de programación y lenguajes con tipos de datos seguro.
• Ejemplos de validación de entrada de datos y sanitización de errores.
  – Desbordamiento de búfer
  – Errores enteros
  – Inyección SQL
  – Vulnerabilidad XSS
• Las condiciones de carrera.
• Manejo correcto de las excepciones y comportamientos inesperados.
• Uso correcto de los componentes de terceros.
• Desplegar eficazmente las actualizaciones de seguridad.
• Información de control de flujo.
• Generando correctamente el azar con fines de seguridad.
• Mecanismos para la detección y mitigación de datos de entrada y errores de sanitización.
• Fuzzing
• El análisis estático y análisis dinámico.
• Programa de verificación.
• Soporte del sistema operativo (por ejemplo, la asignación al azar del espacio de direcciones, canarios)
• El soporte de hardware (por ejemplo, el DEP, TPM)
	• Explicar por que la validación de entrada y desinfección de datos es necesario en el frente del control contencioso del canal de entrada [Usage]
	• Explicar por que uno deberia escoger para desallorrar un programa en un lenguaje tipo seguro como Java, en contraste con un lenguaje de programación no seguro como C/C++ [Usage]
	• Clasificar los errores de validación de entrada común, y escribir correctamente el código de validación de entrada [Usage]
	• Demostrar el uso de un lenguaje de programación de alto nivel cómo prevenir una condición de competencia que ocurran y cómo manejar una excepción [Usage]
	• Demostrar la identificación y el manejo elegante de las condiciones de error [Familiarity]
	• Explique los riesgos de mal uso de las interfaces con código de terceros y cómo utilizar correctamente el código de terceros [Familiarity]
	• Discutir la necesidad de actualizar el software para corregir las vulnerabilidades de seguridad y la gestión del ciclo de vida de la corrección [Familiarity]

Readings : [WL14]	

Unit 4: Ataques y Amenazas (25)	
Competences Expected:	
Topics	**Learning Outcomes**
• Atacante metas, capacidades y motivaciones (como economía sumergida, el espionaje digital, la guerra cibernética, las amenazas internas, hacktivismo, las amenazas persistentes avanzadas) • Los ejemplos de malware (por ejemplo, virus, gusanos, spyware, botnets, troyanos o rootkits) • Denegación de Servicio (DoS) y Denegación de Servicio Distribuida (DDoS) • Ingeniería social (por ejemplo, perscando) • Los ataques a la privacidad y el anonimato . • El malware / comunicaciones no deseadas, tales como canales encubiertos y esteganografía.	• Describir tipos de ataques similares en contra de un sistema en particular [Familiarity] • Discutir los limitantes de las medidas en contra del malware (ejm. detección basada en firmas, detección de comportamiento) [Familiarity] • Identificar las instancias de los ataques de ingeniería social y de los ataques de negación de servicios [Familiarity] • Discutir como los ataques de negación de servicos puede ser identificados y reducido [Familiarity] • Describir los riesgos de la privacidad y del anonimato en aplicaciones comunmente usadas [Familiarity] • Discutir los conceptos de conversión de canales y otros procedimientos de filtrado de datos [Familiarity]
Readings : [WL14]	

Unit 5: Seguridad de Red (25)	
Competences Expected:	
Topics	**Learning Outcomes**
• Red de amenazas y tipos de ataques específicos (por ejemplo, la denegación de servicio, spoofing, olfateando y la redirección del tráfico, el hombre en el medio, ataques integridad de los mensajes, los ataques de enrutamiento, y el análisis de tráfico) • El uso de cifrado de datos y seguridad de la red . • Arquitecturas para redes seguras (por ejemplo, los canales seguros, los protocolos de enrutamiento seguro, DNS seguro, VPN, protocolos de comunicación anónimos, aislamiento) • Los mecanismos de defensa y contramedidas (por ejemplo, monitoreo de red, detección de intrusos, firewalls, suplantación de identidad y protección DoS, honeypots, seguimientos) • Seguridad para redes inalámbricas, celulares . • Otras redes no cableadas (por ejemplo, ad hoc, sensor, y redes vehiculares) • Resistencia a la censura. • Gestión de la seguridad operativa de la red (por ejemplo, control de acceso a la red configure)	• Describir las diferentes categorías de amenazas y ataques en redes [Familiarity] • Describir las arquitecturas de criptografía de clave pública y privada y cómo las ICP brindan apoyo a la seguridad en redes [Familiarity] • Describir ventajas y limitaciones de las tecnologías de seguridad en cada capa de una torre de red [Familiarity] • Identificar los adecuados mecanismos de defensa y sus limitaciones dada una amenaza de red [Usage]
Readings : [WL14]	

Unit 6: Criptografía (25)

Competences Expected:

Topics	Learning Outcomes
• Terminología básica de criptografía cubriendo las nociones relacionadas con los diferentes socios (comunicación), canal seguro / inseguro, los atacantes y sus capacidades, cifrado, descifrado, llaves y sus características, firmas. • Tipos de cifrado (por ejemplo, cifrado César, cifrado affine), junto con los métodos de ataque típicas como el análisis de frecuencia. • Apoyo a la infraestructura de clave pública para la firma digital y el cifrado y sus desafíos. • Criptografía de clave simétrica: – El secreto perfecto y el cojín de una sola vez – Modos de funcionamiento para la seguridad semántica y encriptación autenticada (por ejemplo, cifrar-entonces-MAC, OCB, GCM) – Integridad de los mensajes (por ejemplo, CMAC, HMAC) • La criptografía de clave pública: – Permutación de trampilla, por ejemplo, RSA – Cifrado de clave pública, por ejemplo, el cifrado RSA, cifrado El Gamal – Las firmas digitales – Infraestructura de clave pública (PKI) y certificados – Supuestos de dureza, por ejemplo, Diffie-Hellman, factoring entero • Protocolos de intercambio de claves autenticadas, por ejemplo, TLS . • Primitivas criptográficas: – generadores pseudo-aleatorios y cifrados de flujo – cifrados de bloque (permutaciones pseudo-aleatorios), por ejemplo, AES – funciones de pseudo-aleatorios – funciones de hash, por ejemplo, SHA2, resistencia colisión – códigos de autenticación de mensaje – funciones derivaciones clave	• Describir el propósito de la Criptografía y listar formas en las cuales es usada en comunicación de datos [Familiarity] • Definir los siguientes términos: Cifrado, Criptoanálisis, Algorítmo Criptográfico, y Criptología y describe dos métodos básicos (cifrados) para transformar texto plano en un texto cifrado [Familiarity] • Discutir la importancia de los números primos en criptografía y explicar su uso en algoritmos criptográficos [Familiarity] • Ilustrar como medir la entropía y como generar aleatoriedad criptográfica [Usage] • Usa primitivas de clave pública y sus aplicaciones [Usage] • Explicar como los protocolos de intercambio de claves trabajan y como es que pueden fallar [Familiarity] • Discutir protocolos criptográficos y sus propiedades [Familiarity]

Readings : [WL14]

Unit 7: Seguridad en la Web (25)	
Competences Expected:	
Topics	**Learning Outcomes**
• Modelo de seguridad Web – Modelo de seguridad del navegador incluida la política de mismo origen – Los límites de confianza de cliente-servidor, por ejemplo, no pueden depender de la ejecución segura en el cliente • Gestión de sesiones, la autenticación: – Single Sign-On – HTTPS y certificados • Vulnerabilidades de las aplicaciones y defensas : – Inyección SQL – XSS – CSRF • Seguridad del lado del cliente : – Política de seguridad Cookies – Extensiones de seguridad HTTP, por ejemplo HSTS – Plugins, extensiones y aplicaciones web – Seguimiento de los usuarios Web • Herramientas de seguridad del lado del servidor, por ejemplo, los cortafuegos de aplicación Web (WAFS) y fuzzers	• Describe el modelo de seguridad de los navegadores incluyendo las políticas del mismo origen y modelos de amenazas en seguridad web [Familiarity] • Discutir los conceptos de sesiones web, canales de comunicación seguros tales como Seguridad en la Capa de Transporte(TLS) y la importancia de certificados de seguridad, autenticación incluyendo inicio de sesión único, como OAuth y Lenguaje de Marcado para Confirmaciones de Seguridad($SAML$) [Familiarity] • Investigar los tipos comunes de vulnerabilidades y ataques en las aplicaciones web, y defensas contra ellos [Familiarity] • Utilice las funciones de seguridad del lado del cliente [Usage]
Readings : [WL14]	

Unit 8: Seguridad de plataformas (25)	
Competences Expected:	
Topics	**Learning Outcomes**
• Integridad de código y firma de código. • Arranque seguro, arranque medido, y la raíz de confianza. • Testimonio. • TPM y coprocesadores seguros. • Las amenazas de seguridad de los periféricos, por ejemplo, DMA, IOMMU. • Ataques físicos: troyanos de hardware, sondas de memoria, ataques de arranque en frío. • Seguridad de dispositivos integrados, por ejemplo, dispositivos médicos, automóviles. • Ruta confiable.	• Explica el concepto de integridad de código y firma de códigos, así como el alcance al cual se aplica [Familiarity] • Discute los conceptos del origen de la confidencialidad y el de los procesos de arranque y carga segura [Familiarity] • Describe los mecanismos de arresto remoto de la integridad de un sistema [Familiarity] • Resume las metas y las primitivas claves de los modelos de plataforma confiable (TPM) [Familiarity] • Identifica las amenazas de conectar periféricos en un dispositivo [Familiarity] • Identifica ataques físicos y sus medidas de control [Familiarity] • Identifica ataques en plataformas con hardware que no son del tipo PC [Familiarity] • Discute los conceptos y la importancia de ruta confiable [Familiarity]
Readings : [WL14]	

Unit 9: Investigación digital (Digital Forensics) (25)	
Competences Expected:	
Topics	**Learning Outcomes**
• Principios básicos y metodologías de análisis digital forensico. • Diseñar sistemas con necesidades forenses en mente. • Reglas de Evidencia - conceptos generales y las diferencias entre las jurisdicciones y la Cadena de Custodia. • Búsqueda y captura de comprobación: requisitos legales y de procedimiento. • Métodos y normas de evidencia digital. • Las técnicas y los estándares para la conservación de los datos. • Cuestiones legales y reportes incluyendo el trabajo como perito. • Investigación digital de los sistema de archivos. • Los forenses de aplicación. • Investigación digital en la web. • Investigación digital en redes. • Investigación digital en dispositivos móviles. • Ataques al computador/red/sistema. • Detección e investigación de ataque. • Contra investigación digital.	• Describe qué es una investigación digital, las fuentes de evidencia digital, y los límites de técnicas forenses [Familiarity] • Explica como diseñar software de apoyo a técnicas forenses [Familiarity] • Describe los requisitos legales para usar datos recuperados [Familiarity] • Describe el proceso de recolección de evidencia desde el tiempo en que se identifico el requisito hasta la colocación de los datos [Familiarity] • Describe como se realiza la recolección de datos y el adecuado almacenamiento de los datos originales y de la copia forense [Familiarity] • Realiza recolección de datos en un disco duro [Usage] • Describe la responsabilidad y obligación de una persona mientras testifica como un examinador forense [Familiarity] • Recupera datos basados en un determinado término de búsqueda en una imagen del sistema [Usage] • Reconstruye el historial de una aplicación a partir de los artefactos de la aplicación [Familiarity] • Reconstruye el historial de navegación web de los artefactos web [Familiarity] • Captura e interpreta el tráfico de red [Familiarity] • Discute los retos asociados con técnicas forenses de dispositivos móviles [Familiarity]
Readings : [WL14]	

Unit 10: Seguridad en Ingeniería de Software (25)	
Competences Expected:	
Topics	**Learning Outcomes**
• La construcción de la seguridad en el ciclo de vida de desarrollo de software. • Principios y patrones de diseño seguros. • Especificaciones de software seguros y requisitos. • Prácticas de desarrollo de software de seguros. • Asegure probar el proceso de las pruebas de que se cumplan los requisitos de seguridad (incluyendo análisis estático y dinámico)	• Describir los requisitos para la integración de la seguridad en el SDL [Familiarity] • Aplicar los conceptos de los principios de diseño para mecanismos de protección, los principios para seguridad de software (Viega and McGraw) y los principios de diseño de seguridad (Morrie Gasser) en un proyecto de desarrollo de software [Familiarity] • Desarrollar especificaciones para un esfuerzo de desarrollo de software que especifica completamente los requisitos funcionales y se identifican las rutas de ejecución esperadas [Familiarity]
Readings : [WL14]	

8. WORKPLAN

8.1 Methodology

Individual and team participation is encouraged to present their ideas, motivating them with additional points in the different stages of the course evaluation.

8.2 Theory Sessions

The theory sessions are held in master classes with activities including active learning and roleplay to allow students to internalize the concepts.

8.3 Practical Sessions

The practical sessions are held in class where a series of exercises and/or practical concepts are developed through problem solving, problem solving, specific exercises and/or in application contexts.

9. EVALUATION SYSTEM

********* EVALUATION MISSING ********

10. BASIC BIBLIOGRAPHY

[WL14] Stallings. W and Brown. L. *Computer Security: Principles and Practice*. Pearson Education, Limited, 2014. ISBN: 9780133773927.

National University of Copsta Rica (UNA)
School of Informatics
Sillabus 2024-I

1. COURSE

CS3P1. Parallel and Distributed Computing (Mandatory)

2. GENERAL INFORMATION

2.1 Course	:	CS3P1. Parallel and Distributed Computing
2.2 Semester	:	6^{to} Semestre.
2.3 Credits	:	3
2.4 Horas	:	2 HT; 2 HP;
2.5 Duration of the period	:	16 weeks
2.6 Type of course	:	Mandatory
2.7 Learning modality	:	Face to face
2.8 Prerrequisites	:	CS2S1. Operating systems . (4^{th} Sem) CS2S1. Operating systems . (4^{th} Sem)

3. PROFESSORS

Meetings after coordination with the professor

4. INTRODUCTION TO THE COURSE

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.

5. 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. COMPETENCES

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. TOPICS

Unit 1: Fundamentos de paralelismo (18)	
Competences Expected:	
Topics	**Learning Outcomes**
• Procesamiento Simultáneo Múltiple. • Metas del Paralelismo (ej. rendimineto) frente a Concurrencia (ej. control de acceso a recursos compartidos) • Paralelismo, comunicación, y coordinación: – Paralelismo, comunicación, y coordinación – Necedidad de Sincronización • Errores de Programación ausentes en programación secuencial: – Tipos de Datos (lectura/escritura simultánea o escritura/escritura compartida) – Tipos de Nivél más alto (interleavings violating program intention, no determinismo no deseado) – Falta de vida/progreso (deadlock, starvation)	• Distinguir el uso de recursos computacionales para una respuesta mas rápida para administrar el acceso eficiente a un recurso compartido [Familiarity] • Distinguir múltiples estructuras de programación suficientes para la sincronización que pueden ser interimplementables pero tienen ventajas complementarias [Familiarity] • Distinguir datos de carrera (*data races*) a partir de carreras de mas alto nivel [Familiarity]
Readings : [Pac11], [Mat14], [**quinnz**], [Geo10]	

Unit 2: Arquitecturas paralelas (12)

Competences Expected:

Topics	Learning Outcomes

• Procesadores mutlinúcleo.
• Memoria compartida vs memoria distribuida.
• Multiprocesamiento simétrico.
• SIMD, procesamiento de vectores.
• GPU, coprocesamiento.
• Taxonomia de Flynn.
• Soporte a nivel de instrucciones para programación paralela.
  – Instrucciones atómicas como Compare/Set (Comparar / Establecer)
• Problemas de Memoria:
  – Caches multiprocesador y coherencia de cache
  – Acceso a Memoria no uniforme (NUMA)
• Topologías.
  – Interconecciones
  – Clusters
  – Compartir recursos (p.e., buses e interconexiones)
	• Explicar las diferencias entre memoria distribuida y memoria compartida [Assessment]
	• Describir la arquitectura SMP y observar sus principales caracteristicas [Assessment]
	• Distinguir los tipos de tareas que son adecuadas para máquinas SIMD [Usage]
	• Describir las ventajas y limitaciones de GPUs vs CPUs [Usage]
	• Explicar las caracteristicas de cada clasificación en la taxonomía de Flynn [Usage]
	• Describir los desafíos para mantener la coherencia de la caché [Familiarity]
	• Describir los desafíos clave del desempeño en diferentes memorias y topologías de sistemas distribuidos [Familiarity]

Readings : [Pac11], [KH13], [SK10], [Geo10]

Unit 3: Descomposición en paralelo (18)

Competences Expected:

Topics	Learning Outcomes

• Necesidad de Comunicación y coordinación/sincronización.
• Independencia y Particionamiento.
• Conocimiento Básico del Concepto de Descomposición Paralela.
• Decomposición basada en tareas:
  – Implementación de estrategias como hebras
• Descomposición de Información Paralela
  – Estrategias como SIMD y MapReduce
• Actores y Procesos Reactivos (solicitud de gestores)
	• Explicar por qué la sincronización es necesaria en un programa paralelo especifico [Usage]
	• Identificar oportunidades para particionar un programa serial en módulos paralelos independientes [Familiarity]
	• Escribir un algoritmo paralelo correcto y escalable [Usage]
	• Paralelizar un algoritmo mediante la aplicación de descomposición basada en tareas [Usage]
	• Paralelizar un algoritmo mediante la aplicación de descomposición datos en paralelo [Usage]
	• Escribir un programa usando actores y/o procesos reactivos [Usage]

Readings : [Pac11], [Mat14], [Qui03], [Geo10]

Unit 4: Comunicación y coordinación (18)

Competences Expected:

Topics	**Learning Outcomes**
• Memoria Compartida. • La consistencia, y su papel en los lenguaje de programación garantias para los programas de carrera libre. • Pasos de Mensaje: – Mensajes Punto a Punto versus multicast (o basados en eventos) – Estilos para enviar y recibir mensajes Blocking vs non-blocking – Buffering de mensajes • Atomicidad: – Especificar y probar atomicidad y requerimientos de seguridad – Granularidad de accesos atómicos y actualizaciones, y uso de estructuras como secciones críticas o transacciones para describirlas – Exclusión mutua usando bloques, semáforos, monitores o estructuras relacionadas ∗ Potencial para fallas y bloqueos (*deadlock*) (causas, condiciones, prevención) – Composición ∗ Componiendo acciones atómicas granulares más grandes usando sincronización ∗ Transacciones, incluyendo enfoques optimistas y conservadores • Consensos: – (Ciclicos) barerras, contadores y estructuras relacionadas • Acciones condicionales: – Espera condicional (p.e., empleando variables de condición)	• Usar exclusión mútua para evitar una condición de carrera [Usage] • Dar un ejemplo de una ordenación de accesos entre actividades concurrentes (por ejemplo, un programa con condición de carrera) que no son secuencialmente consistentes [Familiarity] • Dar un ejemplo de un escenario en el que el bloqueo de mensajes enviados pueden dar *deadlock* [Usage] • Explicar cuándo y por qué mensajes de multidifusión (*multicast*) o basado en eventos puede ser preferible a otras alternativas [Familiarity] • Escribir un programa que termine correctamente cuando todo el conjunto de procesos concurrentes hayan sido completados [Usage] • Dar un ejemplo de un escenario en el que un intento optimista de actualización puede nunca completarse [Familiarity] • Usar semaforos o variables de condición para bloquear hebras hasta una necesaria precondición de mantenga [Usage]

Readings : [Pac11], [Mat14], [Qui03], [Geo10]

Unit 5: Análisis y programación de algoritmos paralelos (18)	
Competences Expected:	
Topics	**Learning Outcomes**
• Caminos críticos, el trabajo y la duración y la relación con la ley de Amdahl. • Aceleración y escalabilidad. • Naturalmente (vergonzosamente) algoritmos paralelos. • Patrones Algoritmicos paralelos (divide-y-conquista, map/reduce, amos-trabajadores, otros) – Algortimos específicos (p.e., MergeSort paralelo) • Algoritmos de grafos paralelo (por ejemplo, la ruta más corta en paralelo, árbol de expansión paralela) • Cálculos de matriz paralelas. • Productor-consumidor y algoritmos paralelos segmentados. • Ejemplos de algoritmos paralelos no-escalables.	• Definir: camino crítico, trabajo y *span* [Familiarity] • Calcular el trabajo y el *span* y determinar el camino crítico con respecto a un diagrama de ejecución paralela. [Usage] • Definir *speed-up* y explicar la noción de escalabilidad de un algoritmo en este sentido [Familiarity] • Identificar tareas independientes en un programa que debe ser paralelizado [Usage] • Representar características de una carga de trabajo que permita o evite que sea naturalmente paralelizable [Familiarity] • Implementar un algoritmo dividir y conquistar paralelo (y/o algoritmo de un grafo) y medir empiricamente su desempeño relativo a su analogo secuencial [Usage] • Descomponer un problema (por ejemplo, contar el número de ocurrencias de una palabra en un documento) via operaciones *map* y *reduce* [Usage] • Proporcionar un ejemplo de un problema que se corresponda con el paradigma productor-consumidor [Usage] • Dar ejemplos de problemas donde el uso de *pipelining* sería un medio eficaz para la paralelización [Usage] • Implementar un algoritmo de matriz paralela [Usage] • Identificar los problemas que surgen en los algoritmos del tipo productor-consumidor y los mecanismos que pueden utilizarse para superar dichos problemas [Usage]
Readings : [Mat14], [Qui03], [Geo10]	

Unit 6: Desempeño en paralelo (18)	
Competences Expected:	
Topics	**Learning Outcomes**
• Equilibrio de carga. • La medición del desempeño. • Programación y contención. • Evaluación de la comunicación de arriba. • Gestión de datos: – Costos de comunicación no uniforme debidos a proximidad – Efectos de Cache (p.e., false sharing) – Manteniendo localidad espacial • Consumo de energía y gestión.	• Detectar y corregir un desbalanceo de carga [Usage] • Calcular las implicaciones de la ley de Amdahl para un algoritmo paralelo particular [Usage] • Describir como la distribuición/disposición de datos puede afectar a los costos de comunicación de un algoritmo [Familiarity] • Detectar y corregir una instancia de uso compartido falso (*false sharing*) [Usage] • Explicar el impacto de la planificación en el desempeño paralelo [Familiarity] • Explicar el impacto en el desempeño de la localidad de datos [Familiarity] • Explicar el impacto y los puntos de equilibrio relacionados al uso de energía en el desempeño paralelo [Familiarity]
Readings : [Pac11], [Mat14], [KH13], [SK10], [Geo10]	

8. WORKPLAN

8.1 Methodology

Individual and team participation is encouraged to present their ideas, motivating them with additional points in the different stages of the course evaluation.

8.2 Theory Sessions

The theory sessions are held in master classes with activities including active learning and roleplay to allow students to internalize the concepts.

8.3 Practical Sessions

The practical sessions are held in class where a series of exercises and/or practical concepts are developed through problem solving, problem solving, specific exercises and/or in application contexts.

9. EVALUATION SYSTEM

********* EVALUATION MISSING ********

10. BASIC BIBLIOGRAPHY

[Geo10] Gerhard Wellein Georg Hager. *Introduction to High Performance Computing for Scientists and Engineers (Chapman and Hall/CRC Computational Science)*. Ed. by CRC Press. 1st. 2010. ISBN: 978-1439811924.

[KH13] David B. Kirk and Wen-mei W. Hwu. *Programming Massively Parallel Processors: A Hands-on Approach*. 2nd. Morgan Kaufmann, 2013. ISBN: 978-0-12-415992-1.

[Mat14] Norm Matloff. *Programming on Parallel Machines*. University of California, Davis, 2014. URL: `http://heather.cs.ucdavi`

[Pac11] Peter S. Pacheco. *An Introduction to Parallel Programming*. 1st. Morgan Kaufmann, 2011. ISBN: 978-0-12-374260-5.

[Qui03] Michael J. Quinn. *Parallel Programming in C with MPI and OpenMP*. 1st. McGraw-Hill Education Group, 2003. ISBN: 0071232656.

[SK10] Jason Sanders and Edward Kandrot. *CUDA by Example: An Introduction to General-Purpose GPU Programming*. 1st. Addison-Wesley Professional, 2010. ISBN: 0131387685, 9780131387683.

National University of Copsta Rica (UNA)
School of Informatics
Sillabus 2024-I

1. COURSE

ID105. Technical and professional English V (Mandatory)

2. GENERAL INFORMATION

2.1 Course	:	ID105. Technical and professional English V
2.2 Semester	:	6^{to} Semestre.
2.3 Credits	:	3
2.4 Horas	:	2 HT; 2 HP;
2.5 Duration of the period	:	16 weeks
2.6 Type of course	:	Mandatory
2.7 Learning modality	:	Face to face
2.8 Prerrequisites	:	None None

3. PROFESSORS

Meetings after coordination with the professor

4. INTRODUCTION TO THE COURSE

A fundamental part of the integral formation of a professional is the ability to communicate in a foreign language in addition to the native language itself. It not only broadens its cultural horizon but also allows a more humane and comprehensive view of life. In the case of foreign languages, undoubtedly English is the most practical because it is spoken around the world. There is no country where it is not spoken. In careers related to tourist services, English is perhaps the most important practical tool that the student must master from the outset as part of his / her integral education

5. GOALS

- Increase the ability and fluency of speaking and understanding the English language.
- That the students interact with greater emphasis in the creation of dialogues.

6. COMPETENCES

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

7. TOPICS

Unit 1: It's a wonderful world (0)	
Competences Expected:	
Topics	**Learning Outcomes**
• Auxiliary verbs • Verb tenses • Negative Questions and Prayers • Short answers • Word formation • Colloquial expressions • Error correction	• At the end of the first unit, each student, understanding the grammar of auxiliaries and different types of sentences, is able to express a greater number of expressions of time and also use prepositions to describe varied places and times. He is also able to analyze and express ideas about word formation.
Readings : [**Soars023S**], [**Soars023W**], [**Soars023T**], [**Cambridge06**], [**MacGrew99**]	

Unit 2: Happiness! (0)	
Competences Expected:	
Topics	**Learning Outcomes**
• Simple present • Present continuous • Passive Voice in Present • Verbs for sports and free time • Types of numbers and • Inventions / Modern World • Corrección de errores	• At the end of the second unit, students have identified how to express sports and leisure activities. It uses all kinds of numerical expressions. Express situations and states related to present forms. Explain and apply vocabulary of outdoor activities.
Readings : [**Soars023S**], [**Soars023W**], [**Soars023T**], [**Cambridge06**], [**MacGrew99**]	

Unit 3: Telling tales! (0)	
Competences Expected:	
Topics	**Learning Outcomes**
• Simple Past Time • Past Continuous • Passive Voice in Past • Vocabulary of Art and Literature • Expressions to give and ask opinions • Stories and stories	• At the end of the third unit, students having recognized the characteristics of the past passive forms, they use these make descriptions of various types. Describe art and literature and give indications of opinion. They will use conjunctions to unite type ideas.
Readings : [**Soars023S**], [**Soars023W**], [**Soars023T**], [**Cambridge06**], [**MacGrew99**]	

Unit 4: Doing the right thing! (0)	
Competences Expected:	
Topics	**Learning Outcomes**
• Mode Auxiliary Verbs I. • Affirmative, Negative and Modals Questions • Use of nationalities and other adjectives • Expressions of orders and offers • Guide to Good Manners • Form Fill • Phonetic symbols	• At the conclusion of the fourth unit, the students having identified the idea of expressing ideas of modes of actions that happen at the moment or that are related at any time, structure sentences in the Present. They express ideas of nationalities and make requests and offers varied.
Readings : [**Soars023S**], [**Soars023W**], [**Soars023T**], [**Cambridge06**], [**MacGrew99**]	

Unit 5: On the move! (0)	
Competences Expected:	
Topics	**Learning Outcomes**
• Future with Will • Future Time Prayers with going to • Use of might for future • Climate Expressions • Vocabulary of the climate • Expressions for hotels and transportation • E-mails	• At the end of the fifth unit, students, from the understanding of future time, will elaborate sentences using the necessary elements. They will also assimilate the need to express ideas of the climate. They will acquire vocabulary to describe use of public transportation. Expressions will be presented to order at hotels.
Readings : [**Soars023S**], [**Soars023W**], [**Soars023T**], [**Cambridge06**], [**MacGrew99**]	

Unit 6: I just love it! (0)	
Competences Expected:	
Topics	**Learning Outcomes**
• Questions with Shapes Like • Patrones Verbales II • Vocabulario de Comida, Lugares y ocupaciones • Palabras que van unidas en contexto • Expresiones para vistas y sonidos • Composición de Impresiones personales	• At the end of the sixth unit, students having learned the basics of structuring questions with like and with verbal patterns work applied to appropriate contexts. They emphasize the difference between meals, places and people. Describe sights and sounds. They use expressions to compare daily life in different places. They assume the idea of different lifestyles.
Readings : [**Soars023S**], [**Soars023W**], [**Soars023T**], [**Cambridge06**], [**MacGrew99**]	

8. WORKPLAN

8.1 Methodology

Individual and team participation is encouraged to present their ideas, motivating them with additional points in the different stages of the course evaluation.

8.2 Theory Sessions

The theory sessions are held in master classes with activities including active learning and roleplay to allow students to internalize the concepts.

8.3 Practical Sessions

The practical sessions are held in class where a series of exercises and/or practical concepts are developed through problem solving, problem solving, specific exercises and/or in application contexts.

9. EVALUATION SYSTEM

********* EVALUATION MISSING ********

10. BASIC BIBLIOGRAPHY

National University of Copsta Rica (UNA)
School of Informatics
Sillabus 2024-I

1. COURSE

DS371. Topics in Data Science (Elective)

2. GENERAL INFORMATION

2.1 Course	:	DS371. Topics in Data Science
2.2 Semester	:	6^{to} Semestre.
2.3 Credits	:	3
2.4 Horas	:	2 HT; 2 HP;
2.5 Duration of the period	:	16 weeks
2.6 Type of course	:	Elective
2.7 Learning modality	:	Face to face
2.8 Prerrequisites	:	

- MA203. Statistics and Probabilities. (4^{th} Sem)
- CS271. Databases I. (5^{th} Sem)
- MA203. Statistics and Probabilities. (4^{th} Sem)
- CS271. Databases I. (5^{th} Sem)

3. PROFESSORS

Meetings after coordination with the professor

4. INTRODUCTION TO THE COURSE

Write justification for this course here ...

5. GOALS

- Write your first goal here.
- Write your second goal here.
- Just in case you need more goals write them here

6. COMPETENCES

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

7. TOPICS

Unit 1: title for the unit goes here (5)	
Competences Expected:	
Topics	**Learning Outcomes**
• Topic1 • Topic2 • Topic3	• Learning outcome1 [Levelforthislearningoutcome]. • Apply computing in complex problems [Usage]. • Create a search engine [Assessment]. • Study data structures [Familiarity].
Readings : [**Bibitem1**], [**Bibitem2**]	

Unit 2: another unit goes here (1)	
Competences Expected:	
Topics	**Learning Outcomes**
• Topic1	• Learning outcome xyz [Levelforthislearningoutcome].
Readings : [**Bibitem3**], [**Bibitem1**]	

8. WORKPLAN

8.1 Methodology

Individual and team participation is encouraged to present their ideas, motivating them with additional points in the different stages of the course evaluation.

8.2 Theory Sessions

The theory sessions are held in master classes with activities including active learning and roleplay to allow students to internalize the concepts.

8.3 Practical Sessions

The practical sessions are held in class where a series of exercises and/or practical concepts are developed through problem solving, problem solving, specific exercises and/or in application contexts.

9. EVALUATION SYSTEM

********* EVALUATION MISSING ********

10. BASIC BIBLIOGRAPHY

National University of Copsta Rica (UNA)
School of Informatics
Sillabus 2024-I

1. COURSE

SE302. Elective II (Elective)

2. GENERAL INFORMATION

2.1 Course	:	SE302. Elective II
2.2 Semester	:	6^{to} Semestre.
2.3 Credits	:	3
2.4 Horas	:	4 HT;
2.5 Duration of the period	:	16 weeks
2.6 Type of course	:	Elective
2.7 Learning modality	:	Face to face
2.8 Prerrequisites	:	None None

3. PROFESSORS

Meetings after coordination with the professor

4. INTRODUCTION TO THE COURSE

Write justification for this course here ...

5. GOALS

- Write your first goal here.
- Write your second goal here.
- Just in case you need more goals write them here

6. COMPETENCES

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

7. TOPICS

Unit 1: title for the unit goes here (5)	
Competences Expected:	
Topics	**Learning Outcomes**
• Topic1 • Topic2 • Topic3	• Learning outcome1 [Levelforthislearningoutcome]. • Apply computing in complex problems [Usage]. • Create a search engine [Assessment]. • Study data structures [Familiarity].
Readings : [**Bibitem1**], [**Bibitem2**]	

Unit 2: another unit goes here (1)	
Competences Expected:	
Topics	**Learning Outcomes**
• Topic1	• Learning outcome xyz [Levelforthislearningoutcome].
Readings : [**Bibitem3**], [**Bibitem1**]	

8. WORKPLAN

8.1 Methodology

Individual and team participation is encouraged to present their ideas, motivating them with additional points in the different stages of the course evaluation.

8.2 Theory Sessions

The theory sessions are held in master classes with activities including active learning and roleplay to allow students to internalize the concepts.

8.3 Practical Sessions

The practical sessions are held in class where a series of exercises and/or practical concepts are developed through problem solving, problem solving, specific exercises and/or in application contexts.

9. EVALUATION SYSTEM

********* EVALUATION MISSING ********

10. BASIC BIBLIOGRAPHY

National University of Copsta Rica (UNA)
School of Informatics
Sillabus 2024-I

1. COURSE

CS341. Programming languages (Mandatory)

2. GENERAL INFORMATION

2.1 Course	:	CS341. Programming languages
2.2 Semester	:	7^{mo} Semestre.
2.3 Credits	:	4
2.4 Horas	:	3 HT; 3 HP;
2.5 Duration of the period	:	16 weeks
2.6 Type of course	:	Mandatory
2.7 Learning modality	:	Face to face
2.8 Prerrequisites	:	CS113. Programming II. (3^{rd} Sem) CS113. Programming II. (3^{rd} Sem)

3. PROFESSORS

Meetings after coordination with the professor

4. INTRODUCTION TO THE COURSE

Los lenguajes de programación son el medio a través del cual los programadores describen con precisión los conceptos, formulan algoritmos y representan sus soluciones. Un científico de la computación trabajará con diferentes lenguajes, por separado o en conjunto. Los científicos de la computación deben entender los modelos de programación de los diferentes lenguajes, tomar decisiones de diseño basados en el lenguaje de programación y sus conceptos. El profesional a menudo necesitará aprender nuevos lenguajes y construcciones de programación y debe entender los fundamentos de como las carácteristicas del lenguaje de programación estan definidas, compuestas e implementadas. El uso eficaz de los lenguajes de programación y la apreciación de sus limitaciones, también requiere un conocimiento básico de traducción de lenguajes de programación y su análisis de ambientes estáticos y dinámicos, así como los componentes de tiempo de ejecución tales como la gestión de memoria, entre otros detalles de relevancia.

5. GOALS

- Capacitar a los estudiantes para entender los lenguajes de programación desde diferentes tipos de vista, según el modelo subyacente, los componentes fundamentales presentes en todo lenguaje de programación y como objetos formales dotados de una estructura y un significado según diversos enfoques.

6. COMPETENCES

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. TOPICS

Unit 1: (18)

Competences Expected:

Topics	Learning Outcomes
• Historia de los Lenguajes de Programación • Programas que tienen otros programas como entrada tales como interpretes, compiladores, revisores de tipos y generadores de documentación. • Estructuras de datos que representan código para ejecución, traducción o transmisión. • Estructura de un programa: Léxico, Sintáctico y Semántico • BNF • Interpretación vs. compilación a código nativo vs. compilación de representación portable intermedia. [Familiarity]	• Reconocer el desarrollo histórico de los lenguajes de programación. [Familiarity] • Identificar los paradigmas que agrupan a la mayoría de lenguajes de programación existentes hoy en día. [Familiarity] • Explicar como programas que procesan otros programas tratan a los otros programas como su entrada de datos [Familiarity] • Describir un árbol de sintaxis abstracto para un lenguaje pequeño [Familiarity] • Escribir un programa para procesar alguna representación de código para algún propósito, tales como un interprete, una expresión optimizada, o un generador de documentación [Usage] • Distinguir una definición de un lenguaje de una implementación particular de un lenguaje (compilador vs interprete, tiempo de ejecución de la representación de los objetos de datos, etc) [Familiarity] • Reconocer como funciona un programa a nivel de computador. [Familiarity]

Readings : [Seb12], [Web10]

Unit 2: Pragmática de lenguajes (12)

Competences Expected:

Topics	Learning Outcomes
• Principios de diseño de lenguaje tales como la ortogonalidad. • Orden de evaluación, precedencia y asociatividad. • Evaluación tardía vs. evaluación temprana. • Definiendo controles y constructos de iteración. • Llamadas externas y sistema de librerías.	• Discute el rol de conceptos como ortogonalidad y el buen criterio de selección en el diseño de lenguajes [Usage] • Utiliza criterios objetivos y nítidos para evaluar las decisiones en el diseño de un lenguaje [Usage] • Da un ejemplo de un programa cuyo resultado puede diferir dado diversas reglas de orden de evaluación, precedencia, o asociatividad [Usage] • Muestra el uso de evaluación con retraso, como en el caso de abstracciones definidas y controladas por el usuario [Familiarity] • Discute la necesidad de permitir llamadas a librerias externas y del sistema y las consecuencias de su implementación en un lenguaje [Familiarity]

Readings : [Seb12], [Web10], [RH04]

Unit 3: Sistemas de tipos (18)	
Competences Expected:	
Topics	**Learning Outcomes**
• Constructores de tipo composicional, como tipos de producto (para agregados), tipos de suma (para uniones), tipos de función, tipos cuantificados y tipos recursivos. • Comprobación de tipos. • Seguridad de tipos como preservación más progreso. • Inferencia de tipos. • Sobrecarga estática.	• Definir un sistema de tipo de forma precisa y en su composición [Usage] • Para varias construcciones de tipo fundamental, identificar los valores que describen y las invariantes que hacen que se cumplan [Familiarity] • Precisar las invariantes preservadas por un sistema de tipos seguro (*sound type system*) [Familiarity] • Demostrar la seguridad de tipos para un lenguaje simple en términos de conservación y progreso teoremas [Usage] • Implementar un algoritmo de inferencia de tipos basado en la unificación para un lenguaje básico [Usage] • Explicar cómo la sobrecarga estática y algoritmos de resolución asociados influyen el comportamiento dinámico de los programas [Familiarity]
Readings : [Seb12], [Web10], [RH04]	

Unit 4: Programación orientada a objetos (12)	
Competences Expected:	
Topics	**Learning Outcomes**
• Diseño orientado a objetos: – Descomposicion en objetos que almacenan estados y poseen comportamiento – Diseño basado en jerarquia de clases para modelamiento • Definición de las categorías, campos, métodos y constructores. • Las subclases, herencia y método de alteración temporal. • Asignación dinámica: definición de método de llamada. • Subtipificación: – Polimorfismo artículo Subtipo; upcasts implícitos en lenguajes con tipos. – Noción de reemplazo de comportamiento: los subtipos de actuar como supertipos. – Relación entre subtipos y la herencia. • Lenguajes orientados a objetos para la encapsulación: – privacidad y la visibilidad de miembros de la clase – Interfaces revelan único método de firmas – clases base abstractas • Uso de coleccion de clases, iteradores, y otros componentes de la libreria estandar.	• Diseñar e implementar una clase [Usage] • Usar subclase para diseñar una jerarquía simple de clases que permita al código ser reusable por diferentes subclases [Usage] • Razonar correctamente sobre el flujo de control en un programa mediante el envío dinámico [Usage] • Comparar y contrastar (1) el enfoque procedurar/funcional- definiendo una función por cada operación con el cuerdo de la función proporcionando un caso por cada variación de dato - y (2) el enfoque orientado a objetos - definiendo una clase por cada variación de dato con la definición de la clase proporcionando un método por cada operación. Entender ambos enfoques como una definición de variaciones y operaciones de una matriz [Assessment] • Explicar la relación entre la herencia orientada a objetos (codigo compartido y *overriding*) y subtipificación (la idea de un subtipo es ser utilizable en un contexto en el que espera al supertipo) [Usage] • Usar mecanismos de encapsulación orientada a objetos, tal como interfaces y miembros privados [Usage] • Definir y usar iteradores y otras operaciones sobre agregaciones, incluyendo operaciones que tienen funciones como argumentos, en múltiples lenguajes de programación, selecionar la forma mas natural por cada lenguaje [Usage]
Readings : [Seb12], [Web10], [RH04]	

Unit 5: Programación funcional (18)

Competences Expected:

Topics	**Learning Outcomes**
• El efecto de la programación libre: – Llamadas a función que no tiene efecto secundarios, para facilitar el razonamiento composicional – Variables inmutables, prevencion de cambios no esperados en los datos del programa por otro código. – Datos que pueden ser subnombrados o copiados libremente sin introducir efectos no deseados del cambio • Procesamiento de estructuras de datos (p.e. arboles) a través de fuciones con casos para cada variación de los datos. – Constructores asociados al lenguaje tales como uniones discriminadas y reconocimiento de patrones sobre ellos. – Funciones definidas sobre datos compuestos en términos de funciones aplicadas a las piezas constituidas. • Funciones de primera clase (obtener, retornar y funciones de almacenamiento) • Cierres de función (funciones que usan variables en entornos léxicos cerrados) – Significado y definicion básicos - creacion de cierres en tiempo de ejecución mediante la captura del entorno. – Idiomas canónicos: llamadas de retorno, argumentos de iteradores, código reusable mediante argumentos de función – Uso del cierre para encapsular datos en su entorno – Evaluación y aplicación parcial • Definición de las operaciones de orden superior en los agregados, especialmente en mapa, reducir / doblar, y el filtro.	• Escribir algoritmos básicos que eviten asignación a un estado mutable o considerar igualdad de referencia [Usage] • Escribir funciones útiles que puedan tomar y retornar otras funciones [Usage] • Comparar y contrastar (1) el enfoque procedurar/funcional- definiendo una función por cada operación con el cuerdo de la función proporcionando un caso por cada variación de dato - y (2) el enfoque orientado a objetos - definiendo una clase por cada variación de dato con la definición de la clase proporcionando un método por cada operación. Entender ambos enfoques como una definición de variaciones y operaciones de una matriz [Assessment] • Razonar correctamente sobre variables y el ámbito léxico en un programa usando funciones de cierre (*function closures*) [Usage] • Usar mecanismos de encapsulamiento funcional, tal como *closures* e interfaces modulares [Usage] • Definir y usar iteradores y otras operaciones sobre agregaciones, incluyendo operaciones que tienen funciones como argumentos, en múltiples lenguajes de programación, seleccionar la forma mas natural por cada lenguaje [Usage]

Readings : [Seb12], [Web10], [RH04]

Unit 6: Programación reactiva y dirigida por eventos (12)	
Competences Expected:	
Topics	**Learning Outcomes**

• Eventos y controladores de eventos.
• Usos canónicos como interfaces gráficas de usuario, dispositivos móviles, robots, servidores.
• Uso de frameworks reactivos.
  – Definición de controladores/oyentes (handles/listeners) de eventos.
  – Bucle principal de enventos no controlado po el escritor controlador de eventos (event-handler-writer)
• Eventos y eventos del programa generados externamente generada.
• La separación de modelo, vista y controlador.
	• Escribir manejadores de eventos para su uso en sistemas reactivos tales como GUIs [Usage]
	• Explicar porque el estilo de programación manejada por eventos es natural en dominios donde el programa reacciona a eventos externos [Familiarity]
	• Describir un sistema interactivo en términos de un modelo, una vista y un controlador [Familiarity]

Readings : [Seb12]	

Unit 7: Programación lógica (12)	
Competences Expected:	
Topics	**Learning Outcomes**
• Representación causal de estructura de datos y algoritmos. • Unificación. • Bactracking y busqueda. • Cuts.	• Usa un lenguaje lógico para implementar un algoritmo convencional [Usage] • Usa un lenguaje lógico para implementar un algoritmo empleando búsqueda implícita usando claúsulas, relaciones, y cortes [Usage]
Readings : [Seb12], [Web10], [RH04]	

8. WORKPLAN

8.1 Methodology

Individual and team participation is encouraged to present their ideas, motivating them with additional points in the different stages of the course evaluation.

8.2 Theory Sessions

The theory sessions are held in master classes with activities including active learning and roleplay to allow students to internalize the concepts.

8.3 Practical Sessions

The practical sessions are held in class where a series of exercises and/or practical concepts are developed through problem solving, problem solving, specific exercises and/or in application contexts.

9. EVALUATION SYSTEM

********* EVALUATION MISSING ********

10. BASIC BIBLIOGRAPHY

[RH04] Peter Van Roy and Seif Haridi. *Concepts, Techniques, and Models of Computer Programming*. Cambridge, MA, USA: MIT Press, 2004. ISBN: 0262220695.

[Seb12] Robert W. Sebesta. *Concepts of Programming Languages*. 10th. USA: Addison-Wesley Publishing Company, 2012. ISBN: 0131395319.

[Web10] Adam Brooks Webber. *Modern Programming Languages: A Practical Introduction*. 2nd. Franklin, Beedle and Associates, Inc, 2010. ISBN: 978-1-59028-250-2.

National University of Copsta Rica (UNA)
School of Informatics
Sillabus 2024-I

1. COURSE

CS3P2. Cloud Computing (Mandatory)

2. GENERAL INFORMATION

2.1 Course	:	CS3P2. Cloud Computing
2.2 Semester	:	7^{mo} Semestre.
2.3 Credits	:	3
2.4 Horas	:	2 HT; 2 HP;
2.5 Duration of the period	:	16 weeks
2.6 Type of course	:	Mandatory
2.7 Learning modality	:	Face to face
2.8 Prerrequisites	:	CS3P1. Parallel and Distributed Computing . (6^{th} Sem) CS3P1. Parallel and Distributed Computing . (6^{th} Sem)

3. PROFESSORS

Meetings after coordination with the professor

4. INTRODUCTION TO THE COURSE

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.

5. 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. COMPETENCES

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. TOPICS

Unit 1: Sistemas distribuídos (15)	
Competences Expected:	
Topics	**Learning Outcomes**
• Fallos: – Fallos basados en red (incluyendo particiones) y fallos basados en nodos – Impacto en garantías a nivel de sistema (p.e., disponibilidad) • Envío de mensajes distribuido: – Conversión y transmisión de datos – Sockets – Secuenciamiento de mensajes – Almacenando *Buffering*, renviando y desechando mensajes • Compensaciones de diseño para Sistemas Distribuidos: – Latencia versus rendimiento – Consistencia, disponibilidad, tolerancia de particiones • Diseño de Servicio Distribuido: – Protocolos y servicios Stateful versus stateless – Diseños de Sesión (basados en la conexión) – Diseños reactivos (provocados por E/S) y diseños de múltiples hilos • Algoritmos de Distribución de Núcleos: – Elección, descubrimiento	• Distinguir las fallas de red de otros tipos de fallas [Familiarity] • Explicar por qué estructuras de sincronización como cerraduras simples (*locks*) no son útiles en la presencia de fallas distribuidas [Familiarity] • Escribir un programa que realiza cualquier proceso de *marshalling* requerido y la conversión en unidades de mensajes, tales como paquetes, para comunicar datos importantes entre dos *hosts* [Usage] • Medir el rendimiento observado y la latencia de la respuesta a través de los *hosts* en una red dada [Usage] • Explicar por qué un sistema distribuido no puede ser simultaneamente Consistente (*Consistent*), Disponible (*Available*) y Tolerante a fallas (*Partition tolerant*). [Familiarity] • Implementar un servidor sencillo - por ejemplo, un servicio de corrección ortográfica [Usage] • Explicar las ventajas y desventajas entre: *overhead*, escalabilidad y tolerancia a fallas entre escojer un diseño sin estado (*stateless*) y un diseño con estado (*stateful*) para un determinado servicio [Familiarity] • Describir los desafios en la escalabilidad, asociados con un servicio cresciente para soportar muchos clientes, así como los asociados con un servicio que tendrá transitoriamente muchos clientes [Familiarity] • Dar ejemplos de problemas donde algoritmos de consenso son requeridos, por ejemplo, la elección de líder [Usage]
Readings : [Cou+11]	

Unit 2: Cloud Computing (15)

Competences Expected:

Topics	**Learning Outcomes**

• Visión global de *Cloud Computing*.
• Historia.
• Visión global de las tecnologias que envuelve.
• Beneficios, riesgos y aspectos económicos.
• Servicios en la nube.
  – Infraestructura como servicio
  * Elasticidad de recursos
  * APIs de la Platforma
  – Software como servicio
  – Securidad
  – Administración del Costo
• Computación a Escala de Internet:
  – Particionamiento de Tareas
  – Acceso a datos
  – Clusters, grids y mallas
	• Explicar el concepto de Cloud Computing. [Familiarity]
	• Listar algunas tecnologias relacionadas con Cloud Computing. [Familiarity]
	• Explicar las estrategias para sincronizar una vista comun de datos compartidos a través de una colección de dispositivos [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 : [HDF11], [BVS13]

Unit 3: Centros de Procesamiento de Datos (10)

Competences Expected:

Topics	**Learning Outcomes**
• 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 : [HDF11], [BVS13]

Unit 4: Cloud Computing (20)	
Competences Expected:	
Topics	**Learning Outcomes**
• Virtualización. – Gestión de recursos compartidos – Migración de procesos • Seguridad, recursos y isolamiento de fallas. • Almacenamiento como servicio. • Elasticidad. • Xen y WMware. • Amazon EC2.	• Virtualización. – Gestión de recursos compartidos – Migración de procesos . [Familiarity] • Explicar las ventajas y desventajas de usar una infraestructura vistualizada. [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 : [HDF11], [BVS13]	

Unit 5: Cloud Computing (12)	
Competences Expected:	
Topics	**Learning Outcomes**
• Almacenamiento de datos en la nube: – Acceso compartido a data stores de consistencia débil – Sincronización de datos – Particionamiento de datos – Sistemas de Archivos Distribuidos – Replicación • Visión global sobre tecnologías de almacenamiento. • Conceptos fundamentales sobre almacenamiento en la cloud. • Amazon S3 y EBS. • Sistema de archivos distribuidos. • Sistema de bases de datos NoSQL.	• 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 : [HDF11], [BVS13]	

Unit 6: Modelos de Programación (12)	
Competences Expected:	
Topics	**Learning Outcomes**
• 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 : [HDF11], [BVS13], [Low+12], [Mal+10], [Bal+08]	

8. WORKPLAN

8.1 Methodology

Individual and team participation is encouraged to present their ideas, motivating them with additional points in the different stages of the course evaluation.

8.2 Theory Sessions

The theory sessions are held in master classes with activities including active learning and roleplay to allow students to internalize the concepts.

8.3 Practical Sessions

The practical sessions are held in class where a series of exercises and/or practical concepts are developed through problem solving, problem solving, specific exercises and/or in application contexts.

9. EVALUATION SYSTEM

********* EVALUATION MISSING ********

10. BASIC BIBLIOGRAPHY

[Bal+08] Shumeet Baluja et al. “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. Beijing, China: ACM, 2008, pp. 895–904. ISBN: 978-1-60558-085-2. DOI: 10.1145/1367497.1367618. URL: http://doi.acm.org/10.1145

[BVS13] Rajkumar Buyya, Christian Vecchiola, and S. Thamarai Selvi. *Mastering Cloud Computing: Foundations and Applications Programming*. 1st. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 2013. ISBN: 9780124095397, 9780124114548.

[Cou+11] George Coulouris et al. *Distributed Systems: Concepts and Design*. 5th. USA: Addison-Wesley Publishing Company, 2011. ISBN: 0132143011, 9780132143011.

[HDF11] Kai Hwang, Jack Dongarra, and Geoffrey C. Fox. *Distributed and Cloud Computing: From Parallel Processing to the Internet of Things*. 1st. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 2011. ISBN: 0123858801, 9780123858801.

[Low+12] Yucheng Low et al. “Distributed GraphLab: A Framework for Machine Learning and Data Mining in the Cloud”. In: *Proc. VLDB Endow.* 5.8 (Apr. 2012), pp. 716–727. ISSN: 2150-8097. DOI: 10.14778/2212351.2212354. URL: http://dx.doi.org/10.14778/2212351.2212354.

[Mal+10] Grzegorz Malewicz et al. “Pregel: A System for Large-scale Graph Processing”. In: SIGMOD ’10 (2010), pp. 135–146. DOI: 10.1145/1807167.1807184. URL: http://doi.acm.org/10.1145/1807167.1807184.

National University of Copsta Rica (UNA)
School of Informatics
Sillabus 2024-I

1. COURSE

ET201. Entrerpreneurship I (Mandatory)

2. GENERAL INFORMATION

2.1 Course	:	ET201. Entrerpreneurship I
2.2 Semester	:	7^{mo} Semestre.
2.3 Credits	:	3
2.4 Horas	:	2 HT; 2 HP;
2.5 Duration of the period	:	16 weeks
2.6 Type of course	:	Mandatory
2.7 Learning modality	:	Face to face
2.8 Prerrequisites	:	5tocicloaprobado 5tocicloaprobado

3. PROFESSORS

Meetings after coordination with the professor

4. INTRODUCTION TO THE COURSE

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.

5. 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. COMPETENCES

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. TOPICS

Unit 1: (5)	
Competences Expected:	
Topics	**Learning Outcomes**
• Emprendedor, emprendedurismo e innovación tecnológica • Modelos de negocio • Formación de equipos	• Identificar características de los emprendedores [Familiarity] • Introducir modelos de negocio [Familiarity]
Readings : [BDN10], [OP10], [Gar+14]	

Unit 2: (5)	
Competences Expected:	
Topics	**Learning Outcomes**
• 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 : [BDN10], [BD12], [Gar+14]	

Unit 3: (5)	
Competences Expected:	
Topics	**Learning Outcomes**
• 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 : [BDN10], [OP10], [Rie11], [Gar+14]	

Unit 4: (20)	
Competences Expected:	
Topics	**Learning Outcomes**
• 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 : [OP10], [BD12], [Gar+14]	

Unit 5: (20)	
Competences Expected:	
Topics	**Learning Outcomes**
• Aceleración versus incubación • Customer Development • Lean Startup	• Conocer y aplicar el modelo Customer Development [Usage] • Conocer y aplicar el modelo Lean Startup [Usage]
Readings : [BD12], [Rie11], [Gar+14]	

Unit 6: (5)	
Competences Expected:	
Topics	**Learning Outcomes**
• 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 : [BDN10], [Rie11], [Con96], [Rep97], [Gar+14]	

Unit 7: (5)	
Competences Expected:	
Topics	**Learning Outcomes**
• 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 : [BDN10], [BD12], [Gar+14]	

Unit 8: (5)	
Competences Expected:	
Topics	**Learning Outcomes**
• 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 : [BDN10], [BD12], [Gar+14]	

8. WORKPLAN

8.1 Methodology

Individual and team participation is encouraged to present their ideas, motivating them with additional points in the different stages of the course evaluation.

8.2 Theory Sessions

The theory sessions are held in master classes with activities including active learning and roleplay to allow students to internalize the concepts.

8.3 Practical Sessions

The practical sessions are held in class where a series of exercises and/or practical concepts are developed through problem solving, problem solving, specific exercises and/or in application contexts.

9. EVALUATION SYSTEM

********* EVALUATION MISSING ********

10. BASIC BIBLIOGRAPHY

[BD12] Steve Blank and Bob Dorf. *The Startup Owner's Manual: The Step-By-Step Guide for Building a Great Company*. K and S Ranch, 2012.

[BDN10] Thomas Byers, Richard Dorf, and Andrew Nelson. *Technology Ventures: From Idea to Enterprise*. McGraw-Hill Science, 2010.

[Con96] Congreso de la Republica del Perú. *Decreto Legislativo No 823. Ley de la Propiedad Industrial*. El Peruano, 1996.

[Gar+14] René Garzo-i-Pincay et al. *Planes de Negocios para Emprendedores*. Iniciativa Latinoamericana de Libros de Texto Abiertos (LATIn), 2014.

[OP10] Alexander Osterwalder and Yves Pigneur. *Business Model Generation: A Handbook for Visionaries, Game Changers, and Challengers*. Wiley, 2010.

[Rep97] Congreso de la Republica del Peru. *Ley No 26887. Ley General de Sociedades*. El Peruano, 1997.

[Rie11] Eric Ries. *The Lean Startup: How Today's Entrepreneurs Use Continuous Innovation to Create Radically Successful Businesses*. Crown Business, 2011.

National University of Copsta Rica (UNA)
School of Informatics
Sillabus 2024-I

1. COURSE

SE403. Elective III (Elective)

2. GENERAL INFORMATION

2.1 Course	:	SE403. Elective III
2.2 Semester	:	7^{mo} Semestre.
2.3 Credits	:	3
2.4 Horas	:	4 HT;
2.5 Duration of the period	:	16 weeks
2.6 Type of course	:	Elective
2.7 Learning modality	:	Face to face
2.8 Prerrequisites	:	None None

3. PROFESSORS

Meetings after coordination with the professor

4. INTRODUCTION TO THE COURSE

Write justification for this course here ...

5. GOALS

- Write your first goal here.
- Write your second goal here.
- Just in case you need more goals write them here

6. COMPETENCES

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

7. TOPICS

Unit 1: title for the unit goes here (5)	
Competences Expected:	
Topics	**Learning Outcomes**
• Topic1 • Topic2 • Topic3	• Learning outcome1 [Levelforthislearningoutcome]. • Apply computing in complex problems [Usage]. • Create a search engine [Assessment]. • Study data structures [Familiarity].
Readings : [**Bibitem1**], [**Bibitem2**]	

Unit 2: another unit goes here (1)	
Competences Expected:	
Topics	**Learning Outcomes**
• Topic1	• Learning outcome xyz [Levelforthislearningoutcome].
Readings : [**Bibitem3**], [**Bibitem1**]	

8. WORKPLAN

8.1 Methodology

Individual and team participation is encouraged to present their ideas, motivating them with additional points in the different stages of the course evaluation.

8.2 Theory Sessions

The theory sessions are held in master classes with activities including active learning and roleplay to allow students to internalize the concepts.

8.3 Practical Sessions

The practical sessions are held in class where a series of exercises and/or practical concepts are developed through problem solving, problem solving, specific exercises and/or in application contexts.

9. EVALUATION SYSTEM

********* EVALUATION MISSING ********

10. BASIC BIBLIOGRAPHY

National University of Copsta Rica (UNA)
School of Informatics
Sillabus 2024-I

1. COURSE

CS261. Intelligent Systems (Mandatory)

2. GENERAL INFORMATION

2.1 Course	:	CS261. Intelligent Systems
2.2 Semester	:	8^{vo} Semestre.
2.3 Credits	:	3
2.4 Horas	:	2 HT; 2 HP;
2.5 Duration of the period	:	16 weeks
2.6 Type of course	:	Mandatory
2.7 Learning modality	:	Face to face
2.8 Prerrequisites	:	

- CS212. Analysis and Design of Algorithms. (5^{th} Sem)
- MA203. Statistics and Probabilities. (4^{th} Sem)
- CS212. Analysis and Design of Algorithms. (5^{th} Sem)
- MA203. Statistics and Probabilities. (4^{th} Sem)

3. PROFESSORS

Meetings after coordination with the professor

4. INTRODUCTION TO THE COURSE

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.

5. 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. COMPETENCES

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. TOPICS

Unit 1: Cuestiones fundamentales (2)	
Competences Expected:	
Topics	**Learning Outcomes**
• Descripción general de los problemas de Inteligencia Artificial, ejemplos recientes de aplicaciones de Inteligencia artificial. • ¿Qué es comportamiento inteligente? – El Test de Turing – Razonamiento Racional versus No Racional • Características del Problema: – Observable completamente versus observable parcialmente – Individual versus multi-agente – Deterministico versus estocástico – Estático versus dinámico – Discreto versus continuo • Naturaleza de agentes: – Autónomo versus semi-autónomo – Reflexivo, basado en objetivos, y basado en utilidad – La importancia en percepción e interacciones con el entorno • Cuestiones filosóficas y éticas.	• Describir el test de Turing y el experimento pensado cuarto chino" (*Chinese Room*) [Usage] • Determinando las caracteristicas de un problema dado que sistemas inteligentes deberian resolver [Usage]
Readings : [De 06], [Pon+14]	

Unit 2: Agentes (2)

Competences Expected:

Topics	**Learning Outcomes**

• Definición de Agentes
• Arquitectura de agentes (Ej. reactivo, en capa, cognitivo)
• Teoría de agentes
• Racionalidad, teoría de juegos:
  – Agentes de decisión teórica
  – Procesos de decisión de Markov (MDP)
• Agentes de Software, asistentes personales, y acceso a información:
  – Agentes colaborativos
  – Agentes de recolección de información
  – Agentes creíbles (carácter sintético, modelamiento de emociones en agentes)
• Agentes de aprendizaje
• Sistemas Multi-agente
  – Agentes Colaborativos
  – Equipos de Agentes
  – Agentes Competitivos (ej., subastas, votaciones)
  – Sistemas de enjambre y modelos biológicamente inspirados
	• Lista las características que definen un agente inteligente [Usage]
	• Describe y contrasta las arquitecturas de agente estándares [Usage]
	• Describe las aplicaciones de teoría de agentes para dominios como agentes de software, asistentes personales, y agentes creibles [Usage]
	• Describe los paradigmas primarios usados por agentes de aprendizaje [Usage]
	• Demuestra mediante ejemplos adecuados como los sistemas multi-agente soportan interacción entre agentes [Usage]

Readings : [Nil01], [RN03], [Pon+14]

Unit 3: Estrategias de búsquedas básicas (2)	
Competences Expected:	
Topics	**Learning Outcomes**
• Espacios de Problemas (estados, metas y operadores), solución de problemas mediante búsqueda. • Factored representation (factoring state hacia variables) • Uninformed search (breadth-first, depth-first, depth-first with iterative deepening) • Heurísticas y búsqueda informada (hill-climbing, generic best-first, A*) • El espacio y el tiempo de la eficiencia de búsqueda. • Dos jugadores juegos (introducción a la búsqueda minimax). • Satisfacción de restricciones (backtracking y métodos de búsqueda local).	• Formula el espacio eficiente de un problema para un caso expresado en lenguaje natural (ejm. Inglés) en términos de estados de inicio y final, así como sus operadores [Usage] • Describe el rol de las heurísticas y describe los intercambios entre completitud, óptimo, complejidad de tiempo, y complejidad de espacio [Usage] • Describe el problema de la explosión combinatoria del espacio de búsqueda y sus consecuencias [Usage] • Compara y contrasta tópicos de búsqueda básica con temas jugabilidad de juegos [Usage]
Readings : [Nil01], [Pon+14]	

Unit 4: Búsqueda Avanzada (18)	
Competences Expected:	
Topics	**Learning Outcomes**
• Búsqueda estocástica: – Simulated annealing – Algoritmos genéticos – Búsqueda de árbol Monte-Carlo • Construcción de árboles de búsqueda, espacio de búsqueda dinámico, explosión combinatoria del espacio de búsqueda. • Implementación de búsqueda A *, búsqueda en haz. • Búsqueda Minimax, poda alfa-beta. • Búsqueda Expectimax (MDP-Solving) y los nodos de azar.	• Diseñar e implementar una solución a un problema con algoritmo genético [Usage] • Diseñar e implementar un esquema de recocido simulado (*simulated annealing*) para evitar mínimos locales en un problema [Usage] • Diseñar e implementar una búsqueda A* y búsqueda en haz (*beam search*) para solucionar un problema [Usage] • Aplicar búsqueda minimax con poda alfa-beta para simplifiar el espacio de búsqueda en un juego con dos jugadores [Usage] • Comparar y contrastar los algoritmos genéticos con técnicas clásicas de búsqueda [Usage] • Comparar y contrastar la aplicabilidad de varias heurísticas de búsqueda, para un determinado problema [Usage]
Readings : [Gol89], [Nil01], [RN03], [Pon+14]	

Unit 5: Razonamiento Bajo Incertidumbre (18)	
Competences Expected:	
Topics	**Learning Outcomes**
• Revisión de Probabilidad Básica • Variables aleatorias y distribuciones de probabilidad: – Axiomas de probabilidad – Inferencia probabilística – Regla de Bayes • Independecia Condicional • Representaciones del conocimiento: – Redes bayesianas ∗ Inferencia exacta y su complejidad ∗ Métodos de Muestreo aleatorio (Monte Carlo) (p.e. Muestreo de Gibbs) – Redes Markov – Modelos de probabilidad relacional – Modelos ocultos de Markov	• Aplicar la regla de Bayes para determinar el cumplimiento de una hipótesis [Usage] • Explicar cómo al tener independencia condicional permite una gran eficiencia en sistemas probabilísticos [Usage] • Identificar ejemplos de representación de conocimiento para razonamiento bajo incertidumbre [Usage] • Indicar la complejidad de la inferencia exacta. Identificar métodos para inferencia aproximada [Usage]
Readings : [KF09], [RN03]	

Unit 6: Aprendizaje Automático Básico (4)	
Competences Expected:	
Topics	**Learning Outcomes**
• Definición y ejemplos de la extensa variedad de tareas de aprendizaje de máquina, incluida la clasificación. • Aprendizaje inductivo • Aprendizaje simple basado en estadísticas, como el clasificador ingenuo de Bayes, árboles de decisión. • El problema exceso de ajuste. • Medicion clasificada con exactitud.	• Listar las diferencias entre los tres principales tipos de aprendizaje: supervisado, no supervisado y por refuerzo [Usage] • Identificar ejemplos de tareas de clasificación, considerando las características de entrada disponibles y las salidas a ser predecidas [Usage] • Explicar la diferencia entre aprendizaje inductivo y deductivo [Usage] • Describir el sobre ajuste (*overfitting*) en el contexto de un problema [Usage] • Aplicar un algoritmo de aprendizaje estadístico simple como el Clasificador Naive Bayesiano e un problema de clasificacion y medirla precisión del clasificador [Usage]
Readings : [Mit98], [RN03], [Pon+14]	

Unit 7: Aprendizaje de máquina avanzado (20)

Competences Expected:

Topics	**Learning Outcomes**

• Definición y ejemplos de una amplia variedad de tareas de aprendizaje de máquina
• Aprendizaje general basado en estadística, estimación de parámetros (máxima probabilidad)
• Programación lógica inductiva (*Inductive logic programming ILP*)
• Aprendizaje supervisado
  – Aprendizaje basado en árboles de decisión
  – Aprendizaje basado en redes neuronales
  – Aprendizaje basado en máquinas de soporte vectorial (*Support vector machines SVMs*)
• Aprendizaje y *clustering* no supervisado
  – EM
  – K-means
  – Mapas auto-organizados
• Aprendizaje semi-supervisado.
• Aprendizaje de modelos gráficos
• Evaluación del desempeño (tal como cross-validation, area bajo la curva ROC)
• Aplicación de algoritmos Machine Learning para Minería de datos.
	• Explica las diferencias entre los tres estilos de aprendizaje: supervisado, por refuerzo y no supervisado [Usage]
	• Implementa algoritmos simples para el aprendizaje supervisado, aprendizaje por refuerzo, y aprendizaje no supervisado [Usage]
	• Determina cuál de los tres estilos de aprendizaje es el apropiado para el dominio de un problema en particular [Usage]
	• Compara y contrasta cada una de las siguientes técnicas, dando ejemplo de cuando una estrategia es la mejor: árboles de decisión, redes neuronales, y redes bayesianas [Usage]
	• Evalúa el rendimiento de un sistema de aprendizaje simple en un conjunto de datos reales [Usage]
	• Describe el estado del arte en la teoría del aprendizaje, incluyendo sus logros y limitantes [Usage]
	• Explica el problema del sobreajuste, conjuntamente con técnicas para determinar y manejar el problema [Usage]

Readings : [RN03], [KF09], [Mur12]

Unit 8: Procesamiento del Lenguaje Natural (12)	
Competences Expected:	
Topics	**Learning Outcomes**
• Gramaticas determinísticas y estocásticas • Algoritmos de parseo – Gramáticas libres de contexto (CFGs) y cuadros de parseo (e.g. Cocke-Younger-Kasami CYK) – CFGs probabilísticos y ponderados CYK • Representación del significado / Semántica – Representación de conocimiento basado en lógica – Roles semánticos – Representaciones temporales – Creencias, deseos e intenciones • Metodos basados en el corpus • N-gramas y Modelos ocultos de Markov (HMMs) • Suavizado y back-off • Ejemplos de uso: POS etiquetado y morfologia • Recuperación de la información: – Modelo de espacio vectorial ∗ TF & IDF – Precision y cobertura • Extracción de información • Traducción de lenguaje • Clasificación y categorización de texto: – Modelo de bolsa de palabras	• Define y contrasta gramáticas de tipo estocásticas y determinísticas, dando ejemplos y demostrando como adecuar cada una de ellas [Usage] • Simula, aplica, o implementa algoritmos clásicos y estocásticos para el parseo de un lenguaje natural [Usage] • Identifica los retos de la representación del significado [Usage] • Lista las ventajas de usar corpus estándares. Identifica ejemplos de corpus actuales para una variedad de tareas de PLN [Usage] • Identifica técnicas para la recuperación de la información, traducción de lenguajes, y clasificación de textos [Usage]
Readings : [Nil01], [RN03], [Pon+14]	

Unit 9: Visión y percepción por computador (12)	
Competences Expected:	
Topics	**Learning Outcomes**
• Visión Computacional – Adquisición de imágenes, representación, procesamiento y propiedades – Representación de formas, reconocimiento y segmentación de objetos – Análisis de movimiento • Modularidad en reconocimiento. • Enfoques de reconocimiento de patrones – Algoritmos de clasificación y medidas de calidad de la clasificación. – Técnicas estadísticas.	• Resumir la importancia del reconocimiento de imagenes y objetos en Inteligencia Artificial (AI) e indicar varias aplicaciones significativas de esta tecnologia [Usage] • Listar al menos tres aproximaciones de segmentación de imágenes, tales como algoritmos de limites (thresholding), basado en el borde y basado en regiones, junto con sus características definitorias, fortalezas y debilidades [Usage] • Implementar reconocimiento de objetos en 2d basados en la representación del contorno y/o regiones basadas en formas [Usage] • Proporcionar al menos dos ejemplos de transformación de una fuente de datos de un dominio sensorial a otro, ejemplo, datos táctiles interpretados como imágenes en 2d de una sola banda [Usage] • Implementar un algoritmo para la extracción de caracteristicas en información real, ejemplo, un detector de bordes o esquinas para imágenes o vectores de coeficientes de Fourier describiendo una pequeña porción de señal de audio [Usage] • Implementar un algoritmo de clasificación que segmenta percepciones de entrada en categorias de salida y evalua cuantitativamente la clasificación resultante [Usage] • Evaluar el desempeño de la función de extracción subyacente, en relación con al menos una aproximación alternativa posible (ya sea implementado o no) en su contribución a la tarea de clasificación (8) anterior [Usage]
Readings : [Nil01], [RN03], [Pon+14]	

8. WORKPLAN

8.1 Methodology

Individual and team participation is encouraged to present their ideas, motivating them with additional points in the different stages of the course evaluation.

8.2 Theory Sessions

The theory sessions are held in master classes with activities including active learning and roleplay to allow students to internalize the concepts.

8.3 Practical Sessions

The practical sessions are held in class where a series of exercises and/or practical concepts are developed through problem solving, problem solving, specific exercises and/or in application contexts.

9. EVALUATION SYSTEM

********* EVALUATION MISSING ********

10. BASIC BIBLIOGRAPHY

[De 06] L.N. De Castro. *Fundamentals of natural computing: basic concepts, algorithms, and applications*. CRC Press, 2006.

[Gol89] David Goldberg. *Genetic Algorithms in Search, Optimization and Machine Learning*. Addison Wesley, 1989.

[KF09] Daphne Koller and Nir Friedman. *Probabilistic Graphical Models: Principles and Techniques - Adaptive Computation and Machine Learning*. The MIT Press, 2009. ISBN: 0262013193.

[Mit98] M. Mitchell. *An introduction to genetic algorithms*. The MIT press, 1998.

[Mur12] Kevin P. Murphy. *Machine Learning: A Probabilistic Perspective*. The MIT Press, 2012. ISBN: 0262018020.

[Nil01] Nils Nilsson. *Inteligencia Artificial: Una nueva visión*. McGraw-Hill, 2001.

[Pon+14] Julio Ponce-Gallegos et al. *Inteligencia Artificial*. Iniciativa Latinoamericana de Libros de Texto Abiertos (LATIn), 2014.

[RN03] Stuart Russell and Peter Norvig. *Inteligencia Artifical: Un enfoque moderno*. Prentice Hall, 2003.

National University of Copsta Rica (UNA)
School of Informatics
Sillabus 2024-I

1. COURSE

CS2H1. User Experience (UX) (Mandatory)

2. GENERAL INFORMATION

2.1 Course	:	CS2H1. User Experience (UX)
2.2 Semester	:	8^{vo} Semestre.
2.3 Credits	:	3
2.4 Horas	:	2 HT; 2 HP;
2.5 Duration of the period	:	16 weeks
2.6 Type of course	:	Mandatory
2.7 Learning modality	:	Face to face
2.8 Prerrequisites	:	SE1R1. Requirements and interface design. (2^{nd} Sem) SE1R1. Requirements and interface design. (2^{nd} Sem)

3. PROFESSORS

Meetings after coordination with the professor

4. INTRODUCTION TO THE COURSE

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.

5. 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. COMPETENCES

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. TOPICS

Unit 1: Fundamentos (8)	
Competences Expected:	
Topics	**Learning Outcomes**
• Contextos para IHC (cualquiera relacionado con una interfaz de usuario, p.e., página web, aplicaciones de negocios, aplicaciones móviles y juegos) • Heurística de usabilidad y los principios de pruebas de usabilidad. • Procesos para desarrollo centrado en usuarios, p.e., enfoque inicial en usuarios, pruebas empíricas, diseño iterativo. • Principios del buen diseño y buenos diseñadores; ventajas y desventajas de ingeniería. • Diferentes medidas para evaluación, p.e., utilidad, eficiencia, facilidad de aprendizaje, satisfacción de usuario.	• Discutir por qué el desarrollo de software centrado en el hombre es importante [Familiarity] • Define un proceso de diseño centralizado en el usario que de forma explícita considere el hecho que un usuario no es como un desarrollador o como sus conocimientos [Familiarity] • Resumir los preceptos básicos de la interacción psicológica y social [Familiarity] • Desarrollar y usar un vocabulario conceptual para analizar la interación humana con el software: disponibilidad, modelo conceptual, retroalimentación, y demás [Familiarity]
Readings : [Dix+04], [Sto+05], [RS11]	

Unit 2: Factores Humanos (8)	
Competences Expected:	
Topics	**Learning Outcomes**
• Modelos cognoscitivos que informan diseño de interacciones, p.e., atención, percepción y reconocimiento, movimiento, memoria, golfos de expectativa y ejecución. • Capacidades físicas que informan diseño de interacción, p.e. percepción del color, ergonomía. • Accesibilidad, p.e., interfaces para poblaciones con diferentes habilidades (p.e., invidentes, discapacitados) • Interfaces para grupos de población de diferentes edades (p.e., niños, mayores de 80)	• Crear y dirigir una simple pruebga de usabilidad para una aplicación existente de software [Familiarity]
Readings : [Dix+04], [Sto+05], [RS11], [Mat11], [Nor04]	

Unit 3: Diseño y Testing centrados en el usuario (16)

Competences Expected:

Topics	**Learning Outcomes**
• Enfoque y características del proceso de diseño. • Requerimientos de funcionalidad y usabilidad. • Técnicas de recolección de requerimientos, ej. entrevistas, encuentas, etnografía e investigación contextual. • Técnicas y herramientas para el análisis y presentación de requerimientos ej. reportes, personas. • Análisis de tareas, incluidos los aspectos cualitativos de la generación de modelos de análisis de tareas. • Consideración de IHC como una disciplina de diseño: – Sketching – Diseño participativo – Sketching – Diseño participativo • Técnicas de creación de prototipos y herramientas, ej.bosquejos, *storyboards*, prototipos de baja fidelidad, esquemas de página. • Prototipos de baja fidelidad (papel) • Técnicas de evaluación cuantitativa ej. evaluación Keystroke-level. • Evaluación sin usuarios, usando ambas técnicas cualitativas y cuantitativas. Ej. Revisión estructurada, GOMS, análisis basado en expertos, heurísticas, lineamientos y estándar. • Evaluación con usuarios. Ej. Observación, Método de pensamiento en voz alta, entrevistas, encuentas, experimentación. • Desafíos para la evaluación efectiva, por ejemplo, toma de muestras, la generalización. • Reportar los resultados de las evaluaciones. • Internacionalización, diseño para usuarios de otras culturas, intercultural.	• Llevar a cabo una evaluación cuantitativa y discutir / informar sobre los resultados [Familiarity] • Para un grupo de usuarios determinado, realizar y documentar un análisis de sus necesidades [Familiarity] • Discutir al menos un standard nacional o internacional de diseño de interfaz de usuario [Familiarity] • Explicar cómo el diseño centrado en el usuario complementa a otros modelos de proceso software [Familiarity] • Utilizar *lo-fi* (baja fidelidad) técnicas de prototipado para recopilar y reportar, las respuestas del usuario [Usage] • Elegir los métodos adecuados para apoyar el desarrollo de una específica interfaz de usuario [Assessment] • Utilizar una variedad de técnicas para evaluar una interfaz de usuario dada [Assessment] • Comparar las limitaciones y beneficios de los diferentes métodos de evaluación [Assessment]

Readings : [Dix+04], [Sto+05], [RS11], [Mat11], [Bux07]

Unit 4: Diseño de Interacción (8)	
Competences Expected:	
Topics	**Learning Outcomes**
• Principios de interfaces gráficas de usuario (GUIs) • Elementos de diseño visual (disposición, color, fuentes, etiquetado) • Manejo de fallas humanas/sistema. • Estándares de interfaz de usuario. • Presentación de información: navegación, representación, manipulación. • Técnicas de animación de interfaz (ej. grafo de escena) • Clases Widget y bibliotecas. • Internacionalización, diseño para usuarios de otras culturas, intercultural. • Elección de estilos de interacción y técnicas de interacción.	• Crear una aplicación simple, junto con la ayuda y la documentación, que soporta una interfaz gráfica de usuario [Usage]
Readings : [Dix+04], [Sto+05], [RS11], [Joh10], [Mat11], [LS06]	

Unit 5: Nuevas Tecnologías Interactivas (8)	
Competences Expected:	
Topics	**Learning Outcomes**
• Elección de estilos de interacción y técnicas de interacción. • Enfoques para el diseño, implementación y evaluación de la interacción sin mouse – Interfaces táctiles y multitáctiles. – Interfaces compartidas, incorporadas y grandes – Nuevas modalidades de entrada (tales como datos de sensores y localización) – Nuevas ventanas, por ejemplo, iPhone, Android – Reconocimiento de voz y procesamiento del lenguaje natural – Interfaces utilizables y tangibles – Interacción persuasiva y emoción – Tecnologías de interacción ubicuas y contextuales (Ubicomp) – Inferencia bayesiana (por ejemplo, texto predictivo, orientación guiada) – Visualización e interacción de ambiente / periféricos • Salida: – Sonido – Visualización estereoscópica – Forzar la simulación de retroalimentación, dispositivos hápticos • Arquitectura de Sistemas: – Motores de Juego – Relidad Aumentada móvil – Simuladores de vuelo – CAVEs – Imágenes médicas	• Describe cuando son adecuadas las interfaces sin uso de ratón [Familiarity] • Comprende las posibilidades de interacción que van más allá de las interfaces de ratón y puntero [Familiarity] • Discute las ventajas (y desventajas) de las interfaces no basadas en ratón [Usage] • Describir el modelo óptico realizado por un sistema de gráficos por computadora para sintetizar una visión estereoscópica [Familiarity] • Describir los principios de las diferentes tecnologias de seguimiento de espectador [Familiarity] • Determinar los requerimientos básicos en interfaz, software, hardware, y cofiguraciones de software de un sistema VR para una aplicación específica [Assessment]
Readings : [Dix+04], [Sto+05], [RS11], [WW11], [Mat11]	

Unit 6: Colaboración y Comunicación (8)	
Competences Expected:	
Topics	**Learning Outcomes**
• La comunicación asíncrona en grupo, por ejemplo, el correo electrónico, foros, redes sociales. • Medios de comunicación social, informática social, y el análisis de redes sociales. • Colaboración en línea, espacios "inteligentes" y aspectos de coordinación social de tecnologías de flujo de trabajo. • Comunidades en línea. • Personajes de Software y agentes inteligentes, mundos virtuales y avatares. • Psicología Social	• Describir la diferencia entre la comunicación sincrónica y asincrónica [Familiarity] • Comparar los problemas de IHC en la interacción individual con la interacción del grupo [Familiarity] • Discuta varias problemas de interés social planteados por el software colaborativo [Usage] • Discutir los problemas de IHC en software que personifica la intención humana [Assessment]
Readings : [Dix+04], [Sto+05], [RS11]	

8. WORKPLAN

8.1 Methodology

Individual and team participation is encouraged to present their ideas, motivating them with additional points in the different stages of the course evaluation.

8.2 Theory Sessions

The theory sessions are held in master classes with activities including active learning and roleplay to allow students to internalize the concepts.

8.3 Practical Sessions

The practical sessions are held in class where a series of exercises and/or practical concepts are developed through problem solving, problem solving, specific exercises and/or in application contexts.

9. EVALUATION SYSTEM

********* EVALUATION MISSING ********

10. BASIC BIBLIOGRAPHY

[Bux07] Bill Buxton. *Sketching User Experiences: Getting the Design Right and the Right Design*. Morgan Kaufmann Publishers Inc., 2007.

[Dix+04] Alan Dix et al. *Human-computer Interaction*. 3 ed. Prentice-Hall, Inc, 2004.

[Joh10] Jeff Johnson. *Designing with the Mind in Mind: Simple Guide to Understanding User Interface Design Rules*. 3 ed. Morgan Kaufmann Publishers Inc., 2010.

[LS06] M. Leavitt and B. Shneiderman. *Research-Based Web Design & Usability Guidelines*. Health and Human Services Dept, 2006.

[Mat11] Lukas Mathis. *Designed for Use: Create Usable Interfaces for Applications and the Web*. Pragmatic Bookshelf, 2011.

[Nor04] Donald A. Norman. *Emotional Design: Why We Love (or Hate) Everyday Things*. Basic Book, 2004.

[RS11] Y. Rogers and J Sharp H. & Preece. *Interaction Design: Beyond Human-Computer Interaction*. 3 ed. John Wiley and Sons Ltd, 2011.

[Sto+05] D. Stone et al. *User Interface Design and Evaluation*. Morgan Kaufmann Series in Interactive Technologies, 2005.

[WW11] D. Wigdor and D. Wixon. *Brave NUI World: Designing Natural User Interfaces for Touch and Gesture*. Morgan Kaufmann Publishers Inc, 2011.

National University of Copsta Rica (UNA)
School of Informatics
Sillabus 2024-I

1. COURSE

SE3E1. General and Professional Ethics (Mandatory)

2. GENERAL INFORMATION

2.1 Course	:	SE3E1. General and Professional Ethics
2.2 Semester	:	8^{vo} Semestre.
2.3 Credits	:	3
2.4 Horas	:	3 HT;
2.5 Duration of the period	:	16 weeks
2.6 Type of course	:	Mandatory
2.7 Learning modality	:	Face to face
2.8 Prerrequisites	:	5tocicloaprobado 5tocicloaprobado

3. PROFESSORS

Meetings after coordination with the professor

4. INTRODUCTION TO THE COURSE

Write justification for this course here ...

5. GOALS

- Write your first goal here.
- Write your second goal here.
- Just in case you need more goals write them here

6. COMPETENCES

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

7. TOPICS

Unit 1: title for the unit goes here (5)	
Competences Expected:	
Topics	**Learning Outcomes**
• Topic1 • Topic2 • Topic3	• Learning outcome1 [Levelforthislearningoutcome]. • Apply computing in complex problems [Usage]. • Create a search engine [Assessment]. • Study data structures [Familiarity].
Readings : [**Bibitem1**], [**Bibitem2**]	

Unit 2: another unit goes here (1)	
Competences Expected:	
Topics	**Learning Outcomes**
• Topic1	• Learning outcome xyz [Levelforthislearningoutcome].
Readings : [**Bibitem3**], [**Bibitem1**]	

8. WORKPLAN

8.1 Methodology

Individual and team participation is encouraged to present their ideas, motivating them with additional points in the different stages of the course evaluation.

8.2 Theory Sessions

The theory sessions are held in master classes with activities including active learning and roleplay to allow students to internalize the concepts.

8.3 Practical Sessions

The practical sessions are held in class where a series of exercises and/or practical concepts are developed through problem solving, problem solving, specific exercises and/or in application contexts.

9. EVALUATION SYSTEM

********* EVALUATION MISSING ********

10. BASIC BIBLIOGRAPHY

National University of Copsta Rica (UNA)
School of Informatics
Sillabus 2024-I

1. COURSE

SE3E2. Supervised professional practice (Mandatory)

2. GENERAL INFORMATION

2.1 Course	:	SE3E2. Supervised professional practice
2.2 Semester	:	8^{vo} Semestre.
2.3 Credits	:	5
2.4 Horas	:	3 HT;
2.5 Duration of the period	:	16 weeks
2.6 Type of course	:	Mandatory
2.7 Learning modality	:	Face to face
2.8 Prerrequisites	:	6tocicloaprobado 6tocicloaprobado

3. PROFESSORS

Meetings after coordination with the professor

4. INTRODUCTION TO THE COURSE

Write justification for this course here ...

5. GOALS

- Write your first goal here.
- Write your second goal here.
- Just in case you need more goals write them here

6. COMPETENCES

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

7. TOPICS

Unit 1: title for the unit goes here (5)	
Competences Expected:	
Topics	**Learning Outcomes**
• Topic1 • Topic2 • Topic3	• Learning outcome1 [Levelforthislearningoutcome]. • Apply computing in complex problems [Usage]. • Create a search engine [Assessment]. • Study data structures [Familiarity].
Readings : [**Bibitem1**], [**Bibitem2**]	

Unit 2: another unit goes here (1)	
Competences Expected:	
Topics	**Learning Outcomes**
• Topic1	• Learning outcome xyz [Levelforthislearningoutcome].
Readings : [**Bibitem3**], [**Bibitem1**]	

8. WORKPLAN

8.1 Methodology

Individual and team participation is encouraged to present their ideas, motivating them with additional points in the different stages of the course evaluation.

8.2 Theory Sessions

The theory sessions are held in master classes with activities including active learning and roleplay to allow students to internalize the concepts.

8.3 Practical Sessions

The practical sessions are held in class where a series of exercises and/or practical concepts are developed through problem solving, problem solving, specific exercises and/or in application contexts.

9. EVALUATION SYSTEM

********* EVALUATION MISSING ********

10. BASIC BIBLIOGRAPHY

National University of Copsta Rica (UNA)
School of Informatics
Sillabus 2024-I

1. COURSE

SE404. Elective IV (Mandatory)

2. GENERAL INFORMATION

2.1 Course	:	SE404. Elective IV
2.2 Semester	:	8^{vo} Semestre.
2.3 Credits	:	3
2.4 Horas	:	3 HT;
2.5 Duration of the period	:	16 weeks
2.6 Type of course	:	Mandatory
2.7 Learning modality	:	Face to face
2.8 Prerrequisites	:	None None

3. PROFESSORS

Meetings after coordination with the professor

4. INTRODUCTION TO THE COURSE

Write justification for this course here ...

5. GOALS

- Write your first goal here.
- Write your second goal here.
- Just in case you need more goals write them here

6. COMPETENCES

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

7. TOPICS

Unit 1: title for the unit goes here (5)	
Competences Expected:	
Topics	**Learning Outcomes**
• Topic1 • Topic2 • Topic3	• Learning outcome1 [Levelforthislearningoutcome]. • Apply computing in complex problems [Usage]. • Create a search engine [Assessment]. • Study data structures [Familiarity].
Readings : [**Bibitem1**], [**Bibitem2**]	

Unit 2: another unit goes here (1)	
Competences Expected:	
Topics	**Learning Outcomes**
• Topic1	• Learning outcome xyz [Levelforthislearningoutcome].
Readings : [**Bibitem3**], [**Bibitem1**]	

8. WORKPLAN

8.1 Methodology

Individual and team participation is encouraged to present their ideas, motivating them with additional points in the different stages of the course evaluation.

8.2 Theory Sessions

The theory sessions are held in master classes with activities including active learning and roleplay to allow students to internalize the concepts.

8.3 Practical Sessions

The practical sessions are held in class where a series of exercises and/or practical concepts are developed through problem solving, problem solving, specific exercises and/or in application contexts.

9. EVALUATION SYSTEM

********* EVALUATION MISSING ********

10. BASIC BIBLIOGRAPHY