San Pablo Catholic University (UCSP)
Undergraduate Program in
Computer Science
SILABO

Universidad Católica San Pablo

CS211. Computer Science Theory (Mandatory)

2020-II

1. General information		
1.1 School	:	Ciencia de la Computación
1.2 Course	:	CS211. Computer Science Theory
1.3 Semester	:	4^{to} Semestre.
1.4 Prerrequisites	:	CS1D02. Discrete Structures II. (2^{nd} Sem)
1.5 Type of course	:	Mandatory
1.6 Learning modality	:	Virtual
1.7 Horas	:	2 HT; 2 HP; 2 HL;
1.8 Credits	:	4

2. Professors

3. Course foundation

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

4. Summary

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

5. Generales Goals

- That the student learn the fundamental concepts of the theory of formal languages.

6. Contribution to Outcomes

This discipline contributes to the achievement of the following outcomes:

a) An ability to apply knowledge of mathematics, science. (**Assessment**)

b) An ability to design and conduct experiments, as well as to analyze and interpret data. (**Assessment**)

j) Apply the mathematical basis, principles of algorithms and the theory of Computer Science in the modeling and design of computational systems in such a way as to demonstrate understanding of the equilibrium points involved in the chosen option. (**Assessment**)

7. Content

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

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

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

8. Methodology

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

El profesor del curso presentará demostraciones para fundamentar clases teóricas.

El profesor y los alumnos realizarán prácticas

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

9. Assessment

Continuous Assessment 1 : 20 %

Partial Exam : 30 %

Continuous Assessment 2 : 20 %

Final exam : 30 %

References

Brookshear, J. Glenn (1993). *Teoría de la Computación*. Addison Wesley Iberoamericana.

Hopcroft, John E. and Jeffrey D. Ullman (2013). *Introducción a la Teoría de Autómatas, Lenguajes y Computación*. Pearson Education.

Linz, Peter (2011). *An Introduction to Formal Languages and Automata*. 5th. Jones and Bartlett Learning.

Martin, John (2010). *Introduction to Languages and the Theory of Computation*. 4th. McGraw-Hill.

Sipser, Michael (2012). *Introduction to the Theory of Computation*. 3rd. Cengage Learning.