Universidad Nacional Mayor de San Marcos
School of Computer Science
Syllabus of Course
Academic Period 2018-II

1. **Code and Name:** CS2101. Theory of Computation (Mandatory)
2. **Credits:** 4
3. **Hours of theory and Lab:** 2 HT; 4 HL; (15 weeks)
4. **Professor(s)**

   Meetings after coordination with the professor
5. **Bibliography**

[Bro93] J. Glenn Brookshear. *Teoría de la Computación.* Addison Wesley Iberoamericana, 1993.

[HU93] John E. Hopcroft and Jeffrey D. Ullman. *Introducción a la Teoría de Autómatas, Lenguajes y Computación.* CECSA, 1993.

[Kel95] Dean Kelley. *Teoría de Autómatas y Lenguajes Formales.* Prentice Hall, 1995.

[Kol97] Ross Kolman Busby. *Estructuras de Matemáticas Discretas para la Computación.* Prentice Hall, 1997.

6. **Information about the course**

   (a) **Brief description about the course** This course emphasizes formal languages, computer models and computability, as well as the fundamentals of computational complexity and complete NP problems.

   (b) **Prerrequisites:** CS1D02. Discrete Structures II. (2^{nd} Sem)

   (c) **Type of Course:** Mandatory

   (d) **Modality:** Face to face

7. **Specific goals of the Course**

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

8. **Contribution to 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**)

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**)

9. **Competences (IEEE)**

C8. Understanding of what current technologies can and cannot accomplish. ⇒ **Outcome a**

C9. Understanding of computing's limitations, including the difference between what computing is inherently incapable of doing vs. what may be accomplished via future science and technology.⇒ **Outcome b,j**

C8. Understanding of what current technologies can and cannot accomplish. ⇒ **Outcome a**

C9. Understanding of computing's limitations, including the difference between what computing is inherently incapable of doing vs. what may be accomplished via future science and technology.⇒ **Outcome b,j**

10. List of topics

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

11. Methodology and Evaluation

Methodology:

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

Lab Sessions:
In order to verify their competences, several activities including active learning and roleplay will be developed during lab sessions.

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

Reading:
Throughout the course different readings are provided, which are evaluated. The average of the notes in the readings is considered as the mark of a qualified practice. The use of the UTEC Online virtual campus allows each student to access the course information, and interact outside the classroom with the teacher and with the other students.

Evaluation System:

12. Content

Unit 1: Basic Automata Computability and Complexity (20)	
Competences Expected: C9	
Learning Outcomes	**Topics**
• 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]	• 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
Readings : [Kol97], [Kel95]	

Unit 2: Advanced Computational Complexity (20)	
Competences Expected: C8,C9	
Learning Outcomes	**Topics**
• 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]	• 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
Readings : [Kel95], [HU93]	

<table>
<tr><th colspan="2">Unit 3: Advanced Automata Theory and Computability (20)</th></tr>
<tr><th colspan="2">Competences Expected: C8</th></tr>
<tr><th>Learning Outcomes</th><th>Topics</th></tr>
<tr><td>
• 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]
</td><td>
• 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
</td></tr>
<tr><td colspan="2">Readings : [HU93], [Bro93]</td></tr>
</table>