SOCIEDAD
PERUANA DE
COMPUTACIÓN

**Peruvian Computing Society (SPC)**
School of Computer Science
Sillabus 2021-I

## 1. COURSE

CS1D1. Discrete Structures I (Mandatory)

## 2. GENERAL INFORMATION

| | | |
|---|---|---|
| **2.1 Credits** | : | 4 |
| **2.2 Theory Hours** | : | 2 (Weekly) |
| **2.3 Practice Hours** | : | - |
| **2.4 Duration of the period** | : | 16 weeks |
| **2.5 Type of course** | : | Mandatory |
| **2.6 Modality** | : | Face to face |
| **2.7 Prerrequisites** | : | 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

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

**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. (**Usage**)

## 7. SPECIFIC COMPETENCES

**a1)** Apply demonstration techniques (direct method, contrapositive, induction and contradiction) to demonstrate properties in discrete structures and algorithms. ()

**a2)** Use logical propositions in an orderly manner. ()

**a3)** Apply counting techniques in solving computer problems. ()

**j1)** Solve recurrence problems to simplify algorithmic complexity ()

**j2)** Apply graph and tree theory for optimization and problem solving ()

## 8. TOPICS

| **Unit 1: Sets, Relations, and Functions (22)** | |
|---|---|
| **Competences Expected: a,j** | |
| **Topics** | **Learning Outcomes** |
| • Sets<br>– Venn diagrams<br>– Union, intersection, complement<br>– Cartesian product<br>– Power sets<br>– Cardinality of finite sets<br>• Relations:<br>– Reflexivity, simmetry, transitivity<br>– Equivalence relations<br>– Partial order relations and sets<br>– Extremal elements of a partially ordered sets<br>• Functions<br>– Surjections, injections, bijections<br>– Inverses<br>– Composition | • Explain with examples the basic terminology of functions, relations, and sets [Assessment]<br>• Perform the operations associated with sets, functions, and relations [Assessment]<br>• Relate practical examples to the appropriate set, function, or relation model, and interpret the associated operations and terminology in context [Assessment] |
| **Readings :** [Gri03], [Ros07], [Vel06] | |

**Unit 2: Basic Logic (14)**

**Competences Expected: a,j**

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

**Readings :** [Ros07], [Gri03], [Vel06]

**Unit 3: Proof Techniques (14)**

**Competences Expected: a,j**

| Topics | Learning Outcomes |
|---|---|
| • Notions of implication, equivalence, converse, inverse, contrapositive, negation, and contradiction<br>• The structure of mathematical proofs<br>• Direct proofs<br>• Disproving by counterexample<br>• Proof by contradiction<br>• Induction over natural numbers<br>• Structural induction<br>• Weak and strong induction (i.e., First and Second Principle of Induction)<br>• Recursive mathematical definitions<br>• Well orderings | • Identify the proof technique used in a given proof [Assessment]<br>• Outline the basic structure of each proof technique (direct proof, proof by contradiction, and induction) described in this unit [Usage]<br>• Apply each of the proof techniques (direct proof, proof by contradiction, and induction) correctly in the construction of a sound argument [Usage]<br>• Determine which type of proof is best for a given problem [Assessment]<br>• Explain the parallels between ideas of mathematical and/or structural induction to recursion and recursively defined structures [Familiarity]<br>• Explain the relationship between weak and strong induction and give examples of the appropriate use of each [Assessment]<br>• State the well-ordering principle and its relationship to mathematical induction [Familiarity] |

**Readings :** [Ros07], [Sch12], [Vel06]

| Unit 4: Data Representation (10) | |
|---|---|
| **Competences Expected: a,j** | |
| **Topics** | **Learning Outcomes** |
| • Numerical representation: sign-magnitude, floating point.<br>• Representation of other objects: sets, relations, functions. | • Explain numerical representations such as sign-magnitude and floating point. [Assessment].<br>• Carry out arithmetic operations using different kinds of representations. [Assessment].<br>• Explain the floating point standard IEEE-754 [Familiarity]. |
| **Readings :** [Ros07], [Gri03], [Vel06] | |

## 9. WORKPLAN

### 9.1 Methodology

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

### 9.2 Theory Sessions

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

### 9.3 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.

## 10. EVALUATION SYSTEM

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

## 11. 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. Mc Graw Hill, 2007.

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

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