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

CS212. Algorithm Analysis and Design (Mandatory)

2023-I

1. General information

1.1 School	:	Ciencia de la Computación
1.2 Course	:	CS212. Algorithm Analysis and Design
1.3 Semester	:	5to Semestre.
1.4 Prerrequisites	:	CS210. Algorithms and Data Structures. (4th Sem)
1.5 Type of course	:	Mandatory
1.6 Learning modality	:	Face to face
1.7 Horas	:	2 HT; 4 HP;
1.8 Credits	:	4
1.9 Plan	:	Plan Curricular 2016

2. Professors

Lecturer

- Juan Carlos Gutiérrez Cáceres <jcgutierrezc@ucsp.edu.pe>
  - PhD in Ciencia de la Computación, Universidad Nacional de San Agustín, Perú, 2013.
  - MSc in Ciencia de la Computación, ICMC-USP, Brasil, 2003.

3. Course foundation

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

4. Summary

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

5. Generales Goals

- Develop the ability to evaluate the complexity and quality of algorithms proposed for a given problem.
- Study the most representative, introductory algorithms of the most important classes of problems treated in computation.
- Develop the ability to solve algorithmic problems using the fundamental principles of algorithm design learneds.
- Be able to answer the following questions when a new algorithm is presented: How good is the performance ?, Is there a better way to solve the problem?

6. Contribution to Outcomes

This discipline contributes to the achievement of the following outcomes:

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

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

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

7. Content

UNIT 1: Basic Analysis (10)	
Competences:	
Content	**Generales Goals**
• Differences among best, expected, and worst case behaviors of an algorithm • Asymptotic analysis of upper and expected complexity bounds • Complexity classes, such as constant, logarithmic, linear, quadratic, and exponential • Asymptotic Notation • Analysis of iterative and recursive algorithms • Inductive proofs and correctness of algorithms • Master Theorem and Recursion Trees	• Explain what is meant by "best", "expected", and "worst" case behavior of an algorithm [Assessment] • Determine informally the time and space complexity of different algorithms [Assessment] • List and contrast standard complexity classes [Assessment] • Explain the use of big omega, big theta, and little o notation to describe the amount of work done by an algorithm [Assessment] • Analyze worst-case running times of algorithms using asymptotic analysis [Assessment] • Use recurrence relations to determine the time complexity of recursively defined algorithms [Assessment] • Solve elementary recurrence relations, eg, using some form of a Master Theorem [Assessment] • Argue the correctness of algorithms using inductive proofs [Assessment]
Readings: Kleinberg and Tardos (2005), Dasgupta, Papadimitriou, and Vazirani (2006), Cormen et al. (2009), Sedgewick and Flajolet (2013), Knuth (1997)	

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

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

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

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

8. Methodology

1. El profesor del curso presentará clases teóricas de los temas señalados en el programa propiciando la intervención de los alumnos.
2. El profesor del curso presentará demostraciones para fundamentar clases teóricas.
3. El profesor y los alumnos realizarán prácticas
4. Los alumnos deberán asistir a clase habiendo leído lo que el profesor va a presentar. De esta manera se facilitará la comprensión y los estudiantes estarán en mejores condiciones de hacer consultas en clase.

9. Assessment

Continuous Assessment 1 : 20 %

Partial Exam : 30 %

Continuous Assessment 2 : 20 %

Final exam : 30 %

References

Alsuwaiyel, H. (1999). *Algorithms: Design Techniques and Analysis.* World Scientific. ISBN: 9789810237400.
Cormen, Thomas H. et al. (2009). *Introduction to Algorithms, Third Edition.* 3rd. The MIT Press. ISBN: 0262033844.
Dasgupta, S., C. Papadimitriou, and U. Vazirani (2006). *Algorithms.* McGraw-Hill Education. ISBN: 9780073523408.
Goodrich, Michael T. and Roberto Tamassia (2009). *Algorithm Design: Foundations, Analysis and Internet Examples.* 2nd. John Wiley & Sons, Inc. ISBN: 0470088540, 9780470088548.

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

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

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

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

Sedgewick, R. and K. Wayne (2011). *Algorithms.* Pearson Education. ISBN: 9780132762564.

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