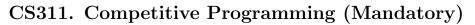
San Pablo Catholic University (UCSP) **Undergraduate** Program in **Computer Science SILABO**



Universidad Católica San Pablo	CS311. Competitive Programming (Mandatory)	
2022-I 1. General information		
1.1 School	: Ciencia de la Computación	
1.2 Course	: CS311. Competitive Programming	
1.3 Semester	: 6^{to} Semestre.	
1.4 Prerrequisites	: CS212. Algorithm Analysis and Design. (5^{th} Sem)	
1.5 Type of course	: Mandatory	
1.6 Learning modality	: Blended	
1.7 Horas	: 2 HT; 2 HP; 2 HL;	
1.8 Credits	: 4	

2. Professors

Lecturer

• Rensso Victor Hugo Mora Colque <rvhmora@ucsp.edu.pe> - MSc in Ciencia de la Computación, Universidade Federal de Ouro Preto, Brasil, 2012.

3. Course foundation

Competitive Programming combines problem-solving challenges with the fun of competing with others. It teaches participants to think faster and develop problem-solving skills that are in high demand in the industry. This course will teach you to solve algorithmic problems quickly by combining theory of algorithms and data structures with practice solving problems.

4. Summary

1. Introduction 2. Data structure 3. Algorithmic Design Paradigms 4. Graphs 5. Advanced topics 6. Domain specific problems

5. Generales Goals

- That the student uses techniques of data structures and complex algorithms..
- That the student apply the concepts learned for the application on a real problem.
- That the student investigate the possibility of creating a new algorithm and / or new technique to solve a real problem.

6. Contribution to Outcomes

This discipline contributes to the achievement of the following outcomes:

- a) An ability to apply knowledge of mathematics, science. (Usage)
- b) An ability to design and conduct experiments, as well as to analyze and interpret data. (Usage)
- i) An ability to use the techniques, skills, and modern computing tools necessary for computing practice. (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. Content				
UNIT 1: Introduction (20)				
Competences: a,b				
Content	Generales Goals			
 Introduction to Competetive Programming Computational model Runtime and space complexity Recurrence and recursion Divide and conquer 	 Identify and learn how to use the resources in the Random Access Machine (RAM) computational model. [Usage] Compute the runtime and space complexity for written algorithms. [Usage] Compute the recurrence relations for recursive algorithms. [Usage] Solve problems related to searching and sorting. [Usage] Learning to select the right algorithms for divide-and-conquer problems. [Usage] Design new algorithms for real-world problem solving.[Usage] 			

Readings: Cormen et al. (2009), Halim (2013), Kulikov (2019), Miguel A. Revilla (2003), Laaksonen (2017), Aziz, Lee, and Prakash (2012)

UNIT 2: Data structure (20)			
Competences: a,b			
Content	Generales Goals		
 Arrays and strings problems Linked lists problems Stacks and queues problems Trees problems Hash tables problems Heaps problems 	 Recognize different data structures, their complexities, uses and restrictions.[Usage] Identify the type of data structure appropriate to the resolution of the problem. [Usage] Recognize types of problems associated with operations on data structures such as searching, inserting, deleting and updating.[Usage] 		

Readings: Cormen et al. (2009), Halim (2013), Kulikov (2019), Miguel A. Revilla (2003), Laaksonen (2017), Aziz, Lee, and Prakash (2012)

Competences: a,b		
Content	Generales Goals	
 Brute force Divide and conquer Backtracking Greedy 	 Learning the different algorithmic design paradigms.[Usage] Learning to select the right algorithms for different problems applying different algorithmic design paradigms.[Usage] 	
• Dynamic Programming		

Lee, and Prakash (2012)

UNIT 4: Graphs (20) Competences: a,b			
Content	Generales Goals		
 Graphs transversal Graphs aplications Shortest path Networks and flows 	 Identify problems classified as graph problems. [Us age] Learn how to select the right algorithms for network problems (transversal, MST, shortest-path, network and flows). [Usage] 		

Readings: Cormen et al. (2009), Halim (2013), Kulikov (2019), Miguel A. Revilla (2003), Laaksonen (2017), Aziz, Lee, and Prakash (2012)

UNIT 5: Advanced topics (20)			
Competences: a,b			
Content	Generales Goals		
 Number theory Probabilities and combinations String algorithms (tries, string hashing, z-algorithm) Geometric algorithms 	 Learning to select the right algorithms for problems in number theory and mathematics as they are im- portant in competitive programming. [Usage] Learning to select the right algorithms for problems about probabilities and combinations, strings and computational geometry. [Usage] 		
Readings: Cormen et al. (2009), Halim (2013), Kulikov (2	 2019), Miguel A. Revilla (2003), Laaksonen (2017), Azi		

Lee, and Prakash (2012)

UNIT 6: Domain specific problems (20) Competences: a,b		
Content	Generales Goals	
• Latency and throughput	• Learning to design systems for different domain-	
• Parallelism	specific problems by applying knowledge about net works, distributed computing, high availability, stor	
• Networks	age and system architecture.[Usage]	
• Storage		
• High availability		
• Caching		
• Proxies		
• Load balancers		
• Key-value stores		
• Replicating and sharing		
• Leader election		
• Rate limiting		
• Logging and monitoring		
 Logging and monitoring Readings: Cormen et al. (2009), Halim (2013), Kulikov (2012) 	2019), Miguel A. Revilla (2003), Laa	

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

- Aziz, A., T.H. Lee, and A. Prakash (2012). Elements of Programming Interviews: The Insiders' Guide. ElementsOfProgrammingInterviews.com. ISBN: 9781479274833.
- Cormen, T. H. et al. (2009). Introduction to Algorithms. MIT Press.
- Halim, Steven (2013). Competitive Programming. 3 rd. Lulu.
- Kulikov, Alexander S. (2019). Learning Algorithms Through Programming and Puzzle Solving. Active Learning Technologies.
- Laaksonen, Antti (2017). Guide to Competitive Programming: Learning and Improving Algorithms Through Contests. Stringer.
- Miguel A. Revilla, Steve Skiena (May 2003). Programming Challenges: The Programming Contest Training Manual. Springer. ISBN: 978-0387001630.