San Cristobal of Huamanga National University (UNSCH)
School of Computer Science
Syllabus 2024-II

1. COURSE

CS113. Computer Science II (Mandatory)

2. GENERAL INFORMATION

2.1 Course	:	CS113. Computer Science II
2.2 Semester	:	3^{rd} Semester.
2.3 Credits	:	4
2.4 Horas	:	2 HT; 4 HP;
2.5 Duration of the period	:	16 weeks
2.6 Type of course	:	Mandatory
2.7 Learning modality	:	Face to face
2.8 Prerrequisites	:	CS112. Computer Science I. (2^{nd} Sem) CS112. Computer Science I. (2^{nd} Sem)

3. PROFESSORS

Meetings after coordination with the professor

4. INTRODUCTION TO THE COURSE

This is the third course in the sequence of introductory courses in computer science. This course is intended to cover Concepts indicated by the Computing Curriculum IEEE (c) -ACM 2001, under the functional-first approach. The object-oriented paradigm allows us to combat complexity by making models from abstractions of the problem elements and using techniques such as encapsulation, modularity, polymorphism and inheritance. The Dominion of these topics will enable participants to provide computational solutions to design problems simple of the real world.

5. GOALS

- Introduce the student in the fundaments of the paradigm of object orientation, allowing the assimilation of concepts necessary to develop an information system

6. COMPETENCES

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

3) Communicate effectively in a variety of professional contexts.. (Usage)

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

7. TOPICS

Unit 1: Advanced STL (8)	
Competences Expected: 1	
Topics	**Learning Outcomes**
• Associative Containers (std::set, std::map, std::unordered_set, std::unordered_map). • Adapters (std::stack, std::queue, std::priority_queue). • Advanced STL Algorithms. • Functors and Predicates.	• Understand the use of associative containers. [Usar] • Implement programs that use STL adapters. [Usar] • Apply advanced STL algorithms. [Usar] • Use functors and predicates with the STL. [Usar]
Readings : [**Stroustrup2013**], [MJo19], [**Deitel17**]	

Unit 2: Advanced Templates (7)	
Competences Expected: 1	
Topics	**Learning Outcomes**
• Template Metaprogramming. • SFINAE (Substitution Failure Is Not An Error). • Perfect Forwarding.	• Apply template metaprogramming to solve complex problems. [Usar] • Understand and use SFINAE for template selection. [Usar] • Use Perfect Forwarding for efficient argument passing. [Usar]
Readings : [**Stroustrup2013**], [MJo19], [**Deitel17**]	

Unit 3: Variadic Templates (12)	
Competences Expected: 1	
Topics	**Learning Outcomes**
• Introduction to Variadic Templates. • Variadic Template Functions. • Variadic Template Methods. • Variadic Template Classes. • Classes inheriting from variable lists of variadic templates. • Example: Implementing a custom tuple class.	• Understand the concept of variadic templates. [Familiarizarse] • Implement variadic functions. [Usar] • Design classes with variadic methods. [Usar] • Create classes with variadic templates. [Usar] • Implement inheritance with variable lists of variadic templates. [Usar] • Apply variadic templates to solve real-world problems. [Usar]
Readings : [**Stroustrup2013**], [MJo19], [**Deitel17**]	

Unit 4: Move Semantics and Rvalue References (5)	
Competences Expected: 1	
Topics	**Learning Outcomes**
• Lvalues and Rvalues. • Rvalue References. • Move Semantics. • Move Constructors and Move Assignment Operators. • Perfect Forwarding (review).	• Explain move semantics and its purpose in C++. [Familiarizarse] • Define and use rvalue references. [Usar] • Analyze the performance implications of using move semantics. [Evaluar] • Implement move constructors and move assignment operators for custom classes. [Usar] • Apply move semantics to optimize resource management in C++ programs. [Usar]
Readings : [**Stroustrup2013**], [MJo19], [**Deitel17**]	

Unit 5: Design Patterns (Creational and Structural) (6)	
Competences Expected: 1	
Topics	**Learning Outcomes**
• Singleton, Factory, Builder. • Adapter, Decorator, Facade.	• Understand and apply creational design patterns: Singleton, Factory, Builder. [Usar] • Understand and apply structural design patterns: Adapter, Decorator, Facade. [Usar]
Readings : [**Stroustrup2013**], [MJo19], [**Deitel17**]	

Unit 6: Functors (3)	
Competences Expected: 1,3	
Topics	**Learning Outcomes**
• Definition of Functors. • Functors and Templates. • Passing Functors to Functions using parameters. • Passing Functors to Functions using templates. • Passing Functors to Classes using parameters. • Passing Functors to Classes using templates. • Examples and Applications.	• Introduction to Functors. [Usar] • Using Functors as parameters to functions and classes. [Usar] • Using Functors in functions and classes through templates. [Usar]
Readings : [**Stroustrup2013**], [MJo19], [**Deitel17**]	

8. WORKPLAN

8.1 Methodology

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

8.2 Theory Sessions

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

8.3 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.

9. EVALUATION SYSTEM

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

10. BASIC BIBLIOGRAPHY

[MJo19] Nicolai M.Josuttis. *C++17-The Complete Guide*. 1st. 2019.