COURSE DESCRIPTION

1. Program Information

1.1 University	**“Alexandru Ioan Cuza”, Iasi**
1.2 Faculty	**Computer Science**
1.3 Department	**Computer Science**
1.4 Field of Study	**Computer Science**
1.5 Type of Degree	**Bachelor**
1.6 Study Program / Qualification	**Computer Science/Bachelor’s Degree in Computer Science**

2. Course Information

2.1 Course Title				Object Oriented Programming			
2.2 Course Coordinator				**Prof. dr. DOREL LUCANU** **Conf. dr. DRAGOŞ GAVRILUŢ**			
2.3 Seminary Coordinator				**Conf. dr. DRAGOŞ GAVRILUŢ**			
2.4 Study Year		**2.5** Semester		**2.6** Evaluation	**Mixed**	**2.7** Course Status*	C

3. Total estimated hours (hours per semester and didactic activities)

3.1 Hours per week	4	in which: **3.2** course	2	**3.3** seminar	2
3.4 Hours in curriculum	56	in which: **3.5** course	28	**3.6** seminar	28

Time Distribution	hours
Individual of study using course support, bibliography, and others	**14**
Individual documentation in library, using electronic tools, field work	**14**
Preparation of eminaries/laboratories, homeworks, reports, portfolios, and essays	**53**
Tutoring	-
Evaluation	**12**
Other activities	**1**

3.7 Total hours individual study	**81**
3.8 Total hours per semester	**150**
3.9 Number of credits	**6**

4. Preconditions (if necessary)

4.1 Curriculum	Introduction to Programming
4.2 Competences	Procedural and imperative programming in C++

5. Conditions (if necessary)

5.1 For Course Operation	**-**
5.2 For Seminary/Laboratory Operation	**-**mandatory attendance

6. Specific Skills Acquired

Professional Skills	Upon successful completion of this discipline, students will be able to: C1. Explains the syntax and semantics of the C ++ language instructions. C2. Describes through UML diagrams software systems of elementary/medium difficulty level. C3. Use POO concepts and OO patterns in designing software systems. C4. Analyze the requirements of software systems of elementary/medium difficulty level. C5. Write C ++ code starting from system specifications written in UML.
Transversal Skills	CT1. Combining OO modeling with OO programming. CT2. Developing the ability to model practical applications in various domains using OO concepts. CT3. The ability to use UML models as a means of communicating with customers in these domains.

7. Course Objectives (from the grid of specific skills acquired)

7.1 General Objectives	Assimilation of object-oriented programming at abstract level using UML, and at the concrete level using the C ++ programming language.
7.2 Specific Objectives	Ability to understand, explain, analyze, use: O1. classes, objects, class hierarchies, polymorphism, abstract classes, interfaces, parameterized classes; O2. modeling in UML at introductory level; O3. C ++ language (ISO Standard), with emphasis on the representation of classes, objects, and relationships between them; O4. design patterns; O5. Standard Generic Type Library (STL).

8. General Description

8.1	**Course**	**Teaching Methods**	**Observations** (hours and bibliographic references)
1.	Introduction to C++. Classes (1)	Lecture	2
2.	Method overloading. Friend specifier	Lecture	2
3.	Constructors & Initialization lists	Lecture	2
4.	Destructor. Operators. Operations with objects	Lecture	2
5.	Inheritance. Virtual methods. Abstract classes	Lecture	2
6.	Casting. Macros. Templates	Lecture	2

7.	STL	Lecture	2
8.	Partial evaluation	Programming test	6
9.	Constant Expressions. For each loops. Type Inference. Static polymorphism. Structured bindings, PODs	Lecture	2
10.	Lambda Expressions	Lecture	2
11.	Exceptions in C++	Lecture	2
12.	Testing (TDD)	Lecture	2
13.	OO modeling and Design	Lecture	2
14.	OO Design Patterns	Lecture	2

BIBLIOGRAPHY

H. Schildt: C++ manual complet, Teora, 2000
D. Kaler, M.J. Tobler, J. Valter: C++, Teora, 2000
Bjarne Stroustrup: The C++ Programming Language, Adisson-Wesley, 3nd edition, 1997
Bruce Eckel **:** Thinking in C++, 2nd Edition
C++ Tutorial Manual, http://www.cplusplus.com/files/tutorial.pdf
C++ Standards 1998 (ISO/IEC 14882:1998, http://tracker.ceph.com/attachments/download/752/cpp98.pdf
C++ Standards 2011 (Draft), http://www.open-std.org/jtc1/sc22/wg21/docs/papers/2011/n3242.pdf
C++ Standards 2013, http://www.open-std.org/jtc1/sc22/wg21/docs/papers/2013/n3690.pdf
C++ Standards 2017, http://www.open-std.org/jtc1/sc22/wg21/docs/papers/2017/n4713.pdf
C++ Standards 2020 (C++20), http://www.open-std.org/jtc1/sc22/wg21/docs/papers/2020/n4849.pdf
C++ Draft 2023 (C++23), https://github.com/cplusplus/draft/releases/download/n4901/n4901.pdf
C++ Tutorial Manual, http://www.cplusplus.com/files/tutorial.pdf

Donald Bell. An introduction to the Unified Modeling Language, https://developer.ibm.com/articles/an-introduction-to-uml/
SGI Standard Template Library Programmer's Guide, https://www.boost.org/sgi/stl/
Introduction to Test Driven Development (TDD). http://agiledata.org/essays/tdd.html
Erich Gamma, Ralph Johnson, John Vlissides, Richard Helm. Design Patterns Elements of Reusable Object-Oriented Software (GoF)
Design Patterns Explained Simply https://sourcemaking.com/design-patterns-book

8.2	**Seminary / Laboratory**	**Teaching/Evaluation methods**	**Observations** (hours and bibliographic references)
1.	Recap C: pointers, files.	Review of topics presented in the course, Individual work on a given set of exercises, Interactive methods in the development of the solutions	

2.	Experimenting method overloading, friend specifier	Experiment, Individual work, Interactive methods	2
3.	Experimenting constructors & Initialization lists	Idem	2
4.	Experimenting Destructor. Operators. Operations with objects	Idem	2
5.	Inheritance. Virtual methods. Abstract classes	Idem	2
6.	Experimenting Casting. Macros. Templates	Idem	2
7.	Experimenting STL	Idem	2
8.	Test	Programming test	2
9.	Experimenting Constant Expressions. For each loops. Type Inference. Static polymorphism. Structured bindings, PODs	Idem	2
10.	Experimenting Lambda Expressions	Idem	2
11.	Experimenting Exceptions in C++	Idem	2
12.	Experimenting Testing (TDD)	Idem	2
13.	Experimenting OO modeling and Design	Idem	2
14.	Test	Programming test	2

Bibliography

That for the course, plus
Visual Studio 2022 Community Edition, https://visualstudio.microsoft.com/downloads/
Compiler explorer (online), https://godbolt.org/
Visual code, https://code.visualstudio.com/download
CodeBlocks portable edition, http://codeblocks.codecutter.org/CodeBlocks-EP.zip

9. Course content synchronization with the expectations of the community representatives, professional associations and employers from the program domain

The content of the discipline is corroborated with the theme of competitions for filling positions in IT companies.

10. Evaluation

Activity Type	10.1 Evaluation criteria	10.2 Evaluation methods	10.3 The weight of each evaluation form (%)
10.4 Course	- the ability to model and design a simple system - the ability to transform a UML diagram into C ++ code - the ability to understand and explain a C ++ code - the quality of the answers	Written test	30%
10.5 Seminary	- the ability to develop POO programs that solve problems of simple and medium level - the ability to apply OO design patterns to finding solutions - quality of written programs	Programming test (30%) Lab activity (10%)	70%

10.6 Minimal performance standards

Passing is conditioned on gaining a minimum of 45 points and should reflect the assimilation of the following concepts & skills:

- the capability to write C++ programs based on specifications
- the capability to correctly apply OOP principles (inheritance, polymorphism, etc)
- the ability to understand OO principles/programming-techniques written in C++
- the ability to detect simple errors in a C++ program and understand them

Date

Course Coordinator
Prof. dr. Dorel Lucanu
Conf. dr. Dragoș Gavriluț

Seminary/Laboratory Coordinator
Conf. dr. Dragoș Gavriluț

Department Date of Approval

Director of the Department

Prof. dr. Dorel Lucanu