Universitatea Naţională de Ştiinţă şi Tehnologie Politehnica Bucureşti
Facultatea de Electronică, Telecomunicaţii şi
Tehnologia Informaţiei

COURSE DESCRIPTION

1. Program identification information

1.1 Higher education institution	**National University of Science and Technology Politehnica Bucharest**
1.2 Faculty	**Electronics, Telecommunications and Information Technology**
1.3 Department	**Telecommunications**
1.4 Domain of studies	Electronic Engineering, Telecommunications and Information Technology
1.5 Cycle of studies	Bachelor/Undergraduate
1.6 Programme of studies	Technologies and Telecommunications Systems

2. Date despre disciplină

2.1 Course name (ro) (en)				Programarea calculatoarelor şi limbaje de programare 2 Computer Programming and Programming Languages 2			
2.2 Course Lecturer				Conf. Dr. Iulian Nastac			
2.3 Instructor for practical activities				Conf. Dr. Iulian Nastac			
2.4 Year of studies	1	2.5 Semester	II	2.6. Evaluation type	E	2.7 Course regime	Ob
2.8 Course type		F	2.9 Course code	04.F.02.O.011		2.10 Tipul de notare	Nota

3. Total estimated time (hours per semester for academic activities)

3.1 Number of hours per week	3	Out of which: 3.2 course	2.00	3.3 seminary/laboratory	1
3.4 Total hours in the curricula	42.00	Out of which: 3.5 course	28	3.6 seminary/laboratory	14
Distribution of time:					hours
Study according to the manual, course support, bibliography and hand notes Supplemental documentation (library, electronic access resources, in the field, etc) Preparation for practical activities, homework, essays, portfolios, etc.					52
Tutoring					0
Examinations					6
Other activities (if any):					0

3.7 Total hours of individual study	58.00
3.8 Total hours per semester	100
3.9 Number of ECTS credit points	4

4. Prerequisites (if applicable) (where applicable)

4.1 Curriculum	Computer programming and programming languages 1

4.2 Results of learning	Ability to solve problems of reduced difficulty in C/C++.

5. Necessary conditions for the optimal development of teaching activities (where applicable)

5.1 Course	The course will be held in a room with computer and projection system.
5.2 Seminary/ Laboratory/Project	The laboratory will be equipped with computers and adequate software. The attendance is mandatory (according to the current regulations imposed by UPB). Individual access to computers and integrated development environments for C/C++, as well as internet access (access to e-learning Moodle).

6. General objective *(Reffering to the teachers' intentions for students and to what the students will be thought during the course. It offers an idea on the position of course in the scientific domain, as well as the role it has for the study programme. The course topics, the justification of including the course in the currcula of the study programme, etc. will be described in a general manner)*

Course: Learning the mechanisms of abstraction of programming problems in the form of classes and objects. Understanding how objects interact and the relationships between different classes. Using the concepts of object-oriented programming to solve problems expressed in natural language.
Laboratory: Practical learning, through the implementation of software programs, of the notions taught in the course. Solving practical problems that include objective-oriented programming elements.

7. Competences *(Proven capacity to use knowledge, aptitudes and personal, social and/or methodological abilities in work or study situations and for personal and proffesional growth. They refflect the empolyers requirements.)*

Specific Competences	- understanding and using fundamental concepts in the field of communications and information transmission. - application of basic knowledge, concepts and methods relating to the architecture of computing systems, microcontrollers, programming languages and techniques. - design and use of computer systems and computer networks.
Transversal (General) Competences	- the ability to make decisions to solve current or unpredictable problems that arise in the process of exploiting computing systems. - the ability to constantly inform and document for personal and professional development by reading the literature. - the ability to communicate and present technical content in both Romanian and English.

8. Learning outcomes *(Synthetic descriptions for what a student will be capable of doing or showing at the completion of a course. The learning outcomes reflect the student's acomplishments and to a lesser extent the teachers' intentions. The learning outcomes inform the students of what is expected from them with respect to performance and to obtain the desired grades and ECTS points. They are defined in concise terms, using verbs similar to the examples below and indicate what will be required for evaluation. The learning outcomes will be formulated so that the correlation with the competences defined in section 7 is highlighted.)*

Knowledge	*The result of knowledge aquisition through learning. The knowledge represents the totality of facts, priciples, theories and practices for a given work or study field. They can be theoretical and/or factual.* Lists the principles of object-oriented programming. Defines domain-specific notions such as class, object, inheritance, polymorphism. Describes the different access specifiers and their implications. Use correctly the operators studied and their overloaded variants. Interpret and implement classes according to textual descriptions formulated in natural language. Describes the relationships between objects in the context of inheritance and polymorphism. Interprets and defines classes according to UML diagrams.
Skills	*The capacity to apply the knowledge and use the know-how for completing tasks and solving problems. The skills are described as being cognitive (requiring the use of logical, intuitive and creative thinking) or practical (implying manual dexterity and the use of methods, materials, tools and intrumentation).* Select and group information about the structure of a class formulated in natural language. It uses the principles of object-oriented programming. Use appropriate scientific language. Test the identified solutions experimentally. Solve practical applications. Interpret causal relationships appropriately. Analyze and compare the results obtained with the desired ones. Explain the identified solutions/ways of solving.
Responsability and autonomy	*The student's capacity to autonomously and responsably apply their knowledge and skills.* Select appropriate bibliographic sources and analyze them. Respects the principles of academic ethics, correctly citing the bibliographic sources used. Demonstrates responsiveness to new learning contexts. Demonstrate autonomy in organizing the learning situation/context or problem-solving situation.

9. Teaching techniques *(Student centric techniques will be considered. The means for students to participate in defining their own study path, the identification of eventual fallbacks and the remedial measures that will be adopted in those cases will be described.)*

Starting from the analysis of student's learning characteristics and their specific needs, the teaching process will explore both expositional (lecture, exposure) and conversational-interactive teaching methods, based on discovery learning models facilitated by direct and indirect exploration of reality (experiment, demonstration, modeling), but also action-based methods, such as exercise, practical activities and problem solving.
Lectures will be used in the teaching activity, based on Power point presentations or different videos that will be made available to students. Each course will start with a review of the chapters already completed, with an emphasis on the notions taken at the last course.
Presentations use images and schemes so that the information presented is easy to understand and assimilate.
This discipline covers information and practical activities designed to support students in their efforts to learn and develop optimal collaboration and communication relationships in a climate conducive to learning by discovery.
It will be considered the practice of active listening and assertive communication skills, as well as feedback building mechanisms, as ways of regulating behavior in various situations and adapting the pedagogical approach to the learning needs of students.
The ability to work in teams will be exercised to solve different learning tasks.

10. Contents

COURSE		
Chapter	**Content**	**No. hours**
1	Introduction: General aspects of object-oriented programming, change from procedural programming language (C) to object-oriented programming language (C++).	2
2	Classes & objects: Introduction of class and object concepts. Specific operators C++; namespace; attributes and methods; setter & getter; constructor & destructors.	4
3	Access specifiers (public, private, protected). Specific class functions, friend functions and classes, solve exercises.	6
4	Inhertitance: Basic concepts, constructors/destructors calling order, Member concealment, function overwriting, virtual functions, ‘overside’, different types of inheritances, solving exercises.	6
5	Casting : static cast, dinamic cast, downcasting & upcasting.	2
6	Polymorphism. Overloading operators and functions, solving exercises.	6
7	Final exam.	2
	Total:	28

Bibliography:

D.I. Năstac, Note de curs POO, UPB, ETTI, Moodle https://curs.upb.ro/;
“The C++ Programming Language”, 4th Edition, Bjarne Stroustrup
„CPP Coding Standards - 101 Rules Guidelines and Best Practices”, Herb Stutter, Andrei Alexandrescu.
„Head First Object-Oriented Design and Analysis”, Brett McLaughlin, Gary Pollice, David West.
„Head First Design Patterns”, Eric Freeman, Elisabeth Robson, Bert Bates, Kathy Sierra.

LABORATORY		
Crt. no.	**Content**	**No. hours**
1	Classes & objects: Introduction of class and object concepts, access specifiers, constructor.	2
2	Classes & objects: Initialization lists, destructors. Access speculators, friend functions.	2
3	Inheritance: Basic concepts, order of calling constructor/destructors, hiding members.	2
4	Virtual concept: Overwriting functions. Overloading of operators.	2
5	Complex inheritance. Polymorphism. Recapitulation.	2
6	Project evaluation.	2
7	Final laboratory exam.	2
	Total:	14

Bibliography:
D.I. Năstac, Note de curs POO, UPB, ETTI, Moodle https://curs.upb.ro;
"The C++ Programming Language", 4th Edition, Bjarne Stroustrup
„CPP Coding Standards - 101 Rules Guidelines and Best Practices", Herb Stutter, Andrei Alexandrescu.
„Head First Object-Oriented Design and Analysis", Brett McLaughlin, Gary Pollice, David West.
„Head First Design Patterns", Eric Freeman, Elisabeth Robson, Bert Bates, Kathy Sierra.

11. Evaluation

Activity type	11.1 Evaluation criteria	11.2 Evaluation methods	11.3 Percentage of final grade
11.4 Course	Check the learning progress.	Written exam in week 14. The topics cover the whole course.	40%
11.5 Seminary/laboratory/project	Practical problems of C++ laguage using object-oriented programming.		30%
	Proof of functionality of the written code		30%
11.6 Passing conditions			
Participation in laboratory work; A minimum of 50% of the discipline score.			

12. Corroborate the content of the course with the expectations of representatives of employers and representative professional associations in the field of the program, as well as with the current state of knowledge in the scientific field approached and practices in higher education institutions in the European Higher Education Area (EHEA)

The course curriculum provides graduates with both the knowledge necessary to understand the principles of object-oriented programming and the foundations of programming using a general language such as C++. The current technological progress of electronic and telecommunications devices is conditioned by the ability to develop and experiment using programming languages. Thus, the object-oriented programming discipline is fundamental in the training of future generations of engineers and researchers in the field;
The curriculum thus provides graduates with skills appropriate to the needs of current qualifications and a modern, quality and competitive scientific and technical training that allows them to engage rapidly after graduation. It is perfectly framed in the politics of the Politehnica University of Bucharest, both in terms of content and structure, as well as in terms of the international skills and openness offered to students. Potential employers target both academia (didactic and research profile) and industrial research and development environment such as organizations/firms of any size, from small ones created by students/master students (e.g. start-up and spin-off) to multinational ones.

Date	Course lecturer	Instructor(s) for practical activities
23.09.2025	Conf. Dr. Iulian Nastac	Conf. Dr. Iulian Nastac

Universitatea Națională de Știință și Tehnologie Politehnica București
Facultatea de Electronică, Telecomunicații și
Tehnologia Informației

Date of department approval	Head of department Conf. Dr. Ing. Serban Obreja
Date of approval in the Faculty Council	Dean Prof. Dr. Ing. Mihnea Udrea