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)				Structuri de date şi algoritmi Algorithms and Data Structures			
2.2 Course Lecturer				Conf. Dr. Iulian Nastac			
2.3 Instructor for practical activities				Conf. Dr. Iulian Nastac			
2.4 Year of studies	2	2.5 Semester	1	2.6. Evaluation type	E	2.7 Course regime	Ob
2.8 Course type		D	2.9 Course code	04.D.03.O.005		2.10 Tipul de notare	Nota

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

3.1 Number of hours per week	3.5	Out of which: 3.2 course	2	3.3 seminary/laboratory	1.5
3.4 Total hours in the curricula	49	Out of which: 3.5 course	28	3.6 seminary/laboratory	21
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.					45
Tutoring					0
Examinations					6
Other activities (if any):					0

3.7 Total hours of individual study	51.00
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
4.2 Results of learning	Ability to write basic scripts 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 projector.

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: learn the mechanisms behind storing, structuring and processing of data with complex composition. Study the basic principles used in the formation of algorithms as an essential step in developing software applications. Criteria for effective design of programs. Case studies and methods for evaluating the performance of algorithms. Laboratory: practical learning through the implementation of software programs, of the notions taught in the course. Solving concrete practical problems the include elements of data structures and algorithms.

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	Ability to apply the fundamental knowledge, concepts and methods with respect to computer architecture, microcontrollers and programming languages.
Transversal (General) Competences	The capacity to get informed, with respect to scientific literature, which supports personal development as well as professional one.

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.* assimilate advanced concepts of computer programming, assimilate advanced programming concepts in C/C++, assimilate the knowledge necessary to solve advanced programming problems, **assimilate** the knowhow to develop complex programming algorithms, assimilate the knowhow to implement a problem from natural language into an algorithm which is implementable.

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).* the ability to understand and explain a program written in C/C++, the ability to build advanced algorithms, the ability to validate the results of a C/C++ program, the ability to identify advanced programming solutions, the ability to communicate and argument solutions.
Responsability and autonomy	*The student's capacity to autonomously and responsably apply their knowledge and skills.* the capacity to select and analyze bibliographical sources, the capacity to propose and contribute with new solutions to problems, the capacity to learn new concepts, the capacity to communicate information to other peers, the development of autonomy in the learning process.

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.)*

Course: teaching is done interactively using both a video projection system for PowerPoint presentations, as well as a black board. The basic concepts are presented and then example cases are discussed. They are solved interactively, going through the steps of understanding requirements, formalizing the algorithm, developing, code, correcting errors and validating results. All course materials are available on the Moodle platform in electronics format. Laboratory: the laboratory relies on individual C/C++ preprogramming on the Moodle platform. Each student has an individual computer. Students have both solved and proposed problems. The laboratory is preceded by short presentations to familiarize with theoretical concepts. All laboratory materials are available on Moodle platform in electronic format.

10. Contents

COURSE		
Chapter	**Content**	**No. hours**
1	Introduction. General notions regarding algorithms.	1
2	Pointers and data structures: working with pointer variables, dynamic allocation of memory (dynamic vs static representation); working with structures and unions; examples and practical applications.	3
3	Lists: defining and working with lists; examples and practical applications. Data queue: define and working with the queue; examples and applications. Data stack: defining and working with data stack, examples and practical applications. Circular lists and practical applications.	6
4	Lists: defining and working with lists; examples and practical applications. Data queue: define and working with the queue; examples and applications. Data stack: defining and working with data stack, examples and practical applications. Circular lists and practical applications.	6
5	Data trees: general information – solving different problems through tree algorithms; defining and working with binary trees; example and applications.	4

6	Special types of trees, heap trees. Transforming trees – vectors.	8
7	Sorting algorithms. Practical applications. Interactive solving of various problems.	0
	Total:	28

Bibliography:

D.I. Năstac, Course Notes for Computer Programming, UPB, online, Moodle ETTI platform
C++, http://www.cplusplus.com
D.I. Năstac, Structuri de date şi algoritmi – Aplicaţii, Editura Printech, Bucureşti.

LABORATORY

Crt. no.	Content	No. hours
1	Pointers. Data structures.	2
2	Linked lists.	2
3	Stack.	2
4	Queue.	2
5	Circulat list.	2
6	Binary trees.	2
7	Applications for algorithms for sorting and searching.	2
8	Recapitulation problems.	2
9	Evaluation of a project/homework.	2
10	Final laboratory exam.	2
	Total:	21

Bibliography:

D.I. Năstac, Course Notes for Computer Programming, UPB, online, Moodle ETTI platform
C++, http://www.cplusplus.com
D.I. Năstac, Structuri de date şi algoritmi – Aplicaţii, Editura Printech, Bucureşti.

11. Evaluation

Activity type	11.1 Evaluation criteria	11.2 Evaluation methods	11.3 Percentage of final grade
11.4 Course	Knowledge with respect to fundamental theoretical notions	Evaluation in 14th week. The subjects cover the complete course.	40%
11.5 Seminary/laboratory/project	Evaluation of laboratory knowhow	Final practical exam (oral and computer programming), with focus on the practical and applied skills.	30%
	Project evaluation (homework)	The focus is on the ability to integrate and apply programming concepts to real problems.	30%
11.6 Passing conditions			
- participation in laboratory work; - cumulation of at least 50% of the score related to the discipline (laboratory and exam).			

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 the ability to solve a medium complexity computational problem, knowledge of classical algorithms for solving computational problems, the ability to transform a natural language computational problem into a programming language, and its solution. The current technological progress of electronic and telecommunications devices is conditioned by the ability to develop and experiment using programming languages. So the discipline of computer programming is fundamental in training 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
25.09.2025	Conf. Dr. Iulian Nastac	Conf. Dr. Iulian Nastac

Date of department approval	Head of department
26.09.2025	Conf. Dr. Serban Georgica Obreja

Date of approval in the Faculty Council	Dean
26.09.2025	Prof. Dr. Mihnea Udrea