**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 | **Applied Electronics and Information Engineering** |
| 1.4 Domain of studies | Computers and Information Technology |
| 1.5 Cycle of studies | Bachelor/Undergraduate |
| 1.6 Programme of studies | Information Engineering |

## 2. Date despre disciplină

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 2.1 Course name (ro)<br>(en) | | | | Curs aplicativ de Linux şi Python<br>Applicative course of Linux and Python<br>(lb.română) | | | |
| 2.2 Course Lecturer | | | | Lect. Phd. Eng. Valentin-Gabriel Voiculescu | | | |
| 2.3 Instructor for practical activities | | | | Lect. Phd. Eng. Valentin-Gabriel Voiculescu | | | |
| 2.4 Year of studies | 3 | 2.5 Semester | II | 2.6. Evaluation type | V | 2.7 Course regime | F |
| 2.8 Course type | | S | 2.9 Course code | 04.S.05.L.032 | | 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 | 1.00 | 3.3 seminary/laboratory | 2 |
| 3.4 Total hours in the curricula | 42.00 | Out of which: 3.5 course | 14 | 3.6 seminary/laboratory | 28 |
| Distribution of time: | | | | | hours |
| Study according to the manual, course support, bibliography and hand notes<br>Supplemental documentation (library, electronic access resources, in the field, etc)<br>Preparation for practical activities, homework, essays, portfolios, etc. | | | | | 28 |
| Tutoring | | | | | 0 |
| Examinations | | | | | 5 |
| Other activities (if any): | | | | | 0 |

| | |
|---|---|
| 3.7 Total hours of individual study | 33.00 |
| 3.8 Total hours per semester | 75 |
| 3.9 Number of ECTS credit points | 3 |

## 4. Prerequisites (if applicable) (where applicable)

| | |
|---|---|
| 4.1 Curriculum | Computer Programming and Programming Languages 1.<br>Computer Programming and Programming Languages 2. |

| | |
|---|---|
| 4.2 Results of learning | Basic requirements of installing programs, using editors to write source code and making computer programs. |

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

| | |
|---|---|
| 5.1 Course | The course will be held in a room equipped, preferably, with a video projector and internet access (to allow simultaneous use in the form of a Teams video conference). Students work on their own computers. |
| 5.2 Seminary/ Laboratory/Project | Attendance of laboratory sessions is mandatory (as stated by the university regulations). Students may work on their own computers during sessions in rooms with internet access. |

**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)*

The discipline has the general objective of familiarizing students with the concept of virtual machines, the Linux operating system and the Python programming language, modern technologies often used in practice.
In class and lab, students will become familiar with the basics and advanced elements of the Python programming language, with the concept of using virtual machines, practiced by running the Linux operating system on the student's computer.
The study of the Python language will combine the teaching of concepts with the completion of exercises by the student, individually, on the personal computer. Students will become familiar with the Linux operating system, its features, and working in the command line. The activities are aimed at developing software abstraction skills, identifying and practicing the skills necessary for computer modeling of some situations from reality.
Based on the knowledge gained from this course, the future electrical engineer will be able to implement or modify programs or command-line scripts specific to modern, scriptable and automatable software development and testing activities, being able to create programs from requirements specification to execution, debugging and interpretation of results.

**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** | **C3.** Application of basic knowledge, concepts and methods regarding the architecture of computer systems, microprocessors, microcontrollers, programming languages and techniques. |
| **Transversal (General) Competences** | **CT1.** Methodical analysis of the problems encountered in the activity, identifying the elements for which there are established solutions, thus ensuring the fulfillment of professional tasks.<br>**CT3.** Adapting to new technologies, professional and personal development, through continuous training using printed documentation sources, specialized software and electronic resources in Romanian and, at least, in one language of international circulation. |

**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.*<br>Is able to answer a series of questions based on a portfolio of activities carried out during the semester, as part of an oral examination.<br>Describes the command (or set of commands) and accompanying arguments required to be used to solve a specific command-line problem. Describes, explains, highlights the consequences of running one (or more) commands in the command line. Describes concepts from the Bash environment needed to solve a given problem under Linux on the command line.<br>Describes the Python program needed to solve a given problem. Highlights the need to use some modules. Describes Python language concepts needed to solve a given problem. |
| **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).*<br>Installing Linux under a virtual machine, starting and interacting with it in the command line. Listing the major variants of Linux distributions and identifying the types of packages used by their included package managers.<br>Designing scripts and programs starting from a requirement, set of commands. Checking and debugging the validity of a given script. Identifying the type of privilege found on the command line, the current location in the file structure, editing and manipulates files, running and chaining commands. Performing searches. Viewing details for active processes and sending signals.<br>Designing, programming, debugging and successfully running a program to solve a given problem. Implementing a program with specified facilities, in the Python language. Checking and debugging the validity of a given program. Working with corresponding basic data types (such as mutable/immutable concept), case sensitiveness, data display and structures (lists, strings, dictionaries, tuples, etc.). Highlighting the utility of the slice operator. Using flow control statements, Boolean expressions, functions, modules, appropriately to solve problems. Identifying situations that benefit from the use of object-oriented programming concepts in Python. |
| **Responsability and autonomy** | *The student's capacity to autonomously and responsably apply their knowledge and skills.*<br>Demonstrating responsiveness to new learning contexts.<br>Conspecting in advance the course materials, laboratory, to the extent that they are made available. In case of absence, going through the material taught, made available, by oneself.<br>Solving homework individually, autonomously, respecting academic ethics.<br>Respecting the principles of academic ethics, individually carrying out the activities marked in this sense, also correctly citing the bibliographic sources used, if the situation requires it.<br>Applying principles of professional ethics/deontology in the analysis of the technological impact of the solutions proposed in the specialized field on the environment. |

**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.)*

The didactic materials used are the course notes and presentations, also available in electronic format.
Starting from the analysis of the students' learning characteristics and their specific needs, the teaching process will explore both expository (lecture, exposition), problem-solving and conversational-interactive teaching methods, based on action-based learning models, such as exercise, practical activities and problem solving. Interactivity with students through the associated applied activities. Intervals are reserved for presentation, analysis and solving of some practical problems (reality modeling).
Lectures will be used in the teaching activity, based on Power Point presentations, which will be presented in front of the students as far as is technically possible, or/and through a videoconferencing environment such as Teams. These will be made available to students. Each course will start with a short recap of the previous lesson to ensure continuity of the concepts covered.
The presentations use, as far as possible, examples of real-life application of the concepts taught, so that the information presented is easy to understand and assimilate.
In the applied section, teaching is based on the use of the expository method (covering the communication and demonstrative function). The dialogue during the course is also extended during the application sessions. These are necessary to prepare students for homework and verification tests along the way.
Feedback will also be used, as a way of adapting the pedagogical approach to the students' learning needs.

**10. Contents**

| **COURSE** | | |
|---|---|---|
| **Chapter** | **Content** | **No. hours** |
| 1 | Linux<br>Installing a Linux virtual machine on the student's computer. Partitioning. Formatting. Checking the installation. Using Linux virtual machines in the browser.<br>Introductory information. The role of the operating system. History. Distributions. Package Managers. Usage scenarios.<br>Working in command line. Shell. Prompt. Commands and arguments. Facilities. Chaining command. Command line editors.<br>The Linux file system. Commands for interacting with it.<br>Processes – view, interact with Linux processes. Signals. | 8 |
| 2 | Python<br>Introductory information. Installing Python on Windows and/or Linux. Using a Python interpreter. Running Python code files.<br>Fundamental data types. Mutable/immutable. Naming variables. Case sensitive. Displaying data.<br>Data structures. Lists. Tuples. Dictionaries. The slice operator.<br>Execution flow control instructions. Boolean expressions. Functions. Modules.<br>Concepts of Object Oriented Programming in Python. Classes. Attributes. Methods. Inheritance. | 6 |
| | **Total:** | 14 |

**Bibliography:**

*V.G.Voiculescu, Curs aplicativ de Linux şi Python, suport de curs electronic, https://archive.curs.upb.ro/2024/course/view.php?id=8648*
*D. Beazley, BK Jones, Python cookbook: Recipes for mastering Python, 3rd edition, O'Reilly, 2013*
*J. Cannon, Linux for Beginners, ACM, 2014*
*W3schools, Python 3, https://www.w3schools.com/python/, 2025*
*W.R. Stevens, S.A. Rago, Advanced programming in the UNIX environment, 3rd edition, 2013*

| **LABORATORY** | | |
|---|---|---|
| **Crt. no.** | **Content** | **No. hours** |
| 1 | Linux.<br>Installation validation.<br>Elements of a shell prompt. Simple commands. Implementing a hierarchical directory structure in the command line. Exercises.<br>Rights and permissions. Modifying them. Exercises.<br>The Linux file system. Interesting portions of the file system hierarchy. Interacting with the file system. | 3 |
| 2 | Linux<br>Processes – visualization and interaction. Process parallelism. Signals. Pipe operators, redirection. Practicing commands on realistic scenarios.<br>Command line editors. Vim. Hello world bash script. Vimtutor.<br>Search. Grep.<br>Scripting using the Linux shell. Verifying the successful execution of a command.<br>Functions. Loops. Regular expressions. | 3 |
| 3 | Python.<br>Data types.<br>Data structures.<br>Execution flow control instructions. | 3 |
| 4 | Python.<br>Functions. Modules.<br>Object oriented programming. Other exercises. | 3 |
| 5 | Linux and Python practical usage scenarios. | 2 |
| | **Total:** | 14 |
| **PROJECT** | | |
| **Crt. no.** | **Content** | **No. hours** |
| 1 | Linux. Fundamental notions.<br>The Linux file system. Commands for interacting with it.<br>Processes. Signals.<br>Preparing the final project. | 6 |
| 2 | Python. Fundamental notions. Object oriented programming notions in Python. Native and browser-based visual interfaces.<br>Preparing the final project. | 6 |
| 3 | Verification | 2 |
| | **Total:** | 14 |

**Bibliography:**
*V.G.Voiculescu, Curs aplicativ de Linux şi Python, electronic course materials, https://archive.curs.upb.ro/2024/course/view.php?id=8648*
*V.G.Voiculescu, Curs aplicativ de Linux şi Python, electronic lab and project materials, https://archive.curs.upb.ro/2024/course/view.php?id=8648*
*D. Beazley, BK Jones, Python cookbook: Recipes for mastering Python, 3rd edition, O'Reilly, 2013*
*J. Cannon, Linux for Beginners, ACM, 2014*
*W3schools, Python 3, https://www.w3schools.com/python/, 2025*
*W.R. Stevens, S.A. Rago, Advanced programming in the UNIX environment, 3rd edition, 2013*

**11. Evaluation**

| Activity type | 11.1 Evaluation criteria | 11.2 Evaluation methods | 11.3 Percentage of final grade |
|---|---|---|---|
| 11.4 Course | Correct identification of the theoretical and practical application contexts of the studied concepts: Python, Linux, as well as the features studied for them. | Oral evaluation based on portofolio/project. Final verification. | 20% |
| 11.5 Seminary/laboratory/project | Ability to model and successfully solve a problem programmatically in Python.<br>Knowledge of the Linux operating system, with its features and working in the command line.<br>Solving various Linux command line issues. | The practical activity is constantly checked throughout the semester. Homework. Verification. | 80% |
| 11.6 Passing conditions | | | |
| Attendance of laboratory sessions is mandatory (as stated by the university regulations). Attending the final verification is mandatory.<br>Installing. Starting up. Interacting with Linux through commands. Creating a script starting from a set of commands. Implementation of a program with specified facilities in Python language, modeling, programming, running successfully in order to get the solution.<br>Obtaining 50% of the total score or the minimum score provided by the regulation. | | | |

**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)**

In our faculty, there is less emphasis on using the Linux operating system itself, only to the small extent that it is used as an operating system in some labs. Linux has become fundamental in many areas: in telecommunications (where we are in the process of transitioning from expensive proprietary equipment to commodity hardware running specific stacks and protocols on top of a generic Linux operating system), similarly in broad and diverse areas in the embedded domain, IoT, including smartphones (Android runs over Linux), in automotive (In vehicle infotainment area), even in outer space (SpaceX/StarLink satellites).
Python is in turn a very popular language, being the basis of some intensively used social media

applications, also having applicability in future fields such as artificial intelligence. And for this language, we are in a process of adoption at the faculty level, but each subject describes it only minimally, in order to introduce the specific study concepts. It is a popular language and in some scripting, testing, building frameworks found in the target fields of our graduates, and with the help of this subject the future electrical engineer will be able to implement or modify programs or command line scripts specific to development and testing activities modern, scriptable and automatable software.
Based on the knowledge gained from this course, the future electronics engineer will be able to implement or modify programs or command-line scripts specific to modern, scriptable and automatable software development and testing activities, being able to create programs from requirements specification to execution, debugging and interpretation of results.

| Date | Course lecturer | Instructor(s) for practical activities |
|---|---|---|
| 22.09.2025 | Lect. Phd. Eng. Valentin-Gabriel Voiculescu | Lect. Phd. Eng. Valentin-Gabriel Voiculescu |

| | |
|---|---|
| Date of department approval | Head of department<br><br>Assoc.Prof.Phd.Eng. Bogdan Cristian Florea |
| Date of approval in the Faculty Council | Dean<br><br>Prof. Phd. Eng. Radu Mihnea Udrea |