**SYLLABUS**

**regarding the qualification cycle FROM 2023TO 2024**

1. Basic Course/Module Information

|  |  |
| --- | --- |
| Course/Module title | Operating Systems 2 |
| Course/Module code \* |  |
| Faculty (name of the unit offering the field of study) | College of Natural Sciences |
| Name of the unit running the course | Institute of Computer Science |
| Field of study | Computer Science & Computer Science and Econometrics |
| Qualification level | First degree |
| Profile | Academic |
| Study mode | Full-time |
| Year and semester of studies | Year 3, semester 5 |
| Course type | Major engineering |
| Language of instruction | English |
| Coordinator | Krzysztof Balicki, PhD |
| Course instructor | Krzysztof Balicki, PhD |

\* - as agreed at the faculty

1.1. Learning format – number of hours and ECTS credits

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Semester  (n0.) | Lectures | Classes | Colloquia | Lab classes | Seminars | Practical classes | Internships | others | **ECTS credits** |
| 5 |  |  |  | 30 |  |  |  |  | 5 |

1.2. Course delivery methods

- conducted in a traditional way

- involving distance education methods and techniques

1.3. Course/Module assessment (exam, pass with a grade, pass without a grade)

exam

2. Prerequisites

|  |
| --- |
| Operating systems 1, The basics of programming in C language, Computer architecture |

3. Objectives, Learning Outcomes, Course Content, and Instructional Methods

3.1. Course/Module objectives

|  |  |
| --- | --- |
| O1 | Identify the role and operation of Unix/Linux operating systems. Learn basic operating system tasks for process scheduling, memory management, and I/O operations. Become familiar with the design of file systems. Understand security and security issues in operating systems. |
| O2 | Learn Unix/Linux shell commands and scripting. |
| O3 | Develop basic programming skills in the Unix/Linux environment. |

3.2. Course/Module Learning Outcomes (to be completed by the coordinator)

|  |  |  |
| --- | --- | --- |
| Learning Outcome | The description of the learning outcome  defined for the course/module | Relation to the degree programme outcomes |
| LO\_01 | The student knows the Windows and Unix/Linux operating system environments. He knows the structure and commands of one or more operating systems and can create shell scripts. He knows the principles of operating systems with particular emphasis on concurrency, security, memory management, task scheduling, synchronization, and avoiding conflicts among processes. | K\_W03  K\_W04  K\_W06 |
| LO\_02 | The student can program in C language in the Linux environment and use the POSIX library on a basic level. | K\_U12 |
| LO\_03 | The student can use system commands of at least one operating system and write scripts in it, as well as make its critical analysis in the context of practical applications and security. | K\_U13 |

**3.3. Course content (to be completed by the coordinator)**

1. Lectures

|  |
| --- |
| Content outline |
| Processes and threads. |
| Process scheduling. |
| Main memory. |
| Virtual memory. |
| Concurrency and synchronization. |
| File system interface. |
| File system implementation. |
| Input/output devices. |
| Protection and security in operating systems. |
| Linux system. |

1. Classes, tutorials/seminars, colloquia, laboratories, practical classes

|  |
| --- |
| Content outline |
| The concept of a system shell. Streams and redirection of streams. The shell as a programming language. Shell commands syntax. |
| Script design using text-based graphical elements. |
| The file structure in Linux. System calls and device drivers. Libraries of functions that support file operations in C language. |
| The standard library of I/O operations. Formatted input and output. Administrative support for files and directories. |
| Linux environment. Environment Variables. Passing arguments to programs. |
| Linux environment. Time and date. Temporary files. User and workstation information. |
| Terminals. Reading from terminals and writing to terminals. Communication with a terminal. The termios structure. Capturing characters from the keyboard. |
| Data management. Management of memory resources. Memory allocation. NULL pointer. Freeing up memory areas. Other memory allocation functions. |
| Development tools. Make command and compiler management files. Debugging process. |
| Processes and signals. Process structure. Process table. Viewing processes. System processes and methods of their scheduling. |
| A new process creation mechanism. Waiting for process. Signals. Sending and receiving signals. |
| POSIX threads. The concept of a thread. Concurrent execution of programs. Synchronization. Threads' termination. |
| Process communication mechanisms. Semaphores. Shared memory and message queues. |

3.4. Methods of Instruction

Lecture: a lecture supported by a multimedia presentation

Laboratory: solving exercises, implementation projects

4. Assessment techniques and criteria

4.1 Methods of evaluating learning outcomes

|  |  |  |
| --- | --- | --- |
| Learning outcome | Methods of assessment of learning outcomes (e.g. test, oral exam, written exam, project, report, observation during classes) | Learning format (lectures, classes,…) |
| LO-01 | exam | lecture |
| LO-o2 | test, project | lab |
| LO-o3 | project, observation during classes | lab |

4.2 Course assessment criteria

|  |
| --- |
| Lab  To complete the lab, you must pass the programming test and prepare a project. The final grade is the average of marks from the test and a project. You must earn at least half of the maximum number of points to complete the test. Test grades are proportional to the number of points scored. Activity in the labs is also considered, as it may lower or increase the final score by half a degree.  Lectures  To pass the exam, you must earn at least half of the maximum number of points. Exam grades are proportional to the number of points scored. |

5. Total student workload needed to achieve the intended learning outcomes

– number of hours and ECTS credits

|  |  |
| --- | --- |
| Activity | Number of hours |
| Scheduled course contact hours | 30 |
| Other contact hours involving the teacher (consultation hours, examinations) | 3 |
| Non-contact hours - student's own work (preparation for classes or examinations, projects, etc.) | 92 |
| Total number of hours | 125 |
| Total number of ECTS credits | 5 |

\* One ECTS point corresponds to 25-30 hours of total student workload

6. Internships related to the course/module

|  |  |
| --- | --- |
| Number of hours | - |
| Internship regulations and procedures | - |

7. Instructional materials

|  |
| --- |
| Compulsory literature:   1. Abraham Silberschatz, Peter B. Galvin, Greg Gagne, Podstawy Systemów Operacyjnych, wyd. 10, PWN 2021 2. Abraham Silberschatz, Podstawy Systemów Operacyjnych, wyd. 7, WNT 2006. 3. Robert Love, Linux. Programowanie systemowe, Wyd. II, Helion, 2014. 4. M. Ben-Ari, Podstawy programowania współbieżnego i rozproszonego, WNT 1996. |
| Complementary literature:   1. Andrew S. Tanenbaum, Herbert Bos, Systemy operacyjne. Wyd IV, Helion, 2016. 2. Christopher Negus, Linux. Biblia. Ubuntu, Fedora, Debian i 15 innych dystrybucji, Helion, 2011. |

Approved by the Head of the Department or an authorised person