| | | | | |
|---|---|---|---|---|
| **Obuda University**<br>Donát Bánki Faculty of Mechanical and Safety Engineering | | | Institute of Mechatronics and Vehicle Engineering | |
| **Name and Neptun-code:** Programming II. BMXPNY4BNE **Credits*:*** 5<br><br>*Full time, Spring Semester of the Academic year 2021/22.* | | | | |
| | | | | |
| Subject lecturer: Peter Juma Ochieng | | | | |
| Prerequisites (with code): | | | | |
| Weekly hours: | Lecture: **2** | Seminar.:0 | Lab. hours: **2** | Consultation:0 |
| Way of assessment: | Exam and Practical | | | |
| **Syllabus:** | | | | |
| **Aim**: Developing algorithmic thinking, introducing the basic tools of programming, which are needed during engineering work. The acquisition of basic algorithms and data structures. Show basic computer programming techniques and approaches. Students learn about the basic algorithms and data structures using an easy to learn programming language. This subject helps to solve complex engineering problems. | | | | |
| **Course description:** The course provides an in-depth introduction to the Python programming language for those who basic knowledge in programming. The course starts with a brief overview of the structure, syntax and building blocks of the Python environment, including data types, data structures and modern tools. This is followed by introducing a few popular libraries, including numpy and pandas. In the second half of the semester, students will learn about object-oriented programming in Python, parallelization and test-driven development. | | | | |

| **Lecture schedule** | |
|---|---|
| *Education week* | *Topic* |
| 1. | Introduction, overview, development environments, basic syntax |
| 2. | List, tuple, dictionary, set. Functions, lambda functions, list comprehension. |
| 3. | IPython. Jupyter Notebooks, Jupyter Lab. Virtual environments. |
| 4. | Introduction to NumPy, ndarray. |
| 5. | Introduction to pandas, dataframe, series. |
| 6. | Files: read/write. Pickle. Exception handling. |
| 7. | OOP in Python: classes, inheritance, polymorphism. |
| 8. | Custom modules. |
| 9. | Parallelism in Python. Synchronization. |
| 10. | Basics of testing. Unit testing in Python. |
| 11. | Practice, use-cases. |
| 12. | Midterm |
| 13. | Midterm re-take |

**Midterm requirements**

| *Education week* | *Topic* |
|---|---|
| 5 | Theory Test 1 |
| 6 | Practical Test 1 |
| 13 | Theory and Practical 2 |
| 14 | Retake Test |

2022. 01. 28.

**Final grade calculation methods**

| Achieved result | Grade |
|---|---|
| 89%-100% | excellent (5) |
| 76%-88<% | good (4) |
| 63%-75<% | average (3) |
| 51%-62<% | satisfactory (2) |
| 0%-50<% | failed (1) |

**Type of exam**

Practical test to solve a given task using Python.

**Type of replacement**

Retake of the midterm on the last week.

**References**

Mandatory:

Recommended: Slatkin, Brett. Effective python: 90 specific ways to write better python. Addison-Wesley Professional, 2019.
Deitel, Harvey, Paul Deitel, and Paul J. Deitel. Python for Programmers. Prentice Hall, 2019.
Danjou, Julien. Serious Python. No Starch Press, 2018.