#### card of course

|  |  |
| --- | --- |
| Subject name | Object-oriented programming paradigm |

1. The placement of the subject in the study system

|  |  |
| --- | --- |
| 1.1. Field of study | Computer science |
| 1.2. Form and path of study | Full-time/Part-time |
| 1.3. Level of education | First-cycle studies |
| 1.4. Study profile | Practical |

|  |  |
| --- | --- |
| 1. 5. Specialty | - |
| 1.6. Subject Coordinator | Dr inż. Sylwester Korga |

2. General characteristics of the subject

|  |  |
| --- | --- |
| 2.1. Belonging to a subject group | Directional/Practical |
| 2.2. Number of ECTS | 4 |
| 2.3. Language of lectures | Polish |
| 2.4. Semesters in which the subject is taught | II |
| 2.5.Criteria for selecting course participants | - |

1. Learning outcomes and course delivery
   1. Subject Objectives

|  |  |
| --- | --- |
| No. | Subject Objectives |
|
| C1 | To introduce students to the basic principles of object-oriented programming (OOP) and their implementation in Python. |
| C2 | Developing skills in designing and implementing applications consistent with the object-oriented paradigm. |
| C3 | Prepare to write scalable and readable OOP-based code in Python. |

* 1. Subject-specific learning outcomes, divided into knowledge , skills and competences , with reference to the directional learning outcomes

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| No. | Description of subject  learning outcomes | Reference to  directional effects  learning (symbols) | Method of implementation (mark "X") | | | |
| ST | | NST/ZD | |
| Classes at the University | Activities on  the platform | Classes at the University | Activities on  the platform |
| After passing the course, the student knows and understands **the knowledge** | | | | | | |
| W1 | Knows the basic principles of object-oriented programming such as encapsulation, inheritance, and polymorphism. | INF\_W07  INF\_W08  INF\_W19 | X |  |  | X |
| W2 | Understands the concepts of classes and objects, including methods, fields, and constructors. | X |  |  | X |
| W3 | Knows advanced OOP concepts such as static methods, abstract base classes, and interfaces. | X |  |  | X |
| W4 | Understands the role of exceptions in OOP and their handling in Python. | X |  |  | X |
| W5 | Knows design patterns such as Singleton, Factory and Observer and their application in OOP. | X |  |  | X |
| After passing the course, the student is **able** to: | | | | | | |
| U1 | Can design and implement classes and create objects in Python. | INF\_U15 INF\_U19  INF\_U23 | X |  | X |  |
| U2 | Is able to apply the principles of inheritance and polymorphism in application design. | X |  | X |  |
| U3 | Can handle exceptions in Python, integrating them with OOP principles. | X |  | X |  |
| U4 | Is able to use design patterns to solve typical programming problems. | X |  | X |  |
| U5 | Can debug and optimize code based on OOP principles. | X |  | X |  |
| After completing the course, the student is ready to take part in **social competences.** | | | | | | |
| K1 | The student is able to plan work in a team and assign tasks to the team. The student is able to use the GIT version control system. | INF\_K04 INF\_K06 | X |  | X |  |
| K2 | The student is able to work in a team and understands what concurrent work control is. | X |  | X |  |

3.3. Forms of teaching and their number of hours - Full-time studies (ST), Part-time studies (NST)

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Path | Lecture | Exercises | Design | Workshop | Laboratory | Seminar | Lecturer | Classes conducted using distance learning methods and techniques in the form of a lecture | Other | **ECTS points** |
| **ST** | 15 |  |  |  | 30 |  |  |  |  | 4 |
| **NST** |  |  |  |  | 15 |  |  | 10 |  | 4 |

3.4 . Content of education (separately for each form of classes: (W, ĆW, PROJ, WAR, LAB, LEK, OTHER). It should be marked (X) how the given content will be implemented (classes at the university or classes on the e-learning platform conducted using distance learning methods and techniques)

TYPE OF CLASS: LECTURE

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| No. | Content of the course | Reference to subject-specific learning outcomes | Method of implementation (mark "X") | | | |
| ST | | NST/ZD | |
| **Classes at the University** | **Activities on  the platform** | **Classes at the University** | **Activities on  the platform** |
| 1. | Coverage of the basics of object-oriented programming, classes and objects, methods and fields in Python. | W1, W2 | X |  |  | X |
| 2. | Explanation of the principles of encapsulation, inheritance, and polymorphism, analyzing examples of implementations in Python. | W1, W2 | X |  |  | X |
| 3. | Presentation of advanced OOP techniques in Python, use of static methods, decorators and abstract classes (abc). | W3 | X |  |  | X |
| 4. | An introduction to exception handling in Python, covering creating your own exception classes and integrating them into the class hierarchy. | W4 | X |  |  | X |
| 5. | Discussing design patterns such as Singleton, Factory, Observer, with Python implementation examples, | W5 | X |  |  | X |
| 6. | An overview of debugging tools and optimization methods for object-oriented applications in Python. | W5 | X |  |  | X |
| 7. | A summary of key object-oriented programming concepts, and a reflection on their application in Python. | W1-W5 | X |  |  | X |

TYPE OF CLASS: LABORATORY

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| No. | Content of the course | Reference to subject-specific learning outcomes | Method of implementation (mark "X") | | | |
| ST | | NST/ZD | |
| **Classes at the University** | **Activities on  the platform** | **Classes at the University** | **Activities on  the platform** |
| 1. | Basics of object-oriented programming, classes and objects, methods and fields in Python. | U1 | X |  |  | X |
| 2. | Encapsulation, inheritance and polymorphism in Python, implementing inheritance and overriding methods. | U2 | X |  |  | X |
| 3. | Advanced OOP techniques in Python, static methods and decorators, abstract classes with abc. | U2 | X |  |  | X |
| 4. | Handling exceptions in Python, creating your own exception classes, integrating exceptions with the class hierarchy . | At 3 | X |  |  | X |
| 5. | Design patterns in Python, Singleton, Factory and Observer implementation. | U4 | X |  |  | X |
| 6. | Debugging and optimizing object-oriented applications, Python debugging tools, analysis of common problems. | U5 | X |  |  | X |
| 7. | Team work on the project, design and implementation of an object-oriented application. | K1, K2 | X |  |  | X |

3.5. Methods of verifying learning outcomes (indication and description of methods of conducting classes and verification of achievement of learning outcomes and method of documentation)

|  |  |  |  |
| --- | --- | --- | --- |
| Subject Effects | Teaching methods | Methods of verifying learning outcomes | Documentation methods |
| KNOWLEDGE | | | |
| W1-W5 | Multimedia presentation | Theoretical exam on the content of lectures on the platform. | Exam archived on the platform |
| SKILLS | | | |
| U1-U4 | Practical classes performed at computer stations | Team Project:  Students work in teams to design and implement an object-oriented application in Python. The project should include key OOP principles such as inheritance, polymorphism, and design patterns. | Files archived on the platform |
| SOCIAL COMPETENCES | | | |
| K1-K2 | Practical classes performed at computer stations | Team Project:  Students work in teams to design and implement an object-oriented application in Python. The project should include key OOP principles such as inheritance, polymorphism, and design patterns. | Files archived on the platform |

3.6. Assessment criteria for the achieved learning outcomes

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Learning effect | For a grade of 3 or "pass."  the student knows and understands/is able to/is ready to | For a grade of 3.5, the student knows and understands/is able to/is ready to | For a grade of 4, the student knows and understands/is able to/is ready to | For a grade of 4.5, the student knows and understands/is able to/is ready to | For a grade of 5, the student knows and understands/is able to/is ready to |
| W | 51-60% of knowledge indicated in learning outcomes | 61-70% of knowledge indicated in learning outcomes | 71-80% of knowledge indicated in learning outcomes | 81-90% of knowledge indicated in learning outcomes | 91-100% of knowledge indicated in learning outcomes |
| U | 51-60% of skills indicated in learning outcomes | 61-70% of skills indicated in learning outcomes | 71-80% of skills indicated in learning outcomes | 81-90% of skills indicated in learning outcomes | 91-100% of skills indicated in learning outcomes |
| K | 51-60% of skills indicated in learning outcomes | 61-70% of skills indicated in learning outcomes | 71-80% of skills indicated in learning outcomes | 81-90% of skills indicated in learning outcomes | 91-100% of skills indicated in learning outcomes |

3.7. Literature

**Basic**

1. Sebesta Robert W, Concepts of programming languages , Pearson Education, Harlow, 2016
2. Baka Benjamin, Python Data Structures and Algorithms. Improve application performance with graphs, stacks, and queues Packt Publishing, Birmingham, 2017.
3. Zaawansowany Python, Luciano Ramalho, 2022;
4. Python wprowadzenie , Mark Lutz, rok wydania 2021;

**Supplementary:**

1. Pierce Benjamin C., Types and programming languages , MIT Press, London, 2002
2. Programowanie dla każdego, Michael Dawson, rok wydania 2020;

4. Student workload - ECTS points balance

|  |  |  |
| --- | --- | --- |
| **Types of student activity** | **Student Load** | |
| **ST** | **NST** |
| **Classes requiring direct contact between the student and the academic teacher at the university premises** | **45** | **25** |
| Classes included in the study plan | 45 | 25 |
| **Student's own work** | **55** | **75** |
| Ongoing preparation for classes, preparation of project work/presentations/etc. | 30 | 40 |
| Preparation for passing classes | 25 | 35 |
| **TOTAL STUDENT HOURLY LOAD** | **100** | **100** |
| **Number of ECTS points** | **4** | **4** |

|  |  |
| --- | --- |
| Last change date | 30/09/2024 |
| The changes were introduced | INF Education Quality Team |
| The changes were approved | Arkadiusz Gwarda, M.A. |