COURSE DESCRIPTION CARD - SYLLABUS

Course name
Object-oriented programming [S1Mech2>PO]

Course

Field of study
Mechatronics

Year/Semester
2/3

Area of study (specialization)
–

Profile of study
general academic

Level of study
first-cycle

Course offered in
Polish

Form of study
full-time

Requirements
compulsory

Number of hours

Lecture
15

Laboratory classes
30

Other
0

Tutorials
0

Projects/seminars
0

Number of credit points

3,00

Coordinators

dr hab. inż. Wojciech Pietrowski prof. PP
wojciech.pietrowski@put.poznan.pl

Lecturers

Prerequisites

Basic knowledge in the field of computer science, operating systems, algorithms and data structures, and programming languages. Proficiency in operating a computer, working with the Windows operating system, and structural programming in C++.

Course objective

Understanding the theoretical and practical aspects of object-oriented programming in C#.

Course-related learning outcomes

Knowledge:
The student possesses fundamental knowledge of object-oriented programming concepts. The student has a basic understanding of object-oriented programming languages, specifically C#. The student is familiar with fundamental object-oriented programming techniques in selected development environments. The student understands program structures, module structures, and selected visual components.

Skills:

The student is capable of acquiring information from technical literature and the Internet regarding object-oriented programming. The student can define and utilize class types, function templates, as well as inheritance and polymorphism techniques. The student is able to develop computer programs in integrated development environments using object-oriented programming techniques. The student can test the developed programs and assess their correctness and functionality.

Social competences:
Understands the importance of enhancing professional, personal, and social competencies; is aware that knowledge and skills in the field of mechatronics are rapidly evolving. Recognizes the significance of knowledge in solving mechatronics-related problems and acknowledges the necessity of consulting experts when addressing engineering tasks beyond their own competencies.

Methods for verifying learning outcomes and assessment criteria

Learning outcomes presented above are verified as follows:

Lecture: The final assessment is based on a written exam consisting of general and test-based questions. Grading scale: 51-60% - (dst), 61-70% - (dst+), 71-80% - (db), 81-90% - (db+), 91-100% - (bdb).
Laboratory: Practical knowledge acquired during previous laboratory exercises is rewarded. The assessment includes verifying practical programming skills in C# (final laboratory exam) and evaluating knowledge and skills related to the completion of individual and group programming projects. Additional points can be earned for active participation in classes, particularly for: the ability to collaborate within a team carrying out a specific task in the laboratory, the use of elements and techniques beyond the scope of the lecture and laboratory exercises, and the aesthetic precision of completed projects.

Programme content

C# language. Object-oriented programming in the Microsoft Visual Studio environment. Source code editors. Debugger. Object-oriented programming. Classes, objects, inheritance, data encapsulation, polymorphism, virtual methods, abstract classes, function templates, and class templates. Creating and processing objects, objects as function arguments. Components, forms, properties, events, exception handling. Application of object-oriented programming for simulating selected operating states of automation actuators.

Course topics

Lecture:
I. Object oriented programming: Application of object-oriented languages, Object-oriented modelling rules, Analysis and design.
II. Phases of software development: Learning and analysis of the problem, Creation of the object model, Design of the program interface and functions, Implementation, Testing.
III. Programming on the dotNET platform: Code structure, Simple types, Type system, Object model, Simple types vs. reference types, boxing and unboxing.
IV. Programming on the dotNET platform: Testing and debugging in C#, Asynchronous programming, Multithreaded programming.
V. Object oriented programming: Classes, Fields, Static fields, Methods, Static methods, Methods - overriding.
VI. Object-oriented programming: Methods - Constructors, Properies/properties, Constants, Indexers, Overriding operators.
VII. Object-oriented programming: Inheritance, Polymorphism, Object destruction - destructor, Object destruction - IDisposable interface, Interfaces, Exceptions, Delegates, Modules.
VIII. Test.
Laboratories:
I. Introduction to C#: Overview of the C# language, Setting up the development environment, First program in C#.
II. Classes, Objects, and Constructors: Definition of classes and objects, Creating and using constructors.
III. Encapsulation and Properties: Access modifiers, Getters and setters, Auto-implemented properties.
IV. Inheritance: Basic principles of inheritance, Base and derived classes, Method overriding.
V. Polymorphism: Method overloading, Method overriding, Abstract classes and interfaces.
VI. Delegates: Definition and use of delegates, Events and event handling.

VII. File Operations - XML: Reading and writing XML files, XML serialization and deserialization.
VIII. Collections: Lists, dictionaries, queues, Introduction to LINQ for data processing.
IX. Exception Handling: Exception mechanism in C#, Standard exception classes, Creating custom exception classes.
X. Asynchronous Programming: Concurrent programming in C#, async and await - managing tasks, Exception handling in asynchronous code.
XI. Multithreading Programming: Threads in C# (Thread, Task), Thread synchronization (lock, Mutex, Semaphore), Inter-thread communication.
XII. Advanced LINQ and Database Operations: Advanced LINQ queries, Introduction to Entity Framework, Basics of database communication.
XIII. Design Patterns in C#: Singleton, Factory, Repository pattern, Dependency Injection in .NET, Best practices in application design.
XIV. Testing and Debugging in C#: Debugging tools in Visual Studio, Unit testing (xUnit, NUnit, MSTest), Creating and running tests.
XV. Exam.

Teaching methods

Lecture: Multimedia presentation and live demonstration of writing and executing selected programs directly in C#.
Laboratory Exercises: Practical exercises on C# language elements, including writing windowed applications.

Bibliography

Basic:
1. Albahari J. C# 9.0 w pigułce, Helion, 2022.
2. Troelsen A., Japikse P. Język C# 6.0 i platforma .NET 4.6, Helion, 2017.
3. Martin R. C. Czysty kod. Podręcznik dobrego programisty, Helion, 2010.

Additional:
1. Gamma E., Helm R., Johnson R., Vlissides J. Wzorce projektowe. Elementy oprogramowania obiektowego wielokrotnego użytku, Helion, 2005.
2. Osherove R. Sztuka Unit Testowania. Przykłady w języku C#, Helion, 2014.

Breakdown of average student's workload

	Hours	ECTS
Total workload	75	3,00
Classes requiring direct contact with the teacher	45	2,00
Student's own work (literature studies, preparation for laboratory classes/ tutorials, preparation for tests/exam, project preparation)	30	1,00