COURSE DESCRIPTION CARD - SYLLABUS

Course name
Visual programming [S1Inf1>PWIZ]

Course

Field of study
Computing

Year/Semester
4/7

Area of study (specialization)
–

Profile of study
general academic

Level of study
first-cycle

Course offered in
polish

Form of study
full-time

Requirements
elective

Number of hours

Lecture
30

Laboratory classes
30

Other (e.g. online)
0

Tutorials
0

Projects/seminars
0

Number of credit points

4,00

Coordinators

dr inż. Paweł Wojciechowski
pawel.wojciechowski@put.poznan.pl

Lecturers

Prerequisites

The student starting this course should have basic knowledge of programming methodology, know the methodology of object-oriented programming and previously discussed object-oriented programming languages, know the basic design patterns and the architecture of modular applications. He should also know the basic concepts of algorithmics, computational complexity, concurrent systems programming and cooperation with databases. He should have the ability to solve basic algorithmic problems, use advanced programming systems, build high-quality code and the ability to obtain information from indicated sources.

Course objective

1. Provide students with basic knowledge of the advanced visual programming environment .NET and .NET Core, in the field of: detailed knowledge of the C # object-oriented programming language, a set of classes and functions provided by the Windows Presentation Foundation libraries, ADO.NET technology. 2. Developing students" skills in solving basic algorithmic problems, designing and implementing window applications. The aim of the course is also to deepen the skills of applying design patterns, creating dynamic DLL libraries, testing developed applications, creating concurrent and distributed applications as well as applications cooperating with databases. 3. Improving students" skills to work with code, the quality of which allows for reuse of part of the code in other programming projects.

Course-related learning outcomes

Knowledge:
1. knows the architecture of the .NET programming environment and has knowledge of the basic technologies provided by this environment and the methods of creating console and window applications as well as preparing dynamic DLLs.
2. knows in detail the object-oriented programming language C # and its advanced structures (including lambda expressions, extension functions, LINQ expressions)
3. knows the basic design patterns used when creating an application, with particular emphasis on the Model-View-ViewModel pattern
4. knows the libraries of classes and functions of the .NET programming environment for creating Windows Presentation Foundation window applications
5. knows the techniques of object-oriented programming, the use of selected design patterns, building a flexible application architecture and user interface. He knows Visual Studio.

Skills:
1.can choose the method of securing the user interface of a window application using the Windows Presentation Foundation libraries against entering erroneous data, critically assess the advantages and disadvantages of the solution used
2. is able to analyze the functioning of the information system with the user interface, assess the application architecture in terms of the ease of making changes to it. Can assess the software architecture from the point of the possibility of changing the data source (multi-layer application), the independence of the application presentation layer from its functional logic (MVVM model)
3. can solving tasks, use properly selected methods of data access, design the application architecture
4.can - in accordance with the given specification - design and implement a simple database based on the Producer - Product relationship using the appropriate methods, techniques and tools
5. The student is able to implement an algorithm for filtering and sorting data in an application with a user interface
6. can find a solution to selected programming problems within .NET and WPF technologies

Social competences:
1. The student understands that the .NET platform is constantly evolving and new elements of the C# language are introduced in subsequent versions
2. is aware of the importance of knowledge and advanced techniques provided by the C# language and WPF libraries in solving engineering problems

Methods for verifying learning outcomes and assessment criteria

Learning outcomes presented above are verified as follows:

Learning outcomes presented above are verified as follows:
Verification of the assumed learning outcomes is carried out by:
- assessment of knowledge and skills related to the implementation of subsequent project / laboratory tasks,
- evaluation of the implementation of the subsequent stages of the final project. The final project consists of four obligatory and one optional stages. Each of them is checked and discussed.
- assessment of the knowledge acquired during the lecture in the form of either short, 10-minute tests carried out during the lecture and covering 1-2 lectures backwards using the moodle platform or in the form of a colloquium after the lectures.
Each of the tests carried out during the lectures is assessed in the same way, with the maximum number of points being the best result. To obtain a credit in this form, the average of the tests should be obtained, min. 50%, with the two lowest test results not taken into account. A student who has not obtained the required points is required to submit a final test after completing the lectures.

Programme content

The lecture covers the following issues:
- Introduction to .NET and C#
Basic elements of C# language: data types, built-in variable types. functions and properties available for data types, strings, arrays, structures and classes. Differences between .NET Frameworks and .NET Core
- Advanced language elements: indexers, type conversion, parametric types (generics), interfaces, iterators, anonymous types and methods, delegations, lambda expressions, events, exceptions, LINQ

queries, concurrency,
- ADO.NET architecture
- WPF library: application architecture, XAML, layouts, events, Dependency Objects, data binding, styles, component development, resources, animations, graphical elements, command system, MVVM model, form data validation, data views.
Lab.
In the initial part of the laboratory classes, students implement simple applications that are an example of the mechanisms discussed in lectures. These programs are implemented by all students as console applications to show advanced C# language structures. The next stage is the implementation of window applications that use the WPF library, taking into account the advanced mechanisms offered by this library (including binding mechanisms and property change notification). At the same time, based on the acquired knowledge, students develop individually or in small teams a more extensive, layered application with a user interface (WPF), implemented in accordance with the MVVM model. This application is carried out in five stages (4 mandatory and one optional), each of which is assessed by the teacher. Additionally, the evaluation is influenced by the timeliness of the implementation of individual stages.

Teaching methods

Lectures: multimedia presentations, presenting code fragments and their results
Laboratories: tasks to be performed, live coding, multimedia presentations

Bibliography

Basic
1. Troelsen, Andrew W., Japikse, Philip F, Język C# 6.0 i platforma .NET 4.6, PWN, 2017.
2. Michaelis, Mark., Lippert, Eric., C# 6.0 : Kompletny przewodnik dla praktyków, Helion, 2016.
3. Raffaele Garofalo, Budowanie aplikacji biznesowych za pomocą Windows Presentation Foundation i wzorca Model View ViewM, PROMISE, 2011
4. Dokumentacja elektroniczna systemu programowania wizualnego Visual Studio.NET
5. Dokumentacja języka C# i bibliotek WPF i WCF
Additional
1. Matthew MacDonald, Pro WPF 4.5 in C#: Windows Presentation Foundation in .NET 4.5, Apress, 2012.
2. Magennis, Troy, LINQ to Objects w C# 4.0 : wygodne operacje na danych!, Helion, 2012.
3. Metsker, Steven John, C# - wzorce projektowe, Helion, 2005.
4. WCF - Getting started tutorial https://docs.microsoft.com/en-us/dotnet/framework/wcf/getting-started-tutorial

Breakdown of average student's workload

	Hours	ECTS
Total workload	100	4,00
Classes requiring direct contact with the teacher	60	2,50
Student's own work (literature studies, preparation for laboratory classes/ tutorials, preparation for tests/exam, project preparation)	40	1,50