

Object-oriented design and programming - opis przedmiotu

Informacje ogólne	
Nazwa przedmiotu	Object-oriented design and programming
Kod przedmiotu	11.3-WE-BizEIP-PiProgrObiek-Er
Wydział	Wydział Informatyki, Elektrotechniki i Automatyki
Kierunek	Biznes elektroniczny
Profil	praktyczny
Rodzaj studiów	Program Erasmus pierwszego stopnia
Semestr rozpoczęcia	semestr zimowy 2020/2021

Informacje o przedmiocie	
Semestr	2
Liczba punktów ECTS do zdobycia	5
Typ przedmiotu	obowiązkowy
Język nauczania	angielski
Sylabus opracował	- dr inż. Tomasz Gratkowski - dr inż. Jacek Tkacz

Formy zajęć					
Forma zajęć	Liczba godzin w semestrze (stacjonarne)	Liczba godzin w tygodniu (stacjonarne)	Liczba godzin w semestrze (niestacjonarne)	Liczba godzin w tygodniu (niestacjonarne)	Forma zaliczenia
Wykład	30	2	-	-	Zaliczenie na ocenę
Laboratorium	30	2	-	-	Zaliczenie na ocenę

Cel przedmiotu

Presentation of the basic concepts related to object-oriented programming and design and their implementation in the Java environment. Knowledge of modern programming environments supporting software development on the Java platform.

Wymagania wstępne

Fundamentals of computer science

Zakres tematyczny

Compiling and running programs on the Java platform. Overview of the Java Development Kit environment and IDE development environments for the Java platform.

Imperative and structured programming in Java. Data types, simple and reference variables, literals, operators, arrays, control instructions, variable visibility range, functions, variable properties.

Basics of object-oriented programming in Java. Classes and instances, enumerated types, packages, class and method properties. Rules for the construction of objects and learning the mechanism of cleaning the memory (garbage collector).

The object-oriented programming principle. Inheritance, polymorphism and encapsulation. Designing complex object types using composition and inheritance.

Advanced object-oriented techniques. Creating programming interfaces using abstract classes and interfaces. Extending interfaces. Internal classes and statically nested classes.

Support for development tools on the Java platform. Creating API documentation in the Java environment. Archiving Java programs and libraries. Debugger support. Basics of creating fault tolerant programs. Data validation methods, handling exceptional situations.

Selected Java implementation issues. Utility classes, stream classes for operating the input and output system, storing objects in collections, creating a graphical user interface.

Metody kształcenia

Lecture - conventional lecture using a video projector.

Laboratory - practical classes in the computer laboratory.

Efekty uczenia się i metody weryfikacji osiągnięcia efektów uczenia się

Opis efektu	Symbole efektów	Metody weryfikacji	Forma zajęć
The student knows the principles of object-oriented design and programming		zaliczenie - ustne, opisowe, testowe i inne	Wykład

Opis efektu	Symbole efektów	Metody weryfikacji	Forma zajęć
The student has knowledge about the process of software development based on Java technology,		• zaliczenie - ustne, opisowe, testowe i inne	• Wykład
The student is able to compile and run a self-written application in Java		• bieżąca kontrola na zajęciach • zaliczenie - ustne, opisowe, testowe i inne	• Laboratorium
The student knows the definitions of basic programming paradigms		• zaliczenie - ustne, opisowe, testowe i inne	• Wykład
The student is able to create and implement application, with the necessary documentation (API, implementation)		• bieżąca kontrola na zajęciach • zaliczenie - ustne, opisowe, testowe i inne	• Laboratorium

Warunki zaliczenia

Lecture - writing and/or oral test, carried out at the end of the semester

Laboratory - the final grade is the weighted sum of the marks obtained for the implementation of individual laboratory exercises and control tests verifying the substantive preparation for the exercises.

Final grade = 50% of the grade in the form of classes **lecture** + 50% of the grade in the form of **laboratory** classes.

Literatura podstawowa

1. Sierra K., Bates B.: Head First Java, 2nd Edition, O'Reilly Media; (February 22, 2005)
2. Horstmann, C.S., Cornell, G., Core Java Volume I--Fundamentals (11th Edition), Prentice Hall; (August 27, 2018)
3. Horstmann, C.S., Cornell, G., Core Java, Volume II--Advanced Features (11th Edition), Prentice Hall; (May 5, 2019)
4. Wróblewski, M.: Algorytmy, struktury danych i techniki programowania. Wydanie V, Helion, 2015
5. Cormen T. H., Leiserson C. E., Rivest R. L., Stein C., Introduction to Algorithms, 3rd Edition, The MIT Press; (July 31, 2009)

Literatura uzupełniająca

1. Martin, R.C.: The Clean Coder: A Code of Conduct for Professional Programmers 1st Edition, Prentice Hall; (May 23, 2011)
2. Eckel, B., Thinking in Java, Wydanie IV, Warszawa, Helion, 2006
3. Lis, M., Praktyczny kurs Java, Wydanie II, Gliwice, Helion, 2004
4. Coldwind G., Zrozumieć programowanie, Wydawnictwo Naukowe PWN, 2018

Uwagi

Zmodyfikowane przez dr hab. inż. Marcin Mrugalski, prof. UZ (ostatnia modyfikacja: 24-04-2020 14:51)

Wygenerowano automatycznie z systemu SylabUZ