KARTA OPISU MODUŁU KSZTAŁCENIA

Nazwa modułu/przedmiotu	Kod
Projektowanie i modelowanie oprogramowania	**1010512321010517859**

Kierunek studiów	Profil kształcenia (ogólnoakademicki, praktyczny)	Rok / Semestr
Informatyka	**ogólnoakademicki**	**1 / 2**
Ścieżka obieralności/specjalność: **Software Engineering (Inżynieria**	Przedmiot oferowany w języku: **polski**	Kurs (obligatoryjny/obieralny) **obligatoryjny**
Stopień studiów: **II stopień**	**Forma studiów** (stacjonarna/niestacjonarna) **stacjonarna**	

Godziny	Liczba punktów
Wykłady: **30** Ćwiczenia: **-** Laboratoria: **30** Projekty/seminaria: **-**	**5**

Status przedmiotu w programie studiów (podstawowy, kierunkowy, inny)	(ogólnouczelniany, z innego kierunku)
kierunkowy	**z danego kierunku**

Obszar(y) kształcenia i dziedzina(y) nauki i sztuki	**Podział ECTS (liczba i %)**
nauki techniczne	**5 100%**
nauki techniczne	**5 100%**

Odpowiedzialny za przedmiot / wykładowca:

Bartosz Walter
email: bartosz.walter@cs.put.poznan.pl
tel. 616652980
Faculty of Computing
ul. Piotrowo 3 60-965 Poznań

Wymagania wstępne w zakresie wiedzy, umiejętności, kompetencji społecznych:

1	**Wiedza:**	Student starting this module should have a basic knowledge of software engineering and object-oriented design.
2	**Umiejętności:**	Should have skills to design and implement of simple software systems and skills of solving basic problems related to requirements analysis, creating software specification, designing systems and skills that are necessary to acquire information from given sources of information.
3	**Kompetencje społeczne**	Student should understand the need to extend his/her competences / has the willingness to work in a team. In addition, in respect to the social skills the student should show attitudes as honesty, responsibility, perseverance, curiosity, creativity, manners, and respect for other people.

Cel przedmiotu:

1. Provide students with knowledge on OOP, in particular the role, responsibility and relationships of objects
2. Present methods of evaluating design quality of object-oriented systems with use of OO metrics and code smells
3. Develop students? teamwork skills in the context of designing software systems
4. Present unit testing as a method for verification if objects properly fulfill their responsibilities
5. Present design patterns as a reusable schemas leading to improving quality of object-oriented design. Teaching students using design patterns in implementing software systems.
6. Present software refactoring as a technique of improving internal quality of software systems.

Efekty kształcenia i odniesienie do kierunkowych efektów kształcenia

Wiedza:

1. ma uporządkowaną i podbudowaną teoretycznie wiedzę ogólną związaną z kluczowymi zagadnieniami z zakresu informatyki - [K2st_W2]
2. ma zaawansowaną wiedzę szczegółową dotyczącą wybranych zagadnień z zakresu informatyki - [K2st_W3]
3. ma zaawansowaną i szczegółową wiedzę o procesach zachodzących w cyklu życia systemów informatycznych sprzętowych lub programowych - [K2st_W5]
4. zna zaawansowane metody, techniki i narzędzia stosowane przy rozwiązywaniu złożonych zadań inżynierskich i prowadzeniu prac badawczych w wybranym obszarze informatyki - [K2st_W6]

Umiejętności:

1. potrafi - przy formułowaniu i rozwiązywaniu zadań inżynierskich -i ntegrować wiedzę z różnych obszarów informatyki (a w razie potrzeby także wiedzę z innych dyscyplin naukowych) oraz zastosować podejście systemowe, uwzględniające także aspekty pozatechniczne - [K2st_U5]

2. potrafi ocenić przydatność i możliwość wykorzystania nowych osiągnięć (metod i narzędzi) oraz nowych produktów informatycznych - [K2st_U6]

3. potrafi - stosując m.in. koncepcyjnie nowe metody - rozwiązywać złożone zadania informatyczne, w tym zadania nietypowe oraz zadania zawierające komponent badawczy - [K2st_U10]

4. potrafi - zgodnie z zadaną specyfikacją, uwzględniającą aspekty pozatechniczne - zaprojektować złożone urządzenie, system informatyczny lub proces oraz zrealizować ten projekt ? co najmniej w części ? używając właściwych metod, technik i narzędzi, w tym przystosowując do tego celu istniejące lub opracowując nowe narzędzia - [K2st_U11]

5. potrafi określić kierunki dalszego uczenia się i zrealizować proces samokształcenia, w tym innych osób - [K2st_U16]

Kompetencje społeczne:

1. rozumie, że w informatyce wiedza i umiejętności bardzo szybko stają się przestarzałe - [K2st_K1]

2. rozumie znaczenie wykorzystywania najnowszej wiedzy z zakresu informatyki w rozwiązywaniu problemów badawczych i praktycznych - [K2st_K2]

Sposoby sprawdzenia efektów kształcenia

Formative assessment:

a) lectures:

? based on the answers to the questions which test understanding of material presented on the lectures

b) laboratory classes / tutorials / projects / seminars:

? based on the assessment of the tasks done during classes and as a homework

Summative assessment:

a) verification of assumed learning objectives related to lectures:

? assessment of knowledge and skills, examined by a written test with multiple choices and problem questions. Student can gain 10.0 pts; passing limit is 5.0 pts

? discussing the results of the examination

b) verification of assumed learning objectives related to laboratory classes / tutorials / projects / seminars:

? assessment of student?s preparation to particular laboratory classes and assessment of student?s skills needed to realize tasks on these classes

? continuous assessment of student?s work during classes ? rewarding ability to use learned principles and methods

? assessment of projects realization, including ability to work in team

Possibility to gain additional points by activity on classes:

? elaboration of additional aspects regarding the subject

? effectiveness of applying acquired knowledge to solve problems

? ability to cooperate with the team during solving problems

? providing additional remarks for the lecturer how to improve teaching materials

? elaboration of an outstanding solution to an assignment ? for use as a case-study

? highlighting the problems with students? perception to improve the teaching process

Treści programowe

The program of the lecture:

The concept of objects and object-oriented perception. Mechanisms of object-oriented programming. Object-oriented languages vs. object-oriented design. Roles of different types of objects in design. Criteria for evaluation of object-oriented design. Metrics and their interpretation. Unit testing. Mock objects. Design patterns ? idea, description, categories. Overview of the catalogue of design patterns, with description of goal, description, participants and consequences ? for each of them. The code decay phenomenon ? reasons, symptoms, consequences. High-level evaluation of design quality with code smells. Methods of identification of code smells. Overview of selected refactorings. Verification methods of refactorings. Aspect-oriented programming and its implementation in different technologies. AspectJ as an aspect-oriented language. Inversion of Control and Dependency Injection.

The course consists of fifteen 2-hour laboratory classes and it starts with an instructional session at the beginning of a semester. Students work individually or in teams of 2-4.

The program of laboratory classes:

Creating preliminary design with CRC cards. Analysis and evaluation of the CRC design. Assigning responsibility to objects. Measuring software with OO metrics. Analysis and interpretation of OO metrics. Implementing unit tests. Applying mock objects in unit tests. Selection and application of appropriate design patterns in design problems. Identification of code smells in code. Comparison of metrics and code smells as tools for evaluation of design quality. Applying software refactorings (both manually and with tools support). Implementation of a simple aspect-oriented program and use of selected capabilities of AspectJ. Design and implementation of a simple program based on a Inversion of Control concept.

Literatura podstawowa:	
Literatura uzupełniająca:	
Bilans nakładu pracy przeciętnego studenta	
Czynność	**Czas (godz.)**
1. participating in laboratory classes / tutorials: 15 x 2 hours, 2. consulting issues related to the subject of the course; especially related to t laboratory classes and projects, 3. implementing, running and verifying software application(s) (in addition to laboratory classes) 4. participating in lectures 5. studying literature / learning aids (10 pages = 1 hour), 60 pages 6. discussing the results of the examination 7. preparing to and participating in exams	30 2 16 30 6 1 15

Obciążenie pracą studenta		
forma aktywności	**godzin**	**ECTS**
Łączny nakład pracy	100	5
Zajęcia wymagające bezpośredniego kontaktu z nauczycielem	62	2
Zajęcia o charakterze praktycznym	56	2