



KARTA OPISU PRZEDMIOTU - SYLABUS

Nazwa przedmiotu

Projektowanie i modelowanie oprogramowania

Przedmiot

Kierunek studiów

Informatyka

Studia w zakresie (specjalność)

Inżynieria oprogramowania

Poziom studiów

drugiego stopnia

Forma studiów

stacjonarne

Rok/semestr

1/1

Profil studiów

ogólnoakademicki

Język oferowanego przedmiotu

angielski

Wymagalność

obieralny

Liczba godzin

Wykład

30

Laboratoria

30

Inne (np. online)

Ćwiczenia

Projekty/seminaria

Liczba punktów ECTS

4

Wykładowcy

Odpowiedzialny za przedmiot/wykładowca:

dr hab. inż. Bartosz Walter

email: bartosz.walter@cs.put.poznan.pl

tel. +48 61 655 2980

Wydział Informatyki i Telekomunikacji

ul. Piotrowo 2, 60-965 Poznań

Odpowiedzialny za przedmiot/wykładowca:

Wymagania wstępne

Student rozpoczynający ten przedmiot powinien posiadać podstawową wiedzę z podstaw programowania oraz inżynierii oprogramowania. Powinien również posiadać umiejętność pozyskiwania informacji ze wskazanych źródeł oraz mieć gotowość do podjęcia współpracy w ramach zespołu.

Cel przedmiotu

Przekazanie studentom wiedzy z metod modelowania oraz obiektowego projektowania systemów informatycznych z wykorzystaniem dobrych praktyk oraz wzorców projektowych. Rozwijanie u studentów umiejętności oceny jakości projektu oraz stosowania wybranych mechanizmów dostępnych w obiektowych językach programowania.

Przedmiotowe efekty uczenia się

Wiedza

1. Posiada ugruntowaną wiedzę teoretyczną dotyczącą cyklu rozwojowego systemu informatycznego



2. Posiada wiedzę na temat wybranych metod, języków i notacji modelowania oprogramowania
3. Posiada wiedzę na temat wzorców projektowych oraz dobrych praktyk w zakresie projektowania oprogramowania
4. Zna wybrane metody pomiaru niektórych charakterystyk oprogramowania (np. rozmiaru, złożoności)

Umiejętności

1. Potrafi zaprojektować system informatyczny korzystając z mechanizmów i cech języków obiektowych
2. Potrafi dokonać oceny jakości projektu systemu informatycznego
3. Potrafi stworzyć model oprogramowania korzystając z wybranych cech języka UML

Kompetencje społeczne

1. Potrafi współpracować w grupie
2. Potrafi poszerzać swoją wiedzę, korzystając z dostępnych źródeł i dokonując ich selekcji

Metody weryfikacji efektów uczenia się i kryteria oceny

Efekty uczenia się przedstawione wyżej weryfikowane są w następujący sposób:

Wiedza przedstawiona w ramach wykładu weryfikowana jest poprzez realizację podczas wykładu, w ramach współpracy w grupach, dwóch zadań projektowych oraz egzamin końcowy -- test wielokrotnego wyboru sprawdzający stopień zrozumienia i przyswojenia treści wykładowych. Oceny uzyskane z tych form są uśredniane z wagami 30% i 70%. Próg zaliczeniowy wynosi 50%. Zagadnienia zaliczeniowe zostaną przedstawione podczas ostatniego wykładu w ramach przedmiotu.

Umiejętności nabyte w ramach laboratorium weryfikowane są poprzez realizację w grupach 3-4 projektów, dotyczących poszczególnych zagadnień omawianych w trakcie zajęć. Próg zaliczeniowy wynosi 50%.

Treści programowe

1. Wykład: Przegląd metod i zagadnień związanych z programowaniem obiektowym. Metody modelowania oprogramowania. Testowanie jednostkowe. Pomiar i metryki związane z kodem programów. Wzorce projektowe. Programowanie aspektowe. Programowanie funkcyjne. Zasada odwrócenia sterowania.
2. Laboratorium: Modelowanie z wykorzystaniem kart CRC oraz elementów języka UML. Testowanie jednostkowe programów obiektowych. Metryki oprogramowania. Dobór i implementacja wzorców projektowych. Zastosowanie wybranych paradygmatów programowania w praktyce. Implementacja odwróconego sterowania.

Metody dydaktyczne

1. Wykład: prezentacja multimedialna



2. Laboratorium: prezentacja ilustrowana przykładami podanymi na tablicy, wykonanie w grupach zadań przedstawionych przez prowadzącego, dyskusja nad rozwiązaniami

Literatura

Podstawowa

1. E. Gamma et al.: Wzorce projektowe. Elementy oprogramowania obiektowego wielokrotnego użytku. Helion, 2012
2. R. C. Martin: Clean code. A Handbook of agile software craftsmanship. Prentice Hall, 2008
3. B. Eckel: Thinking in Java. Edycja polska. Wydanie IV. Helion, 2017.

Uzupełniająca

1. B. Meyer: Programowanie zorientowane obiektowo (Second Edition). Helion, 2005.
2. J. Backfield: Programowanie funkcyjne. Krok po kroku. Helion, 2015

Bilans nakładu pracy przeciętnego studenta

	Godzin	ECTS
Łączny nakład pracy	100	4,0
Zajęcia wymagające bezpośredniego kontaktu z nauczycielem	60	2,5
Praca własna studenta (studia literaturowe, przygotowanie do zajęć laboratoryjnych/ćwiczeń, przygotowanie do kolokwium/egzaminu, wykonanie projektu) ¹	40	1,5

¹ niepotrzebne skreślić lub dopisać inne czynności