KAPITAŁ LUDZKI
NARODOWA STRATEGIA SPÓJNOŚCI

Projekt współfinansowany przez Unię Europejską w ramach Europejskiego Funduszu Społecznego

UNIA EUROPEJSKA
EUROPEJSKI FUNDUSZ SPOŁECZNY

Nazwa przedmiotu

Inżynieria oprogramowania

Kod ECTS

11.3.0747

Nazwa jednostki prowadzącej przedmiot

Faculty of Mathematics, Physics and Informatics

Studia

wydział	kierunek		
Wydział Matematyki, Fizyki i Informatyki	Informatyka	poziom	pierwszego stopnia
		forma	stacjonarne
		moduł specjalnościowy	wszystkie
		specjalizacja	wszystkie

Nazwisko osoby prowadzącej (osób prowadzących)

dr inż. Emilia Lubecka; dr Magdalena Godlewska; dr Paweł Pączkowski

Formy zajęć, sposób ich realizacji i przypisana im liczba godzin

Formy zajęć

Wykład, Ćw. laboratoryjne

Sposób realizacji zajęć

zajęcia w sali dydaktycznej

Liczba godzin

Ćw. laboratoryjne: 15 godz., Wykład: 15 godz.

Liczba punktów ECTS

2
Przedmiot w wymiarze 15h wykładu i 15h laboratorium + praca własna studenta.

Cykl dydaktyczny

2017/2018 letni

Status przedmiotu

obowiązkowy

Język wykładowy

polski

Metody dydaktyczne

- Dyskusja
- Rozwiązywanie zadań
- Samodzielne wykonanie miniprojektów
- Wykład z prezentacją multimedialną

Forma i sposób zaliczenia oraz podstawowe kryteria oceny lub wymagania egzaminacyjne

Sposób zaliczenia

- Zaliczenie na ocenę
- Zaliczenie (zal)

Formy zaliczenia

- ustalenie oceny zaliczeniowej na podstawie ocen cząstkowych otrzymywanych w trakcie trwania semestru
- kolokwium

Podstawowe kryteria oceny

Przedmiot kończy się pisemnym kolokwium zaliczeniowym, zaliczenie od 51% punktów. Przed przystąpieniem do kolokwium konieczność zaliczenia ćwiczeń laboratoryjnych. Ocena z ćwiczeń na podstawie wyników z prac domowych oraz aktywności na zajęciach.

Sposób weryfikacji założonych efektów kształcenia

zakładany efekt kształcenia	egzamin	kolokwium	projekt	reeferat	raport	aktywność w dyskusji	obserwacja
	Wiedza						
K_W02		x					
K_W03		x					
K_W09		x					
K_W12							x
	Umiejętności						
K_U03			x				
K_U04			x				
K_U06			x				
K_U19			x				
K_U22			x				
	Kompetencje						
K_K06							x

Określenie przedmiotów wprowadzających wraz z wymogami wstępnymi

A. Wymagania formalne

Brak

B. Wymagania wstępne

Języki programowania, Programowanie obiektowe

Cele kształcenia

Celem przedmiotu jest zapoznanie studentów z podstawowymi problemami, metodami, technikami i narzędziami produkcji oprogramowania wysokiej jakości. Przedstawienie różnych modeli cyklu życia oprogramowania, metod wytwarzania oprogramowania, podstaw dokumentacji, analizy wymagań, projektowania, testowania oprogramowania. Omówienie zarządzania analizą wymagań, tworzenia specyfikacji ze zdefiniowanymi metrykami. Omawiane zagadnienia przedstawiane z naciskiem na obiektowe podejście do projektowania i modelowanie systemu za pomocą języka UML.

Treści programowe

1. Wprowadzenie; motywacje dla systematycznego wytwarzania oprogramowania.
2. Cykl życia oprogramowania, fazy cyklu klasycznego, miejsce analizy i projektowania w cyklu wytwarzania.
3. Wymagania. Kategorie, pozyskiwanie, analiza i specyfikacja wymagań. Weryfikacja i walidacja; testowanie akceptacyjne.
4. Modelowanie; modelowanie świata a modelowanie systemu, miejsce modelu w analizie i projektowaniu.
5. Przypadki użycia. Diagram, interpretacja, opis ustrukturalizowany, przykłady.
6. Przegląd podstawowych zasad i pojęć paradygmatu obiektowego: obiekt, klasyfikacja, agregacja, dziedzicznie, komunikacja. Obiektowość a naturalny dla człowieka sposób postrzegania rzeczywistości.
7. Wstęp do metodyki UML: model klas, model dynamiczny. Związki pomiędzy modelami.
8. Obiekty, klasy, atrybuty, operacje. Związki pomiędzy klasami (asocjacje). Związki jako klasy. Role. Agregacja jako szczególny przypadek związku. Semantyka agregacji. Propagacja operacji jako kryterium agregacji. Przykłady.
9. Dziedziczenie: specjalizacja i generalizacja. Hierarchia dziedziczenia. Redefiniowanie właściwości obiektów w dół hierarchii dziedziczenia, przesłanianie Klasy abstrakcyjne. Obiekt jako wystąpienie klasy i wszystkich jej nadklas.
10. Jak tworzyć model klas? Przykład tworzenia modelu klas.
11. Model dynamiczny: zdarzenia, czynności, akcje. Diagramy sekwencji (interakcji). Powiązanie z m innymi modelami. Jak tworzyć diagram sekwencji. Przykład.
12. Diagram stanów. Stany: proste, złożone, współbieżne. dziedziczenie stanów. Przejścia między stanami. Powiązanie z modelem klas. Jak tworzyć diagram model dynamiczny? Przykład tworzenia modelu dynamicznego.
13. Przejście od modelu do projektu. Podstawowe fazy projektu obiektowego. Podział na podsystemy i moduły, identyfikacja współbieżności, przydział procesorów, przydział zasobów, priorytety.
14. Przejście od projektu do programu.

Wykaz literatury

Materiały wykładów (umieszczone na Portalu Edukacyjnym)
I. Sommerville: Inżynieria oprogramowania, WNT 2003
Szejko St. (red). Metody wytwarzania oprogramowania. MIKOM, 2002.
Jaszkiewicz A.: Inżynieria oprogramowania. Helion, 1997.
Uzupełniająca:
Booch G., Rumbaugh J., Jacobson I.: UML podręcznk użytkownika. WNT, 2001

Dumnicki R., Kasprzyk A., Kozłowski M.: Analiza i projektowanie obiektowe. Helion, 1998
Eriksson H-E, Penker M.: UML Toolkit. Wiley Computer Publishing, John Wiley & Sons, Inc. 1998.
Pressman R.S.: Software Engineering. A Practitioner's Approach. McGraw-Hill, Inc. 1992.

Efekty kształcenia (obszarowe i kierunkowe)

K_W02: ma uporządkowaną, podbudowaną teoretycznie wiedzę ogólną w zakresie programowania, algorytmów i złożoności, architektury systemów komputerowych, systemów operacyjnych, technologii sieciowych, języków i paradygmatów programowania, grafiki i komunikacji człowiek-komputer, sztucznej inteligencji, baz danych, inżynierii oprogramowania, języków formalnych, K_W03: zna podstawowe metody projektowania, analizowania i programowania algorytmów, K_W09: ma wiedzę na temat inżynierii oprogramowania, projektowania, wzorców projektowych, wykorzystania API, narzędzi i środowisk wytwarzania oprogramowania, cyklu życia projektu informatycznego, specyfikacji oprogramowania, walidacji i weryfikacji, utrzymywania oprogramowania, K_W12: zna podstawowe zasady bezpieczeństwa i higieny pracy w zawodzie informatyka
K_U03: potrafi pracować indywidualnie i w zespole informatyków, w tym także potrafi zarządzać swoim czasem oraz podejmować zobowiązania i dotrzymywać terminów, K_U04: potrafi porozumiewać się przy użyciu różnych technik w środowisku zawodowym oraz w innych środowiskach, w tym w języku angielskim oraz z wykorzystaniem narzędzi informatycznych, K_U06: projektuje, analizuje pod kątem poprawności i złożoności obliczeniowej oraz programuje algorytmy; wykorzystuje podstawowe techniki algorytmiczne i struktury danych, K_U19: tworzy, ocenia i realizuje plan testowania, K_U22: posługuje się wzorcami projektowymi
K_K06: potrafi formułować opinie na temat podstawowych zagadnień informatycznych

Wiedza

Student:
- zna różne modele cyklu życia oprogramowania;
- zna metody wytwarzania oprogramowania;
- posiada wiedzę odnośnie podstaw wytwrzania dokumentacji, analizy wymagań, projektowania, testowania oprogramowania.

Umiejętności

Student:
- potrafi zarządzać analizą wymagań;
- potrafi tworzyć specyfikację ze zdefiniowanymi metrykami;
- potrafi utworzyć projekt oprogramowania za pomocą języka UML.

Kompetencje społeczne (postawy)

Student:
- posiada podstawowe umiejętności komunikacji w zespole informatycznym

Kontakt

elubecka@inf.ug.edu.pl