

**KAPITAŁ LUDZKI**
NARODOWA STRATEGIA SPÓJNOŚCIProjekt współfinansowany przez
Unię Europejską w ramach
Europejskiego Funduszu
Społecznego**UNIA EUROPEJSKA**
EUROPEJSKI
FUNDUSZ SPOŁECZNY

Nazwa przedmiotu		Kod ECTS	
Podstawy inżynierii oprogramowania		11.3.0737	
Nazwa jednostki prowadzącej przedmiot			
Faculty of Mathematics, Physics and Informatics			
Studia			
wydział	kierunek	poziom	pierwszego stopnia
Wydział Matematyki, Fizyki i Informatyki	Informatyka	forma	niestacjonarne (zaoczne)
		moduł	wszystkie
		specjalnościowy	wszystkie
		specjalizacja	wszystkie
Nazwisko osoby prowadzącej (osób prowadzących)			
dr Robert Fidytek; dr Magdalena Godlewska			
Formy zajęć, sposób ich realizacji i przypisana im liczba godzin		Liczba punktów ECTS	
Formy zajęć		2 Przedmiot w wymiarze 10h wykładu i 10h laboratorium + praca własna studenta	
Wykład, Ćw. laboratoryjne			
Sposób realizacji zajęć			
zajęcia on-line, zajęcia w sali dydaktycznej			
Liczba godzin			
Wykład: 10 godz., Ćw. laboratoryjne: 10 godz.			
Cykl dydaktyczny			
2017/2018 letni			
Status przedmiotu		Język wykładowy	
obowiązkowy		polski	
Metody dydaktyczne		Forma i sposób zaliczenia oraz podstawowe kryteria oceny lub wymagania egzaminacyjne	
<ul style="list-style-type: none"> - Dyskusja - Praca w grupach - Rozwiązywanie zadań - Wykład z prezentacją multimedialną 		Sposób zaliczenia	
		<ul style="list-style-type: none"> - Zaliczenie na ocenę - Zaliczenie (zał) 	
		Formy zaliczenia	
		<ul style="list-style-type: none"> - ustalenie oceny zaliczeniowej na podstawie ocen cząstkowych otrzymywanych w trakcie trwania semestru - egzamin pisemny (dłuższa wypowiedź pisemna / rozwiązanie problemu) 	
		Podstawowe kryteria oceny	
		Laboratorium: Ocena zależy od sumy punktów zdobytych za zrealizowane zadania. Wykład: Wynik zaliczenia laboratorium - 50% oceny. Egzamin pisemny lub pisemne zadanie zaliczeniowe - 50% oceny.	
Sposób weryfikacji założonych efektów kształcenia			

zakładany efekt kształcenia	egzamin pisemny/zadanie zaliczeniowe	zadania indywidualne	zadania wykonywane w grupie	obserwacja
Wiedza				
K_W02	x			
K_W03	x			
K_W09	x			
K_W12				x
Umiejętności				
K_U03		x	x	
K_U04			x	
K_U06		x	x	
K_U19			x	
K_U22			x	
Kompetencje				
K_K01				x

Określenie przedmiotów wprowadzających wraz z wymogami wstępnymi**A. Wymagania formalne**

Aktywny udział w zajęciach.

B. Wymagania wstępne

Języki programowania, algorytmy i struktury danych, bazy danych.

Cele kształcenia

Celem przedmiotu jest zapoznanie studentów z podstawowymi problemami, metodami, technikami i narzędziami produkcji oprogramowania wysokiej jakości. Nacisk jest położony na obiektowe podejście do projektowania i modelowanie systemu w UML.

Treści programowe

1. Proces tworzenia Systemów Informatycznych
2. Pojęcie procesu i projektu (budowy) oprogramowania. Modele cyklu życia oprogramowania (model kaskadowy, realizacja kierowana dokumentami, V-model, prototypowanie, model przyrostowy, spiralny)
3. Określanie wymagań i analiza. Rodzaje i role notacji wykorzystywanych w fazie analizy, obiektowe i strukturalne metody analizy.
4. UML 2. – standard modelowania obiektowego: poziomy model i typy diagramów. Przykłady użycia (use case)- elementy diagramów, organizowanie. Model logiczny - diagramy klas, diagramy behawioralne (sekwencji, stanów, aktywności), Model implementacyjny – diagram komponentów (komponenty, podsystemy, porty) i model fizyczny (diagram montażowy).
5. Testowanie i inspekcja. Testowanie ukierunkowane na wyszukiwanie defektów – funkcjonalne, strukturalne (złożoność cyklomatyczna McCabe, pokrycie kodu - ścieżki niezależne, pokrycie danych, testowanie pętli). Testowanie obiektów. Estymacja kosztów. Metoda punktów funkcyjnych, model COCOMO 81 i COCOMO II. Model CMM. Miary niezawodności oprogramowania i techniki programowania dla systemów o dużej niezawodności.

Wykaz literatury

1. I. Sommerville: Inżynieria oprogramowania, WNT 2003
2. Szejko St. (red). Metody wytwarzania oprogramowania. MIKOM, 2002.
3. Jaskiewicz A.: Inżynieria oprogramowania. Helion, 1997.
4. Materiały umieszczone na platformie edukacyjnej.

Uzupełniająca:

5. Booch G., Rumbaugh J., Jacobson I.: UML podręcznik użytkownika. WNT, 2001
6. Dumnicki R., Kasprzyk A., Kozłowski M.: Analiza i projektowanie obiektowe. Helion, 1998
7. Eriksson H-E, Penker M.: UML Toolkit. Wiley Computer Publishing, John Wiley & Sons, Inc. 1998.
8. 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

Wiedza

Student ma uporządkowaną wiedzę dotyczącą metryk procesu tworzenia oprogramowania.

<p>złożoności, architektury systemów komputerowych, systemów operacyjnych, technologii sieciowych, języków i paradygmatów programowania, baz danych, inżynierii oprogramowania, języków formalnych,</p> <p>K_W03: zna podstawowe metody projektowania, analizowania i programowania algorytmów,</p> <p>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,</p> <p>K_W12: zna podstawowe zasady bezpieczeństwa i higieny pracy w zawodzie informatyka</p> <p>K_U03 potrafi pracować indywidualnie i w zespole informatyków, w tym także potrafi zarządzać swoim czasem oraz podejmować zobowiązania i dotrzymywać terminów,</p> <p>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,</p> <p>K_U06 projektuje, analizuje pod kątem poprawności i złożoności obliczeniowej oraz programuje algorytmy; wykorzystuje podstawowe techniki algorytmiczne i struktur danych,</p> <p>K_U19 tworzy, ocenia i realizuje plan testowania,</p> <p>K_U22 posługuje się wzorcami projektowymi</p> <p>K_K01 zna ograniczenia własnej wiedzy i rozumie potrzebę dalszego kształcenia</p> <p>K_K06 potrafi formułować opinie na temat podstawowych zagadnień informatycznych</p>	<p>Student zna techniki i narzędzia do produkcji oprogramowania wysokiej jakości.</p> <p>Student zna podstawy tworzenia dokumentacji projektów informatycznych.</p>
	<p>Umiejętności</p> <p>Student umie tworzyć dokumentację projektów informatycznych.</p> <p>Student umie projektować systemy informatyczne za pomocą języka UML.</p> <p>Student umie wykorzystać narzędzia do produkcji oprogramowania wysokiej jakości.</p>
	<p>Kompetencje społeczne (postawy)</p> <p>Student zna ograniczenia własnej wiedzy i rozumie potrzebę dalszego kształcenia.</p> <p>Student potrafi formułować własne opinie na temat projektowanych systemów.</p>
<p>Kontakt</p> <p>robert.fidytek@inf.ug.edu.pl</p>	