Zał. nr 5 do ZW 13/2019
Załącznik nr … do programu studiów

WYDZIAŁ PODSTAWOWYCH PROBLEMOW TECHNIKI

KARTA PRZEDMIOTU

Nazwa przedmiotu w języku polskim: INŻYNIERIA OPROGRAMOWANIA
Nazwa przedmiotu w języku angielskim: SOFTWARE ENGINEERING

Kierunek studiów (jeśli dotyczy): INŻYNIERIA BIOMEDYCZNA
Specjalność (jeśli dotyczy): INFORMATYKA MEDYCZNA
Poziom i forma studiów: I ~~/ II~~ stopień ~~/ jednolite studia magisterskie*~~, stacjonarna / ~~niestacjonarna*~~
Rodzaj przedmiotu: ~~obowiązkowy /~~ wybieralny ~~/ ogólnouczelniany *~~
Kod przedmiotu INP001035W, INP2017L, INP2017P
Grupa kursów ~~TAK /~~ NIE*

	Wykład	Ćwiczenia	Laboratorium	Projekt	Seminarium
Liczba godzin zajęć zorganizowanych w Uczelni (ZZU)	30		30	15	
Liczba godzin całkowitego nakładu pracy studenta (CNPS)	90		60	60	
Forma zaliczenia	Egzamin / ~~zaliczenie na ocenę*~~	Egzamin / zaliczenie na ocenę*	~~Egzamin /~~ zaliczenie na ocenę*	~~Egzamin /~~ zaliczenie na ocenę*	Egzamin / zaliczenie na ocenę*
Dla grupy kursów zaznaczyć kurs końcowy (X)					
Liczba punktów ECTS	2		2	2	
w tym liczba punktów odpowiadająca zajęciom o charakterze praktycznym (P)			2	2	
w tym liczba punktów ECTS odpowiadająca zajęciom wymagającym bezpośredniego kontaktu (BK)	1		1,5	0,5	

*niepotrzebne skreślić

WYMAGANIA WSTĘPNE W ZAKRESIE WIEDZY, UMIEJĘTNOŚCI I KOMPETENCJI SPOŁECZNYCH

Z zakresu wiedzy:

- Znajomość kluczowych zagadnień z zakresu programowania, w tym tworzenia algorytmów i struktur danych oraz elementarnych współczesnych technik programowania, w tym programowania zorientowanego obiektowo.

Z zakresu umiejętności:

- Umiejętność projektowania, implementowania oraz analizowania programów. realizujących średnio złożone zadanie programistyczne.
- Umiejętność tworzenia programów w paradygmacie obiektowym.
- Umiejętność samodzielnego znajdowania i usuwania błędów w tworzonych programach.
- Umiejętność komunikowania się z użyciem terminologii z zakresu informatyki.

CELE PRZEDMIOTU

C1 Uzyskanie podstawowej wiedzy z zakresu tworzenia oprogramowania oraz kierowania projektem programistycznym.

C2 Nabycie podstawowych umiejętności w zakresie niektórych współczesnych metodyk i technik projektowania oprogramowania oraz zarządzania projektem programistycznym.

PRZEDMIOTOWE EFEKTY UCZENIA SIĘ

Z zakresu wiedzy:

PEU_W01 Ma podstawową wiedzę w zakresie inżynierii oprogramowania.

PEU_W02 Zna wybrane podstawowe metodyki i techniki projektowania oprogramowania oraz zarządzania projektem programistycznym.

Z zakresu umiejętności:

PEU_U01 Potrafi specyfikować wymagania w projekcie programistycznym w sposób właściwy dla Informatyki Medycznej.

PEU_U02 Potrafi zaprojektować system informatyczny.

PEU_U03 Potrafi korzystać ze współczesnych technik tworzenia systemów informatycznych przy realizacji systemu informatycznego z zakresu Informatyki Medycznej.

PEU_U04 Potrafi sprawdzić poprawność oprogramowania.

Z zakresu kompetencji społecznych:

PEU_K01 Jest gotów współdziałać i współpracować w grupie przyjmując w niej różne role oraz wykazując inicjatywę.

PEU_K02 Jest gotów podejmować decyzje dotyczące organizacji realizacji zadań w ramach projektu programistycznego oraz poddawać krytycznej ocenie ich wykonanie.

PEU_K03 Dba o przestrzeganie etyki zawodowej twórcy oprogramowania z uwzględnieniem specyfiki Informatyki Medycznej.

TREŚCI PROGRAMOWE

Forma zajęć - wykład		Liczba godzin
Wy1	**Wprowadzenie do inżynierii oprogramowania** Zasady zaliczanie kursu, przegląd podstawowych pojęć, narzędzi i technik stosowanych w inżynierii oprogramowania	2
Wy2	**Systemy kontroli wersji** Omówienie roli wersjonowania kodu, dokumentacji i środowiska uruchomieniowego. GIT jako przykład rozproszonego systemu kontroli wersji	2
Wy3	**Cykl życia projektu informatycznego.** Omówienie najczęściej spotykanych modeli. Tworzenie dokumentacji projektowej	2
Wy4	**Zbieranie i specyfikacja wymagań w projektach IT** Przegląd stosowanych technik i narzędzi	2
Wy5	**Kaskadowe i zwinne metody zarządzania projektami IT** Omówienie różnić, mocnych i słabych stron różnych metod organizacji pracy w projektach IT	2
Wy6	**Elementy języka UML** Przegląd podstawowych diagramów używanych do zapisu wymagań i architektury	2
Wy7	**Kolokwium sprawdzające I**	2
Wy8	**Testy jednostkowe jako podstawowa technika rozwijania aplikacji** Omówienie podejścia RED, GREEN, REFACTOR	2
Wy9	**Testowanie oprogramowania**	2

	Przegląd metod walidacji oprogramowania: testy integracyjne, wydajnościowe, regresyjne, funkcjonalne, użyteczności, akceptacyjne	
Wy10	**Inne metody dbania o jakość aplikacji. Wirtualizacji środowiska uruchomieniowego aplikacji.** Omówienie metryk i narzędzi wspomagających zarządzanie jakością kodu. Omówienie nowoczesnych technik wirtualizacji zasobów IT, w szczególności metod "konteneryzacji"	2
Wy11	**Ciągła integracja (*continuous integration*), ciągłe dostarczanie (*continuous delivery*)** Kultura DevOps jako niezbędny element nowoczesnego projektu informatycznego	2
Wy12	**Modelowanie domenowe** Projektowanie systemów IT skoncentrowane na domenie biznesowej	2
Wy13	**Architektura aplikacji** Przegląd najczęściej spotykanych obecnie rozwiązań opartych na chmurze obliczeniowej	2
Wy14	**Podsumowanie najważniejszych elementów z inżynierii oprogramowania. Kariera w branży IT, co warto wiedzieć**	2
Wy15	**Kolokwium sprawdzające II**	2
	Suma godzin	**30**

Forma zajęć – laboratorium		**Liczba godzin**
La1	Wprowadzenie do laboratorium. Zasady obowiązujące na zajęciach. Podstawy pracy w systemie Linux. Zintegrowane środowisko programistyczne	2
La2	System kontroli wersji jako podstawowe narzędzie programisty	2
La3	Wirtualizacja środowiska programistycznego z użyciem kontenerów	2
La4-5	Wdrożenie aplikacji w środowisku chmurowym	4
La6-7	Sprawdzanie poprawności oprogramowania za pomocą testów jednostkowych	4
La8-9	Sprawdzanie poprawności oprogramowania za pomocą testów integracyjnych, komponentowych, funkcjonalnych	4
La10	Zbierania metryk obrazujących jakość kodu	2
La11	Kompleksowe podejście do tworzenia dokumentacji projektowej	2
La12	Zbudowanie środowiska ciągłej integracji	2
La13-14	Refaktoryzacja kodu	4
La15	Podsumowanie i zaliczenie	2
	Suma godzin	**30**

Forma zajęć – projekt		**Liczba godzin**
Pr1	Wprowadzenie do projektu. Zasady obowiązujące na zajęciach. Wybór tematu, stworzenie planu pracy	1
Pr2	Ćwiczenia z projektowania systemów informatycznych – metody zbierania i opisywania wymagań systemu.	2
Pr3	Określenie celu, specyfikacja produktu i kryteriów akceptacji, estymacja pracochłonności. Wybór technologii i koncepcji architektury	2
Pr4	Ćwiczenia z budowania procesu zarządzania rozwojem oprogramowania. Przepływ zadań, estymacja, priorytetyzacja, *definition of done*, *definition of ready*.	2
Pr5-6	Implementacja rozwiązania. Retrospekcja postępu prac	4
Pr7-8	Refaktoryzacja rozwiązania. Dokumentacja projektu. Zaliczenie	4
	Suma godzin	**15**

STOSOWANE NARZĘDZIA DYDAKTYCZNE
N1. Wykład – prezentacja multimedialna wspomagana metodą tradycyjną (tablica i pisak) N2. Projekt – zadanie projektowe grupowe w ramach zajęć i godzin CNPS N3. Laboratorium – listy zadań do samodzielnej realizacji w ramach zajęć i godzin CNPS N4. Laboratorium – przykładowe zadania rozwiązywane wspólnie w ramach zajęć N5. Laboratorium – krótkie testy sprawdzające – w formie pisemnej lub elektronicznej N6. Laboratorium i projekt – komputer i oprogramowanie: m.in. zintegrowane środowisko programistyczne, system kontroli wersji

OCENA OSIĄGNIĘCIA PRZEDMIOTOWYCH EFEKTÓW UCZENIA SIĘ

Oceny (F – formująca (w trakcie semestru), P – podsumowująca (na koniec semestru)	Numer efektu kształcenia	Sposób oceny osiągnięcia efektu uczenia się
F1	PEU_W01, PEU_W02	Egzamin
F2	PEU_W01, PEU_W02	Kolokwia sprawdzające
F3	PEU_U01, PEU_U02, PEU_U03, PEU_U04 PEU_K02, PEU_K03	Projekt programistyczny
F4	PEU_U01, PEU_U02 PEU_U03, PEU_U04 PEU_K01	Listy zadań do samodzielnej realizacji
F5	PEU_U01, PEU_U02 PEU_U03, PEU_U04	Krótkie testy sprawdzające
P – wykład = max(F1, F2) **P – projekt = F3** **P – laboratorium = 0.5*F4 + 0.5*F5**		

LITERATURA PODSTAWOWA I UZUPEŁNIAJĄCA
LITERATURA PODSTAWOWA: [1] Jaszkiewicz A., Inżynieria oprogramowania, Helion 2000. [2] Sacha K., Inżynieria oprogramowania, PWN 2010 [3] Somerville J., Inżynieria oprogramowania, WNT 2003. **LITERATURA UZUPEŁNIAJĄCA:** [1] Brooks F.P., Legendarny osobomiesiąc. Opowieści o inżynierii oprogramowania. Wyd.II. Helion 2019. [2] Chrapko M., Scrum. O zwinnym zarządzaniu projektami. Wydanie II rozszerzone. Helion 2015. [3] Fowler M., UML w kropelce, LTP, 2005. [4] Fowler M., i in., Refaktoryzacja: ulepszanie struktury istniejącego kodu, Helion 1991, 2011 [5] Martin R.C., Czysty kod. Podręcznik dobrego programisty, Helion 2010 [6] Pressman R.S., Praktyczne podejście do inżynierii oprogramowania, WNT 2005, 2010. [7] Wiegers K.E., Beatty J., Specyfikacja oprogramowania. Inżynieria wymagań. Wyd. III. Helion 2014.
OPIEKUN PRZEDMIOTU (IMIĘ, NAZWISKO, ADRES E-MAIL)
Mgr inż. Mateusz Milian, mateusz.milian@gmail.com **Dr inż. Witold Dyrka witold.dyrka@pwr.wroc.pl**