Załącznik nr 1.5 do Zarządzenia Rektora UR nr 12/2019

SYLABUS

DOTYCZY CYKLU KSZTAŁCENIA *2020-2024*
(skrajne daty)
Rok akademicki 2022/2023

1. PODSTAWOWE INFORMACJE O PRZEDMIOCIE

Nazwa przedmiotu	***Programowanie zespołowe***
Kod przedmiotu*	
Nazwa jednostki prowadzącej kierunek	*Kolegium Nauk Przyrodniczych*
Nazwa jednostki realizującej przedmiot	*Kolegium Nauk Przyrodniczych, Instytut Informatyki*
Kierunek studiów	*Informatyka i ekonometria*
Poziom studiów	*studia inżynierskie I-go stopnia*
Profil	*praktyczny*
Forma studiów	*stacjonarne*
Rok i semestr/y studiów	*rok III semestr 6*
Rodzaj przedmiotu	*przedmiot kierunkowy*
Język wykładowy	*język polski*
Koordynator	*dr hab. Jan Bazan, prof. UR*
Imię i nazwisko osoby prowadzącej / osób prowadzących	

***** *-opcjonalnie, zgodnie z ustaleniami w Jednostce*

1.1.Formy zajęć dydaktycznych, wymiar godzin i punktów ECTS

Semestr (nr)	Wykł.	Ćw.	Konw.	Lab.	Sem.	ZP	Prakt.	Inne (jakie?)	**Liczba pkt. ECTS**
6	15			30					4

1.2. Sposób realizacji zajęć

☒zajęcia w formie tradycyjnej

☐ zajęcia realizowane z wykorzystaniem metod i technik kształcenia na odległość

1.3 Forma zaliczenia przedmiotu (z toku) (egzamin, zaliczenie z oceną, zaliczenie bez oceny)

ZALICZENIE Z OCENĄ

2.WYMAGANIA WSTĘPNE

Znajomość zagadnień z przedmiotów: programowanie obiektowe, inżynieria oprogramowania, algorytmy i struktury danych, bazy danych

3. CELE, EFEKTY UCZENIA SIĘ , TREŚCI PROGRAMOWE I STOSOWANE METODY DYDAKTYCZNE

3.1 Cele przedmiotu

C1	Poznanie najnowszych narzędzi tworzenia oprogramowania (które mają być używane przez studentów na zajęciach laboratoryjnych).
C2	Nauka organizacji pracy zespołu informatycznego w procesie tworzenia, serwisowania i wdrażania oprogramowania
C3	Wykonanie przez studentów złożonego, praktycznego projektu informatycznego w grupie.

3.2 Efekty uczenia się dla przedmiotu

EK (efekt uczenia się)	Treść efektu uczenia się zdefiniowanego dla przedmiotu	Odniesienie do efektów kierunkowych [1]
EK_01	Student ma wiedzę i umiejętności z języka programowania orientowanego obiektowo umożliwiającą realizację projektów.	K_W03, K_U01, K_U11, K_U13
EK_02	Student zna i potrafi użyć wybrane narzędzia zespołowego wytwarzania oprogramowania.	K_W03, K_U01, K_U11, K_U13, K_K01
EK_03	Student rozumie rolę dokumentacji technicznej zadania informatycznego i ma podstawową wiedzę na temat tworzenia dokumentacji technicznej tworzonego oprogramowania w wybranym narzędziu jej tworzenia.	K_U11, K_U13
EK_04	Student potrafi pracować w zespole nad wspólnym projektem.	K_U01, K_U10, K_U11, K_U13, K_U16, K_K01

3.3 Treści programowe

A. Problematyka wykładu

Treści merytoryczne
Wprowadzenie do projektowania zespołowego (duża waga i trudności programowania zespołowego, duży wysiłek inżynierii oprogramowania celem opracowania efektywnych metod, cykl produkcji oprogramowania).
Metodologia zarządzania Scrum oraz przykładowe narzędzia do zarządzania projektami.
Zasady tworzenia dokumentacji technicznej oraz standardów kodowania oprogramowania.
Przegląd wybranych metod przydatnych do efektywnego programowania (wybrane wzorce projektowe, asercje, dzienniki, techniki testowania).
Ogólne wprowadzenie do narzędzi zespołowego wytwarzania oprogramowania na przykładzie współczesnych narzędzi wytwarzania oprogramowania w języku Java. Narzędzia kompilujące (ANT, MAVEN, GRADLE). Narzędzia kontroli wersji (GIT). Narzędzia ciągłej integracji. Narzędzia mierzenia pokrycie testami jednostkowymi.

[1] W przypadku ścieżki kształcenia prowadzącej do uzyskania kwalifikacji nauczycielskich uwzględnić również efekty uczenia się ze standardów kształcenia przygotowującego do wykonywania zawodu nauczyciela.

Narzędzia automatycznego tworzenia dokumentacji technicznej.
Ogólne zalecenia związane z wykorzystaniem powyższych narzędzi do zespołowego tworzenia oprogramowania.

B. Problematyka ćwiczeń audytoryjnych, konwersatoryjnych, laboratoryjnych, zajęć praktycznych

Treści merytoryczne
Powtórzenie wiadomości z języka Java.
Ustalenie 4-5 osobowych zespołów. Wyłonienie liderów grup. Wybranie tematów projektów.
Rozpoczęcie pracy z systemem kontroli wersji.
Rozpoczęcie pracy z narzędziem do zarządzania projektami.
Ustalenie celu i zakresu systemów tworzonych przez każdą z grup.
Ustalenie wstępnych wymagań tworzonego systemu w każdej grupie.
Modelowanie tworzonego systemu w każdej grupie.
Projektowanie tworzonego systemu w każdej grupie.
Implementacja wszystkich modułów systemu, w tym dokumentowani systemu (JavaDoc) oraz pisanie testów biało-skrzynkowych jednostkowych i integracyjnych dla każdego systemu.
Wykonanie dokumentacji systemu.
Testowanie systemu.
Wdrożenie systemu.
Zaliczenie projektów.

3.4 Metody dydaktyczne

Wykład z prezentacją multimedialną
Laboratorium: wykonywanie ćwiczeń tablicowych i programistycznych.

4. METODY I KRYTERIA OCENY

4.1 Sposoby weryfikacji efektów uczenia się

Symbol efektu	Metody oceny efektów uczenia się (np.: kolokwium, egzamin ustny, egzamin pisemny, projekt, sprawozdanie, obserwacja w trakcie zajęć	Forma zajęć dydaktycznych (w, ćw, ...)
EK_ 01	Dwa kolokwia przy komputerze z programowania obiektowego (podstawy Javy oraz zaawansowana Java). Kolokwium praktyczne przy komputerze z tworzenia złożonych kwerend SQL, weryfikacja podczas przeglądów sprintu oraz ustne zaliczenie projektu.	LAB
EK_ 02	Kolokwium praktyczne przy komputerze z ogólnej znajomości narzędzia GIT, weryfikacja repozytoriów GitLab/GitHub oraz tablic sprintów w Jira podczas przeglądów sprintu oraz przy zaliczeniu projektu.	LAB
EK_ 03	Pisemne kolokwium ze stosowania standardów kodowania i weryfikacja dokumentacji technicznej oprogramowania przy zaliczeniu projektu	LAB
EK_ 04	Weryfikacja repozytoriów GitLab/GitHub oraz tablic sprintów Jira podczas przeglądów sprintu oraz przy zaliczeniu projektu.	LAB

4.2 Warunki zaliczenia przedmiotu (kryteria oceniania)

Efekt	Ocena	Kryteria otrzymania oceny
EK_01	dst	Student ma wiedzę i umiejętności z języka programowania orientowanego obiektowo umożliwiającą realizację projektów. W szczególności wie jak i potrafi utworzyć prosty interfejs graficzny użytkownika, w elementarny sposób skorzystać z bazy danych z poziomu języka oraz utworzyć i użyć prostą bibliotekę funkcji.
	db	Student ma wiedzę i umiejętności z języka programowania orientowanego obiektowo umożliwiającą realizację projektów. W szczególności wie jak i potrafi utworzyć interfejs graficzny użytkownika zawierający bardziej zaawansowane elementy, rutynowo korzystać z bazy danych z poziomu języka oraz utworzyć i użyć bardziej zaawansowana bibliotekę funkcji (np. z użyciem prostych standardowych struktur danych lub zaawansowanych metod konstruowania algorytmów).
	bdb	Student ma wiedzę z języka programowania orientowanego obiektowo umożliwiającą realizację projektów. W szczególności wie jak i potrafi utworzyć interfejs graficzny użytkownika zawierający zaawansowane elementy uwzględniając aspekty intuicyjności i prostoty w użytkowaniu, optymalnie skorzystać z bazy danych z poziomu języka oraz utworzyć i użyć bardziej zaawansowana bibliotekę funkcji (np. z użyciem optymalnie dobranych standardowych struktur danych lub zaawansowanych metod konstruowania algorytmów).
EK_02	dst	Student zna narzędzia zespołowego wytwarzania oprogramowania i potrafi użyć wybrane narzędzie zespołowego wytwarzania oprogramowania.
	db	Student zna narzędzia zespołowego wytwarzania oprogramowania, potrafi porównać dostępne narzędzia oraz potrafi użyć co najmniej dwa wybrane narzędzie zespołowego wytwarzania oprogramowania.
	bdb	Student zna narzędzia zespołowego wytwarzania oprogramowania, potrafi je porównać oraz wskazać optymalne do prac nad danym projektem. Potrafi także użyć jednocześnie (w kooperacji ze sobą) co najmniej dwa wybrane narzędzie zespołowego wytwarzania oprogramowania.
EK_03	dst	Student rozumie rolę dokumentacji technicznej zadania informatycznego i ma podstawową wiedzę na temat tworzenia dokumentacji technicznej tworzonego oprogramowania w wybranym narzędziu jej tworzenia.
	db	Student rozumie rolę dokumentacji technicznej zadania informatycznego i ma rozszerzoną wiedzę na temat tworzenia dokumentacji technicznej tworzonego oprogramowania w wybranym narzędziu jej tworzenia. Jednak nie zawsze rozumie

		potrzebę i użyteczność stosowania niektórych zaawansowanych elementów tworzenia dokumentacji.
	bdb	Student rozumie rolę dokumentacji technicznej zadania informatycznego i ma rozszerzoną wiedzę na temat tworzenia dokumentacji technicznej tworzonego oprogramowania w wybranym narzędziu jej tworzenia oraz rozumie potrzebę i użyteczność stosowania zaawansowanych elementów tworzenia dokumentacji.
EK_04	dst	Student potrafi pracować w zespole nad wspólnym projektem, przyjmując ustaloną rolę. Jednak potrafi w pełni zrealizować tylko podstawowe zadania wyspecyfikowane przez prowadzącego w ramach prac projektowych według współczesnych standardów znanych z literatury.
	db	Student potrafi pracować w zespole nad wspólnym projektem, przyjmując różne role. Jednak nie potrafi w pełni zrealizować wszystkich zadań wyspecyfikowanych przez prowadzącego w ramach prac projektowych według współczesnych standardów znanych z literatury.
	bdb	Student potrafi pracować w zespole nad wspólnym projektem, przyjmując różne role. Potrafi także w pełni zrealizować wszystkie zadania wyspecyfikowane przez prowadzącego w ramach prac projektowych według współczesnych standardów znanych z literatury.

Zasady uzyskania oceny końcowej:

Zaliczenie laboratorium następuje na podstawie zaliczenia wszystkich efektów weryfikowanych przez planowane w danym okresie metody weryfikacji. Przy czym zakłada się, że każda metoda weryfikacji dostarcza osobne oceny dla każdego z weryfikowanych przez nią efektów uczenia się. Jeśli dany efekt jest weryfikowany przez więcej niż jedną metodę, to ocena weryfikująca osiągnięcie tego efektu jest obliczana jako średnia arytmetyczna ocen uzyskanych w poszczególnych metodach weryfikowania tego efektu.

Zaliczenie wykładu i przedmiotu następuje na podstawie zaliczenia laboratorium oraz zajęć projektowych.

Student otrzymuje z zaliczenia ocenę **niedostateczny,** gdy metody weryfikacji wykażą, iż co najmniej jeden z efektów nie został osiągnięty (średnia ocena dla tego efektu jest niższa niż 3.0);

Student otrzymuje ocenę **dostateczny,** gdy przeciętnie każdy z efektów zostanie osiągnięty na poziomie co najmniej 3.0, ale chociaż jeden z efektów został osiągnięty na poziomie mniejszym od 3.75;

Student otrzymuje ocenę **dobry,** gdy przeciętnie każdy z efektów zostanie osiągnięty na poziomie co najmniej 3.75, ale chociaż jeden z efektów został osiągnięty na poziomie mniejszym od 4.75;

Student otrzymuje ocenę **bardzo dobry,** gdy przeciętnie każdy z efektów zostanie osiągnięty na poziomie co najmniej 4.75;

5. CAŁKOWITY NAKŁAD PRACY STUDENTA POTRZEBNY DO OSIĄGNIĘCIA ZAŁOŻONYCH EFEKTÓW W GODZINACH ORAZ PUNKTACH ECTS

Forma aktywności	Średnia liczba godzin na zrealizowanie aktywności
Godziny kontaktowe wynikające z harmonogramu studiów	45
Inne z udziałem nauczyciela akademickiego (udział w konsultacjach, egzaminie)	
Godziny niekontaktowe – praca własna studenta (przygotowanie do zajęć, egzaminu, napisanie referatu itp.)	55
SUMA GODZIN	100
SUMARYCZNA LICZBA PUNKTÓW ECTS	4

** Należy uwzględnić, że 1 pkt ECTS odpowiada 25-30 godzin całkowitego nakładu pracy studenta.*

6. PRAKTYKI ZAWODOWE W RAMACH PRZEDMIOTU

wymiar godzinowy	-
zasady i formy odbywania praktyk	-

7. LITERATURA

Literatura podstawowa:

1. Wykłady: http://fenix.univ.rzeszow.pl/bazan/ oraz Witryny internetowe na temat programowania w języku Java.
2. J. Ferguson Smart, Java: praktyczne narzędzia, Helion, Gliwice (2009)
3. Schildt, H.: Java. Przewodnik dla początkujących, wydanie VI, Gliwice: Helion (2015).
4. Horstmann, C., S., Cornell, G.: Java, Podstawy, wydanie IX, Gliwice: Helion (2016).

Literatura uzupełniająca:

1. Hunt, D. Thomas, JUnit: pragmatyczne testy jednostkowe w Javie, Helion, Gliwice, (2006).
2. Horstmann, C., S., Cornell, G.: Java, Techniki zaawansowane, wydanie IX, Gliwice: Helion (2017).

Akceptacja Kierownika Jednostki lub osoby upoważnionej