



Karta przedmiotu

Nazwa i kod przedmiotu	Inżynieria oprogramowania, PG_00064064						
Kierunek studiów	Fizyka Techniczna						
Data rozpoczęcia studiów	październik 2025 r.	Rok akademicki realizacji przedmiotu			2027/2028		
Poziom kształcenia	I stopnia - inżynierskie	Grupa zajęć			Grupa zajęć fakultatywnych Grupa zajęć powiązanych z prowadzonymi badaniami naukowymi w dziedzinie nauki związanej z kierunkiem - profil ogólnoakademicki		
Forma studiów	stacjonarne	Sposób realizacji			na uczelni		
Rok studiów	3	Język wykładowy			polski		
Semestr studiów	6	Liczba punktów ECTS			7.0		
Profil kształcenia	ogólnoakademicki	Forma zaliczenia			egzamin		
Jednostka prowadząca	Wydziały Politechniki Gdańskiej -> Wydział Fizyki Technicznej i Matematyki Stosowanej -> Instytut Fizyki i Informatyki Stosowanej -> Zakład Fizyki Teoretycznej i Informatyki Kwantowej						
Imię i nazwisko wykładowcy (wykładowców)	Odpowiedzialny za przedmiot	dr hab. inż. Marta Łabuda					
	Prowadzący zajęcia z przedmiotu						
Formy zajęć	Forma zajęć	Wykład	Ćwiczenia	Laboratorium	Projekt	Seminarium	RAZEM
	Liczba godzin zajęć	30.0	0.0	0.0	45.0	0.0	75
	W tym liczba godzin zajęć na odległość: 0.0						
	Dodatkowe informacje:						
	Wykłady i zajęcia projektowe.						
Aktywność studenta i liczba godzin pracy	Aktywność studenta	Udział w zajęciach dydaktycznych, objętych planem studiów		Udział w konsultacjach		Praca własna studenta	RAZEM
	Liczba godzin pracy studenta	75		5.0		95.0	175
Cel przedmiotu	Celem przedmiotu jest zapoznanie studentów z zasadami planowania, projektowania architektury i metod wytwarzania oprogramowania, również naukowego. Celem zajęć projektowych jest praktyczne zastosowanie wiedzy z zakresu inżynierii oprogramowania poprzez realizację zespołowego projektu informatycznego. Studenci uczą się planowania i prowadzenia projektu, analizy i specyfikacji wymagań, doboru odpowiedniego modelu wytwarzania oprogramowania oraz projektowania architektury i struktury systemu. Zajęcia rozwijają umiejętności modelowania systemów z wykorzystaniem UML, pracy zespołowej, dokumentowania projektu oraz stosowania nowoczesnych narzędzi i technologii wspierających proces tworzenia oprogramowania.						

Efekty uczenia się przedmiotu	Efekt kierunkowy	Efekt z przedmiotu	Sposób weryfikacji i oceny efektu
	[K6_K04] współdziała i pracuje w grupie, przyjmując w niej różne role	Student zna podstawowe role zespołowe w projektach informatycznych (np. analityk, projektant, programista, tester, lider zespołu, scrum master, product owner) oraz zasady efektywnej współpracy zespołowej w inżynierii oprogramowania. Student potrafi efektywnie pracować w zespole projektowym, realizując odpowiedzialnie powierzoną rolę i respektując ustalone zasady współpracy oraz dostosowując się do zmieniających się potrzeb zespołu. Student potrafi komunikować się w zespole projektowym, dzielić się zadaniami, raportować postępy oraz współuczestniczyć w podejmowaniu decyzji zespołowych.	[SK1] Ocena umiejętności pracy w grupie [SK3] Ocena umiejętności organizacji pracy
	[K6_K05] prezentuje efekty swojej pracy, przekazuje informacje w sposób powszechnie zrozumiały, komunikuje się, dokonuje samooceny oraz konstruktywnej oceny efektów pracy innych osób	Student potrafi przygotować i zaprezentować wyniki zespołowego projektu informatycznego w sposób zrozumiały dla odbiorców o różnym poziomie wiedzy technicznej. Student potrafi dokonać samooceny własnej pracy oraz konstruktywnie ocenić wkład i efekty pracy innych członków zespołu projektowego.	[SK4] Ocena umiejętności komunikacji, w tym poprawności językowej [SK3] Ocena umiejętności organizacji pracy
	[K6_W05] posiada wiedzę w zakresie metodyki i technik programowania oraz wykorzystywania wybranych narzędzi informatycznych w fizyce i technice	Student zna metodyki wytwarzania oprogramowania (np. kaskadową, iteracyjną, zwinną) oraz podstawowe techniki programowania stosowane w inżynierii oprogramowania. Student zna wybrane narzędzia informatyczne wspomagające proces tworzenia oprogramowania (środowiska programistyczne, systemy kontroli wersji, narzędzia do modelowania (np. CASE do diagramów UML) i testowania, potrafi z nich korzystać oraz rozumie ich zastosowanie w problemach fizyki i techniki.	[SW1] Ocena wiedzy faktograficznej
	[K6_U03] posiada umiejętność programowania w wybranym języku oraz stosowania podstawowych pakietów oprogramowania	Student potrafi tworzyć, kompilować i uruchamiać programy w wybranym języku programowania np. Python, stosując podstawowe konstrukcje językowe (instrukcje sterujące, struktury danych, funkcje/metody). Student potrafi stosować wybrane pakiety oprogramowania oraz narzędzia programistyczne (np. środowisko IDE, system kontroli wersji, repozytorium, biblioteki standardowe, frameworki frontendowe) w procesie tworzenia oprogramowania	[SU4] Ocena umiejętności korzystania z metod i narzędzi [SU1] Ocena realizacji zadania

Treści przedmiotu	Treści przedmiotu - wykład		
	1. Wprowadzenie do inżynierii oprogramowania. Właściwości inżynierii systemów komputerowych 2. Planowanie projektu informatycznego: podstawowe charakterystyki, pojęcia, udziałowcy projektu; cykl życia i zakres projektu. Planowanie zadań. Identyfikacja problemu; Wzbogacony wizerunek. 3. Studium wykonalności projektu informatycznego. Cele, płaszczyzny oceny techniczna, ekonomiczna, organizacyjna i prawna; ryzyko przedsięwzięcia. 4. Proces inżynierii wymagań. Określenie i analiza wymagań. Wymagania stawiane oprogramowaniu i ich dokumentacja. Cechy dobrego wymagania. Metody pozyskiwania wymagań. Podział i klasyfikacja wymagań. Zatwierdzenie wymagań. Zarządzanie wymaganiami. 5. Modelowanie procesu wytwarzania oprogramowania. Cykl życia projektu informatycznego. 6. Strategie i procesy prowadzenia projektów informatycznych: tradycyjne (model kaskadowy, model V, prototypowanie, przyrostowy, spiralny) 7. Zwinne metodyki wytwarzania oprogramowania (Agile, reuse i komponentowość). Programowanie ekstremalne. SCRUM: procesy, artefakty, role. Dobór strategii prowadzenia projektu. 8. Projektowanie architektoniczne. Strukturalizacja systemu, modele sterowania, rozkład na moduły, architektury charakterystyczne dla różnych produktów informatycznych. 9. Projektowanie obiektowe. Obiekty i klasy obiektów, procesy projektowania obiektowego, ewolucja projektu. 10. Język UML Narzędzia CASE do komputerowego wspomaganie projektowania oprogramowania. 11. Projektowanie systemów rozproszonych. Wzorce analizy i projektowania. Klasyfikacja i przykłady. 12. Projektowanie systemów czasu rzeczywistego. Architektura sprzętu. Projektowanie systemów krytycznych. Analiza bezpieczeństwa i awarii. 13. Administracja systemem. Konteneryzacja, mikrousługi, chmura obliczeniowa. 14. Projektowanie dostępu i organizacja danych. 15. Prototypowanie: środowiska twórcze i technologie. AI w projektowaniu systemów.		
	Treści przedmiotu - projekt		
	Ćwiczenia projektowe obejmują identyfikację i analizę wymagań oraz modelowanie obiektowe z użyciem narzędzi CASE i projekt architektury. Praca odbywa się w zespołach kilkusobowych. Każda grupa wykonuje zestaw ćwiczeń w odniesieniu do wybranego przez siebie i przeznaczonego do z informatyzowania obszaru. Efektem jest kompletna dokumentacja (założenia projektowe, raport wykonalności, specyfikacja wymagań na system) i projekt architektury systemu: diagramy UML, administracja systemem, projekt interfejsu oraz zarządzania danymi a także prototypowy zarys implementacji.		
Wymagania wstępne i dodatkowe	Brak		
Sposoby i kryteria oceniania osiągniętych efektów uczenia się	Sposób oceniania (składowe)	Próg zaliczeniowy	Składowa oceny końcowej
	Zadania projektowe	50.0%	75.0%
	Sprawdziany do egzaminu	50.0%	15.0%
	Prezentacja	50.0%	10.0%

Zalecana lista lektur	Podstawowa lista lektur	1. Krzysztof Sacha, Inżynieria Oprogramowania, PWN 2022 2. Ian Sommerville, Inżynieria Oprogramowania, wyd.11, PWN 2019 3. Max Kanat-Alexander, Zrozumieć oprogramowanie, Helion 2019 4. Robert C. Martin, Czysta architektura. Struktura i design oprogramowania. Przewodnik dla profesjonalistów, Helion 2022 5. Mike Cohn, Agile. Metodyki zwinne w planowaniu projektów, Helion 2019 6. Robert C. Martin, Zwinne wytwarzanie oprogramowania. Najlepsze zasady, wzorce i praktyki, Helion 2017
	Uzupełniająca lista lektur	1. Bernd Bruegge, Allen H. Dutoit, Inżynieria oprogramowania w ujęciu obiektowym UML, wzorce projektowe i Java Helion 2011 2. Keeling Michael, Zostań architektem oprogramowania, PWN 2019 3. Piotr Gaczkowski, Adrian Ostrowski, Architektura oprogramowania bez tajemnic, Helion 2022. 4. Praca zbiorowa, <i>Inżynieria oprogramowania w praktyce</i>, PWN, 2022. 5. Strony domowe do wybranych narzędzi informatycznych. Instrukcje obsługi, przykłady. 6. Roger S. Pressman, Bruce R. Maxim, <i>Inżynieria oprogramowania. Praktyczne podejście</i>, Helion, 2021 (wyd. 9)
	Adresy eZasobów	Podstawowe http://cleancoder.com/products - Strona domowa Roberta Martina uznanego programistę i eksperta w zakresie architektury i tworzenia systemów IT
Przykładowe zagadnienia/ przykładowe pytania/ realizowane zadania	1. Planowanie i ocena wykonalności projektu informatycznego analiza techniczna, ekonomiczna, organizacyjna i ryzyka. 2. Inżynieria wymagań metody pozyskiwania, dokumentowania i zarządzania wymaganiami oprogramowania. 3. Cykl życia projektu informatycznego i modele wytwarzania oprogramowania porównanie podejść tradycyjnych i zwinnych. 4. Projektowanie architektury systemów informatycznych struktura systemu, podział na moduły i style architektoniczne. 5. Projektowanie obiektowe i modelowanie UML klasy, obiekty oraz diagramy wspomagające projektowanie systemów. 6. Zwinne metodyki zarządzania projektami IT SCRUM, Agile, programowanie ekstremalne i dobór strategii projektu. 7. Nowoczesne technologie w inżynierii oprogramowania konteneryzacja, mikrouслуги, chmura obliczeniowa i AI w projektowaniu systemów. Wycieczka fakultatywna do Centrum Informatycznego Trójmiejskiej Akademickiej Sieci Komputerowej.	
Zajęcia praktyczne w ramach przedmiotu	Nie dotyczy	

Dokument wygenerowany elektronicznie. Nie wymaga pieczęci ani podpisu.