**karta przedmiotu**

1. Podstawowe informacje o przedmiocie

|  |  |
| --- | --- |
| Nazwa przedmiotu | Inżynieria oprogramowania |
| Rocznik studiów | 2022/2023 |
| Kolegium | Informatyki Stosowanej |
| Kierunek studiów | Informatyka |
| Poziom kształcenia | Studia pierwszego stopnia - inżynierskie |
| Profil kształcenia | Praktyczny |
| Specjalność | ---- |
| Osoba odpowiedzialna | dr inż. Arkadiusz Lewicki |

1. Wymagania wstępne (wynikające z następstwa przedmiotów)

|  |
| --- |
| Wstęp do programowania, Programowanie, Języki i paradygmaty programowania, Systemy operacyjne, Architektura systemów komputerowych |

1. Efekty uczenia się i sposób realizacji zajęć
   1. Cele przedmiotu

|  |  |
| --- | --- |
| C1 | Zapoznanie studenta z metodyką zarządzania poziomem dostarczania produktu i procesami wytwarzania oprogramowania z uwzględnieniem etapów określenia wymagań funkcjonalnych i pozafunkcjonalnych, specyfikacji trzech poziomów zarządzania projektem, projektowania, implementacji, walidacji, testowania i wdrażania oprogramowania. |
| C2 | Kształtowanie praktycznych umiejętności projektowania oprogramowania z zastosowaniem analizy strukturalnej, inżynierii informacyjnej oraz analizy obiektowej. |
| C3 | Kształtowanie umiejętności wyboru właściwych narzędzi wspomagających proces budowy oprogramowania oraz doboru modeli tego procesu do specyfiki realizowanego przedsięwzięcia w rzeczywistych warunkach pracy programisty, a także oceny przydatności różnych paradygmatów i związanych z nim środowisk programistycznych do rozwiązywania różnego typów problemów. |
| C4 | Kształtowanie właściwych umiejętności w zakresie specyfikowania i przeglądu wymagań oraz tworzenia, oceniania i realizowania założonego planu testowania wytwarzanego oprogramowania. |
| C5 | Kształtowanie umiejętności praktycznego posługiwania się wzorcami projektowymi w procesie projektowania oprogramowania. |
| C6 | Kształtowanie umiejętności praktycznego wykorzystania metodyki zarządzania zespołem projektowym oraz aspektami związanymi z jakością projektu. |

* 1. Przedmiotowe efekty uczenia się, z podziałem na wiedzę, umiejętności i kompetencje społeczne, wraz z odniesieniem do efektów uczenia się dla kierunku

|  |  |  |
| --- | --- | --- |
| Lp. | Opis przedmiotowych efektów uczenia się | Odniesienie do efektów uczenia się dla kierunku |
| Po zaliczeniu przedmiotu student w zakresie **wiedzy** | | |
| P\_W01 | omówić techniki oraz metody ewaluacji i testowania oprogramowania. | K\_W04 |
| P\_W02 | przedstawić metody, narzędzia, teorie i praktyki stosowane do projektowania i implementacji oprogramowania z uwzględnieniem etapów określenia wymagań, specyfikacji, walidacji i testowania oprogramowania. | K\_W07 |
| P\_W03 | omówić cykl życia urządzeń i ich oprogramowania. | K\_W08 |
| P\_W04 | przedstawić zasady modelowania i projektowania oprogramowania oraz omówić znaczenie kompromisów w fazie wyboru sposobu rozwiązania projektowego. | K\_W14 |
| Po zaliczeniu przedmiotu student w zakresie **umiejętności** | | |
| P\_U01 | oszacować czas potrzebny na realizację zleconego zadania programistycznego w celu opracowania takiego harmonogramu prac, który zapewni utrzymanie zdefiniowanych wcześniej wartości składowych kryterialnych trójkąta zależności. | K\_U02 |
| P\_U02 | efektywnie wykorzystywać narzędzia stosowane w konstruowaniu i dokumentacji procesu wytwarzania oprogramowania, ze szczególnym uwzględnieniem narzędzi do kontroli oprogramowania w tym kontroli wersji i zarządzania konfiguracją. | K\_U09 |
| P\_U03 | dokonać właściwego doboru środowiska, języka programowania i bibliotek najbardziej dopasowanych zarówno do typu realizowanego przedsięwzięcia, jak i związanej z nim infrastruktury. | K\_U10 |
| P\_U04 | zaproponować rozwiązanie zdefiniowanego praktycznego zadania inżynierskiego, porównując istniejące rozwiązania, określić jego specyfikację, zgodność z istniejącymi standardami, ocenić pozytywne i negatywne aspekty proponowanego rozwiązania, wykonać projekt zgodny ze specyfikacją i przeprowadzić weryfikację uzyskanych wyników oraz zaprezentować rozwiązanie. | K\_U17 |
| Po zaliczeniu przedmiotu student w zakresie **kompetencji społecznych** | | |
| P\_K01 | pełnić rolę kierownika projektu w małym i średnim zespole projektowym i jako taki potrafi określić wymagania projektu, przeprowadzić analizę wymagań, utworzyć jego specyfikację oraz nadzorować pracę zespołu projektowego. | K\_K04 |

* 1. Formy zajęć dydaktycznych oraz wymiar godzin i punktów ECTS (w tabeli wyróżniono zajęcia kształtujące umiejętności praktyczne)

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Studia stacjonarne (ST) | | | | | | | |
| W | K | Ćw | L | ZP | P | eL | ECTS |
| 22 |  |  | 24 |  | 20 |  | 6 |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Studia niestacjonarne (NST) | | | | | | | |
| W | K | Ćw | L | ZP | P | eL | ECTS |
| 14 |  |  | 16 |  | 20 |  | 6 |

* 1. Metody realizacji zajęć dydaktycznych

|  |  |
| --- | --- |
| Formy zajęć | Metoda realizacji |
| Wykład | Wykład informacyjny (treści W1, W2, W4) oraz wykład problemowy (treści W3, W5, W6, W7, W8, W9, W10). |
| Laboratorium | Ćwiczeniowa - ćwiczenia praktyczne przy komputerze z wykorzystaniem dostępnych narzędzi CASE oraz dedykowanego środowiska IDE z zastosowaniem metodyki uwzględniającej rzeczywiste warunki pracy inżyniera oprogramowania. |
| Projekt | Projektowa - realizacja odpowiednio zdefiniowanego przedsięwzięcia programistycznego z zastosowaniem modelu iteracyjno-przyrostowego. |

* 1. Treści kształcenia (oddzielnie dla każdej formy zajęć)

Wykład

|  |  |
| --- | --- |
| Lp. | Treści kształcenia realizowane w ramach wykładów |
| W1 | Geneza, zakres i cele inżynierii oprogramowania. Procesy determinujące sukces przedsięwzięcia programistycznego. |
| W2 | Inżynieria wymagań i proces pozyskiwania i formalizowania wymagań na poziomie biznesowym, funkcjonalnym oraz pozafunkcjonalnym, a także ograniczeń systemowych, integracyjnych i reguł biznesowych dla tworzonego oprogramowania. |
| W3 | Modele zarządzania fazami procesu wytwórczego oprogramowania. Cykl życia oprogramowania. |
| W4 | Walidacja i weryfikacja statyczna oraz weryfikacja i walidacja dynamiczna w procesie realizacji produktu informatycznego. |
| W5 | Projektowanie architektury oprogramowania z wykorzystaniem notacji UML. |
| W6 | Diagramy przypadków użycia, klas, stanów i aktywności w odniesieniu do perspektyw modelu „4+1”. |
| W7 | Dokumentacja API i proces jej tworzenia. |
| W8 | Automatyzacja czynności związanych z procesem wytwarzania oprogramowania. |
| W9 | Wzorce projektowe kreacyjne i strukturalne jako koncepcja rozwiązania wielokrotnie powtarzających się problemów. |
| W10 | Wzorce projektowe czynnościowe jako koncepcja rozwiązania wielokrotnie powtarzających się problemów. |

Laboratorium

|  |  |
| --- | --- |
| Lp. | Treści kształcenia realizowane w ramach laboratorium |
| L1 | Metody budowy testów modułowych z wykorzystaniem dedykowanych narzędzi testowania jednostkowego. Proces inspekcja kodu źródłowego. |
| L2 | Weryfikacja i walidacja wytworzonego produktu programistycznego. |
| L3 | Specyfikacja wymagań. Budowa i modelowanie składowych oprogramowania z wykorzystaniem notacji diagramu przypadków użycia. |
| L4 | Budowa i modelowanie struktury oprogramowania z wykorzystaniem diagramu klas. |
| L5 | Budowa i modelowanie dynamiki oprogramowania z wykorzystaniem diagramu stanów. |
| L6 | Reprezentacja sekwencji kroków i działań z wykorzystaniem diagramu aktywności. |
| L7 | Tworzenie dokumentacji kodowej API. |
| L8 | Wzorce projektowe kreacyjne i ich implementacja. |
| L9 | Wzorce projektowe strukturalne i ich implementacja. |
| L10 | Wzorce projektowe czynnościowe i ich implementacja. |

Projekt

|  |  |
| --- | --- |
| Lp. | Treści kształcenia realizowane w ramach projektu |
| P1 | Opracowywanie dokumentu SRS, z uwzględnieniem uzasadnienia biznesowego dla tworzonego produktu programistycznego; opracowywanie słownika projektu i kontrola jego spójności ze specyfikacją wymagań; opracowywanie diagramu przypadków użycia; wybór modułu realizacji funkcjonalnej i budowa interakcji; projektowanie klas i interfejsów dla wskazanej funkcjonalności; kompilacja i implementacja kodowa; budowa dokumentacji API i testów modułowych. |

* 1. Korelacja pomiędzy efektami uczenia się, celami przedmiotu, a treściami kształcenia

|  |  |  |
| --- | --- | --- |
| Efekt uczenia się | Cele przedmiotu | Treści kształcenia |
| P\_W01 | C1, C3, C4 | W2, W3 |
| P\_W02 | C1, C2, C3, C4 | W2 - W7, W9, W10 |
| P\_W03 | C1, C3 | W2, W9, W10 |
| P\_W04 | C1, C2, C3, C5 | W1, W2, W8 |
| P\_U01 | C1, C2, C4 | P1 |
| P\_U02 | C3, C4, C5 | L7, P1 |
| P\_U03 | C1, C3, C4 | L1, P1 |
| P\_U04 | C1, C2, C3, C4 | L2 - L6, L8-L10 |
| P\_K01 | C1, C2, C4, C6 | W2, W3, W9, W10, P1 |

* 1. Metody weryfikacji efektów uczenia się

|  |  |  |
| --- | --- | --- |
| Efekt  uczenia się | Metoda oceny | Forma zajęć, w ramach której następuje weryfikacja efektu |
| P\_W01 | Egzamin końcowy | Wykład - egzamin przedmiotowy obejmujący pytania problemowe. |
| P\_W02 | Egzamin końcowy | Wykład - egzamin przedmiotowy obejmujący pytania problemowe. |
| P\_W03 | Egzamin końcowy | Wykład - egzamin przedmiotowy obejmujący pytania problemowe. |
| P\_W04 | Egzamin końcowy | Wykład - egzamin przedmiotowy obejmujący pytania problemowe. |
| P\_U01 | Praktyczna praca kontrolna na laboratorium | Laboratorium - zadanie praktyczne do rozwiązania na podstawie dostarczonej przez prowadzącego specyfikacji wymagań w ramach pracy kontrolnej na laboratorium. |
| P\_U02 | Realizacja projektu | Realizacja projektu praktycznego w postaci dedykowanej aplikacji informatycznej, który ma uzmysłowić każdemu studentowi, że sukces realizowanego projektu oparty jest na wszystkich fazach budowy systemu, począwszy od momentu, kiedy podejmowane są strategiczne decyzje o sposobie jego realizacji aż do okresu eksploatacji. |
| P\_U03 | Realizacja projektu | Projekt - realizacja dedykowanego oprogramowania, które należy opracować na podstawie opracowanego dokumentu SRS. |
| P\_U04 | Praktyczna praca kontrolna na laboratorium | Laboratorium- zadanie praktyczne do rozwiązania z wykorzystaniem dostępnych narzędzi CASE oraz środowiska IDE w ramach pracy kontrolnej na laboratorium. |
| P\_K01 | Realizacja projektu | Realizacja projektu praktycznego w postaci dedykowanej aplikacji informatycznej, który ma uzmysłowić każdemu studentowi, że sukces realizowanego projektu oparty jest na wszystkich fazach budowy systemu, począwszy od momentu, kiedy podejmowane są strategiczne decyzje o sposobie jego realizacji aż do okresu eksploatacji. |

* 1. Kryteria oceny stopnia osiągnięcia efektów uczenia się

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Efekt uczenia się | Na ocenę 2  student nie potrafi | Na ocenę 3  student potrafi | Na ocenę 4  student potrafi | Na ocenę 5  student potrafi |
| P\_W01 | przedstawić różnic oraz dedykowanych technik i metod związanych z procesem walidacji i weryfikacji statycznej oraz walidacji i weryfikacji dynamicznej produktów wynikowych i związanych z nimi artefaktów, w odniesieniu do poszczególnych faz procesu wytwórczego oprogramowania. | przedstawić różnice oraz dedykowane techniki i metody związane z procesem walidacji i weryfikacji statycznej oraz walidacji i weryfikacji dynamicznej produktów wynikowych i związanych z nimi artefaktów, w odniesieniu do poszczególnych faz procesu wytwórczego oprogramowania. | prawidłowo określić granice testowania z wykorzystaniem niezbędnych dla tego celu zdefiniowanych aksjomatów. Przedstawić przykładowy plan testów w celu wykrycia błędów ukrytych oraz błędów w interfejsach i interakcjach pomiędzy składowymi oprogramowania, które podlega procesowi ewaluacji. | przedstawić przykładowy plan testów w celu wykrycia błędów ukrytych oraz błędów w interfejsach i interakcjach pomiędzy składowymi oprogramowania, które podlega procesowi ewaluacji. Prawidłowo określić granice testowania z wykorzystaniem niezbędnych dla tego celu zdefiniowanych aksjomatów.  Napisać poprawnie skrypt automatyzacji testowania modułowego dla przedstawionych przez prowadzącego założeń. |
| P\_W02 | omówić wpływu wszystkich znanych modeli cyklu życia oprogramowania na jego proces analizy, projektowania, implementacji, testowania i wdrożenia. | omówić wpływ wszystkich znanych modeli cyklu życia oprogramowania na jego proces analizy, projektowania, implementacji, testowania i wdrożenia. | wskazać obszar tworzonego oprogramowania, który w zależności od zastosowanego modelu oraz wymagań użytkownika końcowego może podlegać potencjalnym zmianom. | sformalizować, w rozumieniu zwięzłości i ścisłości charakterystyki aspekty metod projektowania oprogramowania z uwzględnieniem podejścia „top-down” oraz podejścia „bottom-up”. |
| P\_W03 | przedstawić cyklu życia urządzenia oraz oprogramowania, które jest dla niego dedykowane. | przedstawić cykl życia urządzenia oraz oprogramowania, które jest dla niego dedykowane. | scharakteryzować efekty oraz dokumenty i produkty realizacji każdego z etapów cyklu życia projektu informatycznego dla wskazanego przez siebie modelu. | scharakteryzować efekty oraz dokumenty i produkty realizacji każdego z etapów cyklu życia projektu informatycznego dla modelu wskazanego przez prowadzącego. |
| P\_W04 | przedstawić podstawowych zasad modelowania oprogramowania uwzględniających specyfikę procesów i uwarunkowania architektury urządzeń, na które będzie ono tworzone. | przedstawić podstawowe zasady modelowania oprogramowania uwzględniające specyfikę procesów i uwarunkowania architektury urządzeń, na które będzie ono tworzone. | omówić znaczenie kompromisów w fazie wyboru sposobu rozwiązania projektowego w zależności od uwarunkowań wynikających ze specyfikacji wymagań. | dobierać właściwą metodykę do typu rozwiązywanych problemów związanych ze specyfiką procesów i uwarunkowaniami architektury urządzeń, dla których dedykowane jest tworzone oprogramowanie. |
| P\_U01 | korzystać z dedykowanych narzędzi do tworzenia testów jednostkowych, wraz z ich tworzeniem, uruchamianiem i poprawą logiki kodu. | korzystać z dedykowanych narzędzi do tworzenia testów jednostkowych, wraz z ich tworzeniem, uruchamianiem i poprawą logiki kodu. | pisać testy jednostkowe, wykorzystując zaawansowane techniki, a także korzystać z narzędzi do automatyzacji procesu wytwarzania i testowania oprogramowania. | pisać testy jednostkowe, wykorzystując zaawansowane techniki, a także korzystać z narzędzi do automatyzacji procesu wytwarzania oprogramowania. Potrafi także projektować i implementować zaawansowane strategie automatyzacji procesu wytwarzania oprogramowania |
| P\_U02 | budować podstawowe diagramy architektury oprogramowania. | budować podstawowe diagramy architektury oprogramowania. | projektować architekturę oprogramowania, wykorzystując podstawowe wzorce projektowe. | projektować architekturę oprogramowania, wykorzystując podstawowe wzorce projektowe, a także analizować i optymalizować architekturę oprogramowania pod kątem wydajności i skalowalności. |
| P\_U03 | porównać istniejące rozwiązania i wybrać najlepsze dla wskazanego problemu w postaci zadania programistycznego. | porównać istniejące rozwiązania i wybrać najlepsze dla wskazanego problemu w postaci zadania programistycznego. | dokonać wyboru optymalnego środowiska oraz optymalnych narzędzi do założonej metodyki tworzenia oprogramowania w kontekście dziedziny problemu. | dokonać właściwego doboru środowiska i języka programowania najbardziej dopasowanego zarówno do typu realizowanego przedsięwzięcia, jak i związanej z nim infrastruktury. |
| P\_U04 | zaprezentować osiągnięte wyniki procesu realizacji aplikacji informatycznej zgodnie z założoną metodyką projektową. | zaprezentować osiągnięte wyniki procesu realizacji aplikacji informatycznej zgodnie z założoną metodyką projektową. | przedstawić wady i zalety uzyskanego rozwiązania zaprojektowanego zgodnie z przyjętą metodyką i zaimplantowanego oprogramowania bez umiejętności prawidłowego przeprowadzenia procesu walidacji i weryfikacji aplikacji. | przedstawić wady i zalety uzyskanego rozwiązania zaprojektowanego zgodnie z przyjętą metodyką i zaimplantowanego oprogramowania, jak również dokonać prawidłowego przeprowadzenia procesu walidacji i weryfikacji uzyskanego produktu finalnego realizowanego przedsięwzięcia programistycznego. |
| P\_K01 | poprawnie zamodelować w notacji UML oczekiwań funkcjonalnych i struktury powiązań komponentowych rozwiązania informatycznego związanego z dedykowanym problemem oraz na podstawie stworzonego modelu dobrać odpowiednią metodykę organizacji pracy projektowej. | poprawnie zamodelować w notacji UML oczekiwania funkcjonalne i strukturę powiązań komponentowych realizowanego rozwiązania informatycznego oraz na podstawie stworzonego modelu dobrać odpowiednią metodykę organizacji pracy projektowej. | osiągnąć wynik w postaci działającego prototypu rozwiązania z zachowaniem modelu SOLID, zgodnego z przyjętym modelem. | osiągnąć wynik w postaci działającego prototypu rozwiązania z zachowaniem modelu SOLID, zgodnego z przyjętym modelem oraz przeprowadzić zespołową prezentację stworzonego rozwiązania. |

* 1. Literatura

|  |
| --- |
| Literatura podstawowa |
| Bereza-Jarociński B., Szomański B., Inżynieria oprogramowania: jak zapewnić jakość tworzonym aplikacjom, Wydawnictwo Helion, Gliwice 2009 lub nowsze |
| Sommerville I., Inżynieria oprogramowania, Wydawnictwo Naukowe PWN, 2020 lub nowsze |
| Śmiałek M., Rybiński K., Inżynieria oprogramowania w praktyce. Od wymagań do kodu z językiem UML, Helion, Gliwice, 2024 lub nowsze |
| Hnatkowska B., Huzar Z., Inżynieria oprogramowania: metody wytwarzania i wybrane zastosowania, Wydawnictwo Naukowe PWN, Warszawa 2008 lub nowsze |
| Gamma E, Wzorce projektowe: elementy oprogramowania obiektowego wielokrotnego użytku, Wydawnictwo Helion, Gliwice 2010 lub nowsze. |

|  |
| --- |
| Literatura uzupełniająca |
| Wolff E., Ciągłe dostarczanie oprogramowania. Kompletny przewodnik, Helion, Gliwice 2018 lub nowsze |
| Bass L., Clements P., Kazman R., Architektura oprogramowania w praktyce, Wydawnictwa Naukowo-Techniczne, Warszawa 2006 lub nowsze |
| Sacha K., Inżynieria oprogramowania, Wydawnictwo Naukowe PWN, Warszawa 2010 lub nowsze, IBUK Libra |
| Cockburn A., Jak pisać efektywne przypadki użycia, Wydawnictwo Naukowo-Techniczne, Warszawa 2004 lub nowsze |
| Hunt A., Thomas D., JUnit : pragmatyczne testy jednostkowe w Javie, Wydawnictwo Helion, Gliwice 2006 lub nowsze |
| Graessle P., Baumann H., Baumann P., UML 2.0 w akcji: przewodnik oparty na projektach, Wydawnictwo Helion, Gliwice 2006 lub nowsze |
| Cheesman J., Daniels J., Komponenty w UML, Wydawnictwo Naukowo-Techniczne, Warszawa 2004 lub nowsze |
| Wrycza S., Marcinkowski B., Wyrzykowski K., Język UML 2.0 w modelowaniu systemów informatycznych, Wydawnictwo Helion, Gliwice 2005 lub nowsze |
| DeMarco T., Zdążyć przed terminem, Studio EMKA, Warszawa, 2002 lub nowsze |
| Robertson J., Robertson S., Pełna analiza systemowa, WNT, Warszawa, 1999 lub nowsze |
| Vernon V., DDD dla architektów oprogramowania, Helion, Gliwice 2016 lub nowsza. |

1. Nakład pracy studenta - bilans punktów ECTS

|  |  |  |
| --- | --- | --- |
| **Rodzaje aktywności** | **Obciążenie studenta** | |
| **studia ST** | **studia NST** |
| Udział w W/K (UB) | 22 | 14 |
| Konsultacje do W/K (UB) | 4 | 3 |
| Udział w egzaminie z W (UB) | 2 | 2 |
| Samodzielne studiowanie tematyki W/K, w tym przygotowanie do egzaminu/zaliczenia | 7 | 16 |
| Udział w C/L (UB) | 24 | 16 |
| Konsultacje do C/L (UB) | 5 | 3 |
| Samodzielne przygotowanie się do C/L, w tym przygotowanie do zaliczenia | 31 | 41 |
| Udział w i konsultacje do PS/PN/eL (UB) | 20 | 20 |
| Samodzielne przygotowanie się do zaliczenia PS/PN/eL | 30 | 30 |
| **Sumaryczne obciążenie pracą studenta** | **145** | **145** |
| **Punkty ECTS za przedmiot** | **6** | **6** |
| **Punkty ECTS za zajęcia prowadzone z bezpośrednim udziałem nauczycieli i studentów (UB)** | **3** | **2** |
| **Punkty ECTS za zajęcia kształtujące umiejętności praktyczne (PZ)** | **4** | **4** |