Nazwa przedmiotu						
Inżynieria oprogramowania Software engineering						
Dyscyplina						Oznaczenie przedmiotu
Informatyka						8K_INS1
Rodzaj przedmiotu	Stopień studiów	Tryb studiów	Język zajęć		Rok	Semestr
obowiązkowy	1	stacjonarne	polski		III	VI
Rodzaj zajęć	Wyk.	Ćw.	Lab.	Sem.	Proj.	Liczba punktów ECTS
Liczba godzin w semestrze	18	0	18	0	0	6
Koordynator	Dr inż. Jacek Łyp, jackrat@el.pcz.czest.pl					
Prowadzący	Dr inż. Jacek Łyp, jackrat@el.pcz.czest.pl Dr inż. Piotr Szeląg, szelag@el.pcz.czest.pl					

I. KARTA PRZEDMIOTU

Cel przedmiotu

C1. Przekazanie studentom ogólnej wiedzy o problemach tworzenia oprogramowania natury pozaalgorytmicznej.
C2. Zapoznanie studentów z metodami modelowania systemów informatycznych.
C3. Zapoznanie studentów z technikami budowania oprogramowania w oparciu o wzorce projektowe.
C4. Przekazanie studentom wiedzy o technikach i narzędziach wspomagających walidację, testowanie i ewolucję oprogramowania.

Wymagania wstępne w zakresie wiedzy, umiejętności i innych kompetencji

1. Umiejętności pracy samodzielnej i w grupie.
2. Znajomość podstaw języków programowania obiektowego.

Efekty kształcenia

EK1. Student posiada podstawową wiedzę i umiejętności w zakresie posługiwania się wzorcami projektowymi.
EK2. Student potrafi specyfikować wymagania i modelować projekt systemu informatycznego przy użyciu języka UML.
EK3. Student posługuje się narzędziami wspomagającymi budowę oprogramowania (testowanie, walidacja, systemy kontroli wersji).

Treści programowe: wykłady		Liczba godzin
W 1 -	Zarządzanie projektem informatycznym, modele pracy zespołowej.	1
W 2 -	Zarządzanie konfiguracją, systemy kontroli wersji, Git.	1
W 3 -	Specyfikacja wymagań, diagramy przypadków użycia UML.	1
W 4 -	Projektowanie i modelowanie struktury logicznej systemu informatycznego, diagramy klas UML.	1
W 5 -	Projektowanie i modelowanie interakcji między elementami systemu informatycznego, diagramy sekwencji UML.	1
W 6 -	Projektowanie i modelowanie przepływu sterowania i danych między składnikami systemu informatycznego, diagramy czynności UML.	1
W 7 -	Kreacyjne wzorce projektowe, Builder, Abstract Factory, Singleton.	2
W 8 -	Strukturalne wzorce projektowe, Bridge, Decorator, Composite.	2
W 9 -	Wzorce projektowe zachowania, Mediator, Observer, Strategy.	2
W 10 -	Architektoniczne wzorce projektowe, MVC.	2
W 11 -	Bugtracking, testy jednostkowe, testy automatyczne.	2
Test zaliczeniowy		2
	SUMA	**18**

Treści programowe: laboratorium		Liczba godzin
L 1 -	Ćwiczenia obsługi sytemu kontroli wersji Git.	1
L 2 -	Opracowanie specyfikacji wymagań, diagram przypadków użycia.	1
L 3 -	Opracowanie struktury logicznej systemu, diagram klas.	2
L 4 -	Opracowanie modelu interakcji między elementami systemu informatycznego, diagram sekwencji.	2
L 5 -	Opracowanie modelu w przepływu sterowania i danych systemu informatycznego, diagram czynności.	2
L 6 -	Implementacja kreacyjnych wzorców projektowych.	2
L 7 -	Implementacja strukturalnych wzorców projektowych.	2
L 8 -	Obsługa Bugzilli.	2
L 9 -	Testy jednostkowe.	2
Test zaliczeniowy		2
	SUMA	**18**

Narzędzia dydaktyczne
1. Prezentacja multimedialna
2. Tablica interaktywna
3. Specjalistyczne oprogramowanie
4. Stanowisko komputerowe w laboratorium

Sposoby oceny efektów kształcenia (F – ocena Formująca, P – ocena Podsumowująca)
P1. Pisemny test zaliczeniowy. (100% końcowej oceny z wykładu).
P2. Laboratorium – wykonanie zadań na bieżących zajęciach (50% oceny końcowej).
P3. Laboratorium - praktyczny test zaliczeniowy – (50% oceny końcowej).

Obciążenie pracą studenta

Forma aktywności	Średnia liczba godzin na zrealizowanie aktywności
Godziny kontaktowe z prowadzącym	36
Zapoznanie się ze wskazaną literaturą	34
Przygotowanie do zajęć	40
Przygotowanie do testu	40
Sumaryczna liczba godzin/punktów ECTS dla przedmiotu	**150 / 6 ECTS**

Wykaz literatury podstawowej i uzupełniającej

1. UML i wzorce projektowe. Analiza i projektowanie obiektowe oraz iteracyjny model wytwarzania, Autor: Craig Larman, Helion 2011/03
2. Włodzimierz Gajda. Git. Rozproszony system kontroli wersji. Helion 2013.
3. Sacha Krzysztof. Inżynieria oprogramowania. PWN 2010.

Macierz realizacji efektów kształcenia

Efekt kształcenia	Odniesienie efektu do efektów kształcenia dla dyscypliny naukowej Elektrotechnika*	Cele przedmiotu	Forma zajęć	Narzędzia dydaktyczne	Sposób oceny
EK1	KINF1_W11 KINF1_U17	C1, C3	W, L	1,2,3,4	P1,P2,P3
EK2	KINF1_W11 KINF1_U17	C1, C2	W, L	1,2,3,4	P1,P2,P3
EK3	KINF1_W11 KINF1_U17	C1, C4	W, L	1,2,3,4	P1,P2,P3

* – wg załącznika

II. FORMY OCENY – SZCZEGÓŁY

Ocena	Efekty
EK1	**Student posiada podstawową wiedzę i umiejętności w zakresie posługiwania się wzorcami projektowymi.**
2	Student nie posiada podstawowej wiedzy i umiejętności w zakresie posługiwania się wzorcami projektowymi.
3	Student posiada akceptowalnie minimalną wiedzę i umiejętności w zakresie posługiwania się wzorcami projektowymi.
3.5	Student zna reprezentatywne przypadki dla każdej odmiany wzorców projektowych, potrafi omówić ich specyfikę.
4	Student zna dobrze kilka wzorców projektowych, potrafi wskazać ich obszary zastosowań.
4.5	Student zna najważniejsze wzorce projektowe, potrafi wskazać ich obszary zastosowań i wyjaśnić sposób działania.
5	Student potrafi sprawnie wybrać i zaimplementować wybrane wzorce projektowe w projekcie.
EK2	**Student potrafi specyfikować wymagania i modelować projekt systemu informatycznego przy użyciu języka UML.**
2	Student nie potrafi specyfikować wymagań i nie potrafi modelować projektu systemu informatycznego przy użyciu języka UML.
3	Student potrafi wyspecyfikować wymagania dla projektu stosując diagramy przypadków użycia.
3.5	Student potrafi zaprojektować ogólną strukturę logiczną projektu stosując diagramy klas.
4	Student potrafi zaprojektować szczegółową strukturę logiczną projektu stosując diagramy klas.
4.5	Student potrafi zaprojektować i zamodelować interakcje między poszczególnymi elementami systemu stosując diagramy sekwencji.
5	Student zaprojektować i zamodelować scenariusze przypadków użycia, procesy systemowe, algorytmy stosując diagramy czynności.
EK3	**Student posługuje się narzędziami wspomagającymi budowę oprogramowania (testowanie, walidacja, systemy kontroli wersji).**
2	Student nie posługuje się narzędziami wspomagającymi budowę oprogramowania.
3	Student posługuje się podstawowymi funkcjami systemu kontroli wersji (Git).
3.5	Student posługuje się systemami kontroli wersji (Git) i narzędziami do refaktoringu.
4	Student posługuje się narzędziami do bugtrackingu (Bugzilla).
4.5	Student zna specyfikę i narzędzia stosowania testów jednostkowych oprogramowania.
5	Student zna specyfikę i narzędzia automatyzacji testów oprogramowania.

III. INNE PRZYDATNE INFORMACJE O PRZEDMIOCIE

1. Wszelkie informacje dla studentów na temat planu zajęć dostępne są na tablicy ogłoszeń oraz na stronie el.pcz.pl.
2. Informacje na temat warunków zaliczania zajęć przekazywana jest studentom podczas pierwszych zajęć.