

KARTA OPISU MODUŁU KSZTAŁCENIA		
Nazwa modułu/przedmiotu Języki i paradygmaty programowania		Kod 1010334421010334960
Kierunek studiów Informatyka	Profil kształcenia (ogólnoakademicki, praktyczny) (brak)	Rok / Semestr 1 / 2
Ścieżka obieralności/specjalność -	Przedmiot oferowany w języku: polski	Kurs (obligatoryjny/obieralny) obligatoryjny
Stopień studiów: I stopień	Forma studiów (stacjonarna/niestacjonarna) niestacjonarna	
Godziny Wykłady: 20 Ćwiczenia: - Laboratoria: 20 Projekty/seminaria: -		Liczba punktów 6
Status przedmiotu w programie studiów (podstawowy, kierunkowy, inny) (brak)		(ogólnouczelniany, z innego kierunku) (brak)
Obszar(y) kształcenia i dziedzina(y) nauki i sztuki nauki techniczne		Podział ECTS (liczba i %) 6 100%
Odpowiedzialny za przedmiot / wykładowca:		
<p>dr inż. Beata Jankowska email: beata.jankowska@put.poznan.pl tel. +48 61 665 37 24 Wydział Elektryczny ul. Piotrowo 3A 60-965 Poznań</p>		
Wymagania wstępne w zakresie wiedzy, umiejętności, kompetencji społecznych:		
1	Wiedza:	Student ma podstawową wiedzę w zakresie matematyki, obejmującą algebrę, analizę, logikę, probabilistykę oraz elementy matematyki dyskretnej i stosowanej.
2	Umiejętności:	Student potrafi: posłużyć się środowiskami i platformami programistycznymi do pisania, wykonywania i testowania programów kodowanych w językach programowania imperatywnego; przygotować i przedstawić krótką prezentację poświęconą wynikom realizacji zadania inżynierskiego.
3	Kompetencje społeczne	Student ma świadomość: odpowiedzialności za pracę własną; konieczności podporządkowania się zasadom pracy w zespole; ponoszenia odpowiedzialności za wspólnie realizowane zadania.
Cel przedmiotu:		
<p>Zapoznanie studentów ze stylem programowania obiektowego. Opanowanie przez nich umiejętności posługiwania się konstrukcjami języków obiektowych, w tym - projektowania i implementowania obszernych algorytmów w języku obiektowym C++.</p> <p>Opanowanie zasad doboru stylu i języka programowania do charakteru zadania.</p>		
Efekty kształcenia i odniesienie do kierunkowych efektów kształcenia		
Wiedza:		
<p>1. Student ma uporządkowaną i podbudowaną teoretycznie wiedzę w zakresie podstawowych algorytmów i ich analizy, technik projektowania algorytmów, abstrakcyjnych struktur danych i ich implementacji, problemów obliczeniowo trudnych - [K_W04]</p> <p>2. Student ma uporządkowaną i podbudowaną teoretycznie wiedzę w zakresie podst. konstrukcji programistycznych, implementacji algorytmów, paradygmatów i stylów programowania, metod weryfikacji poprawności programów, języków formalnych, kompilatorów, platform programistycznych - [K_W05]</p>		
Umiejętności:		
<p>1. Student potrafi konstruować algorytmy z wykorzystaniem podstawowych technik algorytmicznych i dokonać analizy ich złożoności - [K_U09]</p> <p>2. Student potrafi posłużyć się środowiskami i platformami programistycznymi do pisania, wykonywania i testowania prostych programów kodowanych w językach programowania imperatywnego, obiektowego i deklaratywnego - [K_U10]</p> <p>3. Student potrafi opracować dokumentację dotyczącą realizacji zadania inżynierskiego i przygotować tekst zawierający omówienie wyników realizacji tego zadania - [K_U03]</p>		
Kompetencje społeczne:		

1. Student ma świadomość ważności dokładnego wykonania projektu, zachowania standardów notacyjnych, przestrzegania poprawności językowej i terminowego oddania prac Konsultacje i egzamin - [K_K07]
2. Student ma świadomość ważności i rozumie pozatechniczne aspekty i skutki działalności inżyniera-informatyka i związaną z tym odpowiedzialność za podejmowane decyzje - [K_K02]

Sposoby sprawdzenia efektów kształcenia		
<p>Wykład: egzamin pisemny.</p> <p>Laboratorium: zaliczenie na podstawie wejściówek, sprawdzianów i aktywności programistycznej na zajęciach, oraz - opcjonalnie - rozwiązania indywidualnego zadania projektowego (implementacja w C++, dokumentacja pisemna).</p> <p>Kryterium egzaminacyjne i zaliczeniowe: od 50,1%.</p>		
Treści programowe		
<p>Wykład.</p> <p>Klasyfikacja stylów programowania. Podstawowe paradygmaty programowania zorientowanego obiektowo (hermetyzacja, dziedziczenie, polimorfizm) i ich realizacja w języku C++. Realizacja operacji wejścia-wyjścia w języku C++. Obsługa błędów i obsługa wyjątków w języku obiektowym. Przeciążanie nazw funkcji i operatorów. Dynamiczne zarządzanie pamięcią w językach obiektowych. Programowanie wielowątkowe.</p> <p>Laboratorium.</p> <p>Projektowanie algorytmów i ich implementacja w języku C++.</p>		
Literatura podstawowa:		
<ol style="list-style-type: none"> 1. Kernighan B., Ritchie D., Język C, WNT, Warszawa, 1988. 2. Stroustrup B., Język C++, WNT, Warszawa, 2002. 3. Grębosz J., Symfonia C++, Oficyna Kallimach, Kraków, 1999. 		
Literatura uzupełniająca:		
<ol style="list-style-type: none"> 1. Kniat J., Programowanie w języku C++, NAKOM, Poznań, 1999. 2. Liberty J., Programowanie C#, Helion, Gliwice, 2006. 3. Eckel B., Thinking in Java. Wydanie 4, edycja polska, Helion, Gliwice, 2006. 		
Bilans nakładu pracy przeciętnego studenta		
Czynność	Czas (godz.)	
1. Wykłady	20	
2. Ćwiczenia laboratoryjne	20	
3. Udział w konsultacjach i i egzaminie	10	
4. Bieżące przygotowanie do ćwiczeń laboratoryjnych	30	
5. Przygotowanie do sprawdzianów	25	
6. Przygotowanie do egzaminu	45	
Obciążenie pracą studenta		
forma aktywności	godzin	ECTS
Łączny nakład pracy	150	6
Zajęcia wymagające bezpośredniego kontaktu z nauczycielem	75	3
Zajęcia o charakterze praktycznym	75	3