

<b>KARTA OPISU MODUŁU KSZTAŁCENIA</b>		
Nazwa modułu/przedmiotu <b>Języki i paradygmaty programowania</b>		Kod <b>1010334521010334960</b>
Kierunek studiów <b>Informatyka</b>	Profil kształcenia (ogólnoakademicki, praktyczny) <b>(brak)</b>	Rok / Semestr <b>1 / 2</b>
Ścieżka obieralności/specjalność <b>-</b>	Przedmiot oferowany w języku: <b>polski</b>	Kurs (obligatoryjny/obieralny) <b>obligatoryjny</b>
Stopień studiów: <b>I stopień</b>	Forma studiów (stacjonarna/niestacjonarna) <b>niestacjonarna</b>	
Godziny Wykłady: <b>20</b> Ćwiczenia: <b>-</b> Laboratoria: <b>20</b> Projekty/seminaria: <b>-</b>		Liczba punktów <b>6</b>
Status przedmiotu w programie studiów (podstawowy, kierunkowy, inny) <b>(brak)</b>		(ogólnouczelniany, z innego kierunku) <b>(brak)</b>
Obszar(y) kształcenia i dziedzina(y) nauki i sztuki <b>nauki techniczne</b> <b>nauki techniczne</b>		Podział ECTS (liczba i %) <b>6 100%</b> <b>6 100%</b>
<b>Odpowiedzialny za przedmiot / wykładowca:</b> dr inż. Beata Jankowska email: beata.jankowska@put.poznan.pl tel. +48 61 665 37 24 Wydział Elektryczny ul. Piotrowo 3A 60-965 Poznań		
<b>Wymagania wstępne w zakresie wiedzy, umiejętności, kompetencji społecznych:</b>		
1	<b>Wiedza:</b>	Student ma podstawową wiedzę w zakresie matematyki, obejmującą algebrę, analizę, logikę, probabilistykę oraz elementy matematyki dyskretnej i stosowanej.
2	<b>Umiejętności:</b>	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	<b>Kompetencje społeczne</b>	Student ma świadomość: odpowiedzialności za pracę własną; konieczności podporządkowania się zasadom pracy w zespole; ponoszenia odpowiedzialności za wspólnie realizowane zadania.
<b>Cel przedmiotu:</b> 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++. Opanowanie zasad doboru stylu i języka programowania do charakteru zadania.		
<b>Efekty kształcenia i odniesienie do kierunkowych efektów kształcenia</b>		
<b>Wiedza:</b>		
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]		
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]		
<b>Umiejętności:</b>		
1. Student potrafi konstruować algorytmy z wykorzystaniem podstawowych technik algorytmicznych i dokonać analizy ich złożoności. - [K_U09]		
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]		
3. Student potrafi opracować dokumentację dotyczącą realizacji zadania inżynierskiego i przygotować tekst zawierający omówienie wyników realizacji tego zadania. - [K_U03]		
<b>Kompetencje społeczne:</b>		

1. Student ma świadomość ważności dokładnego wykonania projektu, zachowania standardów notacyjnych, przestrzegania poprawności językowej i terminowego oddania prac. - [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

Wykład: egzamin pisemny.

Laboratorium: zaliczenie na podstawie wejściówek, sprawdzianów i aktywności programistycznej na zajęciach, oraz - opcjonalnie - rozwiązania zadania projektowego (implementacja w C++, dokumentacja pisemna).

Kryterium egzaminacyjne i zaliczeniowe: od 50,1%.

### Treści programowe

Wykład.

Klasyfikacja stylów programowania. Cele i zasady modelowania obiektowego. Język UML -najczęściej używane diagramy struktur i zachowań. Podstawowe paradygmaty programowania zorientowanego obiektowo (hermetyzacja, dziedziczenie, polimorfizm) i ich implementacja w języku C++. Biblioteki 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 uogólnione, biblioteka STL. Programowanie wielowątkowe. Wyrażenia regularne i klasa boost::regex.

Krótką charakterystyka języka programowania Java.

SOLID - zasady efektywnego programowania w języku obiektowym.

Laboratorium.

Projektowanie algorytmów i ich implementacja w języku C++.

Zastosowane metody kształcenia: wykłady - wykład prowadzony w sposób interaktywny z formułowaniem pytań do grupy studentów lub do wskazywanych konkretnych studentów; wykład uzupełniony materiałami do samodzielnego studiowania w systemie Moodle;

Laboratoria - zajęcia na uczelni uzupełnione materiałami do samodzielnego wykonywania zadań w systemie Moodle; praca w zespołach.

Aktualizacja 2017: Cele i zasady modelowania obiektowego. Język UML - najczęściej używane diagramy struktur i zachowań. Wyrażenia regularne i klasa boost::regex. SOLID - zasady efektywnego programowania w języku obiektowym.

### Literatura podstawowa:

1. Stroustrup B., Język C++. Kompendium wiedzy. Wydanie IV, Helion, 2014.
2. Grębosz J., Symfonia C++ standard. Programowanie w języku C++ orientowane obiektowo. Tom I i II, Wydanie 3B, Helion, 2010.
3. Prata S., Język C++. Szkoła programowania. Wydanie VI, Helion, 2012.

### Literatura uzupełniająca:

1. Prata S., Język C. Szkoła programowania. Wydanie V, Helion, 2006.
2. Eckel B., Thinking in C++. Edycja polska, Helion, 2012.

### 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