KARTA OPISU MODUŁU KSZTAŁCENIA

Nazwa modułu/przedmiotu	Kod
Podstawy programowania	**1010331511010334957**

Kierunek studiów	Profil kształcenia (ogólnoakademicki, praktyczny)	Rok / Semestr
Informatyka	**(brak)**	**1 / 1**
Ścieżka obieralności/specjalność	Przedmiot oferowany w języku:	Kurs (obligatoryjny/obieralny)
-	**polski**	**obligatoryjny**

Stopień studiów:	**Forma studiów** (stacjonarna/niestacjonarna)
I stopień	**stacjonarna**

Godziny	Liczba punktów
Wykłady: **30** Ćwiczenia: **-** Laboratoria: **30** Projekty/seminaria: **-**	**6**

Status przedmiotu w programie studiów (podstawowy, kierunkowy, inny)	(ogólnouczelniany, z innego kierunku)
(brak)	**(brak)**

Obszar(y) kształcenia i dziedzina(y) nauki i sztuki	**Podział ECTS (liczba i %)**
nauki techniczne	**6 100%**

Odpowiedzialny za przedmiot / wykładowca:

dr Jerzy Bartoszek
email: jerzy.bartoszek@put.poznan.pl
tel. 61 665-3713, 61 665-2378
Wydział Elektryczny
ul. Piotrowo 3A 60-965 Poznań

Wymagania wstępne w zakresie wiedzy, umiejętności, kompetencji społecznych:

1	**Wiedza:**	ma podstawową wiedzę wynikającą z programu szkoły średniej [PRK 4]
2	**Umiejętności:**	potrafi realizować zadania wynikające z programu szkoły średniej [PRK 4]
3	**Kompetencje społeczne**	ma kompetencje społeczne wynikające z programu szkoły średniej [PRK 4]

Cel przedmiotu:

Prezentacja podstawowych stylów programowania i konstrukcji programistycznych z przykładami programów w językach C++/C.

Efekty kształcenia i odniesienie do kierunkowych efektów kształcenia

Wiedza:

1. zna i rozumie w zaawansowanym stopniu wiedzę w zakresie podstawowych konstrukcji programistycznych, implementacji algorytmów, paradygmatów i stylów programowania, metod weryfikacji poprawności programów, języków formalnych, kompilatorów i platform - [[K1_W05 (P6S_WG)]]

Umiejętności:

1. potrafi posłużyć się środowiskami i platformami programistycznymi do pisania, wykonywania i testowania prostych programów kodowanych w językach programowania imperatywnego, obiektowego i deklaratywnego, wykorzystać w tym celu metody analityczne, symulacyjne i eksperymentalne - [[K1_U10 (P6S_UW)]]
2. potrafi konstruować algorytmy z wykorzystaniem podstawowych technik algorytmicznych, dokonać analizy ich złożoności i je ocenić - [[K1_U09 (P6S_UW)]]

Kompetencje społeczne:

1. jest gotów do krytycznej oceny posiadanej wiedzy z obszaru informatyki oraz uznawania znaczenia wiedzy w rozwiązywaniu problemów poznawczych i praktycznych z obszaru informatyki - [[K1_K01 (P6S-KK)]]

Sposoby sprawdzenia efektów kształcenia

Wykład: testy pisemne z pytaniami punktowanymi i kryterium zaliczenia od 50% punktów.

Laboratorium: sprawdziany, ocena wykonanych ćwiczeń i sprawozdań.

Treści programowe

Wykłady:

Wprowadzenie: struktura prostych programów, wybrane typy danych, operatory arytmetyczne i logiczne, wyrażenia, instrukcje przypisania, warunkowe i pętle, proste operacje wejścia/wyjścia, przestrzenie nazw. Wprowadzenie do funkcji. Tablice dynamiczne i statyczne. Referencje. Struktury i przeciążanie operatorów. Pliki tekstowych i binarnych. Pliki nagłówkowe. Dynamiczne struktury danych. Wybrane elementy języka C.

Aktualizacja 2017:

Wskaźniki i dynamiczny przydział pamięci: RAII, smart pointers, make_unique, make_shared. Więcej o funkcjach i ich parametrach: przeciążanie funkcji, przekazywanie argumentów, szablony, wyrażenia lambda.

Laboratoria:

Wprowadzenie: funkcja main, int, std::string, operatory arytmetyczne, instrukcje warunkowe if/else, cin/cout, debugger. Typy proste i pętle. SVN. Funkcje. Tablice dynamicznie i statyczne: std::vector, std::array, for_each, auto. Referencje: referencje &, const, const &. Struktury. Pliki tekstowe i binarne: std::fstream, reinterpret_cast. Pliki nagłówkowe. Przestrzenie nazw. Przeciążanie funkcji i operatorów. Wskaźniki i dynamiczny przydział pamięci: RAII, smart pointers, make_unique, make_shared. Wyrażenia lambda. Szablony. Jak czytać programy w języku C?: printf, scanf, malloc, free, tablice statyczne i dynamiczne.

Zastosowane metody kształcenia:

wykłady - z prezentacją multimedialną uzupełniany przykładami podawanymi na tablicy uzupełniony materiałami do samodzielnego studiowania w systemie Moodle

laboratoria - uzupełniane prezentacjami multimedialnymi uzupełnione materiałami do samodzielnego wykonywania zadań w systemie Moodle, korzystanie z narzędzi umożliwiających studentom wykonanie zadań w domu

Literatura podstawowa:

1. Gręboszj J., Symfonia C++ standard, Programowanie w języku C++ orientowane obiektowo, T.1 i 2
2. Stroustrup B., Programming - Principles and Practice Using C++
3. http://en.cppreference.com/w/
4. https://isocpp.org/faq
5. https://msdn.microsoft.com/en-us/library/3bstk3k5.aspx
6. http://www.cplusplus.com/

Literatura uzupełniająca:

1. Banachowski L., Diks K., Rytter W., Algorytmy i struktury danych, WNT, Warszawa, 2006

Bilans nakładu pracy przeciętnego studenta

Czynność	Czas (godz.)
1. wykłady	30
2. laboratoria	30
3. konsultacje i egzamin	15
4. przygotowanie do ćw. lab., wykonanie sprawozdań	45
5. przygotowanie do sprawdzianów i egzaminu	30

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