

KARTA OPISU MODUŁU KSZTAŁCENIA		
Nazwa modułu/przedmiotu Podstawy programowania		Kod 1010334511010334957
Kierunek studiów Informatyka	Profil kształcenia (ogólnoakademicki, praktyczny) ogólnoakademicki	Rok / Semestr 1 / 1
Ścieżka obieralności/specjalność -	Przedmiot oferowany w języku: polski	Kurs (obligatoryjny/obieralny) obligatoryjny
Stopień studiów: I stopień	Forma studiów (stacjonarna/niestacjonarna) niestacjonarna	
Godziny Wykłady: 16 Ćwiczenia: - Laboratoria: 16 Projekty/seminaria: -		Liczba punktów 5
Status przedmiotu w programie studiów (podstawowy, kierunkowy, inny) kierunkowy		(ogólnouczelniany, z innego kierunku) z danego kierunku
Obszar(y) kształcenia i dziedzina(y) nauki i sztuki nauki techniczne		Podział ECTS (liczba i %) 5 100%
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
2	Umiejętności:	potrafi realizować zadania wynikające z programu szkoły średniej
3	Kompetencje społeczne	ma kompetencje społeczne wynikające z programu szkoły średniej
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. 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 - [K_W05]		
Umiejętności: 1. potrafi posłużyć się środowiskami i platformami programistycznymi do pisania, wykonywania i testowania prostych programów kodowanych w językach programowania imperatywnego - [K_U10] 2. potrafi konstruować algorytmy z wykorzystaniem podstawowych technik algorytmicznych i dokonać analizy ich złożoności - [K_U09]		
Kompetencje społeczne: 1. ma świadomość ważności dokładnego wykonania projektu, zachowania standardów notacyjnych, przestrzegania poprawności językowej i terminowego oddania prac - [K_K07]		
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		

<p>Wykłady:</p> <p>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.</p> <p>Aktualizacja 2017:</p> <p>Wskaźniki i dynamiczny przydział pamięci: RAll, smart pointers, make_unique, make_shared. Więcej o funkcjach i ich parametrach: przeciążanie funkcji, przekazywanie argumentów, szablony, wyrażenia lambda.</p> <p>Laboratoria:</p> <p>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: RAll, smart pointers, make_unique, make_shared. Wyrażenia lambda. Szablony. Jak czytać programy w języku C?: printf, scanf, malloc, free, tablice statyczne i dynamiczne.</p> <p>Zastosowane metody kształcenia:</p> <p>wykłady - z prezentacją multimedialną uzupełniany przykładami podawanymi na tablicy uzupełniony materiałami do samodzielnego studiowania w systemie Moodle</p> <p>laboratoria - uzupełniane prezentacjami multimedialnymi uzupełnione materiałami do samodzielnego wykonywania zadań w systemie Moodle, korzystanie z narzędzi umożliwiających studentom wykonanie zadań w domu</p>		
<p>Literatura podstawowa:</p> <ol style="list-style-type: none"> 1. Grębosz 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/ 		
<p>Literatura uzupełniająca:</p> <ol style="list-style-type: none"> 1. Banachowski L., Diks K., Rytter W., Algorytmy i struktury danych, WNT, Warszawa, 2006 		
<p>Bilans nakładu pracy przeciętnego studenta</p>		
<p>Czynność</p>		<p>Czas (godz.)</p>
1. wykłady		16
2. laboratoria		16
3. konsultacje i egzamin		8
4. przygotowanie do ćw. lab., wykonanie sprawozdań		48
5. przygotowanie do sprawdzianów i egzaminu		40
<p>Obciążenie pracą studenta</p>		
<p>forma aktywności</p>	<p>godzin</p>	<p>ECTS</p>
Łączny nakład pracy	128	5
Zajęcia wymagające bezpośredniego kontaktu z nauczycielem	40	2
Zajęcia o charakterze praktycznym	75	3