

<b>KARTA OPISU MODUŁU KSZTAŁCENIA</b>		
Nazwa modułu/przedmiotu <b>Podstawy programowania</b>		Kod <b>1010334511010334957</b>
Kierunek studiów <b>Informatyka</b>	Profil kształcenia (ogólnoakademicki, praktyczny) <b>(brak)</b>	Rok / Semestr <b>1 / 1</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>16</b> Ćwiczenia: <b>-</b> Laboratoria: <b>16</b> Projekty/seminaria: <b>-</b>		Liczba punktów <b>5</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>		Podział ECTS (liczba i %) <b>5 100%</b>
<b>Odpowiedzialny za przedmiot / wykładowca:</b>		
dr Jerzy Bartoszek email: jerzy.bartoszek@put.poznan.pl tel. 61 665-3713, 61 665-2378 Wydział Elektryczny ul. Piotrowo 3A 60-965 Poznań		
<b>Wymagania wstępne w zakresie wiedzy, umiejętności, kompetencji społecznych:</b>		
1	<b>Wiedza:</b>	ma podstawową wiedzę wynikającą z programu szkoły średniej [PRK 4]
2	<b>Umiejętności:</b>	potrafi realizować zadania wynikające z programu szkoły średniej [PRK 4]
3	<b>Kompetencje społeczne</b>	ma kompetencje społeczne wynikające z programu szkoły średniej [PRK 4]
<b>Cel przedmiotu:</b>		
Prezentacja podstawowych stylów programowania i konstrukcji programistycznych z przykładami programów w językach C++/C.		
<b>Efekty kształcenia i odniesienie do kierunkowych efektów kształcenia</b>		
<b>Wiedza:</b>		
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)]]		
<b>Umiejętności:</b>		
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)]]		
<b>Kompetencje społeczne:</b>		
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)]]		
<b>Sposoby sprawdzenia efektów kształcenia</b>		
Wykład: testy pisemne z pytaniami punktowanymi i kryterium zaliczenia od 50% punktów. Laboratorium: sprawdziany, ocena wykonanych ćwiczeń i sprawozdań.		
<b>Treści programowe</b>		

<p><b>Wykłady:</b>  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: RAll, smart pointers, make_unique, make_shared. Więcej o funkcjach i ich parametrach: przeciążanie funkcji, przekazywanie argumentów, szablony, wyrażenia lambda.</p> <p><b>Laboratoria:</b>  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 &amp;, const, const &amp;. 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><b>Zastosowane metody kształcenia:</b>  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</p>		
<p><b>Literatura podstawowa:</b>  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. <a href="http://en.cppreference.com/w/">http://en.cppreference.com/w/</a>  4. <a href="https://isocpp.org/faq">https://isocpp.org/faq</a>  5. <a href="https://msdn.microsoft.com/en-us/library/3bstk3k5.aspx">https://msdn.microsoft.com/en-us/library/3bstk3k5.aspx</a>  6. <a href="http://www.cplusplus.com/">http://www.cplusplus.com/</a></p>		
<p><b>Literatura uzupełniająca:</b>  1. Banachowski L., Diks K., Rytter W., Algorytmy i struktury danych, WNT, Warszawa, 2006</p>		
<p><b>Bilans nakładu pracy przeciętnego studenta</b></p>		
<b>Czynność</b>		<b>Czas (godz.)</b>
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><b>Obciążenie pracą studenta</b></p>		
<b>forma aktywności</b>	<b>godzin</b>	<b>ECTS</b>
Łączny nakład pracy	128	5
Zajęcia wymagające bezpośredniego kontaktu z nauczycielem	40	2
Zajęcia o charakterze praktycznym	75	3