

I. KARTA PRZEDMIOTU

Nazwa przedmiotu/modułu:		Podstawy programowania				Kod przedmiotu PPR
Nazwa angielska:		Fundamentals of programming				
Kierunek studiów:		Edukacja techniczno-informatyczna				
Tryb/Poziom studiów:		Stacjonarne /I-go stopnia – inżynierskie				
Profil studiów		Praktyczny				
Jednostka prowadząca:		Karkonoska Państwowa Szkoła Wyższa w Jeleniej Górze, Wydział Nauk Medycznych i Technicznych Katedra Nauk Informatyczno-Technicznych				
Prowadzący przedmiot:		dr inż. Tadeusz Lewandowski				
I. Formy zajęć, liczba godzin						
Semestr	Wykład	Ćwiczenie	Laboratorium	Projekt	Seminarium	Łącznie
II	30		30			60
Forma zaliczenia	Zaliczenie na ocenę		Zaliczenie na ocenę			
Liczba punktów ECTS	2		2			4
II. Cel przedmiotu:						
C1	Uzyskanie wiedzy z zakresu podstaw programowania zorientowanego strukturalnie w języku wysokiego poziomu oraz zaawansowanych technik programowania.					
C2	Zapoznanie z zasadami konstruowania, zapisu i analizy algorytmów oraz podstawami programowania w języku strukturalnym (struktura programu, typy danych, współpraca z systemem plików, biblioteki funkcji).					
C3	Nabycie umiejętności samodzielnego pisania prostych programów w języku wysokiego poziomu oraz stosowania w praktyce podstawowych algorytmów i struktur danych.					
C4	Poznanie i stosowanie zasad poprawnego pisania programów.					
III. Wymagania wstępne w zakresie wiedzy, umiejętności i innych kompetencji:						
Brak wymagań wstępnych						
IV. Oczekiwane efekty uczenia się:						
Wiedza						
EK1	Ma uporządkowaną wiedzę ogólną na temat podstawowych pojęć związanych z programowaniem oraz zasad i technik programowania strukturalnego w języku wysokiego poziomu, w tym: struktury programu, operatorów, typów danych, wyrażeń i instrukcji sterujących programem, operacji wejścia/wyjścia oraz obsługi systemu plików i bibliotek standardowych, przydatną do formułowania i rozwiązywania zadań inżynierskich związanych z projektowaniem i implementacją różnego rodzaju aplikacji oraz opanowania treści innych przedmiotów.					
Umiejętności						
EK2	Posiada umiejętność czytania i analizowania kodu programów w języku wysokiego poziomu.					

EK3	Ma niezbędne umiejętności implementacji prostych zadań w języku wysokiego poziomu z zachowaniem zasad programowania strukturalnego oraz edycji kodu źródłowego poprzez zastosowanie komentarzy, wcięć i optymalizacji.
EK4	Posiada umiejętności weryfikacji wykonanych rozwiązań oraz radzenia sobie z typowymi błędami programistycznymi.
EK5	Potrafi zastosować w praktyce poznane zasady i techniki programowania strukturalnego w języku wysokiego poziomu.
Kompetencje społeczne	
EK6	Rozumie potrzebę samodzielnego uzupełniania i doskonalenia wiedzy oraz umiejętności związanych z programowaniem strukturalnym oraz potrzebę systematycznej pracy własnej.
EK7	Wykazuje aktywną postawę i chęć współpracy z innymi podczas rozwiązywania trudniejszych zadań oraz potrafi pracować indywidualnie, oraz w zespole.
EK8	Przestrzega zasad etyki i ochrony własności intelektualnej.
V. Treści programowe:	
Forma zajęć: wykład	
Wyk1	Wstęp: Alfabet, składnia, semantyka. Języki formalne. Języki bezkontekstowe jako narzędzie definiowania składni. Notacja BNF i EBNF. Przegląd wybranych języków programowania. Kod maszynowy, kod źródłowy i kod wykonywalny. Translacja: kompilacja i interpretacja. Wstęp do programowania strukturalnego. Język programowania jako sposób zapisu algorytmu.
Wyk2	Typy danych: Pojęcie typu danych. Konwersja typów (niejawna i jawna). Typy proste (całkowite, rzeczywiste, logiczny, znakowy) – omówienie.
Wyk3	Program i jego elementy: stałe, zmienne (lokalne i globalne), zakres ważności nazw, instrukcje proste, instrukcje strukturalne (warunkowe, iteracyjne). Wyrażenia. Priorytety i łączność operatorów. Operatory: przypisania, arytmetyczne, logiczne, relacyjne, połączenia. Przykłady zastosowań.
Wyk4	Funkcje i podprogramy: Przekazywanie parametrów w funkcjach. Funkcje rekurencyjne. Przykładowe programy.
Wyk5	Struktury danych: Tablice, struktury, unie. Przykłady zastosowań.
Wyk6	Wskaźniki: Wskaźniki. Operacje na wskaźnikach. Operatory referencji i dereferencji. Zastosowanie wskaźników. Przykłady.
Wyk7	Struktury danych: Pliki. Schemat przetwarzania plików. Operacje na plikach. Klasa fstream.
Wyk8	Przegląd bibliotek: Przegląd klasycznych bibliotek w języku wysokiego poziomu.
Wyk9	Realizacja zaawansowanych konstrukcji algorytmicznych w programach: Dynamiczne struktury danych: listy, stos, kolejki, kolejki priorytetowe, drzewa i ich reprezentacje, implementacje struktur dynamicznych przy pomocy tablic.
Wyk10	Kolokwium zaliczeniowe. Powtórzenie wiadomości.
Suma godzin	
30	
Forma zajęć: laboratorium	
Lab1	Zajęcia organizacyjne: Zapoznanie ze środowiskiem odpowiednim dla
2	

	języka wysokiego poziomu. Instalacja środowiska. Pierwszy program. Kompilacja i uruchomienie programu. Struktura programu (deklaracje stałych, zmiennych i funkcji). Proste typy danych (logiczny, znakowy, całkowity, rzeczywisty).	
Lab2	Pierwsze programy: Instrukcje proste: instrukcja pusta i instrukcja przypisania. Operacje wejścia/wyjścia. Proste programy. Błędy w programach.	2
Lab3	Instrukcje warunkowe i operatory: Instrukcje IF, IF-ELSE. Operatory arytmetyczne. Operatory logiczne. Priorytety operatorów. Wyrażenia logiczne.	2
Lab4	Funkcje w języku wysokiego poziomu: Argumenty funkcji, wartość funkcji. Przekazywanie argumentów funkcji przez wartość i przez referencję (lub: przez wartość i przez zmienną). Komunikacja między funkcjami. Zmienne lokalne i zmienne globalne.	2
Lab5	Instrukcje iteracyjne: Instrukcja iteracyjna FOR. Instrukcje iteracyjne WHILE i DO-WHILE.	4
Lab6	Rekurencja: Sprawne tworzenie funkcji. Funkcje rekurencyjne.	2
Lab7	Struktury danych: Tablice. Przetwarzanie tablic.	4
Lab8	Napisy: Przetwarzanie napisów. Biblioteka funkcji opartych na napisach. Klasa string.	4
Lab9	Wskaźniki: Operacje na wskaźnikach.	2
Lab10	Struktury danych: Struktury (rekordy, opcjonalnie – unie). Tworzenie (nazywanie) własnych typów danych.	2
Lab11	Struktury danych: pliki. Model logiczny pliku. Typy plików. Rodzaje dostępu do plików. Schemat przetwarzania plików. Klasa fstream.	4
Suma godzin - laboratorium		30
VI. Narzędzia dydaktyczne:		
1.	Prezentacje multimedialne z wykładu.	
2.	Listy ćwiczeń laboratoryjnych (krótkie przykłady programów do analizy).	
3.	Listy ćwiczeń programistycznych (zadania do samodzielnej implementacji).	
4.	Komputer ze zintegrowanym środowiskiem programistycznym (np. Dev Cpp, Code::Blocks).	
5.	Zadania testowe.	
6.	System e-learning – publikowanie materiałów dydaktycznych (prezentacji z wykładów, list laboratoryjnych) i ogłoszeń, gromadzenie prac studenckich z laboratoriów, udostępnianie dodatkowych materiałów.	
VII. Sposoby oceny (F – formująca, P – podsumowująca)		
F1	Ocena punktowa rozwiązań list ćwiczeniowych. Ćwiczeniowe listy zadań to zbiory prostych zadań możliwych do rozwiązania zazwyczaj podczas 1 godziny zajęć. Za rozwiązanie każdej listy zadań student otrzymuje punkty zależne od jakości, samodzielności i tempa wykonanej pracy. Student wysyła rozwiązanie zadań do systemu e-learning.	
F2	Ocena punktowa rozwiązań list programistycznych. Programistyczne listy zadań to zbiory trudniejszych zadań, zazwyczaj możliwych do rozwiązania podczas zajęć obejmujących od dwóch do czterech godzin lekcyjnych. Za rozwiązanie każdej listy zadań student otrzymuje punkty zależne od jakości, samodzielności i tempa wykonanej pracy. Student wysyła rozwiązanie zadań do systemu e-learning.	
F3	Egzamin pisemny	

P1	Ocena końcowa z wykładu ustalana jest na podstawie sumy punktów uzyskanych przez studenta z egzaminu (F3). Ocena pozytywna przyznawana jest studentowi, który zdobył łącznie przynajmniej 50% sumy wszystkich punktów możliwych do uzyskania w ramach oceny F3.
P2	Ocena końcowa z laboratorium wyznaczana jest na podstawie sumy punktów uzyskanych przez studenta ze wszystkich list zadań – ćwiczeniowych (F1) i programistycznych (F2). Ocena pozytywna przyznawana jest studentowi, który zdobył łącznie przynajmniej 50% sumy wszystkich punktów możliwych do uzyskania w ramach ocen F1 oraz F2.

VIII. Obciążenie pracą studenta

Forma aktywności	Łączna i średnia liczba godzin na zrealizowanie aktywności
Godziny kontaktowe z nauczycielem (w trakcie zajęć)	60
Przygotowanie do zajęć laboratoryjnych (średnio na studenta)	25
Przygotowanie się do kolokwium zaliczeniowego	20
Konsultacje	15
SUMA	120
SUMARYCZNA LICZBA PUNKTÓW ECTS DLA PRZEDMIOTU	4

IX. Literatura podstawowa i uzupełniająca

Literatura podstawowa:

1. Grębosz J., *Symfonia C++ standard*, Editions 2000, Kraków 2008.
2. Kernighan B, Ritchie D., *Język ANSI C*, WNT, Warszawa 2001.
3. Stroustrup B., *Język C++*. WNT, Warszawa 1998.

Literatura uzupełniająca:

1. Adamiec-Wójcik I., Guerreiro P., *Elementy programowania obiektowego w C++*. Wydawnictwo Politechniki. Łódzkiej Filii w Bielsku-Białej, Bielsko Biała 1998.
2. Hansen T.L., *C++: zadania i odpowiedzi*. WNT, Warszawa 1994.
3. Liberty J., *C++ dla każdego*. Helion, Gliwice 2002.
4. Liberty J., *C++: księga eksperta*, Helion, Gliwice 1999.
5. Meyers S., *Język C++ bardziej efektywny – 35 praktycznych sposobów ulepszenia programów*. WNT, Warszawa 1998.
6. Strużińska-Walczak A., Walczak K., *Nauka programowania dla początkujących C++*. Wydawnictwo W&W, Warszawa 1999.
7. Vandevoorde D., *Język C++: ćwiczenia i rozwiązania*. WNT, Warszawa 2001.

X. Metody dydaktyczne

M1	Wykład z elementami dyskusji.
M2	Dyskusja nad prezentowanymi przykładowymi rozwiązaniami.
M3	Demonstracje przykładowych rozwiązań zadań.
M4	Ćwiczenia symulacyjne.
M5	Indywidualne konsultacje podczas zajęć.
M6	Zajęcia laboratoryjne – lista zadań wykonywana przez studentów zgodnie z poleceniami, z bieżącym wspomaganie prowadzącego.

XI.Tablica powiązań efektów przedmiotowych i kierunkowych z celami przedmiotu oraz stosowanymi metodami dydaktycznymi					
Efekty uczenia się	Odniesienie danego efektu do efektów zdefiniowanych dla całego programu (PEK)	Cele przedmiotu	Treści programowe	Narzędzia dydaktyczne	Metody dydaktyczne
Wiedza					
EK1	K_W12, K_W13	C1, C2, C4	Wyk1 – Wyk10	1, 4, 5	M1, M2, M3
Umiejętności					
EK2	K_U12	C2, C3, C4	Wyk3 – Wyk7 Lab2 – Lab11	1, 2, 3, 4, 5, 6	M2, M3, M4, M5, M6
EK3	K_U12, K_U13	C3, C4	Lab1 – Lab11	2, 3, 4, 6	M2, M3, M5, M6
EK4	K_U12	C3	Lab2 – Lab11	2, 3, 4, 6	M1, M2, M3, M4, M5, M6
EK5	K_U13	C1, C2, C3, C4	Lab1 – Lab11	2, 3, 4, 6	M2, M3, M4, M5, M6
Kompetencje społeczne					
EK6	K_K01,K_K02	C1, C2, C4	Wyk1 – Wyk10 Lab1 – Lab11	2, 3, 5, 6	M5, M6
EK7	K_K02	C3	Lab1 – Lab11	2, 3, 5	M2, M3, M5, M6
EK8	K_K03	C3	Wyk10 Lab1 – Lab11	2, 3, 5, 6	M5, M6
XII. Zasady weryfikacji oczekiwanych efektów uczenia się					
Efekt kształcenia	Sposób weryfikacji				
EK1	F3, P1				
EK2	F1, F2, F3, P1, P2				
EK3	F1, F2, P2				
EK4	F1, F2, P2				
EK5	F1, F2, P2				
EK6	F1, F2 F3, P1, P2				
EK7	F1, F2, P2				
EK8	F1, F2, P2				
Kryteria weryfikacji					

Sposób weryfikacji	Na ocenę 2.0	Na ocenę 3.0	Na ocenę 3.5	Na ocenę 4.0	Na ocenę 4.5	Na ocenę 5.0
P1	Suma punktów uzyskanych z kolokwium (ocen F3) jest mniejsza niż 50 % sumy wszystkich punktów możliwych do uzyskania w ramach ocen F3.	Suma punktów uzyskanych z kolokwium (ocen F3) jest nie mniejsza niż 50 % sumy wszystkich punktów możliwych do uzyskania w ramach ocen F3.	Suma punktów uzyskanych z kolokwium (ocen F3) jest mniejsza niż 60 % sumy wszystkich punktów możliwych do uzyskania w ramach ocen F3.	Suma punktów uzyskanych z kolokwium (ocen F3) jest mniejsza niż 70 % sumy wszystkich punktów możliwych do uzyskania w ramach ocen F3.	Suma punktów uzyskanych z kolokwium (ocen F3) jest mniejsza niż 80 % sumy wszystkich punktów możliwych do uzyskania w ramach ocen F3.	Suma punktów uzyskanych z kolokwium (ocen F3) jest mniejsza niż 90 % sumy wszystkich punktów możliwych do uzyskania w ramach ocen F3.
P2	Suma punktów uzyskanych z list zadań ćwiczeniowych (ocen F1) oraz z list programistycznych (ocen F2) jest mniejsza niż 50 % sumy wszystkich punktów możliwych do uzyskania w ramach sumy ocen F1 i F2.	Suma punktów uzyskanych z list zadań ćwiczeniowych (ocen F1) oraz z list programistycznych (ocen F2) jest nie mniejsza niż 50 % sumy wszystkich punktów możliwych do uzyskania w ramach sumy ocen F1 i F2.	Suma punktów uzyskanych z list zadań ćwiczeniowych (ocen F1) oraz z list programistycznych (ocen F2) jest nie mniejsza niż 60 % sumy wszystkich punktów możliwych do uzyskania w ramach sumy ocen F1 i F2.	Suma punktów uzyskanych z list zadań ćwiczeniowych (ocen F1) oraz z list programistycznych (ocen F2) jest nie mniejsza niż 70 % sumy wszystkich punktów możliwych do uzyskania w ramach sumy ocen F1 i F2.	Suma punktów uzyskanych z list zadań ćwiczeniowych (ocen F1) oraz z list programistycznych (ocen F2) jest nie mniejsza niż 80 % sumy wszystkich punktów możliwych do uzyskania w ramach sumy ocen F1 i F2.	Suma punktów uzyskanych z list zadań ćwiczeniowych (ocen F1) oraz z list programistycznych (ocen F2) jest nie mniejsza niż 90 % sumy wszystkich punktów możliwych do uzyskania w ramach sumy ocen F1 i F2.
Kompetencje społeczne	Nie rozumie potrzeby samodzielnego uzupełniania i doskonalenia wiedzy oraz umiejętności związanych z programowaniem strukturalnym oraz potrzeby systematycznej pracy własnej.	Rozumie potrzebę samodzielnego uzupełniania i doskonalenia wiedzy oraz umiejętności związanych z programowaniem strukturalnym oraz potrzebę systematycznej pracy własnej.				
	Nie wykazuje aktywnej postawy i chęci współpracy z innymi podczas rozwiązywania trudniejszych zadań oraz nie potrafi pracować indywidualnie, oraz w zespole.	Wykazuje aktywną postawę i chęć współpracy z innymi podczas rozwiązywania trudniejszych zadań oraz potrafi pracować indywidualnie, oraz w zespole.				
	Nie przestrzega zasad etyki i ochrony własności intelektualnej.	Przestrzega zasad etyki i ochrony własności intelektualnej.				

XII.NNE PRZYDATNE INFORMACJE O PRZEDMIOCIE

W systemie nauczania zdalnego e-learning (https://moodle_wpt.kpswjg.pl/login/index.php) publikowane są materiały dydaktyczne dotyczące przedmiotu, w tym wszystkie listy zadań oraz prezentacje z wykładu.