

JĘZYKI PROGRAMOWANIA

Kod modułu: PRG

Rodzaj przedmiotu: kierunkowy; obowiązkowy

Wydział: Informatyki

Kierunek: Informatyka

Poziom studiów: pierwszego stopnia – VI poziom PRK

Profil studiów: praktyczny

Forma studiów: stacjonarna/niestacjonarna

Rok: 1

Semestr: 2

Formy zajęć i liczba godzin:

Forma stacjonarna

wyklady – 30

laboratorium – 35

Forma niestacjonarna

wyklady – 10

laboratorium – 30

Zajęcia prowadzone są w języku polskim.

Liczba punktów ECTS: 4

Osoby prowadzące:

wykład:

laboratorium:

1. Założenia i cele przedmiotu:

Podstawowym celem zajęć jest osiągnięcie przez studentów dobrego poziomu opanowania umiejętności programistycznych, obejmujących umiejętność budowania programów rozwiązujących praktyczne problemy, wykorzystujących pełny zakres możliwości oferowanych przez języki programowania strukturalnego. Dominującą metodyką projektowania i programowania w ramach tego przedmiotu jest metodyka strukturalna.

W ramach zajęć studenci będą łączyć nabywanie umiejętności budowania skutecznych rozwiązań algorytmicznych z efektywną realizacją programową, wykorzystującą potencjał wskazanych języków programowania. W ramach zajęć przewiduje się omówienie podstawowych zagadnień związanych z projektowaniem systemów, testowaniem, lokalizowaniem i usuwaniem błędów z wykorzystaniem debugera. Zajęcia z tego przedmiotu stanowić mają stabilną podbudowę kolejnych zajęć z programowania, realizowanych w następnych semestrach.

Przedmiot Języki Programowania z grupy przedmiotów podstawowych, jest przedmiotem wprowadzającym do przedmiotów Języki Programowania Obiektowego oraz przedmiotów z zakresu specjalizacji Projektowanie systemów informatycznych i programowanie w środowisku internetowym

2. Określenie przedmiotów wprowadzających wraz z wymaganiami wstępnymi:

Języki Programowania - to przedmiot podstawowy, stanowiący kontynuację przedmiotu Podstawy Programowania, wprowadzający do dalszych przedmiotów związanych z programowaniem i inżynierią oprogramowania. Wymogi wstępne dotyczą wiedzy uzyskanej przez studentów w ramach przedmiotu Podstawy Programowania.

3. Opis form zajęć

a) Wykłady

• Treści programowe (tematyka zajęć):

1. Przetwarzanie napisów, reprezentacja, podstawowe operacje, wykorzystanie funkcji z bibliotek standardowych.
2. Zmienne wskaźnikowe, koncepcje, podstawowe operacje, zastosowania.
3. Zmienne wskaźnikowe a przekazywanie parametrów do podprogramów.
4. Zmienne wskaźnikowe a tablice, koncepcja, przykłady zastosowań.
5. Dynamiczny przydział pamięci, zasady zarządzania pamięcią.
6. Struktury w języku, unie, pola bitowe, zastosowania.
7. Pliki, podstawowe operacje, pliki tekstowe i przetwarzanie tekstów.
8. Pliki amorficzne, podstawowe operacje, zastosowania.
9. Tablice, struktury i pliki – koncepcja wspólnego wykorzystania w problemach praktycznych.
10. Rekurencyjne struktury danych, koncepcje, rodzaje, podstawowe operacje.

• Metody dydaktyczne:

Wykład prowadzony metodą tradycyjną z wykorzystaniem rzutnika multimedialnego, obejmować będą również prezentację przykładów z wykorzystaniem wybranego środowiska programistycznego oraz ich dyskusje z aktywnym uczestnictwem studentów. Materiały wspomagające, uzupełniające i związane z pracą własną studenta udostępniane są w wersji elektronicznej.

• Forma i warunki zaliczenia:

Warunkiem zaliczenia całości przedmiotu jest zdanie egzaminu. Forma realizacji egzaminu dostosowywana jest do liczebności grupy studenckiej oraz możliwości wykorzystania wsparcia elektronicznego. W przypadku grup o dużej liczebności przewiduje się formę pisemną, w przypadku grup o niższej liczebności formę sprawdzianu przy stanowisku komputerowym (o ile istnieją takie możliwości infrastrukturalne), również w trybie indywidualnym.

Niezależnie od przyjętej formy realizacji egzaminu prace oceniane są pod kątem stopnia osiągnięcia przez studenta zakładanych efektów kształcenia. W uzasadnionych przypadkach, w porozumieniu z osobami prowadzącymi laboratoria, przewiduje się możliwość zwolnienia z egzaminu, w przypadku uzyskania oceny z zajęć laboratoryjnych, dokumentującej osiągnięcia przez studenta w wysokim stopniu zakładanych efektów kształcenia. Przyjmuje się, że ocena egzaminacyjna będzie nie mniejsza niż ocena laboratoryjna.

• Wykaz literatury podstawowej:

1. Brian W. Kernighan, Dennis M. Ritchie, Język ANSI C, 2004, WNT.
2. Deitel P.J., Deitel H.: Język C. Solidna wiedza w praktyce. Gliwice: Helion, cop. 2020.
3. Stephen Prata, Język C szkoła programowania, Wydanie V, 2006, HELION.
4. Schildt H.: Java. Kompendium programisty. Gliwice: HELION, cop. 2019
5. Niklaus Wirth, Algorytmy + struktury danych = programy wydanie siódme, 2004, WNT.
6. Gaddis T.: Język C++. Gliwice: HELION, cop. 2019

• Wykaz literatury uzupełniającej:

1. Robert Schildt, C Programowanie, 2002, READ ME.
2. Kyle Loudon, Algorytmy w C, 2003, HELION.

b) Ćwiczenia laboratoryjne

• Treści programowe (tematyka zajęć):

1. Tablice znaków, reprezentacja, operacje na łańcuchach znaków, wykorzystanie funkcji z bibliotek standardowych.
2. Ćwiczenia w programowaniu z wykorzystaniem tablic znaków.
3. Typy wskaźnikowe, metody wykorzystania typów wskaźnikowych – ćwiczenia praktyczne.
4. Dynamiczny przydział pamięci, zasady, przykłady, błędy i złe nawyki – ćwiczenia praktyczne.
5. Struktury, unie, pola bitowe, przykłady wykorzystania, ćwiczenia w definiowaniu i ich wykorzystywaniu.
6. Podstawowe operacje na plikach, pliki tekstowe i przetwarzanie tekstów.
7. Operacje blokowe na plikach — pliki amorficzne, zastosowania, przykłady, ćwiczenia.
8. Tworzenie, zarządzanie i wykorzystanie rekurencyjnych struktur danych – ćwiczenia praktyczne.
9. Modularyzacja w językach, koncepcja, zasady, ćwiczenia praktyczne.
10. Tworzenie bibliotek, ćwiczenia praktyczne.

• Metody dydaktyczne:

Każdy blok ćwiczeniowy obejmuje syntezę programu przykładowego, opartego na materiale wykładowym, a pogłębionego o aspekty praktyczne, wprowadzone przez osoby prowadzące ćwiczenia. Po zakończeniu prac nad przykładem, studenci realizują zadania indywidualne pod nadzorem osoby prowadzącej ćwiczenia, konsultując uzyskane rozwiązania. Każdy blok ćwiczeniowy obejmuje zadania dodatkowe, które mogą być realizowane w ramach zajęć własnych studentów oraz mogą być dedykowane studentom wykazującym wyjątkowo dobre opanowanie omawianych zagadnień.

• Forma i warunki zaliczenia:

Ocena jest wypadkową (zgodnie z pkt. 8) oceny sprawdzianów, prac kontrolnych oraz sprawozdania z projektu. Zaliczenie projektu odbywa się na podstawie oceny zrealizowanego programu. Ocenie podlega jego zgodność z założeniami oraz poziom realizacji programowej, adekwatność wybranych rozwiązań, stopień wykorzystania wybranej technologii i narzędzi programowania.

• Wykaz literatury podstawowej:

1. Stephen Prata, Język C szkoła programowania, Wydanie V, 2006, HELION.
2. Niklaus Wirth, Algorytmy + struktury danych = programy wydanie siódme, 2004, WNT.

• Wykaz literatury uzupełniającej:

1. Alfred V. Aho, Jeffrey D. Ullman, Wykłady z informatyki z przykładami w języku C, 2003, HELION.
2. Tłuczek M, Programowanie w języku C. Ćwiczenia praktyczne, 2001, HELION.
3. John Robbins, Debugger usuwanie błędów z programów, 2001, READ ME.

4. Opis sposobu wyznaczania punktów ECTS

a. forma stacjonarna

Forma zajęć	Formy aktywności studenta	Średnia ilość godzin na zrealizowanie aktywności
Wykład	kontakt z nauczycielem	30
	analiza źródeł i materiałów na stronie przedmiotu,	5
	przygotowanie do egzaminu	4
Laboratorium	kontakt z nauczycielem	35
	Realizacja zadań dodatkowych	5

	Projekt indywidualny	15
Konsultacje	Kontakt z nauczycielem	3
Zal./Egzamin	Kontakt z nauczycielem	3
Całkowita ilość godzin aktywności studenta		100
Liczba punktów ECTS dla modułu		4

b. forma niestacjonarna

Forma zajęć	Formy aktywności studenta	Średnia ilość godzin na zrealizowanie aktywności
Wykład	kontakt z nauczycielem	10
	analiza źródeł i materiałów na stronie przedmiotu,	20
	przygotowanie do egzaminu	9
Laboratorium	kontakt z nauczycielem	30
	Realizacja zadań dodatkowych	10
	Projekt indywidualny	15
Konsultacje	Kontakt z nauczycielem	3
Zal./Egzamin	Kontakt z nauczycielem	3
Całkowita ilość godzin aktywności studenta		100
Liczba punktów ECTS dla modułu		4

5. Wskaźniki sumaryczne

a. forma stacjonarna

- liczba godzin dydaktycznych (tzw. kontaktowych) i liczba punktów ECTS na zajęciach wymagających bezpośredniego udziału nauczycieli akademickich
 - Liczba godzin kontaktowych – 71
 - Liczba punktów ECTS – 2,8
- liczba godzin dydaktycznych (tzw. kontaktowych) i liczba punktów ECTS na zajęciach o charakterze praktycznym.
 - Liczba godzin kontaktowych – 35
 - Liczba punktów ECTS – 2,2

b. forma niestacjonarna

- liczba godzin dydaktycznych (tzw. kontaktowych) i liczba punktów ECTS na zajęciach wymagających bezpośredniego udziału nauczycieli akademickich
 - Liczba godzin kontaktowych – 46
 - Liczba punktów ECTS – 1,8
- liczba godzin dydaktycznych (tzw. kontaktowych) i liczba punktów ECTS na zajęciach o charakterze praktycznym.
 - Liczba godzin kontaktowych – 30
 - Liczba punktów ECTS – 2,2

6. Zakładane efekty uczenia się

Numer (Symbol)	Efekty uczenia się dla modułu	Odniesienie do efektów uczenia się dla kierunku
PRG_W_01	... posiada podstawową wiedzę na temat strukturalnego podejścia do programowania, zna złożone struktury danych i rozumie zakres ich stosowalności w programowaniu.	K_W03, K_W04 K_W07 K_W13
PRG_W_02	... posiada wiedzę na temat dekompozycyjnej strategii rozwiązywania problemów złożonych, rozumie metody wykorzystania podprogramów, modułów, rozumie sposoby organizacji programów wielomodułowych.	K_W03 K_W04 K_W07 K_W12
PRG_W_03	... zna podstawowe rodzaje złożonych struktur danych, rozumie metod ich organizacji, wykorzystania, rozumie strategię zarządzania pamięcią w programach komputerowych.	K_W03, K_W04 K_W07 K_W12 K_W13
PRG_U_01	... potrafi programować z wykorzystaniem podprogramów, dokonywać hierarchicznej dekompozycji kodu zgodnie ze strategią programowania strukturalnego.	K_U02 K_U11 K_U23
PRG_U_02	... potrafi dobierać, projektować i implementować złożone struktury danych adekwatnie do specyfiki problemu, skutecznie i efektywnie na nich operować z wykorzystaniem podprogramów.	K_U02 K_U11 K_U17 K_U23
PRG_U_03	... potrafi budować programy wielomodułowe, zarządzać kodem programu wielomodułowego, wykorzystywać środowiska programistyczne dla kompilacji programów wielomodułowych, wykorzystywać systemy kontroli wersji.	K_U02 K_U11 K_U23
PRG_K_01	... potrafi wykorzystywać źródła literaturowe i zasoby sieci Internet do poszerzania wiedzy i samorozwoju zawodowego.	K_K01
PRG_K_02	... posiada kompetencje w zakresie doboru właściwych metod i technik programowania, adekwatnie do specyfiki rozwiązywanego problemu.	K_U23

7. Odniesienie efektów uczenia się do form zajęć i sposób oceny osiągnięcia przez studenta efektów uczenia się

Numer (Symbol)	Forma zajęć			Sposób sprawdzenia osiągnięcia efektu
	wykład	ćwiczenia	laboratorium	
PRG_W_01	x			Egzamin
PRG_W_02	x		x	Egzamin
PRG_W_03	x		x	Egzamin
PRG_U_01			x	Praca kontrolna
PRG_U_02			x	Sprawozdanie z projektu
PRG_U_03			x	Sprawozdanie z projektu
PRG_K_01			x	Dyskusja + obserwacja pracy
PRG_K_02			x	Dyskusja

8. Kryteria uznania osiągnięcia przez studenta efektów uczenia się

Numer (Symbol)	Efekt jest uznawany za osiągnięty gdy:
PRG_W_01	Student poprawnie rozwiąże zadanie egzaminacyjne sprawdzające wiedzę o

	podejściu strukturalnym w programowaniu i wykorzystaniu złożonych struktur danych.
PRG_W_02	Student poprawnie rozwiąże zadanie egzaminacyjne sprawdzające wiedzę na temat budowania programów wielomodułowych.
PRG_W_03	Student poprawnie rozwiąże zadanie egzaminacyjne sprawdzające wiedzę na temat dynamicznie alokowanych złożonych struktur danych..
PRG_U_01	Praca kontrolna zawiera poprawny kod w pełni zgodny ze specyfikacją zadania określonego przez prowadzącego — podprogramy.
PRG_U_02	Sprawozdanie zawiera opis kolejnych etapów realizacji projektu — specyfikację wymagań, założenia projektowe, architekturę systemu, kod programu, opis testów — podprogramy + struktury danych.
PRG_U_03	Sprawozdanie zawiera opis kolejnych etapów realizacji projektu wielomodułowego — specyfikację wymagań, założenia projektowe, architekturę systemu, kod programu, opis testów.
PRG_K_01	... poszukiwał materiałów źródłowych i kreatywnie z ich korzystał, formułując własne rozwiązania postawionych problemów programistycznych, szanuje prawa autorskie.
PRG_K_02	... właściwie dobierał metody i techniki rozwiązania problemów, zadawał merytoryczne pytania i rozumiał otrzymane odpowiedzi, czego wynikiem jest rozwiązanie postawionego zadania.