PODSTAWY PROGRAMOWANIA OBIEKTOWEGO

Kod przedmiotu: PPO

Rodzaj przedmiotu: kierunkowy; obowiązkowy

Wydział: Informatyki

Kierunek: Informatyka

Poziom studiów: pierwszego stopnia – VI poziom PRK

Profil studiów: praktyczny

Forma studiów: stacjonarna/niestacjonarna

Rok: 2

Semestr: 3

Formy zajęć i liczba godzin:

Forma stacjonarna

wykłady – 15

laboratorium – 40

Forma niestacjonarna

wykłady – 10

laboratorium – 25

Zajęcia prowadzone są w języku polskim.

Liczba punktów ECTS: 6

Osoby prowadzące:

wykład:
laboratorium:

---

1. Założenia i cele przedmiotu:

Podstawowym celem zajęć jest osiągnięcie przez studentów dobrego poziomu opanowania podstaw projektowania i programowania obiektowego. Zakłada się wprowadzenie sugestywnych przykładów pozwalających porównać strukturalne podejście do konstruowania programów z podejściem obiektowym. Główny nacisk położony zostanie na prawidłowe zrozumienie podstaw metodyk obiektowych, właściwe zrozumienie i umiejętność praktycznego wykorzystania pojęcia obiektu i klasy, dziedziczenia, związków całość-część, abstrakcji, hermetyzacji oraz polimorfizmu.

2. Określenie przedmiotów wprowadzających wraz z wymaganiami wstępnymi:

Przedmioty wprowadzające to: Podstawy Programowania i Języki Programowania, wymogi wstępne dotyczą osiągniecia efektów kształcenia zakładanych dla modułów Podstawy Programowania i Języki Programowania.

3. Opis form zajęć

a) *Wykłady*

- **Treści programowe:**
  - Koncepcja projektowania i programowania obiektowego. Metodyki strukturalne a obiektowe. Koncepcja obiektu i klasy, abstrakcja, hermetyzacja.
  - Struktura prostych klas w C++ (odniesienie Java i C#), pola, metody, konstruktory, dostęp do składowych obiektów.
  - Dziedziczenie jednobazowe, realizacja C++ (odniesienie Java i C#). Dziedziczenie jako narzędzie implementacji związków gen-spec. Syntaktyczne aspekty dziedziczenia. Dziedziczenie a związki całość-część. Notacje graficzne, przykłady zastosowań.
  - Konstruktory w języku C++. Rodzaje konstruktorów w C++. Destruktory. Konstruktory i destruktory a dziedziczenie i związki całość-cześć.
  - Polimorfizm, wiązanie statyczne i dynamiczne, metody wirtualna. Wykorzystanie metod wirtualnych, polimorfizm a dziedziczenie, abstrakcyjne klasy bazowe.
  - Zarządzanie pamięcią, operatory new i delete, dynamiczna alokacja struktur danych.
  - Mechanizm wyjątków, wyjątki a zarządzanie pamięcią.
- **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 via serwis WWW.
- **Forma i warunki zaliczenia:**
  - Warunkiem zaliczenia całości przedmiotu, w tym wykładu, 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.
- **Wykaz literatury podstawowej:**
  1. Bjarne Stroustrup, Język C++, WNT.
  2. Stroustrup B.: C++. Podróż po języku dla zaawansowanych. Gliwice: HELION, cop. 2019
  3. Stanley B. Lippman, Josee Lajoie, Podstawy języka C++, WNT.
  4. Anggoro W.: C++. Struktury danych i algorytmy. Gliwice: HELION, cop. 2019
- **Wykaz literatury uzupełniającej**
  1. Herb Sutter, Wyjątkowy język C++ 47 łamigłówek zadań, WNT.
  2. Robert Sedgewick, Algorytmy w C++, READ ME.

b) *laboratorium*

- **Treści programowe:**
  - Metodyki strukturalne a obiektowe, koncepcja obiektu i klasy, abstrakcja, hermetyzacja.
  - Tworzenie prostych klas w językach C++ (odniesienie Java i C#) — pola, metody, konstruktory, dostęp do składowych obiektów.
  - Praktyczne aspekty wykorzystania dziedziczenie jednobazowego, realizacja w C++ (odniesienie Java i C#), dziedziczenie a związki całość-część.
  - Konstruktory w języku C++. Rodzaje konstruktorów w C++. Destruktory.
  - Wykorzystanie metod wirtualnych, polimorfizm a dziedziczenie, abstrakcyjne klasy bazowe.
  - Przeciążanie funkcji. Przeciążanie operatorów.
  - Zarządzanie pamięcią, dynamiczna alokacja struktur danych. Obiektowe wersje list, drzew.
  - Wykorzystanie wyjątków, wyjątki a zarządzanie pamięcią.
- **Metody dydaktyczne:**
  - Prezentacje przypadków.
  - Dyskusja.
  - Zespołowe rozwiązywanie problemów, projektów.
  - Indywidualne rozwiązywanie zadań.
- **Forma i warunki zaliczenia:**
  - Sprawdziany cząstkowe.
  - Ocena aktywności studentów podczas zajęć.
  - Ocena projektu programistycznego.
- **Wykaz literatury podstawowej:**
  1. Bruce Eckel, Thinking in C++, edycja polska, HELION.
  2. Gaddis T.: Język C++. Gliwice: HELION, cop. 2019
  3. Grady Booch, James Rumbaugh, Ivar Jacobson , UML przewodnik użytkownika, WNT.
  4. Čukić I.: Programowanie funkcyjne w języku C++. Tworzenie lepszych aplikacji. Gliwice: HELION , cop. 2019
  5. *Weisfeld M.: Myślenie obiektowe w programowaniu. Gliwice: Helion, cop. 2020.*

- **Wykaz literatury uzupełniającej:**
  1. Angelika Langer, Klaus Kreft, C++ biblioteka IOStreams i lokalizacja programów, 2005, HELION.
  2. Joe Dufny, .NET Framework 2.0 zaawansowane programowanie, 2007, HELION.
  3. Michał Śmiałek, Zrozumieć UML 2.0 metody modelowania obiektowego, HELION.

4. Opis sposobu wyznaczania punktów ECTS

a. forma stacjonarna

Forma zajęć	Formy aktywności studenta	Średnia ilość godzin na

		zrealizowanie aktywności
Wykład	Kontakt z nauczycielem	15
	Czytanie wskazanej literatury	20
	Przygotowanie do egzaminu	15
Laboratorium	Kontakt z nauczycielem	40
	Przygotowanie do pracy kontrolnej	20
	Samodzielne rozwiązywanie zadań	25
	Czytanie wskazanej literatury	15

Całkowita ilość godzin aktywności studenta	150
Liczba punktów ECTS dla modułu	6

b. forma niestacjonarna

Forma zajęć	Formy aktywności studenta	Średnia ilość godzin na zrealizowanie aktywności
Wykład	Kontakt z nauczycielem	10
	Czytanie wskazanej literatury	20
	Przygotowanie do egzaminu	20
Laboratorium	Kontakt z nauczycielem	25
	Przygotowanie do pracy kontrolnej	20
	Samodzielne rozwiązywanie zadań	30
	Czytanie wskazanej literatury	25

Całkowita ilość godzin aktywności studenta	150
Liczba punktów ECTS dla modułu	6

5. Wskaźniki sumaryczne

a. forma stacjonarna

a) liczba godzin dydaktycznych (tzw. kontaktowych) i liczba punktów ECTS na zajęciach wymagających bezpośredniego udziału nauczycieli akademickich
   - Liczba godzin kontaktowych – 55
   - Liczba punktów ECTS – 2,2

b) liczba godzin dydaktycznych (tzw. kontaktowych) i liczba punktów ECTS na zajęciach o charakterze praktycznym.
   - Liczba godzin kontaktowych – 40
   - Liczba punktów ECTS – 4,0

b. forma niestacjonarna

a) liczba godzin dydaktycznych (tzw. kontaktowych) i liczba punktów ECTS na zajęciach wymagających bezpośredniego udziału nauczycieli akademickich
   - Liczba godzin kontaktowych – 35
   - Liczba punktów ECTS – 1,4

b) liczba godzin dydaktycznych (tzw. kontaktowych) i liczba punktów ECTS na zajęciach o charakterze praktycznym.
   - Liczba godzin kontaktowych – 25
   - Liczba punktów ECTS – 4,0

5. Zakładane efekty uczenia się

Efekt (Symbol)	Efekty uczenia się dla przedmiotu	Odniesienie do kierunkowych efektów uczenia się
PPO_W1	… zna koncepcję programowania proceduralnego i obiektowego, rozumie podobieństwa i różnice tych podejść, wie jak powinna być zbudowana kompletna klasa, rozumie znaczenie i rolę jej elementów, zna koncepcję dziedziczenia i związków całość-część, rozróżnia poprawnie przypadki ich zastosowania.	K_W04 K_W07 K_W12 K_W13
PPO_W2	… rozumie koncepcję polimorfizmu, zna zasady wykorzystania metod wirtualnych oraz rozumie koncepcje klas abstrakcyjnych. Student zna zasady przeciążania operatorów, rozróżnia ich rodzaje, rozumie przypadki ich stosowania. Rozumie zasady dynamicznego zarządzania pamięcią, obsługę wyjątków, budowanie rekurencyjnych struktur danych.	K_W04 K_W07 K_W12 K_W13
PPO_W3	… rozumie koncepcję programowania sterowanego zdarzeniami w środowiskach GUI, rozróżnia podstawowe komponenty GUI, zna zasady ich wykorzystania, zna zasady tworzenia złożonych okien aplikacji, wie jak programować operacje graficzne i jak wykorzystywać grafikę w aplikacjach GUI	K_W04 K_W07 K_W12 K_W13
PPO_U1	… potrafi programować z wykorzystaniem podprogramów, dokonywać hierarchicznej dekompozycji kodu zgodnie ze strategią programowania strukturalnego. Student potrafi definiować klasy, tworzyć obiekty, budować odpowiednie konstruktory, określać zakresy widoczności pól, wykorzystywać dziedziczenie.	K_U01 K_U02 K_U11 K_U19 K_U24
PPO_U2	… potrafi zaprojektować poprawną hierarchię klas z wykorzystaniem dziedziczenia i związków całość-część, budować klasy abstrakcyjne oraz wykorzystywać polimorfizm, potrafi używać zmiennych wskaźnikowych.	K_U01, K_U02 K_U11, K_U19 K_U24
PPO_U3	… potrafi budować aplikacje GUI, potrafi dobierać odpowiednie dobierać odpowiednie komponenty graficzne i kreatywnie je stosować. Student stosuje programowanie sterowane zdarzeniami, definiuje procedury obsługi zdarzeń. Poprawnie programuje podstawowe operacje graficzne, wykorzystuje elementy multimedialne.	K_U01 K_U02 K_U11 K_U19 K_U24
PPO_K1	… potrafi dobrać metodykę właściwą dla realizacji zadania programistycznego i zastosować ją w praktyce, wykorzystuje podejście obiektowe na etapie analizy, projektu oraz programowania.	K_U03 K_U12
PPO_K2	… posiada kompetencje w zakresie pracy grupowej nad projektem, przejawiające się w umiejętności planowania podzadań, metod ich realizacji oraz zarządzania współdzielonym kodem aplikacji.	K_U03 K_U12 K_K02

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

Efekt (Symbol)	Forma zajęć		Sposób sprawdzenia osiągnięcia efektu
	Wykład	Ćwiczenia	
PPO_W1	v	v	Egzamin i sprawdziany
PPO_W2	v	v	Egzamin i sprawdziany
PPO_W2	v	v	Egzamin i sprawdziany
PPO_U1		v	Sprawdziany przegląd prac projektowych
PPO_U2	v	v	Sprawdziany przegląd prac projektowych
PPO_U3		v	Sprawdziany przegląd prac projektowych
PPO_K1	v	v	Dyskusja + obserwacja pracy
PPO_K2	v	v	Dyskusja

7. Kryteria uznania osiągnięcia przez studenta efektów uczenia się

Efekt (Symbol)	Efekt jest uznawany za osiągnięty, gdy:
PPO_W1	Student poprawnie rozwiąże zadanie egzaminacyjne sprawdzające wiedzę o podejściu obiektowym w programowaniu i wykorzystaniu mechanizmów programowania obiektowego.
PPO_W2	Student poprawnie rozwiąże zadanie egzaminacyjne sprawdzające wiedzę na temat budowania klas.
PPO_W3	Student poprawnie rozwiąże zadanie egzaminacyjne sprawdzające wiedzę na temat wykorzystania dziedziczenia, hermetyzacji i polimorfizmu.
PPO_U1	Sprawdzian kontrolny zawiera poprawny kod w pełni zgodny ze specyfikacją zadania określonego przez prowadzącego — mechanizmy obiektowe.
PPO_U2	Sprawozdanie zawiera opis kolejnych etapów realizacji projektu — specyfikację wymagań, założenia projektowe, architekturę systemu, kod programu, opis testów — mechanizmy obiektowe.
PPO_U3	Sprawozdanie zawiera opis kolejnych etapów realizacji projektu — specyfikację wymagań, założenia projektowe, architekturę systemu, kod programu, opis testów.
PPO_K1	… poszukiwał materiałów źródłowych i kreatywnie z ich korzystał, formułując własne rozwiązania postawionych problemów programistycznych, szanuje prawa autorskie.
PPO_K2	… właściwie dobierał metody i techniki rozwiązania problemów, zadawał merytoryczne pytania i rozumiał otrzymane odpowiedzi, czego wynikiem jest rozwiązanie postawionego zadania.