PROGRAMOWANIE W ŚRODOWISKACH ZINTEGROWANYCH

Kod przedmiotu: PSZ

Rodzaj przedmiotu: kierunkowy, obieralny

Specjalność: Inżynieria Oprogramowania

Wydział: Informatyki

Kierunek: Informatyka

Poziom studiów: pierwszego stopnia – VI poziom PRK

Profil studiów: praktyczny

Forma studiów: stacjonarna/niestacjonarna

Rok: 3

Semestr: 5

Formy zajęć i liczba godzin:

Forma stacjonarna

wykłady – 30

laboratorium – 20

Forma niestacjonarna

wykłady – 20

laboratorium – 15

Zajęcia prowadzone są w języku polskim.

Liczba punktów ECTS: 4

Osoby prowadzące:

wykład:
laboratorium:

---

1. Założenia i cele przedmiotu:

Przedmiot ten poświęcony jest rozwijaniu umiejętności tworzenia oprogramowania z wykorzystaniem najpopularniejszych środowisk programistycznych typu RAD (np. C++ Builder, Visual Studio, QTCreator). Kształcenie ukierunkowane jest na poznanie i praktyczne przećwiczenie budowania programów sterowanych zdarzeniami, wykorzystujących graficzny interfejs użytkownika oraz biblioteki zadaniowo zorientowanych klas i komponentów. Zakłada się, że po zaliczeniu tego przedmiotu student będzie potrafił zrealizować pełnowartościowe, nowoczesne oprogramowanie klasy desktop oraz oprogramowanie klienckie dla systemów w architekturze klient-serwer.

2. Określenie przedmiotów wprowadzających wraz z wymaganiami wstępnymi:

Programowanie w środowiskach zintegrowanych - to przedmiot na specjalizacji Inżyniera Systemów Informatycznych. Stanowi on kontynuację przedmiotów Podstawy Programowania, Języki programowania, Języki Programowania Obiektowego, Wymogi wstępne dotyczą wiedzy uzyskanej

przez studentów właśnie w ramach przedmiotów Podstawy Programowania, Języki programowania i Języki Programowania Obiektowego.

3. Opis form zajęć

a) Wykłady

- **Treści programowe (tematyka zajęć):**

1. Programowanie sterowane przepływem a programowanie sterowane zdarzeniami.
2. Charakterystyka ogólna pakietów RAD
3. Budowa aplikacji w środowiskach RAD.
4. Projektowanie wizualne graficznego interfejsu użytkownika.
5. Programowanie sterowane zdarzeniami w systemach RAD.
6. Charakterystyka wybranych pakietów RAD:
7. Koncepcja organizacji kodu.
8. Elementy typowe dla pakietów RAD:
    - Modelowanie interfejsu użytkownika
    - Biblioteki komponentów graficznych i systemowych.
    - Biblioteki klas kontenerowych, dostępu do baz danych, odwołania do usług systemu operacyjnego.

- **Metody dydaktyczne**:

Wykład prowadzony metodą tradycyjną z wykorzystaniem rzutnika multimedialnego, obejmować będą również prezentację przykładów z wykorzystaniem wybranych środowisk programistycznych 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**:

Literatura podstawowa:

1. Radosław Sokół , Microsoft Visual Studio 2012. Programowanie w C i C++, Helion, 2013.
2. Anggoro W.: C++. Struktury danych i algorytmy. Gliwice: HELION, cop. 2019.
3. Stroustrup B.: C++. Podróż po języku dla zaawansowanych. Gliwice: HELION, cop. 2019.

Literatura uzupełniająca:

1. Owczarek A., Microsoft Visual C++ 2008. Praktyczne przykłady, Helion, 2010.
2. Besta P., Visual Studio 2005. Programowanie z Windows API w języku C++, Helion, 2008.

b) Ćwiczenia laboratoryjne

- **Treści programowe (tematyka zajęć):**

1 Struktura aplikacji w środowisku RAD.
2 Projekt pierwszej aplikacji, projektowanie warstwy wizualnej.
3 Komponenty wizualne, rodzaje, właściwości, funkcje składowe, obsługa zdarzeń.

4 Podstawowe komponenty wizualne, przykłady zastosowań, ćwiczenia.
5 Złożone komponenty wizualne, przykłady zastosowań, ćwiczenia.
6 Dynamiczne tworzenie okien i komponentów wizualnych.
7 Komponenty pojemnikowe i zarządzanie kolekcjami elementów, ćwiczenia.
8 Grafika i programowanie operacji graficznych.
9 Komponenty systemowe, dostępu do baz danych, komponenty sieciowe, przykłady zastosowań, ćwiczenia.
10 Implementacja projektu indywidualnego.

- **Metody dydaktyczne**:

Przedmiot ten realizowany jest w ramach zajęć wykładowych oraz ćwiczeń laboratoryjnych. Wykład stanowi podbudowę ćwiczeń, wprowadzając wszystkie niezbędne zagadnienia. Na tej podstawie realizowane są ćwiczenia, w ramach których zakłada się realizację bloków tematycznych, obejmujących spójne treściowo przykłady oraz zadania do indywidualnego wykonania..

- **Forma i warunki zaliczenia**:

Warunkiem zaliczenia przedmiotu jest uzyskanie pozytywnych ocen z elementów, zgodnie z pkt. 8. Ocena jest wypadkową oceny sprawdzianów, prac kontrolnych oraz sprawozdania z projektu.

- **Wykaz literatury podstawowej**:

1. Čukić I.: Programowanie funkcyjne w języku C++. Tworzenie lepszych aplikacji. Gliwice: HELION, cop. 2019.
2. Bancila M.: Nowoczesny C++. Zbiór praktycznych zadań dla przyszłych ekspertów. Gliwice: Helion, cop. 2019.

- **Wykaz literatury uzupełniająca**:

1. Peter Van Roy, Seif Haridi, Programowanie koncepcje techniki i modele, Helion, 2004.
2. Stasiewicz A., C++ Builder. 20 efektownych programów, Helion, 2002

4. Opis sposobu wyznaczania punktów ECTS

a. forma stacjonarna

Forma zajęć	Formy aktywności studenta	Średnia ilość godzin na zrealizowanie aktywności
Wykład	Kontakt z prowadzącym, dyskusja zagadnień (w tym konsultacje: 5)	30
	Analiza źródeł i strony internetowej przedmiotu, przykłady dodatkowe	20
Ćwiczenia	Kontakt z nauczycielem, dyskusja przykładów	20
	Czytanie wskazanej literatury	5
	Realizacja zadań dodatkowych	15
	Projekt indywidualny	10

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 prowadzącym, dyskusja zagadnień (w tym konsultacje: 5)	20
	Analiza źródeł i strony internetowej przedmiotu, przykłady dodatkowe	30
Ćwiczenia	Kontakt z nauczycielem, dyskusja przykładów	15
	Czytanie wskazanej literatury	5
	Realizacja zadań dodatkowych	15
	Projekt indywidualny	15

Całkowita ilość godzin aktywności studenta	100
Liczba punktów ECTS dla modułu	4

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 – 50
   - Liczba punktów ECTS – 2,0

b) liczba godzin dydaktycznych (tzw. kontaktowych) i liczba punktów ECTS na zajęciach o charakterze praktycznym.
   - Liczba godzin kontaktowych – 20
   - Liczba punktów ECTS – 2,0

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 – 15
   - Liczba punktów ECTS – 2,0

6. Zakładane efekty uczenia się

Numer (Symbol)	Efekty uczenia się dla przedmiotu	Odniesienie do efektów uczenia się dla kierunku
PSZ_W_01	… zna koncepcję obiektowego, rozumie w jaki sposób wykorzystać techniki programowania obiektowego, zna koncepcję dziedziczenia jedno i wielobazowego, związków całość-część, rozróżnia poprawnie przypadki ich zastosowania, rozumie koncepcję polimorfizmu, zna zasady wykorzystania metod wirtualnych oraz rozumie koncepcje klas abstrakcyjnych.	K_W04 K_W07 K_W13
PSZ_W_02	… rozumie koncepcję programowania sterowanego zdarzeniami w środowiskach, rozróżnia podstawowe komponenty GUI oraz zna zasady ich wykorzystania, zna zasady tworzenia, wie jak programować operacje graficzne i jak wykorzystywać grafikę w aplikacjach. Zna zasady tworzenia GUI.	K_W04 K_W07 K_W11 K_W13 K_U21

PSZ_W_03	… zna i rozumie metody wykorzystania zintegrowanych środowisk programistycznych do tworzenia aplikacji klasy desktop i WWW.	K_W04 K_W07 K_W13
PSZ_U_01	… potrafi programować z wykorzystaniem podejścia obiektowego, potrafi stosować dziedziczenie jedno i wielobazowe, związki całość-część, potrafi poprawnie je stosować, potrafi wykorzystywać polimorfizm, stosować metody wirtualne oraz potrafi umiejętnie budować klasy abstrakcyjne i bazujące na nich hierarchie klas.	K_W04 K_W07 K_W13 K_U02 K_U11 K_U24
PSZ_U_02	… 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_W04 K_W07 K_W11 K_W13 K_U02 K_U11 K_U21 K_U24
PSZ_U_03	… potrafi wykorzystywać zintegrowane środowiska programistyczne do projektowania, tworzenia, testowania i uruchamiania aplikacji, optymalizacji jej działania, organizacji pracy grupowej i wersjonowania kodu.	K_W04 K_W07 K_W13 K_U02 K_U03 K_U11 K_U24 K_K02
PSZ_K_01	… 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_U19 K_U24 K_K02
PSZ_K_02	… potrafi tworzyć złożone a ergonomiczne aplikacje GUI dostosowane do wymagań użytkowników. Posiada kompetencje w zakresie współdziałania z użytkownikiem w zakresie ustalania i formułowania wymagań funkcjonalnych i niefunkcjonalnych, w tym wymagań w zakresie warstwy wizualnej aplikacji.	K_W11 K_W14 K_U02 K_U03 K_U09 K_U11 K_U13 K_U23 K_U24 K_K03 K_K05

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

Efekt nr	Forma zajęć			Sposób sprawdzenia osiągnięcia efektu
	wykład	ćwiczenia	laboratorium	
PSZ_W_01	x			Egzamin
PSZ_W_02	x		x	Egzamin
PSZ_W_03	x		x	Egzamin
PSZ_U_01			x	Praca kontrolna
PSZ_U_02			x	Sprawozdanie z projektu
PSZ_U_03			x	Sprawozdanie z projektu
PSZ_K_01			x	Dyskusja + obserwacja pracy
PSZ_K_02			x	Dyskusja

8. Kryteria uznania osiągnięcia przez studenta efektów uczenia się

Efekt nr	Efekt jest uznawany za osiągnięty gdy:
PSZ _W_01	Student poprawnie rozwiąże zadanie egzaminacyjne sprawdzające wiedzę o podejściu obiektowym w programowaniu i wykorzystaniu mechanizmów programowania wizualnego.
PSZ _W_02	Student poprawnie rozwiąże zadanie egzaminacyjne sprawdzające wiedzę na temat budowania klas z wykorzystaniem GUI.
PSZ _W_03	Student poprawnie rozwiąże zadanie egzaminacyjne sprawdzające wiedzę na temat wykorzystania dziedziczenia, hermetyzacji i polimorfizmu w aplikacjach z GUI.
PSZ _U_01	Praca kontrolna zawiera poprawny kod w pełni zgodny ze specyfikacją zadania określonego przez prowadzącego — wykorzystanie elementów GUI.
PSZ _U_02	Sprawozdanie zawiera opis kolejnych etapów realizacji projektu — specyfikację wymagań, założenia projektowe, architekturę systemu, kod programu, opis testów — komponenty wizualne.
PSZ _U_03	Sprawozdanie zawiera opis kolejnych etapów realizacji projektu GUI — specyfikację wymagań, założenia projektowe, architekturę systemu, kod programu, opis testów.
Student w trakcie zajęć laboratoryjnych...	
PSZ _K_01	… poszukiwał materiałów źródłowych i kreatywnie z ich korzystał, formułując własne rozwiązania postawionych problemów programistycznych, szanuje prawa autorskie.
PSZ _K_02	… właściwie dobierał metody i techniki rozwiązania problemów, zadawał merytoryczne pytania i rozumiał otrzymane odpowiedzi, czego wynikiem jest rozwiązanie postawionego zadania.