**Nazwa przedmiotu:**

Programowanie obiektowe

**Koordynator przedmiotu:**

Rajmund Kożuszek

**Status przedmiotu:**

Fakultatywny ograniczonego wyboru

**Poziom kształcenia:**

Studia II stopnia

**Program:**

Informatyka

**Grupa przedmiotów:**

Przedmioty techniczne - podstawowe

**Kod przedmiotu:**

PROI

**Semestr nominalny:**

1 / rok ak. 2021/2022

**Liczba punktów ECTS:**

4

**Liczba godzin pracy studenta związanych z osiągnięciem efektów uczenia się:**

3. liczba godzin kontaktowych – 55 godz., w tym
 obecność na wykładach: 30 godz.,
 obecność na zajęciach laboratoryjnych:15 godz.,
 udział w konsultacjach związanych z problematyką poruszaną na wykładzie//laboratorium/zajęcia wprowadzające do projektu: 2 godz.,
 udział w konsultacjach projektowych: 8 godz.,
4. praca własna studenta – 60 godz., w tym
 rozwiązywanie zadań i problemów przedstawianych na zajęciach laboratoryjnych 20 godz.,
 udział w dyskusji w trakcie wykładu: 1 godz.,
 analiza literatury i materiałów wykładowych związana z przygotowaniem do kolejnych wykładów, zajęć laboratoryjnych, projektu, instalacja oprogramowania: 10 godz.,
 realizacja zadań projektowych, przygotowanie dokumentacji: 25 godz. ,
 przygotowanie do kolokwium: 4 godz.
Łączny nakład pracy studenta wynosi 115 godz., co odpowiada 4 pkt. ECTS.

**Liczba punktów ECTS na zajęciach wymagających bezpośredniego udziału nauczycieli akademickich:**

2 pkt. ECTS, co odpowiada 55 godz. kontaktowym

**Język prowadzenia zajęć:**

polski

**Liczba punktów ECTS, którą student uzyskuje w ramach zajęć o charakterze praktycznym:**

2 pkt. ECTS, co odpowiada 50 godz. zajęć laboratoryjnych i projektowych przygotowaniu do tych zajęć, oraz przygotowanie dokumentacji projektowej

**Formy zajęć i ich wymiar w semestrze:**

|  |  |
| --- | --- |
| Wykład: | 30h |
| Ćwiczenia: | 0h |
| Laboratorium: | 15h |
| Projekt: | 15h |
| Lekcje komputerowe: | 0h |

**Wymagania wstępne:**

Konieczna jest znajomość podstawowych algorytmów i umiejętność projektowania prostych struktur danych, a także umiejętność opracowania algorytmów na podstawie opisu słownego oraz programowania w językach strukturalnych.

**Limit liczby studentów:**

150

**Cel przedmiotu:**

Celem przedmiotu jest zapoznanie studenta z metodyką programowania i modelowania obiektowego oraz rodzajowego z wykorzystaniem mechanizmów języka C++.
Studenci nauczą się tworzyć abstrakcje bytów rzeczywistych w taki sposób aby ukrywać szczegóły techniczne, a jednocześnie pozwalać na łatwą rozbudowę oprogramowania. Podczas wkładu omawiane są konkretne mechanizmy języka, które używane są w trakcie realizacji zadań laboratoryjnych. Projekt, realizowany w zespołach kilkuosobowych, pozwala zdobyć umiejętności projektowania obiektowego aplikacji oraz rozwija umiejętność pracy grupowej.

**Treści kształcenia:**

WYKŁADY:
1. Ewolucja metod programowania: programowanie proceduralne, sterowane danymi, modularne. Zasada ukrywania szczegółów, abstrakcyjne typy danych. Pojęcia podstawowe programowania obiektowego: obiekty, klasy, dziedziczenie, polimorfizm; struktury dziedziczenia. Informacje wprowadzające o programowaniu rodzajowym. Przykład migracji rozwiązania problemu od wersji proceduralnej do obiektowej. (3 godz.)
2. Projektowanie obiektowe: modelowanie obiektowe, podstawy modelowania, diagramy klas UML, omówienie relacji dziedziczenia, agregacji i kompozycji. Narzędzia kontroli wersji i pracy grupowej, określenie zadań, punktów kontrolnych. Podstawy testowania – testy jednostkowe, TDD. Przenośność kodu, refaktoryzacja kodu. Budowa aplikacji C++, kompilacja, konsolidacja, biblioteki, środowiska IDE. (3 godz.)
3. Mechanizmy pomocnicze języka C++: podstawy składni języka (instrukcje warunkowe, pętle, kontenery iteratory - przykłady użycia), przegląd systemu typów, typ referencyjny; podstawy współpracy z WE/WY strumieniowym. Elementy programowania niskopoziomowego – język C. Przeciążenia funkcji i operatorów (polimorfizm statyczny), prototypy funkcji, listy parametrów formalnych, parametry predefiniowane; funkcje rozwijane. Tworzenie obiektów dynamicznych, operatory new, new [], delete, delete [].(4 godz.)
4. Klasy autonomiczne: definiowanie klas samodzielnych, składowe klasy, kontrola dostępu; zaprzyjaźnienia. Konstruktory i destruktory. Różnice pomiędzy klasami z wyróżnikiem class, struct i union. Projektowanie klas autonomicznych, klasa Fraction. Składowe statyczne. Klasy ze zmienną strukturą wewnętrzną, postać kanoniczna. Klasy zagnieżdżone, zastosowanie. (3 godz.)
5. Szablony i podstawy programowania rodzajowego. Definiowanie szablonów klas i funkcji, parametryzacja szablonów. Specjalizacje szablonów. Funkcje składowe szablonowe. Zasady programowania rodzajowego. (3 godz.)
6. Dziedziczenie i polimorfizm: dziedziczenie bezpośrednie, pośrednie, pojedyncze, wielobazowe. Klasy bazowe wirtualne. Dziedziczenie a zawieranie. Polimorfizm dynamiczny, funkcje wirtualne, mechanizm aktywacji funkcji wirtualnych; obiekty polimorficzne. Funkcje wirtualne czyste, klasy abstrakcyjne i interfejsy. Szablony w dziedziczeniu. Model obiektu w C++. Tworzenie i destrukcja obiektów polimorficznych. (4 godz.)
7. Obsługa sytuacji wyjątkowych: mechanizm reagowania na sytuacje wyjątkowe, składnia i semantyka bloku try i bloków obsługi catch, aktywowanie wyjątku przez throw. Funkcje standardowe terminate(), unexpected (), set\_terminate (), set\_unexpected (). Zasady projektowania programów bezpiecznych, metodologia RAII, klasa szablonowa auto\_ptr/unique\_ptr. Wyjątki standardowe. (3 godz.)
8. Polimorfizm i RTTI: Mechanizmy C++ do identyfikacji typów w czasie wykonania programu. Kontrolowane konwersje polimorficzne, zastosowania operatorów dynamic\_cast i typeid; wyjątek bad\_cast. (4 godz.)
9. Biblioteka standardowa C++: Podstawowe akcesoria biblioteki - kontenery, iteratory i algorytmy. Klasyfikacja usług biblioteki standardowej; przegląd kontenerów sekwencyjnych i asocjacyjnych. Wsparcie dla współpracy ze strumieniami i przetwarzania tekstów. Elementy wielowątkowości, rozszerzenia C++14, C++17, proste wzorce projektowe(3 godz.)
LABORATORIUM:
Studenci opracowują programy uwzględniające istotne aspekty programowania obiektowego. Problemy do rozwiązania obejmują: projektowanie klas autonomicznych, przeciążenie funkcji i operatorów, projektowanie klas ze zmienną strukturą obiektów, wykorzystanie reprezentacji grupowej obiektów, definiowanie szablonów klas i funkcji, wykorzystanie dziedziczenia i funkcji wirtualnych, obsługę sytuacji wyjątkowych, współpracę ze strumieniami, projektowanie klas kontenerowych i iteratorów a także wykorzystanie akcesoriów biblioteki standardowej.
PROJEKT:
W ramach projektu 2-3 osobowy zespół ma za zadanie przygotować kompletną aplikację.
Wymagana jest odpowiednia dekompozycja projektu, podział zadań, projekt podziału na klasy, opracowanie mechanizmu komunikacji między klasami. Niezbędne jest korzystanie z narzędzi do wersjonowania kodu i pracy grupowej, oraz przygotowanie testów jednostkowych.

**Metody oceny:**

Realizacja przedmiotu obejmuje następujące formy zajęć:
– wykład prowadzony w wymiarze 2 godz. tygodniowo;
– zajęcia laboratoryjne; w ramach tych zajęć student, korzystając z oprogramowania i sprzętu będzie – pod opieka prowadzącego zajęcia – realizował wskazane zadania związane tematycznie z treścią wykładu w wymiarze 1 godz. tygodniowo;
– zajęcia projektowe; w ramach tych zajęć student – korzystając z konsultacji prowadzącego zajęcia – będzie wykonywał zadanie związane z realizacją prostej aplikacji.
Sprawdzanie założonych efektów kształcenia realizowane jest przez:
– ocenę wiedzy i umiejętności związanych z realizacją zadań laboratoryjnych – ocenę sprawozdań z realizacji zadań;.
– ocenę wiedzy i umiejętności związanych z realizacją zadań projektowych – ocenę prezentacji, dokumentacji i raportów z systemu;
– ocenę wiedzy i umiejętności wykazanych na kolokwiach oraz – w przypadkach szczególnych – na kolokwium ustnym,

**Egzamin:**

nie

**Literatura:**

1. B. Stroustrup: Język C++, WNT, 2002, 2008
2. B. Stroustrup: Język C++. Kompendium wiedzy, WNT, 2014
2. S. B. Lippman: Podstawy języka C++, WNT 2001, 2003.
3. Jerzy Grębosz: Opus magnum C++11. Programowanie w języku C++, Helion 2017
4. Bruce Eckel: Thinking in C++. Edycja polska, tom1, tom2 Helion 2002, 2004.

**Witryna www przedmiotu:**

https://usosweb.usos.pw.edu.pl/kontroler.php?\_action=katalog2/przedmioty/pokazPrzedmiot&prz\_kod=103B-INxxx-ISP-PROI

**Uwagi:**

(-)

## Charakterystyki przedmiotowe