# Politechnika Krakowska im. Tadeusza Kościuszki

# Karta przedmiotu

obowiązuje studentów rozpoczynających studia w roku akademickim 2020/2021

Wydział Informatyki i Telekomunikacji

Kierunek studiów: Informatyka | Profil: Ogólnoakademicki

Forma sudiów: niestacjonarne | Kod kierunku: I

Stopień studiów: I

Specjalności: Brak specjalności

## 1 Informacje o przedmiocie

| | |
|---|---|
| Nazwa przedmiotu | Języki i paradygmaty programowania II |
| Nazwa przedmiotu w języku angielskim | programming languages and paradigms II |
| Kod przedmiotu | WIiT I oIN C12 20/21 |
| Kategoria przedmiotu | Przedmioty kierunkowe |
| Liczba punktów ECTS | 6.00 |
| Semestry | 3 |

## 2 Rodzaj zajęć, liczba godzin w planie studiów

| Semestr | Wykład | Ćwiczenia | Laboratorium | Laboratorium komputerowe | Seminarium | Projekt |
|---|---|---|---|---|---|---|
| 3 | 18 | 0 | 18 | 0 | 0 | 0 |

## 3 Cele przedmiotu

**Cel 1** Poznanie podstawowych paradygmatów programowania,w tym paradygmatu imperatywnego (proceduralnego) oraz obiektowego, funkcyjnego i programowania w logice. Osiągnięcia umiejętności w ocenie przydatności paradygmatów do rozwiązywania rożnego typu problemów; projektowania, implementacji, testowania i debugowania prostych programów obiektowych.

**Cel 2** Podstawy obiektowego podejścia do programowania - abstrakcja danych, enkapsulacja, ukrywanie implementacji. Rozszerzenia w C++ pozwalające na programowanie obiektowe.

**Cel 3** Projektowanie klas, tworzenie programów obiektowych.

# 4 Wymagania wstępne w zakresie wiedzy, umiejętności i innych kompetencji

**1** zaliczenie przedmiotu języki i paradygmaty programowania I

# 5 Efekty kształcenia

**EK1 Wiedza** Student rozumie pojęcie paradygmat programowania. Ma wiedzę ogólną w zakresie języków i paradygmatów programowania, programowania obiektowego. Potrafi wybrać pardygmat właściwy dla problemu, który rozwiązuje oraz zna środowiska (języki) umożliwiające implementacje rozwiązania problemu

**EK2 Wiedza** Student zna składnię C++, rozumie na czym polega obiektowe podejście do programowania. Ma wiedzę nt. algorytmiki, projektowania i programowania obiektowego.

**EK3 Umiejętności** Student potrafi czytać ze zrozumieniem programy napisane w C++, potrafi napisać i uruchomić własny program w C++, który rozwiązuje postawiony przed nim problem.

**EK4 Umiejętności** Umie stworzyć model obiektowy prostych programów w języku C++ przy użyciu klas, dziedziczenia, wielodziedziczenia, przeciążenia funkcji i operatorów, szablonów funkcji i klas, identyfikacji typów, wyjątków.

# 6 Treści programowe

| Laboratorium | | |
|---|---|---|
| Lp | Tematyka zajęć<br>Opis szczegółowy bloków tematycznych | Liczba godzin |
| **L1** | Hierarchia obiektów. Obsługa tablicy obiektów Triangle, zawierajacych obiekt Point (wierzchołek trójkata). Generalna zasada: kontener (tablica) nie powinna zalezec od typu danych. Technika: stosowanie wskazników void * oraz wskazników do funkcji. | 2 |
| **L2** | Hierarchia obiektów. Obsługa tablicy obiektów Triangle, zawierajacych obiekt Point (wierzchołek trójkata). Point wymaga dynamicznego alokowania pamieci dla tablicy współrzednych oraz jej zwolnienia. | 2 |
| **L3** | Tworzenie dla poprzedniego zadania menedzer pamieci, odpowiadajacy za 1 alokowanie, zwolnienie, re-alokowanie w celu przechwytywania ucieczek pamieci. | 2 |
| **L4** | Techniki obsługi macierzy i wektorów. Rozwijanie petli dla algorytmu obliczenia 1 iloczynu skalarnego dwóch wektorów, funkcji pomiaru czasu. | 1 |
| **L5** | Przepisanie jednego obiektu do innego. Powstajace problemy i sposoby rozwiazania. | 1 |
| **L6** | Inicjowanie i niszczenie obiektu. Konstruktorzy i destruktory. | 1 |
| **L7** | Hierarchia klas, wprowadzenie w dziedziczenie. Funkcji inline. | 1 |

| Laboratorium | | |
|---|---|---|
| Lp | Tematyka zajęć<br>Opis szczegółowy bloków tematycznych | Liczba godzin |
| **L8** | Funkcji zaprzyjaźnione do klasy | 1 |
| **L9** | Operatory new, delete. Alokowanie i zwolnienie pamięci. | 1 |
| **L10** | Przekazywanie obiektów do funkcji (przez wartość, przez wskaźnik, przez 1 referencje). Konstruktory kopii. | 1 |
| **L11** | Przeciążenie operatorów binarnych. Przeciążenie operatora przypisania. | 1 |
| **L12** | Funkcje-szablony. | 1 |
| **L13** | Klasa-szablon my_vect. Tworzenie projektu z wieloma plikami. Klasy obsługi 2 komunikatów i interfejsu. Klasy danych. | 1 |
| **L14** | Praca z plikami tekstowymi i binarnymi. Odczyt i zapis dokumentu. | 1 |
| **L15** | Funkcje wirtualne, identyfikacje typu RTTI. | 1 |

| Wykład | | |
|---|---|---|
| Lp | Tematyka zajęć<br>Opis szczegółowy bloków tematycznych | Liczba godzin |
| **W1** | Przegląd paradygmatów i języków. Paradygmat obiektowy. Mechanizmy bezpieczeństwa języków programowania (typizacja itd.) | 2 |
| **W2** | Rozszerzenia względem C (nowe mechanizmy, operatory, klasy) | 2 |
| **W3** | Klasy. Tworzenie i niszczenie obiektów. Konstruktorzy i destruktory. | 2 |
| **W4** | Wprowadzenie w dziedziczenie. Specyfikatory private, protected, public. Funkcje inline. Przypisanie obiektów. Przekazywanie obiektów do funkcji. Funkcje zaprzyjaźnione | 1 |
| **W5** | Przeciążenie funkcji, konstruktorzy kopii, argumenty domyślne. | 1 |
| **W6** | Przeciążenie operatorów. | 1 |
| **W7** | Dynamiczne alokowanie pamięci. Operatory new, delete. | 2 |
| **W8** | Dziedziczenie. Specyfikatory dostępu. Wielodziedziczenie. | 1 |
| **W9** | Polimorfizm dynamiczny. Funkcje wirtualne, abstrakcyjne, klasy abstrakcyjne. Identyfikacje typu na etapie wykonania (RTTI) . | 2 |
| **W10** | Szablony funkcji i klas. | 1 |
| **W11** | Wejście-wyjście w C++. Wprowadzenie do klas kontenerowych / STL | 1 |

| Wykład | | |
|---|---|---|
| Lp | Tematyka zajęć<br>Opis szczegółowy bloków tematycznych | Liczba godzin |
| **W12** | Obsługa wyjątków. Statyczne składowe klasy. Specyfikatory const, volatile. Niepolimorficzne rzutowanie typów. | 2 |

# 7 Narzędzia dydaktyczne

**N1** Wykłady

**N2** Ćwiczenia laboratoryjne

**N3** Konsultacje

# 8 Obciążenie pracą studenta

| Forma aktywności | Średnia liczba godzin na zrealizowanie aktywności |
|---|---|
| **Godziny kontaktowe z nauczycielem akademickim, w tym:** | |
| Godziny wynikające z planu studiów | 36 |
| Konsultacje przedmiotowe | 15 |
| Egzaminy i zaliczenia w sesji | 6 |
| **Godziny bez udziału nauczyciela akademickiego wynikające z nakładu pracy studenta, w tym:** | |
| Przygotowanie się do zajęć, w tym studiowanie zalecanej literatury | 63 |
| Opracowanie wyników | 0 |
| Przygotowanie raportu, projektu, prezentacji, dyskusji | 60 |
| **Sumaryczna liczba godzin dla przedmiotu wynikająca z całego nakładu pracy studenta** | **180** |
| Sumaryczna liczba punktów ECTS dla przedmiotu | 6.00 |

# 9 Sposoby oceny

### Ocena formująca

**F1** Odpowiedź ustna

**F2** Ocena projektu laboratoryjnego

**F3** Kolokwium

### Ocena podsumowująca

**P1** Ocena z egzaminu

### Warunki zaliczenia przedmiotu

**W1** Zaliczenie wszystkich projektów laboratoryjnych

**W2** Zaliczenie egzaminu

## Kryteria oceny

| Efekt kształcenia 1 | |
|---|---|
| Na ocenę 2.0 | Student nie wie, co to jest paradygmat programowania, nie zna podstawowych paradygmatów, nie zna ważnych języków programowania,nie wie jakie paradygmaty można zrealizować w danym języku, nie potrafi dobrać właściwego paradygmatu i języka do rozwiązania postawionego przed nim zadania |
| Na ocenę 3.0 | Student wie, co to jest paradygmat programowania, zna podstawowe paradygmy |
| Na ocenę 3.5 | Student wie, co to jest paradygmat programowania, zna podstawowe paradygmaty, zna ważne języków programowania, |
| Na ocenę 4.0 | Student wie, co to jest paradygmat programowania, zna podstawowe paradygmaty, zna ważne języków programowania , wie jakie paradygmaty można zrealizować w danym języku, |
| Na ocenę 4.5 | Student wie, co to jest paradygmat programowania, zna podstawowe paradygmaty, zna ważne języków programowania , wie jakie paradygmaty można zrealizować w danym języku, potrafi dobrać właściwy paradygmat |
| Na ocenę 5.0 | Student wie, co to jest paradygmat programowania, zna podstawowe paradygmaty, zna ważne języków programowania , wie jakie paradygmaty można zrealizować w danym języku, potrafi dobrać właściwy paradygmat i język do rozwiązania postawionego przed nim zadania |
| Efekt kształcenia 2 | |
| Na ocenę 2.0 | Student nie zna wielu podstawowych elementów składni języka C/C++, nie rozumie na czym polega programowanie obiektowe |
| Na ocenę 3.0 | Student nie zna wielu elemementów składni, ale potrafi je zastąpić innymi; potrafi napisać prosty program w wersji proceduralnej, ma kłopoty ze zrozumieniem działania programów w C/C++ na podstawie analizy kodu źródłowego napisanego przez innych programistów |
| Na ocenę 3.5 | Student opanował składnię C/C++ w stopniu pozwalającym na zwarty zapis kodu prostych programów, potrafi analizowac kod prostych programów, nie potrafi napisac programu obiektowego |
| Na ocenę 4.0 | Student potrafi napisać definicję prostej klasy będącej zawierającej podstawowe elementy opisu i wykonującego podstawowe działania na danych |
| Na ocenę 4.5 | Student zna wszystkie elementy skladni C/C++ , potrafi zaprojekować klasy potrzebne w implementacji programu w wersji obiektowej, |

| | |
|---|---|
| Na ocenę 5.0 | Student zna skladnię C/C++ w stopniu pozwalającym na tworzenie zwartych, szybkich i poprawnie działających programów, potrafi zaprojetować i dokonać implementacji metod klas w podejściu obiektowym. |
| Efekt kształcenia 3 | |
| Na ocenę 2.0 | Student nie potrafi analizować kodu źródłowego napisanego przez innych |
| Na ocenę 3.0 | Student rozumie działanie programu na podstawie lektury kodu źródeł. |
| Na ocenę 3.5 | Potrafi napisać program w C/C++ realizujący przyjęte założenia |
| Na ocenę 4.0 | Potrafi zrobić analizę pozwalającą ustalić wymagania jakie stawiamy programowi |
| Na ocenę 4.5 | Potrafi zastosować kod uzywający dynamicznych struktur danych |
| Na ocenę 5.0 | Potrafi napisać kod optymalny pod względem wydajności i uzasadnić jego optymalność |
| Efekt kształcenia 4 | |
| Na ocenę 2.0 | Student nie potrafi napisać definicji klasy |
| Na ocenę 3.0 | Potrafi napisać definicję prostej klasy bez odniesiena sie do potrzeb programu |
| Na ocenę 3.5 | Student umie zaprojektować klasę dopasowaną do potrzeb programu |
| Na ocenę 4.0 | Potrafi wykorzystać klasy biblioteczne |
| Na ocenę 4.5 | Potrafi zrobić projekt współdziałania wszystkich klas współdziałających w programie |
| Na ocenę 5.0 | Student potrafi zaprojektować klasę modelującą rzeczywiste obiekty i umie stosować w programie klasy napisane przez siebie oraz klasy biblioteczne. |

## 10 Macierz realizacji przedmiotu

| Efekt kształcenia | Odniesienie danego efektu do szczegółowych efektów zdefiniowanych dla programu | Cele przedmiotu | Treści programowe | Narzędzia dydaktyczne | Sposoby oceny |
|---|---|---|---|---|---|
| EK1 | I1_W06 I1_W08 I1_W09 I1_W11 I1_W12 I1_U22 | Cel 1 Cel 2 Cel 3 | L1 L2 L3 L4 L5 L6 L7 L8 L9 L10 L11 L12 L13 L14 L15 W1 W2 W3 W4 W5 W6 W7 W8 W9 W10 W11 W12 | N1 N2 N3 | F1 F2 F3 P1 |
| EK2 | I1_W06 I1_W08 I1_W11 I1_W12 I1_U09 | Cel 1 Cel 2 Cel 3 | L1 L2 L3 L4 L5 L6 L7 L8 L9 L10 L11 L12 L13 L14 L15 W1 W2 W3 W4 W5 W6 W7 W8 W9 W10 W11 W12 | N1 N2 N3 | F1 F2 F3 P1 |
| EK3 | I1_W04 I1_W06 I1_W08 I1_W09 I1_W11 I1_U09 | Cel 1 Cel 2 Cel 3 | L1 L2 L3 L4 L5 L6 L7 L8 L9 L10 L11 L12 L13 L14 L15 W1 W2 W3 W4 W5 W6 W7 W8 W9 W10 W11 W12 | N1 N2 N3 | F1 F2 F3 P1 |
| EK4 | I1_W04 I1_W06 I1_W08 I1_W11 I1_U09 | Cel 1 Cel 2 Cel 3 | L1 L2 L3 L4 L5 L6 L7 L8 L9 L10 L11 L12 L13 L14 L15 W1 W2 W3 W4 W5 W6 W7 W8 W9 W10 W11 W12 | N1 N2 N3 | F1 F2 F3 P1 |

# 11 Wykaz literatury

## Literatura podstawowa

**[1 ] J.Grębosz** — *Symfonia C++ Standard*, Kraków, 2006, Editions 2000

## Literatura uzupełniająca

**[1 ] B.Eckel** — *Thinking in C++*, Gliwice, 2002, Helion

# 12 Informacje o nauczycielach akademickich

## Osoba odpowiedzialna za kartę

dr Adam Marszałek (kontakt: amarszalek@pk.edu.pl)

## Osoby prowadzące przedmiot

**1** dr inż. Piotr Poznański (kontakt: ppoznanski@pk.edu.pl)

# 13 Zatwierdzenie karty przedmiotu do realizacji

(miejscowość, data) (odpowiedzialny za przedmiot) (dziekan)

**Przyjmuję do realizacji** (data i podpisy osób prowadzących przedmiot)

. . . . . .