Politechnika Krakowska im. Tadeusza Kościuszki

Karta przedmiotu

obowiązuje studentów rozpoczynających studia w roku akademickim 2016/2017

Wydział Fizyki, Matematyki i Informatyki

Kierunek studiów: Informatyka

Profil: Ogólnoakademicki

Forma sudiów: stacjonarne

Kod kierunku: I

Stopień studiów: I

Specjalności: Brak specjalności

1 Informacje o przedmiocie

Nazwa przedmiotu	Języki i paradygmaty programowania 2
Nazwa przedmiotu w języku angielskim	Programming Languages and Paradigms
Kod przedmiotu	WFMiI I oIS C4 16/17
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	30	0	30	0	0	0

3 Cele przedmiotu

Cel 1 Osiągnięcia umiejętności w ocenie przydatności paradygmatów i związanych z nimi środowisk programistycznych do rozwiązywania rożnego typu problemów; projektowania, implementacji, testowania i debugowania prostych programów obiektowych.

4 Wymagania wstępne w zakresie wiedzy, umiejętności i innych kompetencji

1 Język C, podstawowa wiedza z analizy matematycznej, algebry liniowej

2 Podstawy C++ (semestr I)

5 Efekty kształcenia

EK1 Wiedza Ma wiedzę ogólną w zakresie języków i paradygmatów programowania, programowania obiektowego i funkcyjnego

EK2 Wiedza Ma szczegółową wiedzę nt. algorytmiki, projektowania i programowania obiektowego.

EK3 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.

EK4 Umiejętności Ma umiejętność formułowania algorytmów i ich programowania. Potrafi tworzyć, jak i debugować programy

6 Treści programowe

Laboratorium		
Lp	Tematyka zajęć Opis szczegółowy bloków tematycznych	Liczba godzin
L1	Wprowadzenie w C++. Pierwsze programy.	2
L2	Inkapsulacja. Pojęcia klasy. Dane i metody klasy	2
L3	Konstrukcja i dekonstrukcja obiektów - konstruktor i destruktor klasy	2
L5	Funkcji zaprzyjaźnione do klasy	2
L6	Dynamiczna alokacja - 0peratory new, delete.	2
L7	Przekazywanie obiektów do funkcji (przez wartość, przez wskaźnik, przez referencje). Konstruktory kopii.	2
L8	Przeciążenie operatorów.	2
L9	Wprowadzenie w dziedziczenie. Wielodziedziczenie, konwersje.	2
L10	Polimorfizm. Funkcje wirtualne, abstrakcyjne. KLasy abstrakcyjne.	4
L11	Szablony funkcji i klas Klasa-szablon my_vect. Tworzenie projektu z wieloma plikami. Klasy obsługi komunikatów i interfejsu. Klasy danych.	4
L12	Wprowadzenie w programowanie funkcyjne - Haskell. Typy i wyrażenia, funkcje, rekurencja, listy, zakresy i kompozycje	2
L13	funkcje wyższego rzędu. Leniwe wartościowanie	2
L14	Klasy, polimorfizm. Monady	2

Wykład		
Lp	Tematyka zajęć Opis szczegółowy bloków tematycznych	Liczba godzin
W1	Oprogramowanie obiektowe: inkapsulacja, polimorfizm i dziedziczenie. 2. Zasieg deklaracji i czas trwania obiektów. Przestrzen nazw.	2
W2	Przeładowania operatorów, przeładowanie operatorów [],(),->, new, delete, >.	4
W3	Wprowadzenie w dziedziczenie. Wielodziedziczenie. Specyfikatory private, protected, public. Funkcji inline. Przypisanie obiektów. Przekazywanie obiektów do funkcji.	2
W4	Dziedziczenie - konwersje.	2
W5	Polimorfizm. Funckje wirtualne, abstrakcyjne. Klasy abstrakcyjne.	2
W6	Obsługa sytuacji wyjątkowych.	2
W7	Szablony funkcji i klas	4
W8	Klasy pojemnikowe	2
W9	Wprowadzenie w programowanie funkcyjne (Haskell)	2
W10	Wyrażenia, typy, wartości logiczne, listy, funkcje rekurencyjne, zakresy i kompozycje.	2
W11	Funkcje wyższego rzędu. Leniwe wartościowanie.	2
W12	Klasy i typy. Funkcje i polimorfizm. Typy rekurencyjne.	2
W13	Monady	2

7 Narzędzia dydaktyczne

N1 Wykłady

N2 Ćwiczenia laboratoryjne

N3 Konsultacje

N4 Prezentacje multimedialne

N5 Inne

8 Obciążenie pracą studenta

Forma aktywności	Średnia liczba godzin na zrealizowanie aktywności
Godziny kontaktowe z nauczycielem akademickim, w tym:	
Godziny wynikające z planu studiów	60
Konsultacje przedmiotowe	10
Egzaminy i zaliczenia w sesji	10
Godziny bez udziału nauczyciela akademickiego wynikające z nakładu pracy studenta, w tym:	
Przygotowanie się do zajęć, w tym studiowanie zalecanej literatury	30
Opracowanie wyników	15
Przygotowanie raportu, projektu, prezentacji, dyskusji	55
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 Kolokwium

F2 Zadanie tablicowe

F3 Projekt indywidualny

F4 Test

Ocena podsumowująca

P1 Egzamin

Ocena aktywności bez udziału nauczyciela

B1 Projekt indywidualny

Kryteria oceny

Efekt kształcenia 1	
Na ocenę 2.0	Student nie zna podstawowych pojęć z zakresu albo inkapsulacji, albo dziedziczenia albo polimorfizmu albo paradygmatów programowania

Na ocenę 3.0	Student zna podstawowe pojęcia z zakresu inkapsulacji, dziedziczenia i polimorfizmu, podstawowe paradygmaty programowania, podstawowe pojęcia programowania funkcyjnego.
Na ocenę 3.5	Student zna podstawowe pojęcia z zakresu inkapsulacji, dziedziczenia i polimorfizmu, podstawowe paradygmaty programowania, podstawowe pojęcia programowania funkcyjnego, wykazuje znajomość tworzenia prostych klas C++ oraz przeciążenia funkcji i operatorów, prostych programów funkcyjnych.
Na ocenę 4.0	Student zna podstawowe pojęcia z zakresu inkapsulacji, dziedziczenia i polimorfizmu, podstawowe paradygmaty programowania, wykazuje znajomość tworzenia klasy C++, zawierającej wskaźniki do obiektów prostych typów, przeciążenia funkcji i operatorów, tworzenia klas pochodnych na podstawie wielodziedziczenia, prostych programów funkcyjnych i wykorzytywania w nich klas.
Na ocenę 4.5	Student zna podstawowe pojęcia z zakresu inkapsulacji, dziedziczenia i polimorfizmu, podstawowe paradygmaty programowania, wykazuje znajomość tworzenia klasy C++, zawierającej wskaźniki do obiektów prostych typów, przeciążenia funkcji i operatory, tworzenia klas pochodnych na podstawie wielodziedziczenia, funkcji wirtualnych oraz klas abstrakcyjnych,prostych programów funkcyjnych i wykorzytywania w nich klas i monad.
Na ocenę 5.0	Student zna podstawowe pojęcia z zakresu inkapsulacji, dziedziczenia i polimorfizmu, paradygmatów programowania, wykazuje znajomość tworzenia klasy C++, zawierającej wskaźniki do obiektów prostych typów oraz typów innych klas i obiektów STL, przeciążenia funkcji i operatory, tworzenia klas pochodnych na podstawie wielodziedziczenia, funkcji wirtualnych oraz klas abstrakcyjnych, prostych programów funkcyjnych i wykorzytywania w nich klas i monad, potrafi przekonująco wytłumaczyć, dla czego w podanym przypadku trzeba postępować tak, a nie inaczej, różnice pomiędzy poznanymi językami
Efekt kształcenia 2	
Na ocenę 2.0	Student nie zna albo podstawowych pojęć z zakresu szablonów funkcji i klas, albo formatowanego We/Wy.
Na ocenę 3.0	Student zna podstawowe pojęcia z zakresu szablonów funkcji i klas i formatowanego We/Wy.
Na ocenę 3.5	Student zna podstawowe pojęcia z zakresu szablonów funkcji i klas, formatowanego We/Wy, wykazuje podstawowe znajomości z klasami-kontenerami STL oraz z algorytmami STL.
Na ocenę 4.0	Student zna podstawowe pojęcia z zakresu szablonów funkcji i klas, zidentyfikowania typów w czasie wykonania programu, formatowanego oraz nieformatowanego We/Wy, wykazuje podstawowe znajomości z klasami-kontenerami STL oraz z algorytmami STL.
Na ocenę 4.5	Student zna podstawowe pojęcia z zakresu szablonów funkcji i klas, zidentyfikowania typów w czasie wykonania programu, formatowanego oraz nieformatowanego We/Wy, wykazuje podstawowe znajomości z klasami-kontenerami STL oraz z algorytmami STL, wyjątkami.

Na ocenę 5.0	Student doskonałe zna podstawowe pojęcia z zakresu szablonów funkcji i klas, zidentyfikowania typów w czasie wykonania programu, formatowanego oraz nieformatowanego We/Wy, wykazuje podstawowe znajomości z klasami-kontenerami np. STL oraz z algorytmami STL, wyjątkami, potrafi przekonująco wytłumaczyć, dla czego w podanym przypadku trzeba postępować tak, a nie inaczej.
Efekt kształcenia 3	
Na ocenę 2.0	Student nie umie tworzyć prostych programów przy użyciu klas, prostego dziedziczenia, przeciążenia funkcji i operatorów.
Na ocenę 3.0	Student umie tworzyć proste programy przy użyciu klas, prostego dziedziczenia, przeciążenia funkcji.
Na ocenę 3.5	Student umie tworzyć proste programy przy użyciu klas, prostego dziedziczenia, przeciążenia funkcji i operatorów, przekazywać obiekty do funkcji przez wartość, przez wskaźnik i przez referencje.
Na ocenę 4.0	Student umie tworzyć proste programy przy użyciu klas oraz klas, zawierających wskaźniki do obiektów prostych typów, używać wielodziedziczenie, przeciążenie funkcji i operatorów, tworzyć konstruktory kopiujące, przekazywać obiekty do funkcji przez wartość, przez wskaźnik i przez referencje.
Na ocenę 4.5	Student umie tworzyć proste programy, składające się z kilku plików, przy użyciu klas prostych oraz klas, zawierających wskaźniki do obiektów prostych typów, używać wielodziedziczenie, przeciążenie funkcji i operatorów, tworzyć konstruktory kopiujące, przekazywać obiekty do funkcji przez wartość, przez wskaźnik i przez referencje, tworzyć funkcje wirtualne, destruktory wirtualne, klasy abstrakcyjne, stosować polimorfizm dynamiczny.
Na ocenę 5.0	Student umie tworzyć proste programy, składające się z kilku plików, przy użyciu klas prostych oraz klas, zawierających wskaźniki do obiektów prostych typów oraz typów innych klas i obiektów STL, używać wielodziedziczenie, przeciążenie funkcji i operatorów, tworzyć konstruktory kopiujące, przekazywać obiekty do funkcji przez wartość, przez wskaźnik i przez referencje, tworzyć funkcje wirtualne, destruktory wirtualne, klasy abstrakcyjne, stosować polimorfizm dynamiczny, potrafi przekonująco wytłumaczyć, dla czego w podanym przypadku trzeba postępować tak, a nie inaczej.
Efekt kształcenia 4	
Na ocenę 2.0	Student nie umie tworzyć szablony funkcji i klas albo nie umie stosować formatowane We/Wy.
Na ocenę 3.0	Student umie tworzyć szablony funkcji i klas, stosować formatowane We/Wy.
Na ocenę 3.5	Student umie tworzyć szablony funkcji i klas, stosować formatowane oraz nieformatowane We/Wy, używać klasy-kontenery oraz algorytmy STL do obiektów prostych typów.
Na ocenę 4.0	Student umie tworzyć szablony funkcji i klas, stosować formatowane We/Wy, używać klasy-kontenery oraz algorytmy STL do obiektów prostych typów, umie używać operatory zidentyfikowania typów RTTI, stosować obiekty klas polimorficznych.

Na ocenę 4.5	Student umie tworzyć szablony funkcji i klas, stosować formatowane We/Wy, używać klasy-kontenery oraz algorytmy STL do obiektów klas, umie używać operatory zidentyfikowania typów RTTI, stosować obiekty klas polimorficznych, stosować wyjątki.
Na ocenę 5.0	Student umie tworzyć szablony funkcji i klas, stosować formatowane We/Wy, używać klasy-kontenery oraz algorytmy STL do obiektów klas, umie używać operatory zidentyfikowania typów RTTI, stosować obiekty klas polimorficznych, stosować wyjątki, potrafi przekonująco wytłumaczyć, dla czego w podanym przypadku trzeba postępować tak, a nie inaczej.

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_W10 I1_W12 I1_W13 I1_U03 I1_U07 I1_U08 I1_U15 I1_U22 I1_U23 I1_U24	Cel 1	W1 W2 W3 W4 W5 W6 W7	N1 N2 N3 N4	F1 F2 F4 P1
EK2	I1_W04 I1_W06 I1_W08 I1_W10 I1_W12 I1_W13 I1_U03 I1_U07 I1_U08 I1_U15	Cel 1	W7 W8 W9 W10 W11 W12 W13	N1 N2 N3 N4 N5	F1 F2 F3 F4 P1
EK3	I1_W06 I1_W08 I1_W10	Cel 1	L1 L2 L3 L5 L6 L7 L8	N1 N2 N3 N5	F1 F2 F4 P1
EK4	I1_W04 I1_W06 I1_W08 I1_W10 I1_W12 I1_W13 I1_U01 I1_U05 I1_U14	Cel 1	L9 L10 L11 L12 L13 L14	N1 N2 N3 N5	F1 F2 F3 F4 P1

11 Wykaz literatury

Literatura podstawowa

[1] B. Stroustrup — *Jezyk C++*, Warszawa, 2000, WNT

[2] B. Eckel — *Thinking in C++*, Gliwice, 2014, Helion

[3] H. Schildt — *Programowanie C++*, Warszawa, 2014, WNT

[4] B. Kernighan, D. Ritchie — *Język ANSI C*, Warszawa, 2002, WNT

Literatura uzupełniająca

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

[2] S. Prata — *Język C. Szkoła programowania*, Gliwice, 2006, Helion

[3] S. B. Lippman, J. Lajoie — *Podstawy jezyka C++*, Warszawa, 2003, WNT

12 Informacje o nauczycielach akademickich

Osoba odpowiedzialna za kartę

dr inż. Piotr Poznański (kontakt: `poznan@pk.edu.pl`)

Osoby prowadzące przedmiot

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

2 mgr inż. Marek Kosiorowski (kontakt: `mkosiorowski@pk.edu.pl`)

3 mgr inż. Dawid Barnaś (kontakt: `dbarnas@pk.edu.pl`)

13 Zatwierdzenie karty przedmiotu do realizacji

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

Przyjmuję do realizacji (data i podpisy osób prowadzących przedmiot)

. . . . . .

. . . . . .

. . . . . .