Politechnika Krakowska im. Tadeusza Kościuszki

Karta przedmiotu

obowiązuje studentów rozpoczynających studia w roku akademickim 2017/2018

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
Nazwa przedmiotu w języku angielskim	
Kod przedmiotu	WFMiI I oIS C4 17/18
Kategoria przedmiotu	Przedmioty kierunkowe
Liczba punktów ECTS	10.00
Semestry	2 3

2 Rodzaj zajęć, liczba godzin w planie studiów

Semestr	Wykład	Ćwiczenia	Laboratorium	Laboratorium komputerowe	Seminarium	Projekt
2	30	0	30	0	0	0
3	30	0	30	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 Poznanie składni języka C jako przykładu języka strukturalnego pozwalającego na programowanie proceduralne

Cel 3 Podstawy obiektowego podejścia do programowania - abstrakcja danych, enkapsulacja, ukrywanie implementacji. Rozszerzenia w C++ pozwalające na programowanie obiektowe.

Cel 4 Projektowanie klas, tworzenie programów obiektowych

4 Wymagania wstępne w zakresie wiedzy, umiejętności i innych kompetencji

1 zaliczenie przedmiotu: Wstęp do programowania

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 i C++, rozumie na czym polega obiektowe podejście do programowania. Ma szczegółową wiedzę nt. algorytmiki, projektowania i programowania obiektowego.

EK3 Umiejętności Student potrafi czytać ze zrozumieniem programy napisane w C i C++, potrafi napisać i uruchomić własny program w C i 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

Wykład		
Lp	Tematyka zajęć Opis szczegółowy bloków tematycznych	Liczba godzin
W1	Pojęcie paradygmat programowania, przegląd najważniejszych paradygmatów. Formalna definicja języka programowania. Przegląd języków i paradymatów, które można w nich realizować	2
W2	Historia języka C, jego ogólna chrakterystyka. Zasady przetwarzania programu. Instrukcje we/wy, słowa kluczowe języka, Przykłady prostych programów.	2
W3	Typy w C. Problem unkalności n - czas zycia i zakres ważności nazwy. Pojęcie przestrzeni nazw.	2
W4	Operatory w C; sposób działania, ilość argumentów, priorytet, typ łączności. Specyfikatory dostępu do pamięci. Konwersje typów - jawne i niejawne	2
W5	Funkcje i struktura programu. Definicja i wywołanie funkcji, przekazywanie argumentów. Zmienne globalne i lokalne. Sposób podziału kodu źródłowego na pliki. Domyślne argumenty funkcji, funkcje inline. Rekursja.	2

Wykład		
Lp	Tematyka zajęć Opis szczegółowy bloków tematycznych	Liczba godzin
W6	Tablice definicje, inicjalizacja, dostęp do składowych. Tablice jak argumenty funkcji. Tablice wielowymiarowe, tablice znakowe, C-stringi.	2
W7	Wskaźniki i ich zastosowanie do przekazywania argumentów funkcji, usprawnienia pracy z tablicami, dynamicznej alokacji pamięci, uzyskania dostępu do wybranych komórek pamięci.	3
W8	8.Dynamiczna alokacja tablic dwuwymiarowych. Wskaźniki do funkcji. Zasady tworzenia i interpretacji deklaracji złożonych. Przekazywanie argumentów programu (funkcji main()) z wiersza poleceń	3
W9	Przeładowanie (przeciążenie) nazw funkcji (2 godz). Praca z plikami zewnętrznymi	2
W10	Struktury w C. Dynamicze struktury danych (listy, drzewa). Unie, pola bitowe.	2
W11	Problemy programowania proceduralnego i ich źróda. Obiekt - abstrakcja rzeczywistego obiektu. Klasa - nowy typ danych - opis stanu i zachowań.	2
W12	Definicja klasy. Pojęcie danych i funkcji składowych. Enkapsulacja, etykiety dostępu. Sposoby definiowania funkcji składowych	2
W13	Używanie obiektów (danych i funkcji składowych). Projektowanie klas. Inicjalizacja obiektów klas - konstruktory. Przeładowanie konstruktorów	2
W14	Likwiadacja obiektu klasy. Destruktory. Składowa statyczna klasy. Metody statyczne, const i volatile. Funkcje i klasy zaprzyjaźnione	2
W15	Oprogramowanie obiektowe: inkapsulacja, polimorfizm i dziedziczenie. 2. Zasieg deklaracji i czas trwania obiektów. Przestrzen nazw.	2
W16	Klasy. Tworzenie i niszczenie obiektów. Konstruktorzy i destruktory.	2
W17	Wprowadzenie w dziedziczenie. Specyfikatory private, protected, public. Funkcji inline. Przypisanie obiektów. Przekazywanie obiektów do funkcji.	2
W18	Zwracanie obiektu przez funkcje. Funkcje zaprzyjaźnione.	2
W19	Przeciążenie funkcji, konstruktorzy kopii, argumenty domyślne.	2
W20	Przeciążenie operatorów.	2
W21	Dynamiczne alokowanie pamięci. Operatory new, delete.	2
W22	Dziedziczenie. Specyfikatory dostępu. Wielodziedziczenie. Klasy wirtualne.	2
W23	Polimorfizm dynamiczny. Funkcje wirtualne, abstrakcyjne, klasy abstrakcyjne.	2
W24	Szablony funkcji i klas.	2
W25	Identyfikacje typu na etapie wykonania (RTTI) .	3

Wykład		
Lp	Tematyka zajęć Opis szczegółowy bloków tematycznych	Liczba godzin
W26	Wejście-wyjście w C++.	3
W27	Obsługa wyjątków. Statyczne składowe klasy. Specyfikatory const, volatile. Niepolimorficzne rzutowanie typów.	2
W28	Wprowadzenie w STL	2

Laboratorium		
Lp	Tematyka zajęć Opis szczegółowy bloków tematycznych	Liczba godzin
L1	Powtórzenie wiadomości o podstawowych instrukcjach sterujących, instrukcjach we/wy oraz operatorach w języku C.	2
L2	Proste programy wyrabiające biegłość w stosowaniu instrukcji formatowanego we/wy, pętli i instrukcji warunkowych	2
L3	Programy przetwarzające znaki i łańcuchy znaków (zliczanie ilości linii, słów, znaków, białych znaków, cyfr, liter). Odwracanie łąńcucha znaków. konwersja liczba całkowita - ciąg znaków	2
L4	Operatory bitowe. Napisanie funkcji, które: wyświeltają obraz bitowy słowa, ustawiają, kasują lub negują grupę bitów, realizują 32-bitowy bufor cykliczny z przesunięciem w lewo	2
L5	Tablice dwuwymiarowe; definicja statyczna i dynamiczna program wczytujący tablicę (macierz) 4 x 4 liczb i obliczający wyznacznik, wykorzystujący statyczną i dynamiczną deklarację tablic.	2
L6	Wskaźniki do funkcji: program obliczający całkę oznaczoną z gładkiej funkcji jednej zmiennej rzeczywistej.	2
L7	Rekursja, wskaźniki, dynamiczna alokacja pamięci: program wykorzystujący listę jednokierunkową nazwisk, pozwalający na sortowane, drukowanie wskazanego kawałka listy, wybranie elementu spełniającego postawione wymagania	2
L8	Klasy w C++ : zaprojektowanie klasy opisującej dane studenta. Klasa ma być wyposażone w metody pozwalające na sortowanie wg. dowolnego pola, drukowanie listy, wyznaczenie najmłodszego/najstarszego studenta w grupie. Dynamicznie utworzone obiekty tej klasy zestawiamy w listę jednokierunkową.	4
L9	Projektowanie klas: należy zaprojektowanić i zaimplementować klasę, która ma służyć do przechowywania danych bazy PESEL mieszkańców miasta. Wymagane dane: imie, nazwisko, data i miejsce urodzenia (śmierci), numer PESEL, dane rodziców. Należy wyposażyć klasę w metody pozwalające na odczyt/wydruk danych, uzupełnienie czy tez edycję danych.	4

Laboratorium		
Lp	Tematyka zajęć Opis szczegółowy bloków tematycznych	Liczba godzin
L10	Struktury drzewiaste: należy zaprojektować klasę do opisu drzewa genealogicznego. Metody klasy mają pozwolić na: dodawanie osoby do drzewa, edycję danych, zapis/odczyt do/z pliku tekstowego.	4
L11	Kolokwium 1	2
L12	Kolokwium 2	2
L13	Wprowadzenie w C++. Pierwsze programy.	2
L14	Inkapsulacja. Pojęcia klasy. Dane i metody klasy	2
L15	Inicjowanie i niszczenie obiektu. Konstruktorzy i destruktory.	2
L16	Hierarchia klas, wprowadzenie w dziedziczenie. Funkcji inline.	2
L17	Funkcji zaprzyjaźnione do klasy	2
L18	Operatory new, delete. Alokowanie i zwolnienie pamięci.	2
L19	Przekazywanie obiektów do funkcji (przez wartość, przez wskaźnik, przez referencje). Konstruktory kopii.	2
L20	Przeciążenie operatorów binarnych. Przeciążenie operatora przypisania.	2
L21	Funkcje-szablony.	2
L22	Klasa-szablon my_vect. Tworzenie projektu z wieloma plikami. Klasy obsługi komunikatów i interfejsu. Klasy danych.	2
L23	Praca z plikami tekstowymi i binarnymi. Odczyt i zapis dokumentu.	2
L24	Funkcje wirtualne, identyfikacje typu RTTI.	4
L25	Kolokwium 3	2
L26	Kolokwium 4	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	120
Konsultacje przedmiotowe	5
Egzaminy i zaliczenia w sesji	15
Godziny bez udziału nauczyciela akademickiego wynikające z nakładu pracy studenta, w tym:	
Przygotowanie się do zajęć, w tym studiowanie zalecanej literatury	60
Opracowanie wyników	0
Przygotowanie raportu, projektu, prezentacji, dyskusji	30
Sumaryczna liczba godzin dla przedmiotu wynikająca z całego nakładu pracy studenta	**230**
Sumaryczna liczba punktów ECTS dla przedmiotu	10.00

9 Sposoby oceny

Ocena formująca

F1 Odpowiedź ustna

F2 Projekt indywidualny

F3 Kolokwium

Ocena podsumowująca

P1 Średnia ważona ocen formujących

Warunki zaliczenia przedmiotu

W1 Zaliczenie wszystkich projektów indywidualnych

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 zna paradygmat imperatywny programowania i potrafi go zaimplementować w proceduralnym języku strukturalnym
Na ocenę 3.5	Student - oprócz pradygmatu imperatywnego - zna obiektowy paradygmat programowania, wie jakich języków powinien użyć do wykonania implementacji
Na ocenę 4.0	Student potrafi - oprócz programowania imperatywnego i obiektowego - programować w logice i zna języki umożliwiające takie podejście do programowania.
Na ocenę 4.5	Student zna i potrafi zastosować wszystkie podstawowe paradygmaty programowania
Na ocenę 5.0	Student zna i potrafi zastosować wszystkie podstawowe paradygmaty programowania, potrafi dokonać analizy i wybrać właściwy paradygmat i język programowania do jego implementacji oraz potrafi swój wybór przekonywująco uzasadnić/
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		Cel 1 Cel 2 Cel 3 Cel 4	W1 W2 W11 W14 W15 W16 W17 W18 W19 W20 W21 W22 W23 W24 W25 W26 W27 W28 L1 L2 L3 L4 L5 L6 L7 L8 L9 L10 L11 L12 L13 L14 L15 L16 L17 L18 L19 L20 L21 L22 L23 L24 L25 L26	N1 N3	F2

Efekt kształcenia	Odniesienie danego efektu do szczegółowych efektów zdefiniowanych dla programu	Cele przedmiotu	Treści programowe	Narzędzia dydaktyczne	Sposoby oceny
EK2		Cel 2 Cel 3 Cel 4	W1 W2 W3 W4 W5 W6 W7 W8 W9 W10 W11 W12 W13 W14 W15 W16 W17 W18 W19 W20 W21 W22 W23 W24 W25 W26 W27 W28 L1 L2 L3 L4 L5 L6 L7 L8 L9 L10 L11 L12 L13 L14 L15 L16 L17 L18 L19 L20 L21 L22 L23 L24 L25 L26	N1 N2 N3	F1 F3
EK3		Cel 2	W2 W3 W4 W5 W6 W7 W8 W9 W10 W11 W12 W13 W14 W15 W16 W17 W18 W19 W20 W21 W22 W23 W24 W25 W26 W27 W28 L1 L2 L3 L4 L5 L6 L7 L8 L9 L10 L11 L12 L13 L14 L15 L16 L17 L18 L19 L20 L21 L22 L23 L24 L25 L26	N1 N2 N3	F1 F3
EK4		Cel 3 Cel 4	W1 W2 W3 W11 W12 W13 W14 W15 W16 W17 W18 W19 W20 W21 W22 W23 W24 W25 W26 W27 W28 L8 L9 L10 L11 L12 L13 L14 L15 L16 L17 L18 L19 L20 L21 L22 L23 L24 L25 L26	N1 N2 N3	F1

11 Wykaz literatury

Literatura podstawowa

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

[2] J.Gręboszc — *Symfonia C++ Standard*, Kraków, 2006, Editions 2000

Literatura uzupełniająca

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

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

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: `poznan@pk.edu.pl`)

2 mgr inż. Marek Kosiorowski (kontakt:)

13 Zatwierdzenie karty przedmiotu do realizacji

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

Przyjmuję do realizacji (data i podpisy osób prowadzących przedmiot)

. . . . . .

. . . . . .