KARTA OPISU PRZEDMIOTU - SYLABUS

Nazwa przedmiotu

Algorytmy i struktury danych 2

Przedmiot

Kierunek studiów
Teleinformatyka

Rok/semestr
2/3

Studia w zakresie (specjalność)

Profil studiów
ogólnoakademicki

Poziom studiów
pierwszy

Język oferowanego przedmiotu
polski

Forma studiów
stacjonarne

Wymagalność
obowiązkowy

Liczba godzin

Wykład
30

Laboratoria
30

Inne (np. online)

Ćwiczenia
0

Projekty/seminaria
0/0

Liczba punktów ECTS

5

Wykładowcy

Odpowiedzialny za przedmiot/wykładowca:

prof. dr hab. inż. Jerzy Tyszer
e-mail: jerzy.tyszer@put.poznan.pl
tel. +48 61 665 3814
Instytut Radiokomunikacji
Wydział Informatyki i Telekomunikacji

Odpowiedzialny za przedmiot/wykładowca:

Wymagania wstępne

Student powinien posiadać podstawową wiedzę z zakresu matematyki dyskretnej, kombinatoryki i rachunku prawdopodobieństwa. Powinien posiadać umiejętność wykonywania obliczeń za pomocą aparatu matematycznego z zakresu analizy matematycznej i rachunku prawdopodobieństwa oraz pozyskiwania informacji ze wskazanych źródeł.

Cel przedmiotu

Przekazanie studentom podstawowej wiedzy dotyczącej podstawowych algorytmów matematyki dyskretnej i metod numerycznych, statycznych i dynamicznych struktur danych oraz zasad programowania obiektowego w języku C++.

Przedmiotowe efekty uczenia się

Wiedza

Student ma podstawową wiedzę teoretyczną i praktyczną w zakresie programowania w językach C i C++, z naciskiem na projektowanie programów poprawnie zbudowanych, projektowanie programów złożonych oraz wykorzystywanie oprogramowania bibliotecznego. Sudent ma także wiedzę o podstawowych algorytmach i strukturach danych wykorzystywanych w codziennej praktyce programisty.

Umiejętności

Przy projektowaniu oprogramowania student potrafi przeprowadzić analizę problemu z punktu widzenia postępowania algorytmicznego stosując kryteria złożoności obliczeniowej, skalowalności zastosowanych rozwiązań, oraz adekwatności przyjętych metod. Potrafi także krytycznie analizować dostępne oprogramowanie biblioteczne pod kątem zastosowania w realizowanym projekcie oraz zaproponować zasady współpracy w konfiguracji zbiorowego programisty.

Kompetencje społeczne

Student ma świadomość możliwości i ograniczeń współczesnej informatyki przy jednoczesnym otwarciu na możliwość zastosowań w nowych dziedzinach życia codziennego, gospodarki, techniki i nauki.

Metody weryfikacji efektów uczenia się i kryteria oceny

Efekty uczenia się przedstawione wyżej weryfikowane są w następujący sposób:

Wiedza nabyta w ramach wykładu jest weryfikowana poprzez egzamin pisemny składający się z kilku zadań problemowych obejmujących treść wykładu i/lub rozwiązanie testu wyboru obejmującego około 15 pytań. Czas trwania egzaminu: 2h 30 min. Umiejętności nabyte w ramach ćwiczeń laboratoryjnych są weryfikowane na podstawie dwóch kolokwiów pisemnych obejmujących zadania wykonywane w ramach zajęć. Ponadto umiejętności nabyte w ramach zajęć są na bieżąco oceniane na podstawie ćwiczeń projektowych i odpowiedzi ustnych. Ocenie podlega także aktywność na zajęciach.

Treści programowe

Wykład: Programowanie obiektowe. Klasy w C++. Konstruktor i destruktor. Klasa liczb zespolonych. Konstruktor jako konwerter. Funkcje zaprzyjaźnione. Przeładowanie operatorów. Dziedziczenie. Hierarchia klas. Wzorce klas. Wskaźniki, operatory new i delete. Dynamiczne kreowanie obiektów. Wskaźniki do obiektów złożonych. Polimorfizm. Funkcje wirtualne. Klasy abstrakcyjne. Konstruktor kopiujący. Pojemniki. Lista liniowa jednokierunkowa, operacje wstawiania i usuwania. Złożone operacje na liście. Iteratory i ich wykorzystanie. Listy uporządkowane, dwukierunkowe, cykliczne, kolejki. Odwrotna notacja polska. Drzewa binarne, podstawowe operacje, metody przeszukiwania. Drzewa poszukiwań binarnych – operacje wstawiania i usuwania. Drzewa AVL – lokalne równoważenie drzewa, operacje obracania. Wstawianie i usuwanie z drzew AVL. Reprezentacja grafów. Przeszukiwanie grafu w głąb, cykl Eulera, grafy Eulera, algorytm znajdywania cyklu Eulera. Cykl Hamiltona. Problem komiwojażera, symulowane wyżarzanie. Przeszukiwanie grafu wszerz. Drzewa spinające. Algorytmy Prima i Kruskala. Algorytm Dijkstry, metryka euklidesowa.

Laboratoria: Definiowanie prostych klas. Powoływanie obiektów. Rola konstruktora i destruktora. Przeładowanie operatorów. Funkcje zaprzyjaźnione. Dziedziczenie i wzorce klas. Polimorfizm i funkcje wirtualne. Pojemniki – proste przykłady. Budowa pojemnika dla wektorów jednowymiarowych. Konstruowanie i operowanie na listach liniowych. Drzewa binarne – algorytmy in-order, pre-order i post-order. Podstawowe operacje na drzewach poszukiwań binarnych – wstawianie i usuwanie. Drzewa AVL – operacje obrotu. Algorytmy Prima, Kruskala i Dijkstry.

Metody dydaktyczne

Wykład: prezentacja multimedialna, wspomagana przykładami podawanymi na tablicy. Laboratoria: rozwiązywanie zadań podanych przez prowadzącego, projektowanie prostych algorytmów, kodowanie algorytmów w języku C++.

Literatura

Podstawowa

1. R. Sedgewick, Algorytmy w C++, Oficyna Wydawnicza READ ME, Łódź, 1999
2. N. Wirth, Algorytmy + struktury danych = programy, WNT, Warszawa, 1980.
3. T.H. Cormen, C.E. Leiserson, R.L. Rivest, C. Stein, Wprowadzenie do algorytmów, WNT, Warszawa, 2004
4. E.W. Dijkstra, Umiejętność programowania, WNT, Warszawa, 1985.

Uzupełniająca

1. J. Grębosz, Symfonia C++, Oficyna Kallimach, Kraków 2008.
2. W. Lipski, Kombinatoryka dla programistów, WNT, Warszawa, 1982.

Bilans nakładu pracy przeciętnego studenta

	Godzin	ECTS
Łączny nakład pracy	120	5.0
Zajęcia wymagające bezpośredniego kontaktu z nauczycielem	64	3.0
Praca własna studenta (przygotowanie do zaliczenia, przygotowanie do laboratorium, przygotowanie do egzaminu, studia literaturowe)	56	2.0