KARTA OPISU MODUŁU KSZTAŁCENIA

Nazwa modułu/przedmiotu	Kod
Algorytmy i struktury danych	**1010514321010510077**

Kierunek studiów	Profil kształcenia (ogólnoakademicki, praktyczny)	Rok / Semestr
Informatyka	**ogólnoakademicki**	**1 / 2**
Ścieżka obieralności/specjalność	Przedmiot oferowany w języku:	Kurs (obligatoryjny/obieralny)
-	**polski**	**obieralny**

Stopień studiów:	**Forma studiów** (stacjonarna/niestacjonarna)
I stopień	**niestacjonarna**

Godziny	Liczba punktów
Wykłady: **20** Ćwiczenia: **-** Laboratoria: **20** Projekty/seminaria: **-**	**5**

Status przedmiotu w programie studiów (podstawowy, kierunkowy, inny)	(ogólnouczelniany, z innego kierunku)
kierunkowy	**z danego kierunku**

Obszar(y) kształcenia i dziedzina(y) nauki i sztuki	**Podział ECTS (liczba i %)**
nauki techniczne	**5 100%**

Odpowiedzialny za przedmiot / wykładowca:

Dr inż. Maciej Machowiak
email: maciej.machowiak@put.poznan.pl
tel. 665-2982
Instytut Informatyki
ul. Piotrowo 2, 60-965 Poznań

Odpowiedzialny za przedmiot / wykładowca:

Dr hab. inż. Grzegorz Pawlak
email: grzegorz.pawlak@cs.put.poznan.pl
tel. 665-2982
Instytut Informatyki
ul. Piotrowo 2, 60-965 Poznań

Wymagania wstępne w zakresie wiedzy, umiejętności, kompetencji społecznych:

1	**Wiedza:**	Student rozpoczynający ten przedmiot powinien posiadać podstawową wiedzę z zakresu implementacji prostych programów w języku C, C++ lub Pascal.
2	**Umiejętności:**	Powinien posiadać umiejętność rozwiązywania podstawowych problemów programistycznych oraz testowania i poprawiania błędów w zaimplementowanych przez siebie programach. Dodatkowo student powinien posiadać umiejętność pozyskiwania informacji ze wskazanych źródeł. Powinien również rozumieć konieczność poszerzania swoich kompetencji.
3	**Kompetencje społeczne**	W zakresie kompetencji społecznych student musi prezentować takie postawy jak uczciwość, odpowiedzialność, wytrwałość, ciekawość poznawcza, kreatywność, kultura osobista, szacunek dla innych ludzi.

Cel przedmiotu:

1. Przekazanie studentom podstawowej wiedzy o złożoności obliczeniowej w zakresie jej analizy, działania deterministycznej i niedeterministycznej maszyny Turinga, maszyny RAM, klasyfikacji problemów i algorytmów oraz klas złożoności P i NP.
2. Przekazanie studentom podstawowej wiedzy z algorytmiki w zakresie sortowania ciągów danych z różną złożonością obliczeniową, programowania zachłannego i dynamicznego, przeszukiwania z nawracaniem oraz podstawowych algorytmów grafowych takich jak BFS, DFS, znajdowanie cyklu Eulera i Hamiltona.
3. Przekazanie studentom podstawowej wiedzy o strukturach danych obejmujące sposób działania drzew, drzew BST, kopców i grafów oraz analizę ich złożoności.
4. Rozwijanie u studentów umiejętności dowodzenia NP-zupełności problemów.
5. Rozwijanie u studentów umiejętności implementacji programistycznej poznanych algorytmów oraz struktur danych.
6. Rozwijanie u studentów umiejętności doboru odpowiedniego algorytmu i struktury danych do rozwiązywanego problemu oraz ocenę złożoności obliczeniowej i pamięciowej ich implementacji.
7. Rozwijanie u studentów umiejętności testowania zaimplementowanych algorytmów oraz ich oceny.

Efekty kształcenia i odniesienie do kierunkowych efektów kształcenia

Wiedza:

1. Ma rozszerzoną i pogłębioną wiedzę z matematyki przydatną do formułowania i rozwiązywania złożonych zadań informatycznych dotyczących analizy i formalnych dowodów poprawności oraz złożoności obliczeniowej algorytmów. - [K_W1]
2. Ma uporządkowaną, podbudowaną teoretycznie wiedzę ogólną w zakresie algorytmów i złożoności. - [K_W4]
3. Ma szczegółową wiedzę nt. algorytmiki, struktur danych oraz analizy złożoności obliczeniowej i pamięciowej. - [K_W5]
4. Zna podstawowe metody, techniki i narzędzia stosowane przy rozwiązywaniu prostych zadań informatycznych z zakresu analizy złożoności obliczeniowej algorytmów i problemów. - [K_W8]

Umiejętności:

1. potrafi planować i przeprowadzać eksperymenty, w tym pomiary czasu działania algorytmów, interpretować uzyskane wyniki i wyciągać wnioski o poprawności doboru i złożoności algorytmów. - [K_U7]

2. potrafi wykorzystać do formułowania i rozwiązywania zadań informatycznych metody analityczne i eksperymentalne w celu dobrania odpowiednich algorytmów i struktur danych. - [K_U8]

3. potrafi ocenić złożoność obliczeniową algorytmów i problemów. - [K_U13]

4. potrafi wybrać język programowania odpowiedni do implementacji zadania programistycznego na podstawie wiedzy o algorytmach, które będą wykorzystywane oraz znajomości wymagań o wielkości instancji wejściowych oraz wymaganej wydajności. - [K_U20]

5. ma umiejętność formułowania algorytmów i ich programowania z użyciem przynajmniej jednego z podstawowych języków programowania (C, C++ i Pascal). - [K_U22]

Kompetencje społeczne:

1. Potrafi odpowiednio określić priorytety służące realizacji określonego przez siebie lub innych zadania poprzez rozstrzyganie dylematu, czy implementacja bardziej wydajnych algorytmów warta jest zwiększonego nakładu pracy na ich implementację. - [K_K6]

Sposoby sprawdzenia efektów kształcenia

Efekty kształcenia przedstawione wyżej weryfikowane są w następujący sposób:

Ocena formująca:

a) w zakresie wykładów weryfikowanie założonych efektów kształcenia realizowane jest przez:

- ocenę dwóch zadań rozdawanych studentom w czasie wykładów: jedno dotyczy implementacji i analizy struktury grafu, drugie dotyczy analizy złożoności obliczeniowej
- premiowanie aktywności studentów na wykładach

b) w zakresie laboratoriów weryfikowanie założonych efektów kształcenia realizowane jest przez:

- ocenę sprawozdań z wynikami projektów, których celem jest implementacja i analiza algorytmów i struktur danych
- ocenę rozwiązań zadań demonstrujących sposób działania algorytmów prezentowanych przez studentów na tablicy

Ocena podsumowująca:

Sprawdzanie założonych efektów kształcenia realizowane jest przez:

- ocenę sprawozdań z wynikami projektów, których celem jest implementacja i analiza algorytmów i struktur danych,
- ocenę wiedzy i umiejętności związanych z realizacją zadań laboratoryjnych poprzez 2 kolokwia w semestrze,
- ocenę wiedzy i umiejętności wykazanych na egzaminie pisemnym o charakterze problemowym:
- w formie 4 zadań zamkniętych, które polegają na wpisaniu w wolne miejsca wyniku obliczeń i analiz sprawdzających umiejętności studentów w zakresie rozwiązywania problemów algorytmicznych,
- dodatkowego 1 zadania otwartego sprawdzającego umiejętność dowodzenia przynależności algorytmów do klasy NP,
- zadania punktowane są w skali 0-5 punktów, ze skokiem co 0,25 punktu,
- do zaliczenia egzaminu wymagane jest 50% punktów.

Aktywność podczas zajęć premiowana jest przyznawaniem punktów, które uwzględniane są w czasie wystawiania oceny podsumowującej pracę w semestrze.

Treści programowe

Wykłady z przedmiotu rozpoczynają się od wyjaśnienia podstawowych terminów z zakresu algorytmiki, takich jak problem i algorytm, dane i operacje na danych, instancja, pojęcie typu. Poruszona zostaje tematyka poprawności algorytmów, jej definiowanie oraz weryfikacja. Przedstawiony zostaje podział problemów na decyzyjne i optymalizacyjne, wraz z charakterystyką tych klas i przykładami należących do nich problemów. Przed przystąpieniem do omawiania implementacji algorytmów we współczesnych językach programowania omawiana jest deterministyczna i niedeterministyczna maszyna Turinga oraz maszyna RAM, jako przykłady abstrakcyjnego modelu komputera służącego do wykonywania algorytmów. W oparciu o ten materiał wyjaśniona zostaje idea i definicja klas problemów decyzyjnych P oraz NP, wraz z podklasami problemów NP-zupełnych i silnie NP-zupełnych oraz przedstawione sposoby dowodzenia przynależności problemów do tych klas. Omówiona zostaje złożoność obliczeniowa problemów oraz złożoność czasowa i pamięciowa algorytmów wraz ze sposobami jej wyznaczania oraz zapisywania w notacji O(). Poruszony zostaje problem złożoności w najgorszym i najlepszym przypadku oraz złożoność średnia. W trakcie wykładu szczegółowo prezentowane są ogólne metody konstruowania algorytmów takie jak metoda zstępująca, dziel i rządź oraz przeszukiwanie z nawrotami. Przedstawiane jest również porównanie metody zachłannej oraz programowania dynamicznego wraz z omówieniem złożoności pseudowielomianowej. W tym celu wykorzystywana jest szczegółowa analiza problemu plecakowego. W czasie wykładu prezentowane są również możliwe sposoby komputerowej reprezentacji grafów uwzględniając macierz i listę incydencji, listę następników oraz macierz grafu wraz ze szczegółową analizą ich złożoności czasowej i pamięciowej w zależności od liczby wierzchołków i krawędzi w grafie oraz wykonywanych operacji.

Zajęcia laboratoryjne kładą duży nacisk na zastosowanie w praktyce algorytmów i struktur danych prezentowanych na wykładzie poprzez realizację projektów oraz rozwiązywanie zadań na tablicy. Zajęcia podzielone są na kilka grup tematycznych, z których każda zakończona jest realizacją projektu implementującego omawiane algorytmy. W pierwszej grupie tematycznej prezentowane są algorytmy sortowania począwszy od najprostszych, działających ze złożonością kwadratową, takich jak sortowanie bąbelkowe, przez wybór i wstawianie, przez szybsze sortowanie QuickSort, przez scalanie oraz Shella, po sortowania w czasie liniowym za pomocą algorytmu kubełkowego i przez zliczanie. Dla każdego algorytmu analizowana jest jego złożoność w najlepszym, średnim i najgorszym przypadku. Na podstawie algorytmów sortowania zademonstrowana zostaje również koncepcja rekurencji. Kolejna grupa tematyczna obejmuje złożone struktury danych, takie jak lista jedno i dwu kierunkowa, drzewa, w tym drzewa BST oraz kopce. Dla każdej struktury przedstawiony jest algorytm dodawania i usuwana z nich elementów oraz możliwe sposoby ich przeszukiwania. Analizowana jest również ich złożoność oraz problemy, w których należy je wykorzystać. Trzecia grupa tematyczna to algorytmy grafowe obejmująca algorytmy dla grafów skierowanych i nieskierowanych, takie jak BFS, DFS, sortowanie topologiczne, drzewa rozpinające oraz wyszukiwanie cyklu Eulera i Hamiltona prezentujące również algorytmy z nawracaniem. W trakcie omawiania grafów szczegółowo poruszana jest tematyka implementacji reprezentacji grafów przedstawionych na wykładzie. Ostatnia grupa tematyczna obejmuje implementację algorytmu zachłannego oraz dynamicznego dla problemu plecakowego, ich porównanie oraz analizę.

Cześć wymienionych wyżej treści programowych realizowana jest w ramach pracy własnej studenta.

Metody dydaktyczne:

1. Wykład: prezentacja ilustrowana przykładami podawanymi na tablicy, rozwiązywanie zadań.
2. Ćwiczenia laboratoryjne: rozwiązywanie zadań, ćwiczenia praktyczne, wykonywanie eksperymentów programistycznych, dyskusja.

Literatura podstawowa:

1. Elementy analizy algorytmów, L. Banachowski, A. Kreczmar, WNT, W-wa, 1982
2. Algorytmy + struktury danych = programy, N. Wirth, WNT, W-wa, 2004
3. Złożoność obliczeniowa problemów kombinatorycznych, J. Błażewicz, WNT, W-wa, 1988
4. Wprowadzenie do algorytmów, T.H. Cormen, Ch.E.Leiserson, R.L. Rivest, C. Stein, PWN, W-wa, 2012

Literatura uzupełniająca:

1. Algorytmika praktyczna nie tylko dla mistrzów, P. Stańczyk, PWN, 2009

Bilans nakładu pracy przeciętnego studenta

Czynność	Czas (godz.)
1. Udział w zajęciach laboratoryjnych:	20
2. Przygotowanie do ćwiczeń laboratoryjnych:	20
3. Udział w konsultacjach związanych z realizacją procesu kształcenia: ćwiczeń laboratoryjnych i projektów.	1
4. Napisanie programów, uruchomienie, weryfikacja i analiza (czas poza zajęciami laboratoryjnymi).	35
5. Przygotowanie do kolokwiów.	15
6. Udział w wykładach.	20
7. Przygotowanie do egzaminu i obecność na egzaminie: 13 godz. + 2 godz.	15

Obciążenie pracą studenta

forma aktywności	godzin	ECTS
Łączny nakład pracy	126	5
Zajęcia wymagające bezpośredniego kontaktu z nauczycielem	42	2

Zajęcia o charakterze praktycznym	55	2