

Algorytmy i struktury danych

Zadania obowiązkowe

Informatyka WIEiT - 2018/2019

1. Ćwiczenia - Sortowania proste

Brak zadań obowiązkowych

2. Ćwiczenia - MergeSort

1. Implementacja algorytmu MergeSort dla sortowania list
2. Proszę zaproponować/zaimplementować algorytm scalający k posortowanych tablic o łącznej długości n w jedną posortowaną tablicę w czasie $O(n * \log(k))$.
3. Proszę zaproponować strukturę przechowującą liczby naturalne, w której operacje: Insert i GetMedian mają złożoność $O(\log(n))$. Proszę zaimplementować w/w operacje.
4. Proszę zaimplementować algorytm zliczający liczbę inwersji w tablicy (Inwersja to para indeksów i, j taka, że $i < j$ oraz $T[i] > T[j]$)

3. Ćwiczenia - QuickSort

1. Proszę zaimplementować algorytm QuickSort do sortowania listy jednokierunkowej.
2. Proszę zaimplementować algorytm, który w czasie liniowym sortuje tablicę A zawierającą n liczb ze zbioru $0, 1, \dots, n^2 - 1$.
3. Mamy serię pojemników z wodą, połączonych (każdy z każdym) rurami. Pojemniki mają kształty prostokątów (2d), rury nie mają objętości (powierzchni). Każdy pojemnik opisany jest przez współrzędne lewego górnego rogu i prawego dolnego rogu. Wiemy, że do pojemników nalano A wody (oczywiście woda rurami spłynęła do najniższych pojemników). Obliczyć ile pojemników zostało w pełni zalanych.
4. Dany jest ciąg przedziałów domkniętych $[a_1, b_1], \dots, [a_n, b_n]$. Proszę zaproponować algorytm, który znajduje taki przedział $[a_t, b_t]$, w którym w całości zawiera się jak najwięcej innych przedziałów.

4. Ćwiczenia - Zastosowania sortowań

1. Dana jest posortowana tablica `int A[N]` oraz liczba x . Napisać program, który stwierdza czy istnieją indeksy i oraz j , takie że $A[i] + A[j] = x$ (powinno działać w czasie $O(N)$).
2. Zaimplementować algorytm, który dla tablicy `int A[N]` wyznacza rekurencyjną medianę median (magiczne piątki).
3. Mamy daną tablicę A z n liczbami. Proszę zaproponować algorytm o złożoności $O(n)$, który stwierdza, czy w tablicy ponad połowa elementów ma jednakową wartość.
4. Proszę zaproponować algorytm sortujący ciąg słów o różnych długościach w czasie proporcjonalnym do sumy długości tych słów.

5. Ćwiczenia - Struktury danych

1. Proszę zaimplementować dodawanie elementu do SkipListy.
2. Proszę zaimplementować kolejkę przy użyciu dwóch stosów.

6. Ćwiczenia - Tablice z haszowaniem

1. Proszę zaimplementować następujące operacje na tablicy z haszowaniem:
 - wstawianie
 - usuwanie
 - wyszukiwanie
 - reorganizacja (usunięcie kluczy zaznaczonych do skasowania)
2. Dana jest nieposortowana tablica `int A[N]` oraz liczba x . Proszę napisać funkcję, która sprawdza na ile sposobów można przedstawić x jako sumę $A[i] + A[j]$ takiego że $i < j$.

7. Ćwiczenia - Drzewa BST

1. Proszę podać modyfikację drzewa BST, która pozwala na efektywne wykonywanie następujących operacji:
 - (a) znalezienie i -tego co do wielkości elementu w drzewie BST
 - (b) wyznaczenie, którym co do wielkości w drzewie jest zadany węzeł

Proszę zaimplementować obie operacje.

8. Ćwiczenia - Drzewa czerwono-czarne

1. Zaimplementować funkcję, która koloruje węzły drzewa aby spełniało warunek drzewa RB, funkcja powinna zwracać informację czy udało się pokolorować drzewo.
2. Udowodnić, że każde drzewo AVL jest drzewem RB.

9. Ćwiczenia - Drzewa B-tree

1. Dana jest tablica `bool A[N][N]`; Gracz początkowo znajduje się na (zadanej) pozycji (x, y) , dla której zachodzi `A[y][x] == true`. Z danej pozycji wolno bezpośrednio przejść jedynie na pozycję, której dokładnie jedna współrzędna różni się o 1, oraz której wartość w tablicy *A* wynosi `true`. Proszę napisać funkcję obliczającą do ilu różnych pozycji może dojść gracz startując z zadanej pozycji (x, y) .
2. Dana jest struktura węzła drzewa B-tree przechowującego unikalne klucze:

```
struct node {  
    int n; // liczba kluczy zawarta w węźle  
    int key[N]; // tablica kluczy w węźle  
    node* child[N+1]; // wskaźniki do synów węzła  
    bool leaf; // czy węzeł jest liściem  
};
```

Proszę napisać funkcję `bool is_b_tree(node* p)`; sprawdzającą czy wskaźnik *p* wskazuje na poprawne drzewo B-tree.

10. Ćwiczenia - Grafy - BFS, DFS

1. Proszę zaimplementować następujące algorytmy:
 - (a) Sprawdzanie czy graf jest dwudzielny
 - (b) Policzyć liczbę spójnych składowych w grafie

Graf jest reprezentowany jako macierz sąsiedztwa albo listy sąsiadów albo listy krawędzi.

2. Mówimy, że wierzchołek *t* w grafie skierowanym jest ujściem, jeśli:
 - z każdego innego wierzchołka *v* istnieje krawędź z *v* do *t*,
 - nie istnieje żadna krawędź wychodząca z *t*.

Podać algorytm znajdujący ujście (jeśli istnieje) przy macierzowej reprezentacji grafu.

11. Ćwiczenia - Kolokwium nr 2

Brak zadań obowiązkowych