Kopiec

Laboratorium 10. Kopiec binarny typu MAX i MIN.

Zad. 1 Wprowadź kolejno następujące dane do kopca typu MIN:
3.5, 11, 21, 4, -4, 2, 3,-5, 32, 30, 41

a) Zaprezentuj w postaci grafu otrzymane drzewo.

b) Wykonaj trawersowanie drzewa metodą PreOrder i InOrder.

c) Przedstaw kopiec po usunięciu jednej danej.

Zad. 2 Wprowadź następujące dane typu łańcuchowego:

a) „do", „re", „mi", „fa", „sol", „la", „si", „do"

b) 3.14, 0, 16, 2, 8, 97, 2.72, 1, 16, 8051, 11, 0, 1.62, -1

do kopca typu MAX. Następnie przedstaw stan kopca w postaci grafu oraz przeprowadź trawersowanie InOrder i PostOrder. Usuń dwukrotnie daną z kopca, po każdym usunięciu przedstaw stan kopca.

Zad. 3 Zapoznaj się z implementacją kopca THeapMax<E> przedstawioną na wykładzie. Następnie rozszerz implementację o metody:

a) int find(E element) – znajdującą indeks elementu przechowywanego w kopcu, albo -1 gdy brak danej

b) void preorder(), void inorder() i void postorder() – trawersowania, w razie potrzeby zaimplementuj metody pomocnicze,

c) bool checkHeap(ArrayList<E> array) – statyczną, sprawdzającą czy dane zawarte w tablicowej implementacji listy stanowią strukturę danych kopiec.

d) bool insertHeap(E element, ArrayList<E> array) – statyczną, pozwalającą na wstawienie danej element do tablicowej implemenacji listy array tak jak do kopca.

Zad 4**.** Czy dane w tablicy tworzą kopiec typu *max*:
a) [100,44, 88, 33, 43, 72, 66, 22, 32, 38, 5, 19, 70, 60, 1],
b) [100,44, 88, 33, 43, 72, 66, 22, 32, 19, 5, 38, 70, 60, 1],
c) [100,43, 88, 33, 44, 72, 66, 22, 32, 38, 5, 19, 70, 60, 1].

Zad 5. Przy wykorzystaniu struktury danych THeapMax<E> zaimplementuj strukturę danych kolejka priorytetowa TPriorityQueue<E>, udostępniają metody służące do: dodawania elementów, usuwania elementu o maksymalnym priorytecie oraz metody sprawdzającej liczbę elementów przechowywanych w kolejce.