LABORATORIUM 5

Kopce i kolejki priorytetowe

Zajęcia poświęcone są implementacji kopców i kolejek priorytetowych oraz wybranych algorytmów opartych na tych strukturach.

4.1 Kopce

Kopcem nazywamy drzewo binarne o własności: dla każdego węzła v

- jeśli $v\uparrow.left \neq \text{NULL}$, to $v\uparrow.left\uparrow.key < v\uparrow.key$
- jeśli $v\uparrow.right \neq \text{NULL}$, to $v\uparrow.right\uparrow.key < v\uparrow.key$

Reprezentacja:

Tablica jednowymiarowa $K[1..n]$ o własności:

- $K[1]$ jest korzeniem kopca
- jeśli $2i \leqslant n$, to $K[2i]$ jest lewym następnikiem $K[i]$
- jeśli $2i+1 \leqslant n$, to $K[2i+1]$ jest prawym następnikiem $K[i]$.

Poprawianie struktury kopca

Warunek $Kopiec(i, j)$:

$$(2i \leqslant j \implies K[2i] \leq K[i]) \wedge (2i+1 \leqslant j \implies K[2i+1] \leq K[i]).$$

NAGŁÓWEK: $Heapify(l, r)$
DANE: $K[1..n]$ spełnia warunek $Kopiec(i, r)$ dla $i = l+1, l+2, \ldots, r$.
WYNIK: $K[1..n]$ spełnia warunek $Kopiec(i, r)$ dla $i = l, l+1, \ldots, r$.

Algorytm 3.1: *Heapify*

```
1   Heapify(l, r : int);
2   var i, j : int;
3   begin
4     j := l; k := 2j; x := A_j;
5     while k ⩽ r do;
6       if k+1 ⩽ r then
7         if K[k] < K[k + 1] then k := k+1 fi fi;
8       if x < A_k
9       then K[j] := K[k]; j := k; k := 2j
10     else k := r+1 fi;
11    od;
12    K[j] := x
13  end Heapify;
```

Budowa kopca

DANE: Tablica $K[1..n]$ elementów typu U.

WYNIK: Tablica $K[1..m]$ reprezentująca kopiec.

Algorytm 3.2: *BuildHeap*

```
1   BuildHeap(n : int);
2   var i : int;
3   begin
4     for i := ⌊n/2⌋ downto 1 do
5       Heapify(i, n);
6     od
7   end BuildHeap;
```

Wstawianie do kopca

DANE: Kopiec K o $n \geqslant 0$ elementach oraz $x \in U$.

WYNIK: Kopiec K o $n+1$ elementach z elementem x.

Algorytm 3.3: *InsertHeap*

```
1   InsertHeap(n : int; x : U);
2   begin
3     K[n + 1] := x;
4     i := n+1; j := ⌊i/2⌋;
5     while j>0 do
6       if K[j] ⩾ x then j := 0 fi;
7       K[i] := K[j];
8       i := j; j := ⌊j/2⌋;
9     od;
10    K[i] := x
5   end InsertHeap;
```

Usuwanie elementu maksymalnego

DANE: tablica $K[1..n]$ reprezentująca kopiec.
WYNIK: tablica $K'[1..n-1]$ reprezentująca kopiec po usunięciu elementu korzenia kopca wejściowego.

Algorytm 3.4: *DeleteMax*

```
1   DeleteMax(n : int);
2   begin
3     K[1] := K[n]; n := n−1;
4     Heapify(1, n)
5   end DeleteMax;
```

4.2 Kolejki priorytetowe

Kolejką priorytetową nazywamy strukturę danych zbioru $A \subseteq U$, gdzie $(U, \leqslant)$, umożliwiającą wykonanie następujących operacji podstawowych:

- wstawienie elementu do kolejki
- dostęp do elementu o najwyższym priorytecie
- usunięcie elementu o najwyższym priorytecie.

Typowe realizacje:

- kopiec
- lista jednokierunkowa
- tablica cykliczna.

4.3 Zadania

4.3.1

Zaimplementować strukturę kopca liczb rzeczywistych reprezentowanego 1–wymiarową tablicą K wraz z operacją $Change(i, x) ::= K[i] := x$.

4.3.1

Zaimplementować kolejkę priorytetową realizowaną przy pomocy listy jednokierunkowej. Elementami kolejki są pakiety danych o wynikach studentów uzyskanych na laboratorium z AiSD postaci:

- *numer indeksy studenta*
- *wyniki* z 7 laboratoriów.

Zakładamy, że na poszczególnych zajęciach każdy student uzyskuje o, 1, 2 albo 3 punkty. Priorytetem jest suma punktów uzyskanych w trakcie wszystkich 7 zajęć.