Listy, stosy, kolejki

Listy

Definicja *Lista* jest to struktura danych zawierająca pewne elementy ustawione w jakiejś kolejności.

Typowe operacje na liście to: *wstaw, usuń, szukaj*

Implementacje listy: tablicowa i dowiązaniowa

Listy dowiązaniowe

Warianty list

- jednokierunkowa/dwukierunkowa
- cykliczna/niecykliczna
- bez wartowników/z wartownikami

Notacja

`NIL`	pusty wskaźnik
`L.head`	początek listy `L`
`x.next`	następny element po węźle `x`
`x.prev`	poprzedni element przed węzłem `x`
`x.key`	wartość (klucz) identyfikujący węzeł `x`
`L.nil`	wartownik listy `L`

```
List-search(L,k)
// szuka węzła listy L zawierającego klucz k
// lista niecykliczna bez wartownika

x = L.head
while x!=NIL and x.key!=k
   x = x.next
return x // NIL oznacza, że szukanego klucza
         // nie było na liście


List-insert(L,x)
// wstawia węzeł x do listy L
// lista dwukierunkowa niecykliczna bez wartownika

  x.next = L.head
  if L.head != NIL
     L.head.prev = x
  L.head = x
  x.prev = NIL
```

```
List-delete(L,x)
// usuwa węzeł x z listy L
// lista dwukierunkowa niecykliczna bez wartownika

  if x.prev != NIL
     x.prev.next = x.next
  else
     L.head = x.next
  if x.next != NIL
     x.next.prev = x.prev


List-delete(L,x)
// usuwa węzeł x z listy L
// lista dwukierunkowa cykliczna z wartownikiem

   x.prev.next = x.next
   x.next.prev = x.prev


List-search(L,k)
// szuka węzła listy L zawierającego klucz k
// lista dwukierunkowa cykliczna z wartownikiem

x = L.nil.next
while x!=L.nil and x.key!=k
   x = x.next
return x // zwrócone L.nil oznacza, że szukanego
         // klucza nie było na liście
```

Stosy

Definicja *Stos* jest to struktura danych w której można przechowywać pewne elementy i na której określone są następujące operacje

`Push` – włożnie elementu na stos

`Pop` – zdjęcie elementu ze stosu (zdejmowany jest ten element,który był najpóźniej włożony (LIFO)); zdjęty element jest zwracany jako wynik tej operacji

`EmptyStack` – zwraca wartość logiczną `true` jeżęli stos jest pusty a wartość `false` jeżeli nie jest pusty

Kolejki

Definicja *Kolejka* (Queue) jest to struktura danych w której można przechowywać pewne elementy i na której określone są następujące operacje

`Enqueue` – wstawienie elementu do kolejki

`Dequeue` – pobranie (i usunięcie) elementu z kolejki (pobierany jest ten element, który był najwcześniej wstawiony (FIFO))

`EmptyQueue` – zwraca wartość logiczną `true` jeżeli kolejka jest pusta a wartość `false` jeżeli nie jest pusta

```
Enqueue(Q,x)
// wstawia element x do kolejki Q
  if  FullQueue(Q)
    return  błąd
  Q[Q.tail] = x
  if Q.tail == Q.length
    Q.tail = 1
  else
    Q.tail = Q.tail+1
```

```
Dequeue(Q)
// pobiera element z kolejki Q
// i usuwa go z niej
  if  EmptyQueue(Q)
    return  błąd
  x = Q[Q.head]
  if Q.head == Q.length
    Q.head = 1
  else
    Q.head = Q.head+1
  return x
```

```
EmptyQueue(Q)
// zwraca true jeżeli kolejka Q
// jest pusta
  return Q.tail == Q.head
```

```
FullQueue(Q)
// zwraca true jeżeli kolejka Q
// jest pełna
  if Q.tail == Q.length and Q.tail == 1
    return true
  else
    return Q.tail+1 == Q.head
```