

UVOD U STRUKTURE PODATAKA

Uvod

- Struktura podataka je složeni podatak ili složena promjenljiva koja nije ugrađena u programski jezik pa se mora realizovati sa onim što je na raspolaganju
- Pregled prezentacije
 - Lista
 - Red
 - Stek

Lista

- Lista je konačan niz x_1, x_2, \dots, x_n , gdje je $n \geq 0$, dužina liste je n , članovi liste x_k su podaci istog tipa, fizički raspored elemenata ne odgovara logičkom
 - jednostruko i dvostruko povezana lista
- Implementacija liste nizom, unaprijed se zadaje veličina niza = statičko alociranje memorije

Realizacije liste pokazivačima

- Realizacija liste pomoću pokazivača, element liste je struktura sastavljena iz dva polja, prvi je podatak a drugi pokazivač na sljedeći element liste, memorija se alocira dinamički
- Posebnu ulogu ima promjenljiva GLAVA LISTE koja je pokazivač na prvi član liste, ako je NIL onda je lista prazna

Pokazivači u PASCALu

- Deklaracija `var p: ^integer`
- Alokacija memorije `new(p)`, u memoriji se pravi prostor veličine 2 bajta, a adresa tog prostora dodjeljuje se promjenljivoj `p`
- Upisivanje vrijednosti `p^ := 50`
- Oslobađanje memorije i postavljanje pokazivača na nil `dispose(p)`

Pokazivači na record

- Lista cijelih brojeva u PASCALu je pokazivač na strukturu sa dva polja: cijeli broj + pokazivač na sljedeći član u listi

```
type lista = ^cjelina;  
cjelina = record  
  
        podatak: integer;  
        nastavak: lista  
end;  
var p: lista;
```

Operacija sa listom INSERT

- Dodavanje novog člana na početak liste

```
p: ^lista;  
new(p);  
p^.podatak := readln(broj);  
if (glava = nil)  
    p^.nastavak = nil;  
    glava := p;  
else  
    p^.nastavak := glava;  
    glava := p;
```

Brisanje člana iz liste

```
p := glava; q:= glava;
while (p <> nil)
begin
  if (p^.podatak = broj)
    if (p = glava)
      glava := p^.nastavak;
      free(p);
    else
      q^.nastavak := p^.nastavak;
      free(p);
  else
    q := p;
    p := p^.nastavak;
end
```


Stek

- Stek je konačan niz x_1, x_2, \dots, x_n , gdje je $n \geq 0$ a sve operacije sa stekom rade se na njegovom kraju tj. vrhu
- Operacija POP – uklanjanje elementa sa vrha steka, novo stanje steka je x_1, x_2, \dots, x_{n-1}
- Operacija EMPTY - provjerava da li je stek prazan
- Operacija WRITE – čitanje elementa sa vrha steka (bez uklanjanja)
- Operacija PUSH(a) – dodaj a na vrh steka, novo stanje steka x_1, x_2, \dots, x_n, a

Implementacija steka

- Lista
 - PUSH povlači operaciju NEW, POP povlači operaciju DISPOSE
- Niz
 - Statički se alocira memorija m članova * k BAJTa
 - Promjenljiva TOP da označi vrh steka, posljednji zauzeti element u nizu, ako je $TOP = 0$ stek je prazan, ako je $TOP = m$ stek je pun
 - Inicijalizacija steka

Red

- Za red je karakteristično da se umetanje izvodi na kraju, a brisanje na početku reda (ili obratno)
- Ako je red x_1, x_2, \dots, x_n , poslije dodavanja elementa a stanje reda je x_1, x_2, \dots, x_n, a ; ako se sada izvrši operacija brisanja stanje reda je x_2, \dots, x_n, a
- Implementacija reda je uglavnom pomoću niza, potrebne su promjenljive za početak b i za kraj e , ako je $b = e = 0$ red je prazan, kružno dodavanje

Domaći zadatak

- Strana 386, zadaci 3, 4, 5, 6, 7, 8, 10