

Diskretne i kombinatorne metode za računarsku grafiku

Data je procedura stepen koja za graf smešten u niz listi susedstva $G[]$ sa n čvorova nalazi stepen svakog čvora $s[]$.

```

1 void stepen(grana G[], int n,
2   int s[])
3 {
4   int i;
5   grana gr;
6   for(i=0; i<n; i++){
7     gr = G[i];
8     s[i] = 0;
9     while(gr){
10      (s[i])++;
11      gr = gr->next;
12    }
13  }
14
15 }
```

1. Naći vreme izvršavanja $T(n, m)$ procedure stepen u zavisnosti od broja čvorova grafa

n i broja grana grafa m i vremena c_k izvršavanja linije k .

Dati asimptotsku ocenu za $T(n, m)$.

2. Dati definiciju malog o ponašanja. Da li je za sve $f = f(n)$ i $g = g(n)$ tačno:
 $f = o(g) \Rightarrow f = O(g)$?
 $f = \Theta(g) \Rightarrow f = O(g)$?
 $f = \Omega(g) \Rightarrow f = \Theta(g)$?
3. Napisati pseudo kod algoritma HORNER za računanje vrednosti polinoma Hornerovom šemom. Polinom $a_n x^n + \dots + a_1 x + a_0$ je smešten u niz $a = [a_0, a_1, \dots, a_n]$.
4. Izračunati broj sabiranja $S_H(n)$ i broj množenja $M_H(n)$ potrebnih da se izračuna vrednost polinoma stepena n Hornerovom šemom:

	a_n	a_{n-1}	$\dots$	a_1	a_0
x	a_n	$x \cdot a_n + a_{n-1}$	$\dots$	$\dots$	$p_n(x)$

5. Napisati u programskom jeziku C proceduru printstack iz implementacije ADT stack preko povezanih listi.

```

typedef char listdata;
typedef struct _node node;
typedef node *stack;
```

```

struct _node
{
    listdata data;
    node *next;
};
void printstack(stack S)
{
    // Ovaj kod napisati
}
```

```

#include <stdio.h>
#include <stdlib.h>
int key[] = { 5, 1, 7, 4, 2, 6, 3};
int LC[] = { 1,-1,-1, 4,-1,-1,-1};
int RC[] = { 2, 3, 5,-1, 6,-1,-1};
void infix_print(int root)
{ // Ovaj kod napisati }
int main()
{
    int i, root = 0;
    printf("Infix_print: \n");
    infix_print(root);
    printf("\n");
    return 0;
}
```

Levo je dat program u programskom jeziku C koji treba da ispiše infix prolaz kroz korensko drvo dato LC-RC reprezentacijom u programu.

6. Napisati nedostajući kod procedure infix_print.
7. Rekonstruisati korensko drvo (nacrtati) iz LC-RC reprezentacije date u programu.
8. Ispisati izlaz posle izvršavanja datog programa (dopunjenog procedurom infix_print).

Bodovi: 1→10, 2→10, 3→10, 4→10, 5→10, 6→10, 7→10, 8→10

$$1. T(n, m) = c_4 + c_5 + (n + 1)c_6 + n(c_7 + c_8) + (n + m)c_9 + m(c_{10} + c_{11}) = \Theta(n + m)$$

$$2. o(g) = \{f | (\forall c > 0)(\exists n_0 \in \mathbb{N})(\forall n) (n \geq n_0) \Rightarrow (0 \leq f(n) < cg(n))\}$$

$$f = o(g) \Rightarrow f = O(g)? \quad \text{DA}$$

$$f = \Theta(g) \Rightarrow f = O(g)? \quad \text{DA}$$

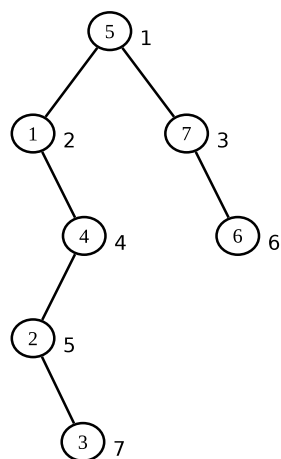
$$f = \Omega(g) \Rightarrow f = \Theta(g)? \quad \text{NE}$$

3. **function** HORNER(a, x)
 $n \leftarrow \text{length}(a)$
 $y \leftarrow a_n$
 for $i \leftarrow n - 1$ **downto** 0 **do**
 $y \leftarrow x \cdot y + a_i$
 end for
 return y
end function

4. Ukupno n množenja i n sabiranja.

5. **void** printstack(stack S)
 {
 while(S)
 {
 printf("%c\n", S->data);
 S = S -> next;
 }
 }

6. **void** infix_print(**int** koren)
 {
 if(koren > -1){
 infix_print(LC[koren]);
 printf("%3d", key[koren]);
 infix_print(RC[koren]);
 }
 }



7.

8. 1 2 3 4 5 7 6