ALGORITMI I STRUKTURE PODATAKA

RAČUNSKE VEŽBE - TERMIN BR. 9 - BINARNA STABLA - drugi deo

ALDINA AVDIĆ, DIPL. INŽ. - apljaskovic@np.ac.rs

RAČUNARSKA TEHNIKA, SOFTVERSKO INŽENJERSTVO, INFORMATIKA I MATEMATIKA

Binarno stablo - implementacija u C-u

```
#include<stdlib.h>
#include<stdio.h>

struct node{
int data;
struct node * right, * left;
};
```

- Binarno stablo je ono stablo čiji svaki čvor može imati maksimalno dva potomka, znači nijedan, jedan ili oba
- Stablo se u C-u implementira preko ulančane liste, ali se binarno stablo može implementirati i preko niza, ako je kompletno, a to ćemo učiti kasnije
- Data je ono što se čuva u čvoru, dok je lef pokazivač na levi potomak, a right na desni

Binarno stablo - implementacija u C-u

Kreiranje novog čvora

```
struct node* newNode(int data)
{
// Alociranje memorije za novi čvor
struct node* node = (struct node*)malloc(sizeof(struct node));

// Dodavanje data dela tom novom čvoru
node->data = data;
```

```
// incijalizacija desnog i levog potomka na NULL
node->left = NULL;
node->right = NULL;
return(node);
}
```

Obilasci

Inorder

```
void print_inorder(struct node * tree)
{
    if (tree)
    {
        print_inorder(tree->left);
        printf("%d\n",tree->data);
        print_inorder(tree->right);
    }
}
```

Objašnjenje

- Stabla su pogodna za rekurziju jer je svaki čvor koren sopstvenog podstabla
- To znači da na svako podstablo stabla možemo da se odnosimo kao na celo stablo
- To važi i za levo podstablo stabla i za desno, što nam olakšava obliske stabla
- Kod inorder obilaska, prvo se obilazi levo podstablo, koren pa desno podstablo

Obilasci

Objašnjenje:

- Stablo možemo da obilazimo ako postoji bar jedan čvor
- Preorder obilazak znači koren, levo podstablo, pa desno podstablo
- Počinjemo od zadatog čvora, i štampamo data - ono što je sačuvano u njemu
- Zatim isti algoritam prvo primenjujemo na njegovo levo, a zatim na desno podstablo

Preorder

```
void print_preorder(struct node *
tree)
{
    if (tree)
    {
        printf("%d\n",tree->data);
        print_preorder(tree->left);
        print_preorder(tree->right);
    } }
```

Obilasci i brisanje

Postorder

```
void print_postorder(struct node * tree)
{
    if (tree)
    {
        print_postorder(tree->left);
        print_postorder(tree->right);
        printf("%d\n",tree->data);
    }
}
```

Brisanje elementa

```
void deltree(struct node * tree)
{
    if (tree)
    {
        deltree(tree->left);
        deltree(tree->right);
        free(tree);
    }}
```

Main funkcija

```
int main()
{
struct node *root =
newNode(1);
root->left      = newNode(2);
root->right     = newNode(3);
root->left->left =
newNode(4);
printf("preorder\n");
print_preorder(root);
printf("inorder\n");
print_inorder(root);
printf("postorder\n");
print_postorder(root);
printf("deletion");
deltree(root->left);
getchar();
return 0;
```

Zadaci

- **Zadatak 1.** Napisati rekurzivnu funkciju u pseudokodu kojom se ispisuje sadržaj informacionih polja u listovima binarnog stabla.

```
procedure Listovi(koren:pokazivac);
 begin
  if koren<>nil
   then
    begin
     Listovi(koren^.leva_veza);
     if(koren^.leva_veza=nil)and(koren^.desna_veza=nil)
      then writeln(koren^.inf)
     Listovi(koren^.desna_veza);
    end;
 end;
```

Zadaci

- **Zadatak 2.** Napisati rekurzivnu funkciju kojom se izračunava zbir elemenata nepraznog binarnog stabla.

```
function Suma(koren:pokazivac):real;
 begin
  if koren=nil
   then
    Suma:=0
   else
    Suma:=koren^.inf+Suma(koren^.leva_veza)
          +Suma(koren^.desna_veza);
 end;
```

Zadaci

- **Zadatak 3.** Napisati rekurzivnu funkciju kojom se izračunava broj elemenata nepraznog binarnog stabla.

```
function BrojElem(koren:pokazivac):integer;
 begin
  if koren=nil
   then
    BrojElem:=0
   else
    BrojElem:=1+BrojElem(koren^.leva_veza)
               +BrojElem(koren^.desna_veza);
 end;
```

Zadaci

- **Zadatak 4.** Napisati rekurzivnu funkciju kojom se ispituje da li se element E nalazi u binarnom stablu.

```
function Postoji(E:char; koren:pokazivac):boolean;
 begin
  if koren=nil
   then Postoji:=false
   else if koren^.inf=E
            then Postoji:=true
            else Postoji:=Postoji(E,koren^.leva_veza) or
                 Postoji(E,koren^.desna_veza);
 end;
```

Zadaci

- **Zadatak 5.** Napisati rekurzivnu funkciju kojom se izračunarava maksimalna dubina stabla.

```
function Dubina(koren:pokazivac):integer;
 begin
  if koren=nil
   then
    Dubina:=0
   else
    Dubina:=1+Max(Dubina(koren^.leva_veza),
                  Dubina(koren^.desna_veza));
 end;
```

Zadaci

- **Zadatak 6.** Napisati rekurzivnu funkciju kojom se izračunarava broj elemenata na n-tom nivou stabla, pri čemu se koren stabla tretira kao nulti nivo.

```
function Nivo(koren:pokazivac; n:integer):integer;
 begin
  if koren<>nil
   then
    if n<>0
     then Nivo:= Nivo(koren^.leva_veza,n-1)+
                 Nivo(koren^.desna_veza,n-1)
     else Nivo:=1
   else Nivo:=0;
 end;
```

Zadaci

- **Zadatak 7.** Napisati rekurzivnu funkciju kojom se ispisuju elemenati na n-tom nivou stabla, pri čemu se koren stabla tretira kao nulti nivo.

```
procedure Nivo(koren:pokazivac; n:pokazivac);
  if koren<>nil
   then
    if n<>0
     then
      begin
       Nivo(koren^.leva_veza,n-1);
       Nivo(koren^.desna_veza,n-1);
      end;
     else write(' ', koren^.inf);
end;
```

Zadaci

- **Zadatak 8.** Napisati rekurzivnu funkciju kojom se izračunava zbir elemenata na n-tom nivou stabla, pri čemu se koren stabla tretira kao nulti nivo.

```
function Nivo(koren:pokazivac; n:integer):integer;
 begin
  if koren<>nil
   then
    if n<>0
     then Nivo:= Nivo(koren^.leva_veza,n-1)+
                 Nivo(koren^.desna_veza,n-1)
     else Nivo:=koren^.inf
   else Nivo:=0;
 end;
```

Zadaci

- **Zadatak 9.** Napisati rekurzivnu funkciju kojom se izračunava zbir elemenata u listovima binarnog stabla.

```
function ZbirL(koren:pokazivac):integer;
 begin
  if koren<>nil
   then
    if (koren^.leva_veza=nil)and(koren^.desna_veza=nil)
      then ZbirL:=Koren^.inf
      else ZbirL:= ZbirL(koren^.leva_veza)
                   + ZbirL(koren^.desna_veza)
   else Zbir:=0;
 end;
```

Zadaci

- **Zadatak 10.** Napisati rekurzivnu funkciju kojom se izračunava maksimalni element binarnog stabla.

```
function Max(koren:pokazivac):real;
 var
  Max1:real;
 begin
  Max1:=koren^.inf;
  if koren^.leva_veza<>nil
   then
    begin
     if Max1<Max(koren^.leva_veza)
      then Max1:= Max(koren^.leva_veza);
    end;
   if koren^.desna_veza<>nil
   then
    begin
     if Max1<Max(koren^.desna_veza)
      then Max1:= Max(koren^.desna_veza);
    end;
   Max:=Max1;
 end;
```

Test

1. Kako se binarno stablo implemetira u C-u?
2. Ispisati kod u C-u za preorder obliska stabla.
3. Ispisati kod u C-u za postorder obliska stabla.
4. Ispisati kod u C-u za inorder obliska stabla.
5. Šta je root?
6. Zašto je rekurzija pogodna za f-je za rad sa stablima?
7. Kako glasi kod u C-u za brisanje stabla?
8. Kako glasi kod iz zadataka 1 u C-u?
9. Kako glasi kod iz zadataka 2 u C-u?
10. Kako glasi kod iz zadataka 10 u C-u?

Test

- Test poslati do 04.05.2020. u 14h na mejl apljaskovic@np.ac.rs prema uputstvima sa sajta univerziteta

Hvala na pažnji!