

Strukture podataka i algoritmi 2
Januar, 2015

1. Napisati funkciju za uređivanje proizvoljnog binarnog stabla celih brojeva u binarno stablo za pretragu korišćenjem SelectionSort algoritma (uzastopni izbor minimuma), bez korišćenja pomoćnih struktura podataka. Čvorovi binarnog stabla sadrže samo podatak i pokazivače na potomke.

sa pravljenjem novog stabla	-	10 poena
bez pravljenja novog stabla	-	20 poena
bez promene oblika stabla	-	30 poena

U nastavku je dato jedno od rešenja studenata.

Rešenje 1

(Tijana Ristović, Lazar Krstić)

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

typedef struct stablo
{
    int broj;
    struct stablo *levi,*desni;
} stablo;

#define alloc(x) x=(stablo*)malloc(sizeof(stablo))

stablo* Min(stablo *p, stablo *tren, int *ind);
void Zameni(stablo *p,stablo *q);
void SelectionSort(stablo *p,stablo *tren);
void Sort(stablo *p);
void Stampaj(stablo *p);
void Dodaj(stablo **p,int x);
void Formiranje(stablo **p);
void UpisivanjeBrojeva(stablo *p);

main()
{
    stablo *p=NULL;

    Formiranje(&p);
    printf("\nUnesite brojeve koje zelite da upisete u stablo redosledom LEVO-KOREN-DESNO:\n");
    UpisivanjeBrojeva(p);

    printf("\nStablo pre sortiranja:\n");
    Stampaj(p);

    Sort(p);
    printf("\n\nStablo posle sortiranja:\n");
    Stampaj(p);

    printf("\n");
}

void Sort(stablo *p)
{
    SelectionSort(p,p);
}
```

```

void SelectionSort(stablo *p, stablo *tren)
{
    int ind=0;
    stablo *pom;

    if(tren)
    {
        if(tren->levi) SelectionSort(p, tren->levi);

        pom=Min(p, tren, &ind);
        Zameni(pom, tren);

        if(tren->desni) SelectionSort(p, tren->desni);
    }
}

void Zameni(stablo *p, stablo *q)
{
    int t;
    t=p->broj;
    p->broj=q->broj;
    q->broj=t;
}

stablo* Min(stablo *p, stablo *tren, int *ind)
{
    stablo *min=NULL, *min1;

    if(p)
    {
        if(p->desni)
        {
            min1=Min(p->desni, tren, ind);
            if(*ind==0 && (min1==NULL || min1->broj>p->broj)) min=p;
            else min=min1;
        }
        else min=p;

        if(p==tren) *ind=1;

        if(p->levi && *ind==0)
        {
            min1=Min(p->levi, tren, ind);
            if(min1 && min1->broj<min->broj) min=min1;
        }

        return min;
    }
}

void Stampaj(stablo *p)
{
    if(p)
    {
        Stampaj(p->levi);
        printf("%d ", p->broj);
        Stampaj(p->desni);
    }
}

void Dodaj(stablo **p, int x)
{
    stablo *pom1, *pom2, *t;

    alloc(t);
    t->broj=x;
    t->levi=t->desni=NULL;

    if(*p==NULL)

```

```

    {
        *p=t;
    }
    else
    {
        pom1=*p;
        while(pom1)
        {
            pom2=pom1;
            if(x<pom1->broj) pom1=pom1->levi;
            else pom1=pom1->desni;
        }

        if(x<pom2->broj) pom2->levi=t;
        else pom2->desni=t;
    }
}

void Formiranje(stablo **p)
{
    int x,n,i;

    printf("\nUnesite broj cvorova stabla: \n");
    scanf("%d",&n);

    printf("\nUnesite brojeve koji formiraju oblik stabla: \n");

    for(i=0;i<n;i++)
    {
        scanf("%d",&x);
        Dodaj(p,x);
    }
}

void UpisivanjeBrojeva(stablo *p)
{
    int n;
    if(p)
    {
        UpisivanjeBrojeva(p->levi);
        scanf("%d",&n);
        p->broj=n;
        UpisivanjeBrojeva(p->desni);
    }
}

```