

Programiranje i programski jezici

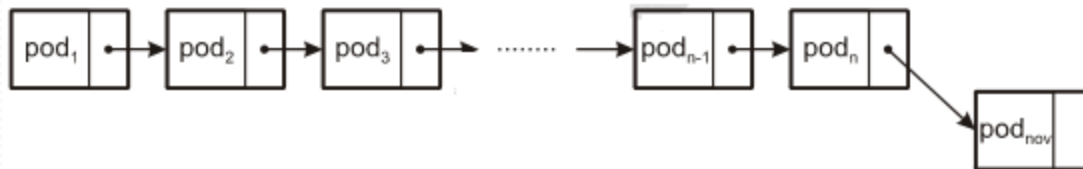


Zadatak 1:

Povezana lista od n prirodnih brojeva

```
#include <stdio.h>
#include <stdlib.h>
struct element{
    int podatak;
    struct element *sledeci;
};
```

Red



Stek



```
main()
{
    int n,i,k;
    int x,broj;
    struct element *glava=NULL, *p=NULL, *pom;
    printf("Unesi broj elemenata liste\n");
    scanf("%d",&n);
    for (i=0;i<n;i++)
    {
        printf("Unesi %d. element liste:\n",i+1);
        scanf("%d",&x);
        p=(struct element *)malloc(sizeof (struct element));
        p->podatak=x;
        p->sledeci=NULL;
        if(glava==NULL)
            glava=p;
        else
        {
            pom=glava;
            while(pom->sledeci!=NULL)
                pom=pom->sledeci;
            pom->sledeci=p;
        }
    }
}
```

Dodavanje na kraj liste

```
printf("stampana lista je:\n");
    broj=0;
    pom=glava;
    if (glava!=NULL)
    {
        while(pom!=NULL)
        {
            printf("%d\n",pom->podatak);
            pom=pom->sledeci;
            broj++;
        }
    }
    else
    {
        printf("Lista je prazna");
    }

    printf("duzina liste je:%d",broj);

}
```

STEK

```
for (i=0;i<n;i++)  
{  
    printf("Unesi %d. element liste:\n",i+1);  
    scanf("%d",&x);  
    p=(struct element *)malloc(sizeof (struct element));  
    p->podatak=x;  
    p->sledeci=glava;  
    glava=p;  
}
```

Brisanje elementa iz povezane liste

```
if (glava==NULL)
{
    printf("Lista je prazna");
}
else
{
    pom=glava;
    if (pom->podatak==k)
    {
        glava=pom->sledeci;
        free(pom);
    }
    else
    {
        while(pom->sledeci)
        {if(pom->sledeci->podatak==k)
        {
            temp=pom->sledeci;
            pom->sledeci=temp->sledeci;
            free(temp); }
        else pom=pom->sledeci;
        }
    }
}
```

Brisanje prvog elementa liste

Brisanje elementa liste

Zadatak 2:

Povezana lista od n boja (reči CRVENA, ZUTA...)

Zadatak 3:

Napisati funkciju koja formiranu povezanu listu brojeva sortira.

```
p=(struct element *)malloc(sizeof (struct element));
p->podatak=k;
p->sledeci=NULL;
if(glava==NULL)
    glava=p;
else{
    res=glava;
    if (res-> podatak >k) {p->sledeci=res;
                        res=p;
                        }
    else{
        pos=res;
        while((pos->sledeci) && (pos->sledeci-> podatak
<k))pos=pos->sledeci;
        p->sledeci=pos->sledeci;
        pos->sledeci=p;
    }
}
```


Zadatak *:

- Definirati strukturu za rad koja predstavlja povezanu listu koja sadrži sledeće podatke o studentima:

- Ime
- Prezime
- Broj indeksa
- Godina studija
- Prosek

Napisati program koji za uneti prirodan broj n i podatke za n studenata formira: sortiranu listu, ali tako da se na početku liste nalaze studenti sa najvećim prosekom, pri čemu za studente koji imaju isti prosek prednost imaju oni koji su na višoj godini studija. Na izlazu ispisati podatke za one studente koji imaju prosek veći od zadatog.

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

```
struct studenti{
    char *ime,*prezime,*br_ind;
    int godina;
    float prosek;
};
```

```
struct lista{
    struct studenti *glava;
    struct lista *rep;
};
```

```
#define novi(x) x=(struct lista *)malloc(sizeof(struct lista))
```

```
struct lista *form_sort(){
    int i,n;
    struct lista *p=NULL;
    struct studenti *q;
    p=NULL;
    scanf("%d",&n);
    for(i=0;i<n;i++){
        q=ucitaj_studenta();
        printf("Ucitan\n");
        p=dodaj_sort(p,q);
    }
    return p;
}
```

```
struct studenti* ucitaj_studenta(){
    struct studenti *st;
    char s[100];

    st=(struct studenti*)malloc(sizeof(struct studenti));
    printf("indeks: "); scanf("%s",s);
    st->br_ind=(char*)malloc(strlen(s)+1);
    strcpy(st->br_ind,s);
    printf("prezime: "); scanf("%s",s);
    st->prezime=(char*)malloc(strlen(s)+1);
    strcpy(st->prezime,s);
    printf("ime: "); scanf("%s",s);
    st->ime=(char*)malloc(strlen(s)+1);
    strcpy(st->ime,s);
    printf("godina studija: "); scanf("%d",&st->godina);
    printf("prosek: "); scanf("%f",&st->prosek);
    return st;
}
```

```

struct lista *dodaj_sort(struct lista *p, struct studenti *q){
    struct lista *temp,*res,*pos;
    novi(temp);
    if (!temp){
        printf("Greska pri alokaciji memorije\n");
        exit(0);
    }
    temp->glava=q;
    temp->rep=NULL;
    res=p;
    if(!res) res=temp;
    else{
        if ((res->glava->prosek<q->prosek) ||
            ((res->glava->prosek==q->prosek) &&
             (res->glava->godina<q->godina))){
            temp->rep=res;
            res=temp;
        }
        else {
            pos=res;
            while((pos->rep) &&
                  ((pos->rep->glava->prosek>q->prosek) ||
                   ((pos->rep->glava->prosek==q->prosek) &&
                    (pos->rep->glava->godina>q->godina))))
                pos=pos->rep;
            temp->rep=pos->rep; pos->rep=temp;
        }
    }
    return res;
}

```

```
void ispis(struct lista *p){
    struct lista *pom;
    pom=p;
    struct studenti *st;
    while(pom) {
        st=pom->glava;
        printf("%s  %s  ",st->br_ind,st->prezime);
        printf("%s ",st->ime);
        printf("%d  %f\n",st->godina,st->prosek);
        pom=pom->rep;
    }
}

main(){
    struct lista *p;
    int k;
    p=form_sort();
    printf("Ispis liste:\n");
    ispis(p);
}
```