Strukture podataka i algoritmi 1

Jun, 2015

Tri fakulteta organizuju zajednički upis studenata. Za svaki fakultet se zna koliko studenta upisuje. Svi kandidati polažu isti prijemni i na prijavi pišu najviše tri želje koji fakultet žele da upišu, po prioritetu. Pravi se zbirna rang lista svih kandidata, a zatim se počev od najboljeg kandidata vriši upis. Rang lista se ureduje u nerastuću listu po broju poena, a kandidati sa istim brojem poena se uredjuju po imenu i prezimenu. Za svakog kandidata važi isto pravilo:

- Ukoliko ima mesta na fakultetu koji je napisan kao prva želja, kandidat se upisuje, a ako ne, onda se gleda druga, pa treća želja, ako postoje;
- Ukoliko ni na jednom od navedenih fakulteta nema mesta kandidat se stavlja na spisak neupisanih kandidata.

Nakon raspoređivanja svih kandidata, neki kandidati mogu da se predomisle (bez obzira da li su upisani ili ne) i izraze želju da se upišu na drugi fakultet. Ako na željenom fakultetu ima mesta, želja će im biti ispunjena, a u suprotnom postaju neupisani.

Napisati program u kome se najpre učitavaju imena fakulteta (jedna reč) koji organizuju upis i broj studenta koji svaki od fakulteta prima. Potom se učitava broj kandidata koji su polagali prijemni i za svakog kandidata ime i prezime (jedinstven podatak), broj ostvarenih poena i jedna do tri želje (jedna do tri reči u istoj liniji), pri čemu se formira rang lista. Za svaki od fakulteta ispisati koliko kandidata želi da se upiše na taj fakultet, bez obzira da je fakultet naveden kao prva, druga ili treća želja. Kandidati se, zatim, raspoređuju po fakultetima. Svaki fakultet predstvlja niz pokazivača na kandidate koji su primljeni. Nakon rasporedjivanja, za ime fakulteta koje se zadaje na ulazu ispisuju se imena kandidata koji su upisani na taj fakultet. Zatim se unosi broj kandidata koji su žele da promene fakultet, njihovo ime i prezime i koji fakultet žele da upišu, a promene želja se razmatraju po redosledu u kom su predate. Na kraju se ispisuje ukupan broj preostalih slobodnih mesta i spisak neupisanih kandidata.
(3 poena)

Za rešavanje problema napisati sledeće funkcije:

a) Napisati funkciju **DodajKandidata** koja iz u već sortiranu rang listu kandidata dodaje novog kandidata, tako da lista ostane sortirana.
(3+2 poena)

b) Napisati funkciju **UcitajKandidate** koja iz datoteke učitava podatke o kandidatima i formira uredjenu rang listu.
(2 poena)

c) Napisati funkciju **BrojPrijavljenih** koja za zadato ime fakulteta odredjuje broj kandidata kod kojih se taj fakultet nalazi u listi želja, nezavisno da li je prva, druga ili treća želja.
(3 poena)

d) Napisati funkciju **PrebaciUNeupisane** koja zadatog kandidata briše iz rang liste i dodaje u niz neupisanih kandidata.
(3 poena)

e) Napisati funkciju **UpisiKandidata** koja odredjuje na koji fakultet kandidat može biti upisan. Ukoliko ni na jednom fakultetu sa spiska želja nema mesta, kandidat se briše iz rang liste i smesta u niz neupisanih kandidata.
(4 poena)

f) Napisati funkciju **RasporediKandidate** koja sve kandidate sa rang liste rasporedjuje po fakultetima, pri čemu formira i niz neupisanih za one koji na osnovu želja ne mogu biti upisani.
(2 poena)

g) Napisati funkciju **IspisiUpisane** koja za dato ime fakulteta ispisuje imena primljenih kandidata.
(2 poena)

h) Napisati funkciju **PromeniFakultet** koja za zadato ime i prezime kandidata i novu želju briše kandidata iz spiska sa već upisanog fakulteta i upisuje ga, ako je moguće, na nov fakultet. Ukoliko navedeni kandidat u prvom rasporedjivanju nije bio upisan, a sada se upiše, on se iz niza neupisanih prebacuje nazad u rang listu, pri čemu rang lista ne mora ostati sortirana.
(6 poena)

Zadatak rešiti bez korišćenja globalnih promenljivih i bez unapred definisanih dužina korišćenih nizova.

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

#define BROJ_F 3

struct student
{
	char *ime_prezime;
	int poeni;
	int broj_z;
	char *zelja[3];
};

struct fakultet
{
	char *ime;
	int broj;
	int broj_upisanih;
	struct student **upisani;
};

void DodajKandidata(struct student , struct student *, int );
int UcitajKandidate(struct student **);
void zelje(char *, struct student *);
int BrojPrijavljenih(char *,struct student *, int);
void PrebaciUNeupisane(struct student*, struct student *, int *, struct student *, int *);
int UpisiKandidata(struct student *, struct student *, int *, struct student *, int *,
                   struct fakultet *, int );
void RasporediKandidate(struct student *, int *, struct student *, int *,
                        struct fakultet *, int);
void IspisiUpisane(char*, struct fakultet*, int);
void PromeniFakultet(struct student, struct student *, int *, struct student *, int *,
                     struct fakultet *, int );
void PrebaciNaKraj(struct student, struct student *, struct student *, int,
                   struct fakultet *,int);
void PrepisiStudenta(struct student, struct student *);
int BrojPreostalihMesta(struct fakultet *,int);
```

```
main()
{
	struct fakultet niz[BROJ_F];
	struct student *niz_st, *neupisani;
	struct student s;
	int i,k, br_st, br_neup=0,d;
	char rec[100];
	for(i = 0; i < BROJ_F; i++)
	{
		scanf("%s",rec);
		niz[i].ime=(char*)malloc(strlen(rec)+1);
		strcpy(niz[i].ime,rec);
	}
	for(i = 0; i < BROJ_F; i++)
	{
		scanf("%d",&niz[i].broj);
		niz[i].upisani=(struct student **)malloc(sizeof(struct student*)*niz[i].broj);
		niz[i].broj_upisanih=0;
	}

	br_st=UcitajKandidate(&niz_st);

	for(i=0;i<BROJ_F;i++)
	    printf("%s %d\n",niz[i].ime,BrojPrijavljenih(niz[i].ime,niz_st,br_st));

	neupisani=(struct student *)malloc(sizeof(struct student)*br_st);
	RasporediKandidate(niz_st,&br_st,neupisani,&br_neup,niz,BROJ_F);

	for(i=0;i<BROJ_F;i++)
	{
	    printf("\n Upisani na %s\n",niz[i].ime);
	    IspisiUpisane(niz[i].ime,niz,BROJ_F);
	}
	printf("\nNeupisani\n");


	for(i=0; i<br_neup; i++) printf("%s\n",neupisani[i].ime_prezime);
	scanf("%d",&k);
	getchar();
	for(i=0;i<k;i++)
	{
	     gets(rec);
	     d=strlen(rec);
	     rec[d]='\0';
	     s.ime_prezime=(char*)malloc(sizeof(char)*(d+1));
	     strcpy(s.ime_prezime,rec);
	     gets(rec);
	     d=strlen(rec);
	     rec[d]='\0';
	     s.broj_z=1;
	     s.zelja[0]=(char*)malloc(sizeof(char)*(d+1));
	     strcpy(s.zelja[0],rec);
	     PromeniFakultet(s,niz_st,&br_st,neupisani,&br_neup,niz,BROJ_F);
	}

	printf("\n\n");
	for(i=0;i<BROJ_F;i++)
	{
	    printf("\n Upisani na %s\n",niz[i].ime);
	    IspisiUpisane(niz[i].ime,niz,BROJ_F);
	}
	printf("\n Proj preostalih mesta %d\n",BrojPreostalihMesta(niz, BROJ_F));
	printf("\nNeupisani\n");
	for(i=0; i<br_neup; i++) printf("%s\n",neupisani[i].ime_prezime);
}
```

```
void DodajKandidata(struct student s, struct student niz[], int n)
{
    int i = 0,j,k;
    while (i < n && niz[i].poeni > s.poeni) i++;
    while (i < n && niz[i].poeni == s.poeni && strcmp(s.ime_prezime,niz[i].ime_prezime)>0) i++;
    if(i < n)
         for(j=n;j>i;j--) PrepisiStudenta(niz[j-1],&niz[j]);
    PrepisiStudenta(s,&niz[i]);
}


int UcitajKandidate(struct student **niz)
{
    FILE *f = fopen("kandidati.txt","r");
    int i,j,n,d;
    char rec[100];
    struct student s;
    fscanf(f,"%d",&n);
    fgetc(f);
    *niz=(struct student*)malloc(sizeof(struct student*)*n);
    for (i = 0; i < n; i++)
    {
        fgets(rec,100,f);
        d=strlen(rec);
        rec[d-1]='\0';
        s.ime_prezime=(char*)malloc(sizeof(char)*d);
        strcpy(s.ime_prezime,rec);
        fscanf(f,"%d",&s.poeni);
        fgetc(f);
        fgets(rec,100,f);
        s.broj_z=0;
        zelje(rec,&s);
        DodajKandidata(s,*niz,i);
    }
    fclose(f);
    return n;
}

void zelje(char *r, struct student *s)
{
     int i=0,j;
     char w[100];
     while (r[i])
     {
              j=0;
              while(isalnum(r[i])) w[j++]=r[i++];
              w[j]='\0';
              if(j>1){
                      s->zelja[s->broj_z]=(char*)malloc(sizeof(char)*j);
                      strcpy(s->zelja[s->broj_z++],w);
              }
              i++;
     }
}
```

```
int BrojPrijavljenih(char *fakultet, struct student *niz, int n)
{
    int i,j, broj=0;
    for(i=0; i<n; i++)
             for(j=0;j<niz[i].broj_z;j++)
                 if(!strcmp(fakultet,niz[i].zelja[j]))
                 {
                     broj++;
                     break;
                 }
    return broj;

}

void PrebaciUNeupisane(struct student *s, struct student *niz_st, int *n,
                       struct student *niz_n, int *m)
{
     struct student *t;
     int i;
     PrepisiStudenta(*s,&niz_n[*m]);
     (*m)++;
     for(t=s; t<niz_st+(*n)-1; t++)
         PrepisiStudenta(*t,t+1);
     (*n)--;
}


int UpisiKandidata(struct student *s, struct student *niz_s, int *n, struct student *niz_n,
                   int *m, struct fakultet *niz_f, int k)
{
    int i,j;
    for(i=0; i<s->broj_z; i++)
    {
             j=0;
             while(j<k && strcmp(s->zelja[i],niz_f[j].ime)!=0) j++;
             if(j<k)
             {
                    if(niz_f[j].broj>niz_f[j].broj_upisanih)
                    {
                        niz_f[j].upisani[niz_f[j].broj_upisanih++]=s;
                        break;
                    }

             }
    }
    if(i < s->broj_z)
        return 1;
    else
    {
        PrebaciUNeupisane(s,niz_s,n,niz_n,m);
        return 0;
    }
}


void RasporediKandidate(struct student *niz_st, int *n, struct student *niz_n, int *m, struct
fakultet *niz_f, int k)
{
     int i=0,br_st;
     br_st=*n;
     while(i<*n)
         if(UpisiKandidata(&niz_st[i],niz_st,n,niz_n,m,niz_f,k)) i++;
}
```

```
void IspisiUpisane(char *ime, struct fakultet *niz_f, int n)
{
     int i=0,j;
     while(strcmp(ime,niz_f[i].ime)!=0) i++;
     for(j=0;j<niz_f[i].broj_upisanih; j++)
         printf("%s  %d\n",niz_f[i].upisani[j]->ime_prezime, niz_f[i].upisani[j]->poeni);
}


void PromeniFakultet(struct student s, struct student *niz_st, int *n, struct student *niz_n,
                     int *m, struct fakultet *niz_f, int k)
{
     int i=0,j;
     struct student t;
     while(i<*m && strcmp(s.ime_prezime,niz_n[i].ime_prezime)!=0) i++;
     if(i<*m)
     {
             s.poeni=niz_n[i].poeni;
             PrepisiStudenta(s,&niz_st[*n]);
             (*n)++;
             PrepisiStudenta(niz_n[*m-1],&niz_n[i]);
             (*m)--;
             UpisiKandidata(&niz_st[*n-1],niz_st,n,niz_n,m,niz_f,k);
             return;
     }
     for (i=0; i<k; i++)
     {
         j=0;
         while(j<niz_f[i].broj_upisanih &&
               strcmp(niz_f[i].upisani[j]->ime_prezime,s.ime_prezime)!=0) j++;
         if(j<niz_f[i].broj_upisanih)
         {
             s.poeni=niz_f[i].upisani[j]->poeni;
             PrebaciNaKraj(s,niz_f[i].upisani[j],niz_st,*n,niz_f,k);
             niz_f[i].upisani[j]=niz_f[i].upisani[niz_f[i].broj_upisanih-1];
             niz_f[i].broj_upisanih--;
             UpisiKandidata(&niz_st[*n-1],niz_st,n,niz_n,m,niz_f,k);
             return;
         }
     }
}


void PrebaciNaKraj(struct student s, struct student *t, struct student *niz_st, int n,
                   struct fakultet *niz_f,int k)
{
     int i,j;
     for(i=0; i<k; i++)
     {
              j=0;
              while (j<niz_f[i].broj_upisanih &&
                     strcmp(niz_f[i].upisani[j]->ime_prezime,niz_st[n-1].ime_prezime)!=0) j++;
              if (j<niz_f[i].broj_upisanih)
              {
                  PrepisiStudenta(niz_st[n-1],t);
                  niz_f[i].upisani[j]=t;
                  PrepisiStudenta(s,&niz_st[n-1]);
              }
     }
}
```

```
int BrojPreostalihMesta(struct fakultet *niz, int n)
{
    int i,b=0;
    for (i=0; i<n; i++)
        b+=niz[i].broj-niz[i].broj_upisanih;
    return b;
}


void PrepisiStudenta(struct student s, struct student *d)
{
     int i;
     d->ime_prezime=(char*)malloc(sizeof(char)*(strlen(s.ime_prezime)+1));
     strcpy(d->ime_prezime,s.ime_prezime);
     d->poeni=s.poeni;
     d->broj_z=s.broj_z;
     for(i=0; i<s.broj_z; i++)
     {
              d->zelja[i]=(char*)malloc(sizeof(char)*(strlen(s.zelja[i])+1));
              strcpy(d->zelja[i],s.zelja[i]);
     }
}
```