# Strukture podataka i algoritmi 1
# (zadatak - max 30 poena)

Jun, 2020

Na festival IMIFest prijavljuju se izvođači i pravi se raspored nastupa. Festival traje 3 dana i svakog dana se organizuju nastupi na svim lokacijama. Program na svim lokacijama počinje istovremeno, a između dva izvođača pravi se pauza od 15 minuta. Za svaku lokaciju se učitava ID (ceo broj), njen naziv i kratak opis (nizovi karaktera). Prilikom prijave izvođači navode svoj ID (ceo broj), ime (niz karatera), ID lokacije na kojoj nastupaju i vreme trajanja nastupa (minuti, ceo broj). Prijavljeni izvođač se raspoređuje u onaj dan festivala koji, za izabranu lopkaciju, trenutno najmanje zauzet, a termini se popunjavaju redom, po redosledu prijave. Po završenoj prijavi vrši se izbor štampe jedne od 3 vrste rasporeda nastupa: Kompletan raspored po danima i lokacijama i svim izvođačima, Raspored za izabrani dan za sve lokacije, Raspored za izabranu lokaciju za sve dane. Pri svakoj štampi vidi se dan, lokacija, naziv izvođača i početak nastupa.
Naknadno, može da se desi da neko od izvođača odustane od nastupa, pri čemu se izvođači koji su trebali da nastupaju na toj lokaciji nakon njega pomeraju na ranije.
**(3 poena)**

Za rešavanje problema napisati sledeće funkcije:

a) Definisati sve potrebne složene tipove podataka neophodne za rešavanje opisanog problema.
**(3 poena, obavezno!)**
b) Napisati funkciju **UcitajLokacije** koja iz datoteke *Lokacije.txt* učitava podatke o lokacijama i formira listu/niz lokacija.
**(3 poena, obavezno)**
c) Napisati funkciju **NadjiLokaciju** koja za dati ID lokacije vraća pokazivač na određenu lokaciju.
**(2 poena, obavezno)**
d) Napisati funkciju **UcitajIzvodjace** koja iz datoteke *Izvodjaci.txt* učitava podatke o izvođačima i formira listu/niz izvođača.
**(4 poena)**
e) Napisati funkciju **Rasporedjivanje** koja rasporedjuje izvođače po lokacijama i danima.
**(5 poena)**
f) Napisati funkcije za štampanje **StampaSve**, **StampaDan**, **StampaLokacija** koje štampaju kompletan program ili program za izabrani dan ili program za izabranu Lokaciju
**(5 poena)**
g) Napisati funkciju **ObrisiIzvodjaca** koja za zadati ID izvođača briše izvođača iz spiska izvođača.
**(5 poena)**

Pri učitavanju podataka podrazumevati da su svi podaci korektno zadati: ID-jevi jedinstveni i korektni, trajanje nastupna u prihvaljivim okvirima…
Funkcije **UcitajIzvodjace i Rasporedjivanje** možete rešiti i drugačijom implementacijom funkcija.
Dozvoljeno je proširivanje struktura i definisanje novih.

Zadatak se može rešavati korišćenjem povezanih lista, nizova ili kombinacijom.
Zadatak rešiti bez korišćenja globalnih promenljivih i bez unapred definisanih dužina korišćenih nizova.

Rešenje zadatka

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


struct lokacije{
   int ID;
   char *ime;
   char *opis;
   int vreme[3];
   struct izvodjaci *dan[3];
   struct lokacije *sledeci;
};

struct izvodjaci{
   int ID;
   char *ime;
   int dan;
   int pocetak;
   struct izvodjaci *sledeci;
};

struct lokacije* UcitajLokacije(){
   int i,n;
   char unos[50];
   struct lokacije *temp,*lst,*glava=NULL;
   FILE *in;
   in=fopen("Lokacije.txt","r");
   fscanf(in,"%d",&n);
   for (i=0;i<n;i++){
         temp=(struct lokacije*)malloc(sizeof(struct lokacije));
         fscanf(in,"%d",&(temp->ID)); fgetc(in);
         fgets(unos,50,in);
         unos[strlen(unos)-1]='\0';
         temp->ime=(char*)malloc(strlen(unos)+1);
         strcpy(temp->ime,unos);
         fgets(unos,50,in);
         unos[strlen(unos)-1]='\0';
         temp->opis=(char*)malloc(strlen(unos)+1);
         strcpy(temp->opis,unos);

         temp->vreme[0]=-15;
         temp->vreme[1]=-15;
         temp->vreme[2]=-15;

         temp->dan[0]=NULL;
         temp->dan[1]=NULL;
         temp->dan[2]=NULL;

         temp->sledeci=NULL;

         if(glava == NULL){
               glava=temp;
               lst=temp;
         }
         else{
               lst->sledeci=temp;
```

```
                lst=temp;
          }
   }
   fclose(in);
   return glava;
}

struct lokacije* NadjiLokaciju(struct lokacije *glava,int ID){
   while(glava && glava->ID!=ID)
      glava=glava->sledeci;
   return glava;
}

int NadjiDan(struct lokacije *l){
   int min,d=0;
   min=l->vreme[0];
   if(l->vreme[1]<min){
         min=l->vreme[1];
         d=1;
   }
   if(l->vreme[2]<min){
         min=l->vreme[2];
         d=2;
   }
   return d;
}

struct izvodjaci* UcitajIzvodjace(struct lokacije *pocetak){
   struct izvodjaci *glava,*temp,*lst;
   struct lokacije *lok;
   int i,n,d,l,t;
   FILE *in;
   char unos[50];
   glava=NULL;
   in=fopen("Izvodjaci.txt","r");
   fscanf(in,"%d",&n);
   for (i=0;i<n;i++){
         temp=(struct izvodjaci*)malloc(sizeof(struct izvodjaci));
         fscanf(in,"%d",&(temp->ID)); fgetc(in);
         fgets(unos,50,in);
         unos[strlen(unos)-1]='\0';
         temp->ime=(char*)malloc(strlen(unos)+1);
         strcpy(temp->ime,unos);

         temp->sledeci=NULL;

         fscanf(in,"%d",&l);
         fscanf(in,"%d",&t);
         lok=NadjiLokaciju(pocetak,l);
         d=NadjiDan(lok);

         temp->dan=d;
         lok->vreme[d]+=15;
         temp->pocetak=lok->vreme[d];
         lok->vreme[d]+=t;

         if(lok->dan[d] == NULL)
               lok->dan[d]=temp;
         else{
```

```
                lst=lok->dan[d];
                while(lst->sledeci)
                      lst=lst->sledeci;
                lst->sledeci=temp;
          }
    }
    fclose(in);
    return glava;

}

void StampaSve(struct lokacije *glava){
    struct izvodjaci *temp;
    int i;
    while(glava){
          printf("%s\n",glava->ime);
          for(i=0;i<3;i++){
                printf("Dan %d\n",i+1);
                temp=glava->dan[i];
                while(temp){
                      printf("%2d:%2d    %s\n",temp->pocetak/60+20,
                                        temp->pocetak%60,temp->ime);
                      temp=temp->sledeci;
                }

          }
    glava=glava->sledeci;
    }
}

void StampaLokacija(struct lokacije *glava,int ID){
    struct lokacije *l;
    struct izvodjaci *temp;
    int i;
    l=NadjiLokaciju(glava,ID);
    printf("%s\n",l->ime);
    for(i=0;i<3;i++){
          printf("Dan %d\n",i+1);
          temp=l->dan[i];
          while(temp){
                printf("%2d:%2d    %s\n",temp->pocetak/60+20,temp->pocetak%60,
                                        temp->ime);
                temp=temp->sledeci;
          }

    }
}

void StampaDan(struct lokacije *glava,int d){
    struct izvodjaci *temp;
    while(glava){
          printf("Dan %d\n",d);
          temp=glava->dan[d-1];
          while(temp){
                printf("%2d:%2d    %s\n",temp->pocetak/60+20,temp->pocetak%60,
                                        temp->ime);
                temp=temp->sledeci;
          }
    glava=glava->sledeci;
```

```
   }
}

void Obrisi(struct lokacije *glava, int ID){
   struct izvodjaci *preth,*temp,*rest;
   int i,t;
   while(glava){
         printf("Lokacija %d\n",glava->ID);
         for(i=0;i<3;i++){
               if(glava->dan[i]){

                     if(glava->dan[i]->ID==ID){
                           rest=glava->dan[i]->sledeci;
                           t=rest->pocetak;
                           temp=glava->dan[i];
                           glava->dan[i]=rest;
                           free(temp);
                           break;
                     }
                     else{
                           preth=glava->dan[i];
                           temp=preth->sledeci;
                           while(temp && temp->ID!=ID){
                                 preth=preth->sledeci;
                                 temp=temp->sledeci;
                           }
                           if(temp){
                                 rest=temp->sledeci;
                                 if(rest) t=temp->sledeci->pocetak-temp->pocetak;
                                 preth->sledeci=temp->sledeci;
                                 free(temp);
                                 break;
                           }
                     }
               }
         }
         glava=glava->sledeci;
   }
   while(rest){
         rest->pocetak-=t;
         rest=rest->sledeci;
   }

}

main(){
   struct lokacije *lok;
   struct izvodjaci *izv;
   int s, k;
   lok=UcitajLokacije();
   izv=UcitajIzvodjace(lok);
   printf("Stampa: 1 - sve,  2 - izabranu lokaciju,  3 - izabrani dan\n");
   scanf("%d",&s);
   if (s==1) StampaSve(lok);
   if (s==2){
         printf("Unesi ID lokacije: ");
         scanf("%d",&k);
         StampaLokacija(lok,k);
   }
```

```
   if (s==3){
         printf("Unesi dan: ");
         scanf("%d",&k);
         StampaDan(lok,k);
   }
   printf("Unesi ID izvodjaca za brisanje:");
   scanf("%d",&k);
   Obrisi(lok,k);
   StampaSve(lok);
}
```