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

Avgust, 2019

Lanac restorana za dostavu hrane ima svoje restorane na nekoliko lokacija. Lanac ima objedinjeni jelovnik za sve restorane. Jelovnik sadrži listu jela koja su u ponudi, organizovanu po tipu jela (niz karaktera) i za svaki tip se zna koliko ima jela tog tipa (ceo broj), a zatim za svako jelo su dati naziv jela (niz karaktera) i cena jela (realan broj). Ne pripremaju svi restorani sve tipove jela, ali za određeni tip pripremaju sva jela iz jelovnika koja su tog tipa. Za svaki restoran je poznat njegov naziv (niz karaktera), koliko tipova jela priprema (ceo broj), tipovi jela koja priprema (nizovi karaktera) i njegova lokacija (dva realna broja, koordinate u pravouglom koordinatnom sistemu). Takođe svaki restoran ima evidenciju o ukupnoj vrednosti isporučenih jela tog dana. Korisnik usluga ovih restorana, prilikom naručivanja, navodi svoju lokaciju (svoje koordinate), koliko različitih jela poručuje, za svako jelo navodi, tip jela, naziv jela i koliko komada tog jela naručuje. Za svaki tip jela koje je korisnik poručio određuje se najbliži restoran koji priprema taj tip, a dostava može biti objedinjena i iz više restorana. Ukoliko se sva naručena jela nalaze u ponudi, korisnik dobija informaciju o ceni kreirane porudžbine i ukoliko neka jela iz porudžbine ne postoje, njihov spisak. Porudžbine se unose čitavog dana i ne zna se unapred koliko će ih biti. U nekom trenutku neki restoran može promeniti spisak tipova jela koje nudi, koji se tog trenutka ažurira. Na kraju dana određuje se restoran koji ima najveću vrednost isporučenih jela.
(4 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 **UcitajJelovnik** koja iz datoteke *Jelovnik.txt* učitava podatke o jelima koja se nalaze u ponudi i formira listu/niz jela.
(4 poena, obavezno!)

c) Napisati funkciju **UcitajRestorane** koja iz datoteke *Restorani.txt* učitava podatke o restoranim i formira listu/niz restorana, sa odgovarajućom ponudom.
(4 poena)

d) Napisati funkciju **NadjiJelo** koja za dati tip jela i naziv jela vraća pokazivač na odgovarajuće jelo u listi/nizu jela.. Ukoliko jelo ne postoji vratiti prazan pokazivač.
(3 poena, obavezno)

e) Napisati funkciju **Porudzbina** koja kreira porudžbinu korisnika i vraća vrednost porudžbine.
(4+2 poena)

f) Napisati funkciju **PromenaPonude** koja menja ponudu nekog restorana.
(3 poena)

g) Napisati funkciju **NajboljiRestoran** koja određuje koji restoran ima nejveću vrednost isporučenih jela.
(3 poena)

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

Rastojanje tačke $A(x_1, y_1)$ od tačke $B(x_2, y_2)$ je dato sa $r = \sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2}$

Strukture podataka i algoritmi 1
Test – max 20 poena

Avgust, 2019

1. **(Obavezno!)** Svi obavezni delovi iz zadatka o restoranima.
2. Napisati funkciju koja u datu pointersku listu realnih brojeva, iza svaka dva elementa ubacuje element koji sadži prosečnu vrednost prethodna dva.
   Primer. Ulaz: 3 4 2 8 9 Izlaz: 3 4 3.5 2 7 5 9
3. Data je struktura

```
struct element{
    int d;
    struct element *par;
};
```

   I niz `struct element *a`. Svaki element niza `a`, može da pokazuje na neki element istog niza. Napisati funkciju koja će za datu poziciju elemeta u nizu `a` ispisati vrednost i indeks elemeta na koji on pokazuje.
4. Za strukturu i niz `a` iz zadatka 3 napisati funkciju koja će polazeći od prvog elementa niza `a` formirati zbir elemenata kroz koje prolazi koristeći pokazivače, dok zbir ne premaši vrednost 100 ili dok se ne stigne do elementa koji ne pokazuje nigde. Pretpostaviti da je niz dobro zadat, pa će bar jedna od uslova biti ispunjen.
5. Šta je rezultat sledećeg koda

```
#include <stdio.h>
main()
{
      int x, y = 17, d =0;
      scanf("%d%d", &x, &y);

      while (x>0)
      {
            y &= x--;
            if (!y) continue;
            d++;
      }
      printf("%d\n",d);
 }
```

   Ako se kao vrednosti promenljivih `x` i `y` unesu:

   a. 3 17 b. 15 31 c. 16 32

Rešenje studenta

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

typedef struct jelo{
   char *ime;
   int cena;
   struct jelo *sl;
}jelo;

typedef struct jelovnik{
   char *tip;
   int kolikoJ;
   jelo *nizJ;
   struct jelovnik *sl;
}jelovnik;

typedef struct restoran{
   char *ime;
   int brTipova;
   jelovnik **Tip;
   float x;
   float y;
   int ukupnaV;
   struct restoran *sl;
}restoran;

jelovnik *UcitajJelovnik(){
   int i, j, n;
   char buf[50], c;
   jelovnik *glava, *pom, *nov;
   jelo *nizJ, *pomJ, *novJ;
   FILE *f = fopen("Jelovnik.txt", "r");

   fscanf(f, "%d", &n);
   while(fgetc(f) != '\n');


   for(i = 0 ; i < n ; i++){
          nov = (jelovnik *) malloc (sizeof(jelovnik));
          ///
          fgets(buf, 50, f);
          buf[strlen(buf) - 1] = '\0';
          nov->tip = (char *) malloc (strlen(buf) * sizeof(char));
          strcpy(nov->tip, buf);
          ///
          fscanf(f, "%d", &nov->kolikoJ);
          while(fgetc(f) != '\n');
          ///
          nizJ = NULL;
          for(j = 0 ; j < nov->kolikoJ ; j++){
                 novJ = (jelo *) malloc (sizeof(jelo));
                 ///
                 fgets(buf, 50, f);
                 buf[strlen(buf) - 1] = '\0';
                 novJ->ime = (char *) malloc (strlen(buf) * sizeof(char));
                 strcpy(novJ->ime, buf);
                 ///
                 fscanf(f, "%d", &novJ->cena);
                 while((c = fgetc(f)) != '\n' && c != EOF);
                 ///
```

```
                novJ->sl = NULL;
                if(nizJ == NULL)
                        nizJ = novJ;
                else
                        pomJ->sl = novJ;
                pomJ = novJ;
        }
        nov->nizJ = nizJ;
        ///
        nov->sl = NULL;
        if(glava == NULL)
                glava = nov;
        else
                pom->sl = nov;
        pom = nov;
    }
    fclose(f);
    return glava;
}
restoran *UcitajRestorane(jelovnik *jel){
    int i, j, n;
    char buf[50], c;
    restoran *nizR, *pomR, *novR;
    jelovnik *pomJ;
    FILE *f = fopen("Restorani.txt", "r");

    fscanf(f, "%d", &n);
    while(fgetc(f) != '\n');

    for(i = 0 ; i < n ; i++){
        novR = (restoran *) malloc (sizeof(restoran));
        ///
        fgets(buf, 50, f);
        buf[strlen(buf) - 1] = '\0';
        novR->ime = (char *) malloc (strlen(buf) * sizeof(char));
        strcpy(novR->ime, buf);
        ///
        fscanf(f, "%d", &novR->brTipova);
        while(fgetc(f) != '\n');
        ///
        novR->Tip = (jelovnik **) malloc (sizeof(jelovnik *));
        for(j = 0 ; j < novR->brTipova ; j++){
                fgets(buf, 50, f);
                buf[strlen(buf) - 1] = '\0';
                pomJ = jel;
                while(strcmp(pomJ->tip, buf) != 0)
                        pomJ = pomJ->sl;
                novR->Tip[j] = pomJ;
        }
        ///
        fscanf(f, "%f%f", &novR->x, &novR->y);
        while((c = fgetc(f)) != '\n' && c != EOF);
        novR->ukupnaV = 0;
        ///
        novR->sl = NULL;
        if(nizR = NULL)
                nizR = novR;
        else
                pomR->sl = novR;
        pomR = novR;
    }
    return nizR;
}
jelo *NadjiJelo(jelovnik *jel, char *tipJ, char *imeJ){
    jelo *jelo = NULL, *pomJ;
```

```
   while(jel != NULL && strcmp(jel->tip, tipJ) != 0)
          jel = jel->sl;
   if(jel != NULL){
          pomJ = jel->nizJ;
          while(pomJ != NULL)
                if(strcmp(pomJ->ime, imeJ) == 0){
                      jelo = pomJ;
                      break;
                }
   }
   return jelo;
}
float distance(float x, float y, float a, float b){
   return sqrt((x - a)*(x - a) + (y - b)*(y - b));
}
int Porudzbina(jelovnik *jel, restoran *nizR){
   int i, j, brK, brJ, cena = 0;
   float x, y, d;
   char tip[50], imeJ[50];
   jelo *spisak = NULL, *pomJ, *novJ, *trazenoJ;
   restoran *najR, *pomR;

   scanf("%f%f%d", &x, &y, &brJ);
   while(fgetc(stdin) != '\n');

   for(i = 0 ; i < brJ ; i++){
          fgets(tip, 50, stdin);
          tip[strlen(tip) - 1] = '\0';
          fgets(imeJ, 50, stdin);
          tip[strlen(imeJ) - 1] = '\0';
          scanf("%d", &brK);
          while(fgetc(stdin) != '\n');
          ///
          trazenoJ = NadjiJelo(jel, tip, imeJ);
          if(trazenoJ == NULL){
                novJ = (jelo *) malloc (sizeof(jelo));
                novJ->ime = (char *) malloc (strlen(imeJ) * sizeof(char));
                strcpy(novJ->ime, imeJ);
                if(spisak == NULL)
                      spisak = novJ;
                else
                      pomJ->sl = novJ;
                pomJ = novJ;
          }
          else{
                cena += trazenoJ->cena * brK;
                pomR = nizR; najR = nizR;
                d = distance(x, y, pomR->x, pomR->y);

                while(pomR != NULL){
                      for(j = 0 ; j < pomR->brTipova ; j++)
                            if(strcmp(pomR->Tip[j]->tip, tip) == 0){
                                  if(distance(x, y, pomR->x, pomR->y) < d){
                                        d = distance(x, y, pomR->x, pomR->y);
                                        najR = pomR;
                                  }
                                  break;
                            }
                      pomR = pomR->sl;
                }
                najR->ukupnaV += trazenoJ->cena * brK;
          }
   }
   printf("Ukupna cena porudjbine iznosi : %d\n", cena);
```

```
   printf("Sledeca porucena jela ne postoje :\n");
   pomJ = spisak;
   while(pomJ != NULL){
         printf("%s\n", pomJ->ime);
         pomJ = pomJ->sl;
   }
   return cena;
}
void PromenaPonude(jelovnik *jel, restoran **R){
   int i;
   char buf[50];
   jelovnik *pomJ;

   free((*R)->Tip);
   scanf("%d", &(*R)->brTipova);
   (*R)->Tip = (jelovnik **) malloc (sizeof(jelovnik *));

   for(i = 0 ; i < (*R)->brTipova ; i++){
         fgets(buf, 50, stdin);
         buf[strlen(buf) - 1] = '\0';
         pomJ = jel;
         while(strcmp(pomJ->tip, buf) != 0)
               pomJ = pomJ->sl;
         (*R)->Tip[i] = pomJ;
   }
}
void NajboljiRestoran(restoran *nizR){
   restoran *najR = nizR;

   while(nizR != NULL){
         if(nizR->ukupnaV > najR->ukupnaV)
               najR = nizR;
         nizR = nizR->sl;
   }
   printf("Najbolji restoran je %s sa zaradom od %d", najR->ime, najR->ukupnaV);
}

int main(){
    char buf[50];
   jelovnik *J = UcitajJelovnik();
   restoran *R = UcitajRestorane(J), *pomR;

    while(1){
        scanf("%s", buf);
        if(strcmp(buf, "PromenaPonude") == 0){
            scanf("%s", buf);
            pomR = R;
            while(pomR != NULL && strcmp(pomR->ime, buf) != 0)
                pomR = pomR->sl;
            if(pomR == NULL)
                printf("dati restoran ne postoji");
            else
                PromenaPonude(J, &pomR);
        }
        else
            Porudzbina(J, R);
    }
    NajboljiRestoran(R);
}
```