JMBAG

IME I PREZIME

Programiranje 2 – drugi kolokvij, 27. 6. 2025.

Upute: Na skraćenom ispitu je dozvoljeno koristiti samo pribor za pisanje i brisanje, te službeni podsjetnik. Kalkulatori, razne neslužbene tablice, papiri i sl., nisu dozvoljeni! **Mobitele isključite i spremite!** Sva rješenja napišite isključivo na papire sa zadacima, jer jedino njih predajete. Obavezno predajte sve papire sa zadacima, čak i ako neke zadatke niste rješavali. Ne zaboravite se **potpisati** na svim papirima! Skice smijete raditi i na drugim papirima koje će vam dati dežurni asistent.

Zadatak 1 (12 bodova) Napišite program koji učitava nazive ulazne i izlazne datoteke te dva stringa, `stari` i `novi`. Program u ulaznoj datoteci zamjenjuje sva pojavljivanja stringa `stari` stringom `novi` i zapisuje rezultat u izlaznu datoteku. Smijete pretpostaviti da string `stari` ne sadrži znak `'\n'` (može sadržavati razmake) te da su sve linije u ulaznoj datoteci kraće od 100 znakova. **Za dodatnih 5 bodova**, neka program prima nazive ulazne i izlazne datoteke te dva stringa kao argumente komandne linije.

Primjer:

`stari`="taubeka"
`novi`="goluba"

`Ulazna.txt`

Bledi je mesec hodil po krovu,
A ja sem popeval popevkicu svoju:
Kak taubeka dva mi srečni smo bili,
Z jednoga peharčeka ljubav smo pili.
Kak taubeka dva mi srečni smo bili,
Z jednoga peharčeka ljubav smo pili

`Izlazna.txt`

Bledi je mesec hodil po krovu,
A ja sem popeval popevkicu svoju:
Kak goluba dva mi srečni smo bili,
Z jednoga peharčeka ljubav smo pili.
Kak goluba dva mi srečni smo bili,
Z jednoga peharčeka ljubav smo pili

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

int main(int argc, char *argv[]) {
    const char *ulazna = argv[1];
    const char *izlazna = argv[2];
    const char *stari = argv[3];
    const char *novi = argv[4];

    FILE *fin = fopen(ulazna, "r");
    if (!fin) {
        perror("Greska pri otvaranju ulazne datoteke");
        return 1;
    }
    FILE *fout = fopen(izlazna, "w");
    if (!fout) {
        perror("Greska pri otvaranju izlazne datoteke");
        return 1;
    }

    char linija[200];
    char linija_novi[200];
    int length_stari = strlen(stari);
    int length_novi = strlen(novi);
    while (fgets(linija, 200, fin)) {

        int i=0;
        int i_novi=0;
        while(linija[i] != '\n'){
            if (linija[i] == stari[0]){
                int contr=1;
                for (int j=0; j < length_stari ; ++j)
                    if (linija[i+j] != stari[j]) {contr=0; break;}
                if (contr) {
                    i+=length_stari;
                    for (int j=0; j<length_novi; ++j)
                        linija_novi[i_novi+j]=novi[j];
                    i_novi += length_novi;
                }
                else {
                    linija_novi[i_novi]=linija[i];
                    i_novi++;
                    i++;
                }

            }
            linija_novi[i_novi]=linija[i];
            i_novi++;
            i++;

        }

        fputs(linija_novi, fout);
    }

    fclose(fin);
    fclose(fout);
    return 0;
}
```

JMBAG

IME I PREZIME

Programiranje 2 – drugi kolokvij, 27. 6. 2025.

Zadatak 2 (2+6+6 bodova) Na jednom sveučilištu studenti imaju pravo na subvencioniranu prehranu. Podaci o subvenciji nalaze se na tzv. "iksici", koja sadrži identifikacijski broj studenta - JMBAG (string duljine 10) te iznos subvencije za troškove prehrane na koji student ima pravo (nenegativan realan broj). Prilikom korištenja iksice, 70% iznosa računa podmiruje se sredstvima s iksice (tada se sredstva na iksici umanjuju za taj iskorišteni iznos). Jelovnik u restoranu predstavljen je kao niz jela, pri čemu svako jelo ima svoje ime (string duljine najviše 30) te cijenu (pozitivan realan broj). Svaki restoran bilježi promet tijekom jednog dana u obliku niza računa. Račun sadrži broj odabranih jela (prirodan broj), popis odabranih jela (niz jela), vrijeme kupnje zapisano u formatu `hh:mm` (string duljine 5) te ukupan iznos računa (nenegativan realan broj).

(a) Definirajte tipove `iksica`, `jelo` i `racun`, tako da bude moguća deklaracija `iksica x`, `jelo j` i `racun r`.

(b) Napišite funkciju

```
racun *provuci_iksicu(iksica *x, char vrijeme[], jelo jelovnik[], int m, char* odabir[], int k)
```

koja prima pokazivač na iksicu, string duljine 5 u kojem je zapisano vrijeme u formatu `hh:mm`, jelovnik duljine m te niz stringova duljine k. Ti stringovi predstavljaju imena odabranih jela s jelovnika. Funkcija izrađuje novi račun i vraća pokazivač na njega te ispisuje: `Za platiti:x` gdje je `x` iznos koji **student treba platiti** (nakon uračunate subvencije). Funkcija također treba smanjiti iznos subvencije na iksici za 70% iznosa računa, kako je prethodno opisano. Ako student **nema dovoljno sredstava** na iksici, funkcija vraća `NULL` i ispisuje: `ODBIJENO`

(c) Napišite funkciju `void vrijeme_sort(racun restoran[], int n)` koja prima niz računa restorana duljine n tijekom jednog dana i sortira ga uzlazno prema vremenu izdavanja.

Napomene:

- Zabranjeno je korištenje dodatnih nizova. U (b) dijelu zadatku možete alocirati memoriju samo za potrebu izrade računa. Ne treba provjeravati uspješnost (re)alokacije. Uz standardnu biblioteku `stdio.h`, dozvoljeno je korištenje biblioteke `stdlib.h` i `string.h`.
- Podzadatak (c) nosi **6** bodova, međutim ukoliko se umjesto algoritma sortiranja složenosti $\mathcal{O}(n \log n)$ koriste algoritmi složenosti $\mathcal{O}(n^2)$, zadatak nosi maksimalno **2** bodova. Korištenje quickSort algoritma iz biblioteke `stdlib.h` nosi dodatnih **4** boda, odnosno tako riješen zadatak u sumi nosi **10** bodova.

```
racun *provuci_iksicu(iksica *x, char vrijeme[], jelo jelovnik[], int m, char* odabir[], int k){

    racun *novi = (racun*)malloc(sizeof(racun));
    novi->br_jela = k;
    strcpy(novi->vrijeme, vrijeme);

    // Ostaje dodati jela i iznos racuna.
    jelo *p = novi->popis;
    p = NULL;
    double iznos = 0.0;
    int i, j;
    for(i = 0; i < k; i++){
        p = realloc(p, (i+1)*sizeof(jelo));
        for(j = 0; j < m; j++){
            if(!strcmp(jelovnik[j].ime, odabir[i])){
                p[i] = jelovnik[j];
                iznos += jelovnik[j].cijena;
            }
        }
    }
    novi->popis = p;
    novi->iznos = iznos;

    // Ako nema dovoljno subvencije.
    if(0.7*iznos > x->subvencija){
        free(p);
        free(novi);
        printf("ODBIJENO\n");
        return NULL;
    }

    x->subvencija -= 0.7*iznos;
    printf("Za platiti: %.2f\n", 0.3*iznos);
    return novi;
}

void vrijeme_sort(racun restoran[], int n){
    qsort(restoran, n, sizeof(racun), comp);
    return;
}
```

JMBAG

IME I PREZIME

Programiranje 2 – drugi kolokvij, 27. 6. 2025.

Zadatak 3 (2+4+6 bodova) Sve pjesme koje pripadaju popisu za reproduciranje (playlisti) su spremljene u vezanoj listi. Za svaku pjesmu pamtimo naziv (string s najviše 20 znakova), izvođača (string s najviše 30 znakova), trajanje u sekundama (cijeli broj) i je li označena kao omiljena (cijeli broj koji ima vrijednost 1 ako je i 0 ako nije).

(a) Napišite deklaraciju odgovarajućeg tipa podataka za pjesmu na način da bude moguće definirati varijablu naredbom `pjesma p;`. Definirajte samo podatke koji su nužni za čuvanje takve liste u memoriji.

(b) Napišite funkciju `pjesma* izdvoji_omiljene(pjesma* playlist)` koja prima pokazivač na listu pjesama. Funkcija stvara novu vezanu listu koja sadrži sve pjesme iz liste `playlist` koje su označene kao omiljene i vraća pokazivač na početak novostvorene liste, a ako nema takvih pjesama onda vraća `NULL`. Za elemente nove liste je potrebno alocirati memoriju, te izvorna lista `playlist` ne smije biti promijenjena.

(c) Napišite funkciju `void premjesti_kratke(pjesma** playlist, pjesma** playlist_kratke, int max_sec)`. Funkcija prima dvostruki pokazivač na popis pjesama `playlist`, dvostruki pokazivač `playlist_kratke` preko kojeg će se vratiti lista kratkih pjesama, te `max_sec` koji definira maksimalno trajanje u sekundama za koje se pjesma smatra *kratkom* (kratka je ako traje `<= max_sec`). Lista na koju pokazuje `*playlist_kratke` inicijalno je prazna (`NULL`) i funkcija je popunjava.

Funkcija treba premjestiti sve kratke pjesme iz liste `*playlist` u listu na koju pokazuje `*playlist_kratke`. Premještanje treba izvršiti **bez alociranja dodatne memorije**, tj. preslagivanjem postojećih čvorova.

Napomena:

Dozvoljeno je korištenje pomoćnih funkcija. Nije dozvoljeno korištenje pomoćnih nizova. Možete pretpostaviti da su sve ulazne liste ispravno formirane. Jedine dozvoljene biblioteke u ovom zadatku su `stdlib.h` i `string.h`.

Primjer:

- `moja_playlista -> {"HeyJude", "TheBeatles", 431, 1} -> {"Stairway", "LedZeppelin", 482, 0} -> {"Imagine", "JohnLennon", 183, 1} -> {"Yesterday", "TheBeatles", 125, 0} -> {"LightMyFire", "TheDoors", 428, 1} -> NULL`

```
pjesma* omiljene = izdvoji_omiljene(moja_playlista);
```

- `omiljene->{"HeyJude", "TheBeatles", 431, 1} -> {"Imagine", "JohnLennon", 183, 1} -> {"LightMyFire", "TheDoors", 428, 1} -> NULL`

```
premjesti_kratke(&moja_playlista, &lista_kratkih, 200);
```

- `lista_kratkih -> {"Imagine", "JohnLennon", 183, 1} -> {"Yesterday", "TheBeatles", 125, 0} -> NULL`
- `moja_playlista -> {"HeyJude", "TheBeatles", 431, 1} -> {"Stairway", "LedZeppelin", 482, 0} -> {"LightMyFire", "TheDoors", 428, 1} -> NULL`

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

typedef struct _pjesma
{
    char naziv[21];
    char izvodac[31];
    int trajanje;
    int omiljena;

    struct _pjesma *next;
} pjesma;

pjesma *izdvoji_omiljene(pjesma *playlist)
{
    if (playlist == NULL)
        return NULL;

    pjesma *ret_first = NULL;
    pjesma *ret_last = NULL;

    for (pjesma *p = playlist; p != NULL; p = p->next) {
        if (p->omiljena) {
```

```
            pjesma *nova = (pjesma *) malloc(sizeof(pjesma));
            strcpy(nova->naziv, p->naziv);
            strcpy(nova->izvodac, p->izvodac);
            nova->trajanje = p->trajanje;
            nova->omiljena = p->omiljena;
            nova->next = NULL;

            if (ret_last == NULL) {
                // ret lista je trenutno prazna
                ret_first = ret_last = nova;
            } else {
                ret_last->next = nova;
                ret_last = nova;
            }
        }
    }

    return ret_first;
}

void premjesti_kratke(pjesma **playlist, pjesma **playlist_kratke, int max_sec)
{
    pjesma *kratke_first = NULL;
    pjesma *kratke_last = NULL;

    // Iteriramo po playlist
    pjesma *prev = NULL;
    pjesma *p = *playlist;
    while (p) {
        if (p->trajanje > max_sec) {
            // Nije kratka, ne moramo nista raditi s pjesmom p
            // nastavljamo s iteriranjem
            prev = p;
            p = prev->next;
            continue;
        }

        // Moramo premjestiti pjesmu p na kraj liste kratkih
        pjesma *tmp = p->next;
        p->next = NULL;
        if (kratke_first == NULL) {
            kratke_first = kratke_last = p;
        } else {
            kratke_last->next = p;
            kratke_last = p;
        }

        p = tmp;
        if (prev == NULL) {
            // Prvi element se promijenio
            *playlist = p;
        } else {
            prev->next = p;
        }
    }

    *playlist_kratke = kratke_first;
}
```

JMBAG

IME I PREZIME

Programiranje 2 – drugi kolokvij, 27. 6. 2025.

Zadatak 4 (5+8 bodova) Nakon sajma čokolada, posjetitelji su isprobanim čokoladama dodjeljivali ocjene od 1 do 5. Podaci o ocjenjenoj čokoladi su spremljeni u strukturi `cokolada` koja sadrži informacije o udjelu kakaa (cjelobrojna varijabla) koji može biti broj od 0 do 100 uključivo, dodatku (string od najviše 20 znakova) koji može biti točno jedan od sljedećih stringova: `ljesnjak`, `badem`, `keks` te dodijeljenoj ocjeni (cjelobrojna varijabla). Udio kakaa i dodatak **jedinstveno** određuju čokoladu.

Jedna je tvornica čokolade spremila rezultate anketiranja u binarnu datoteku `anketa.bin` u obliku niza struktura tipa `cokolada`.

(a) Definirajte strukturu `cokolada` čiji su podaci opisani ranije. Struktura mora biti definirana tako da bude moguća deklaracija oblika `cokolada c;` te smije sadržavati samo zadane podatke.

(b) Napišite implementaciju funkcije s prototipom

```
void najdraza(FILE* in, FILE* out)
```

koja kao argumente prima pokazivače na datoteke. Pri tome, `in` je pokazivač na binarnu datoteku `anketa.bin` otvorenu za čitanje, a `out` je pokazivač na tekstualnu datoteku otvorenu za pisanje. Funkcija treba u datoteku na koju pokazuje `out` upisati maksimalnu prosječnu ocjenu ostvarenu među svim čokoladama. Preciznije, prosječnu ocjenu čokolade računamo kao omjer sume ocjena za tu čokoladu kroz broj ocjena za tu čokoladu.

Primjer:
Za binarnu datoteku `anketa.bin` sa podacima:

`{ljesnjak,40,5}, {keks,30,4}, {ljesnjak,60,3}, {ljesnjak,40,3}, {badem,80,2}, {ljesnjak,60,5}`

maksimalna prosječna ocjena je 4 (ostvarena za čokolade `ljesnjak,40`, `ljesnjak,60` i `keks,30`) pa u tekstualnu datoteku na koju pokazuje `out` treba upisati 4.000 (nije bitno na koliko decimala).

Napomena: Nije dozvoljeno korištenje pomoćnih polja ili datoteka. Dozvoljeno je korištenje pomoćnih varijabli tipa `cokolada`, kao i implementacija pomoćnih funkcija.

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

typedef struct
{
    int udio;
    char dodatak[21];
    int ocjena;
}cokolada;

double prosjek(FILE* in, cokolada tmp)
{
    cokolada pom;
    int br=0;
    double suma=0;

    rewind(in);

    while(fread(&pom,sizeof(cokolada),1,in)==1)
    {
        if(strcmp(pom.dodatak,tmp.dodatak)==0 && pom.udio==tmp.udio)
        {
            br++;
            suma+=pom.ocjena;
        }
    }

        suma/= (double)br;
        return suma;

}
```

```c
void najdraza(FILE* in, FILE* out)
{
    int poz;
    double max=0, max_tren;
    cokolada pom;

    rewind(in);
    poz=ftell(in);

    while(fread(&pom,sizeof(cokolada),1,in)==1)
    {
        poz=ftell(in);
        max_tren=prosjek(in,pom);

        if(max<max_tren) max=max_tren;

        fseek(in,poz,SEEK_SET);
    }

    fprintf(out,"%f",max);
}
```