Uvod u programiranje

Završni ispit 2021/22

drugi termin

9 bodova

Napisati definiciju funkcije `nulovanje` koja će pronaći redak i stupac u kojem se nalazi najveća vrijednost dvodimenzionalne cjelobrojne matrice s `m` redaka i `n` stupaca čiji su elementi tipa `int`, te postaviti sve elemente u tom retku i tom stupcu na nula.

Ako više elemenata matrice ima maksimalnu vrijednost, odabrati onaj s **najvećim** indeksom retka, a ako imaju i isti indeks retka, onda onaj s **najmanjim** indeksom stupca.

Funkcija preko imena treba vratiti zbroj vrijednosti svih elemenata koji su zamijenjeni nulama.

Primjer 1:

Zadano polje:

```
1 2 3 4
5 6 7 8
9 0 9 2
1 2 3 4
```

treba promijeniti u:

```
0 2 3 4
0 6 7 8
0 0 0 0
0 2 3 4
```

pri čemu funkcija vraća vrijednost 27.

Primjer 2:

Zadano polje:

```
7 2 7 5 7
6 7 3 7 -1
```

treba promijeniti u:

```
7 0 7 5 7
0 0 0 0 0
```

pri čemu funkcija vraća vrijednost 24.

Napomena: Kao rješenje dostaviti samo definiciju funkcije `nulovanje`, odnosno dio programskog koda koji se u dolje prikazanom modulu nalazi između `/* POCETAK DEFINICIJE */` i `/* KRAJ DEFINICIJE */`.

```c
#include <stdio.h>

/* POCETAK DEFINICIJE */
// m - broj redaka matrice
ovdje_navesti_tip_funkcije nulovanje(ovdje_navesti_prvi_parametar, int m, int n) {
   ovdje_napisati_tijelo_funkcije
}
/* KRAJ DEFINICIJE */

int main(void) {
   ...
}
```

Correct answer:

```
int nulovanje(int *p, int m, int n) {

   int r = m-1, s = 0, sum = 0;

   for (int i = m-1; i >= 0; --i) {
      for (int j = 0; j < n; ++j) {
         if (p[i * n + j] > p[r * n + s]) {
            r = i;
            s = j;
         }
      }
   }
   for (int i = 0; i < m; ++i) {
      sum += p[i * n + s];
      p[i * n + s] = 0;
   }
   for (int j = 0; j < n ; ++j) {
      sum += p[r * n + j];
      p[r * n + j] = 0;
   }
   return sum;
}
```

9 bodova

Potrebno je napisati funkciju koja zbraja n vektora definiranih u Kartezijevom koordinatnom sustavu preko vektorskih komponenti paralelnih s osima. Svaka vektorska komponenta predstavljena je umnoškom skalarne komponente i jediničnog vektora, primjerice $\vec{a} = 5 \cdot \vec{i} + 3 \cdot \vec{j}$. Za pohranu podataka o skalarnim komponentama vektora koristi se struktura

```
struct vektor_s {
    int komp_i;
    int komp_j;
};
```

Funkcija kao parametar prima pokazivač na prvi član polja koje sadrži vektore i definirano je u glavnom programu i broj vektora u polju, te preko pokazivača vraća rezultat. Prototip funkcije je:

```
void SumaVektora(struct vektor_s *pok, int n, struct vektor_s *rez);
```

Napomena: sumacija vektora provodi se sumiranjem skalarnih komponenti odgovarajućeg jediničnog vektora.

Primjer

Ako je u glavnom programu definirano polje sa 7 vektora ($n = 7$) sljedećih skalarnih komponenti:

(1, -3), (-2, 8), (4, 5), (3, -14), (2, 2), (6, -6), (1, -1)

funkcija u pozivajući program vraća vektor sa skalarnim komponentama (15,-9).

Correct answer:

```
void SumaVektora(struct vektor_s *pok, int n, struct vektor_s *rez) {
        rez->komp_i = 0;
        rez->komp_j = 0;
        int i;
        for (i = 0; i < n; i = i + 1) {
            rez->komp_i += (pok + i)->komp_i;
            rez->komp_j += (pok + i)->komp_j;
        }
}

```

9 bodova

Napisati funkciju s prototipom `void igra(int seed, int brojBacanja, int *pobjednik, int *razlika);` koja vraća pobjednika igre bacanja kocke između dva igrača, igrača 1 i igrača 2, te ostvarenu (uvijek nenegativnu) razliku bodova. Igrači naizmjence bacaju jednu kocku, svaki od njih zadani broj puta (`brojBacanja`), počevši s igračem 1.
Pobjednik je igrač koji ukupno ostvari više bodova. Ako nema pobjednika, rezultat je 0.
`seed` je inicijalna vrijednost za generator slučajnih brojeva.

Važno: za transformaciju pseudoslučajnog broja u željeni interval koristiti operator `%`, a postavljanje inicijalne vrijednosti u funkciji napraviti samo jednom, na zadanu vrijednost.

Predati samo definiciju funkcije, bez naredbi `#include` i glavnog programa.

Napomena: konkretan slijed pseudolučajnih brojeva, a time i konkretan rezultat, razlikovat će se u Edgaru i na lokalnom računalu za isti `seed` , ali za korektno napisanu funkciju rezultat će u Edgaru odgovarati očekivanome.

Primjer:

Za zadani broj bacanja jednak 3 (`brojBacanja`), ako redoslijed ishoda bude npr. 1, 2, 3, 4, 5, 6, funkcija kao pobjednika vraća cijeli broj 2 i razliku 3 jer je prvi igrač ostvario rezultat (1 + 3 + 5) = 9, a drugi (2 + 4 + 6) = 12, pa je pobjednik igrač 2.

Correct answer:

```c
void igra(int seed, int brojBacanja, int *pobjednik, int *razlika) {
    int r1 = 0, r2 = 0, r;
    srand(seed);
    for (int i = 0; i < brojBacanja; i++) {
        r1 += rand() % 6 + 1;
        r2 += rand() % 6 + 1;
    }
    r = r1 - r2;
    if (r == 0) {
        *pobjednik = 0;
    } else if (r > 0) {
        *pobjednik = 1;
    } else {
        *pobjednik = 2;
    }
    *razlika = abs(r);
}
```

9 bodova

Načiniti funkciju s prototipom `_Bool txt2bin(char *inputFile, char *outputFile)` koja iz tekstne datoteke s imenom *inputFile* stvara binarnu datoteku s imenom *outputFile*. Funkcija treba vratiti *1* ako je obavljena uspješno, a *0* ako nije uspjelo otvaranje nekog od potrebnih tokova.

Zapisi u *inputFile* su oblika ccxxxyyy gdje je cc šifra točke (dvije znamenke, cijeli broj), xxx x-koordinata točke (cijeli broj), yyy y-koordinata točke (cijeli broj).

Zapisi u *outputFile* trebaju biti oblika

```
struct record {
    int code;
    int x;
    int y;
};
```

Redni broj zapisa odgovara šifri točke. Zapis sa šifrom 1 treba biti na samom početku direktne datoteke.

Napomena: predati samo izvorni kod funkcije, bez deklaracije strukture i naredbi `#include`. Glavni program, deklaracija strukture i naredbe `#include` kojima se testira funkcionalnost funkcije već su pripremljeni u Edgaru. U funkciji obvezno zatvoriti tokove, jer inače načinjene promjene neće biti spremljene ni vidljive programu koji testira funkciju. **Nije dozvoljeno korištenje polja.**

Primjer ulazne datoteke nalazi se u privitku.

Correct answer:

```c
_Bool txt2bin(char *inputFile, char *outputFile) {
    FILE *fi, *fo;
    struct record r;
    fi = fopen(inputFile, "r");
    if (!fi)
        return 0;
    fo = fopen(outputFile, "wb");
    if (!fo)
        return 0;
    while (fscanf(fi, "%2d%3d%3d", &r.code, &r.x, &r.y) == 3) {
        fseek(fo, (r.code - 1) * sizeof(r), SEEK_SET);
        fwrite(&r, sizeof(r), 1, fo);
    }
    fclose(fi);
    fclose(fo);
    return 1;
}
```

2 boda

Koju naredbu treba dodati na mjesto znakova `XXXXXX` kako bi omogućili da funkcija `f` tijekom prvih 5 poziva vraća broj 1, a za svaki od sljedećih poziva broj 0?

```c
#include <stdio.h>

int f(void) {
    XXXXXX
    int rez = 1;
    if (i > 5) rez = 0;
    else i++;
    return rez;
}

int main(void) {
    int i;
    for (i = 0; i < 10; i++) printf("%d", f());  // treba ispisati 1111100000
    return 0;
}
```

Student's answer:

```c
static int i = 1;
```

Hint: Correct. Well done!

2 boda

Funkcija

```
void maxDjelIMinVisek (int broj1, int broj2, int *djelitelj, int *visekratnik);
```

izračunava i preko parametra vraća najveći zajednički djelitelj (`djelitelj`) i najmanji zajednički višekratnik (`visekratnik`) dva broja koje funkcija prima kao parametre `broj1` i `broj2`.
Čime u donjem programskom odsječku treba zamijeniti POZIVFUNKCIJE, kako bi se printf naredbom ispisao najveći zajednički djelitelj i najmanji zajednički višekratnik brojeva `prvi` i `drugi`:

```
int main(void) {
    int prvi, drugi, djel, visek;
    scanf("%d %d", &prvi, &drugi);
    POZIVFUNKCIJE
    printf("Djelitelj: %d, višekratnik: %d. \n", djel, visek);
    return 0;
}
```

Student's answer:

```
1 maxDjelIMinVisek(prvi, drugi, &djel, &visek);
```

Hint: Correct. Well done!

Correct answer:

maxDjelIMinVisek (prvi, drugi, &djel, &visek);
ili
maxDjelIMinVisek (drugi, prvi, &djel, &visek);

2 boda

Što u pozivu funkcije fopen treba napisati kao drugi argument ako se želi dodavati podatke na kraj postojeće tekstne datoteke (bez potrebe za pozivom funkcije `fseek`), ali i čitati iz te datoteke? U odgovoru napisati odgovarajući konstantni znakovni niz, s navodnicima.

Student's answer:

```
1 "a+"
```

Hint: Correct. Well done!