Ispit iz Programiranja 2

Ispit traje 135 minuta

Napomene:

a) Pažljivo proučite Uputstvo pre popunjavanja Obrasca za odgovore.

b) Vrednost odgovora: tačan = **5**; netačan = **-1.25**; nevažeći (nula ili više zacrnjenih kružića) = **0.**

c) Na pitanjima se može osvojiti najviše **20** poena. Prvi zadatak nosi **20** poena, dok drugi nosi **25** poena.

I ZADACI

1)Napisati program na programskom jeziku C koji rotira matricu za 90 stepeni u smeru kazaljke na satu. Program na početku učitava dimenzije matrice, a zatim i njene elemente po vrstama. Smatrati da su svi uneti podaci korektni. Za učitavanje matrice neophodno je realizovati funkciju `int** readMatrix(int *numrows, int *numcols);`. Nakon toga je potrebno izvršiti zahtevanu rotaciju matrice realizacijom funkcije `int** rotateMatrix90(int **matrix, int numrows, int numcols);`, koja treba da vrati novu, rotiranu matricu. Glavni program treba da pozove funkciju za učitavanje matrice sa standardnog ulaza, zatim pozove funkciju za formiranje nove, rotirane matrice i na kraju ispiše rotiranu matricu na standardnom izlazu. Voditi računa o ispravnom korišćenju dinamičke memorije.

Primer ulaza:	Primer izlaza:
2 3 1 1 1 2 2 2	2 1 2 1 2 1

2)Maturant Miki je polagao prijemni ispit i želi da nakon objavljenih preliminarnih rezultata proceni svoj uspeh. Preliminarne rezultate prijemnog ispita komisija je objavila u datoteci `prijemni.txt`. U datoteci su za svakog kandidata u zasebnom redu zapisani šifra kandidata (tačno 5 cifara), rezultat na prijemnom iz matematike i rezultat na prijemnom iz fizike, razdvojeni jednim znakom razmaka. Ukoliko neko od kandidata nije polagao prijemni ispit iz nekog predmeta, broj poena je 0. Ocene iz srednje škole zapisane su u datoteci `ocene.txt`. U datoteci su za svakog kandidata u zasebnom redu zapisani šifra kandidata i prosečna ocena za svaki od četiri razreda. Ukupan broj poena računa se kao zbir poena na prijemnom ispitu (pri čemu se računa bolji rezultat ako je kandidat polagao i matematiku i fiziku) i poena iz škole, koji se računaju kao suma prosečnih ocena pomnožena brojem dva. Napisati program na programskom jeziku C koji, nakon što Miki unese svoju šifru preko standardnog ulaza, proveri i na standardni izlaz ispiše da li je Miki prema preliminarnim rezultatima ostvario dovoljan rezultat za upis na budžet (400 mesta), samofinansiranje (100 mesta), ili nije ostvario dovoljan broj poena za upis. Voditi računa o ispravnoj upotrebi resursa.

II PITANJA

1)Koje od ponuđenih tvrdnji su tačne za programski jezik C?

A) Strukturna promenljiva ne može biti povratna vrednost funkcije.

(B) Adresni operator `&` se može koristiti za uzimanje adrese polja strukturne promenljive.

C) Za dve strukturne promenljive se smatra da su istog tipa ukoliko imaju polja istog tipa i istog naziva.

2)Šta ispisuje sledeći program napisan na programskom jeziku C?

```
#include <stdio.h>
#include <string.h>
#define ABC 26
int main() {
  char *imena[] =
  { "marko", "zika", "vladimir",
    "milana", "dragana", "jovan", 0}, **s1;
  int niz[ABC] = { 0 }, c;
  s1 = imena;
  while (*s1) {
    char *s3 = *s1++;
    while (*s3) (*(niz + *s3++ - 'a'))++;
  }
  c = *niz;
  for (int i = 1; i < ABC; i++)
    c = c < *(niz + i) ? *(niz + i) : c;
  printf("%d", c);
}
```

(A) 9 B) 4 C) 2

3)Šta ispisuje sledeći program napisan na programskom jeziku C?

```
#include <stdio.h>
#include <stdlib.h>
#define ALLOC(X) \
  malloc(sizeof(X))
int* f1(int* a) {
  int* b = ALLOC(int);
  *b = *a + 1; return b;
}
int* f2(int* a) {
  int* b = ALLOC(int);
  *b = *a + 2;
  *a = *b; return b;
}
```

```
void run(int* data1, int* (*job)(int *), int** data2) {
  if (data1 != NULL && data2 != NULL) *data2 = job(data1);
}
int main() {
  int* (*array[])(int*)= {f1, f2, f2, f1, f2, f1};
  int i, *data, n = sizeof(array) / sizeof(int* (*)(int*));
  for (int i = 0; i < n; i++) {
    run(&i, array[i], &data);
    printf("%d ", *data);
    free(data);
  }
  return 0;
}
```

A) 1 3 4 4 6 6 B) 1 4 6 **(C)** 1 3 6

4)Šta ispisuje sledeći program na programskom jeziku C, ako se pokrene komandom: **./program.exe 4 2 1 7**

```
#include <stdlib.h>
#include <stdio.h>
int fun(int *p, int *q) {
  int sum = 0, (*r)[2] = (int(*)[2]) p;
  while (p <= q) sum += **r++ + *q--;
  return sum;
}
```

```
int main(int argc, char *argv[]) {
  int *niz = calloc(argc, sizeof(int)), i;
  for (i = 1; i < argc; i++)
    niz[i] = atoi(argv[i]);
  printf("%d\n", fun(niz, niz + argc / 2));
  free(niz); return 0;
}
```

(A) 15 B) 16 C) 14

5)Šta ispisuje sledeći program napisana na programskom jeziku C? Smatrati da je alokacija memorije uvek uspešna.

```
#include <stdlib.h>
#include <stdio.h>
typedef struct node{
int num; struct node *next;
} Node;
void foo(Node *lst) {
  if (lst == NULL) return;
  Node *cur = lst;
  while (cur != NULL) {
    if (cur->num & 1) lst->num += 2;
    else cur->num /= 2;
    cur = cur->next;
  }
  foo(lst->next);
}
```

```
void addNode(Node **lst, int n) {
  Node *tmp = malloc(sizeof(Node));
  tmp->num = n;
  tmp->next = (*lst); (*lst) = tmp;
}
int main(){
  Node *lst=NULL; int i;
  for(i=0;i<5;i++) addNode(&lst,i);
  foo(lst);
  while (lst != NULL) {
    Node *old = lst;
    printf("%d ", lst->num);
    lst = lst->next; free(old);
  } return 0;
}
```

A) 6 5 4 3 0 **(B)** 6 9 5 3 0 C) 6 9 4 3 0

Rešenje 1. Zadatka

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

#define CALLOCATE(ptr, cnt)\
{ if(!(ptr = malloc(cnt * sizeof(*ptr)))) { printf("Greska u alokaciji!\n"); exit(1); }}
```

```c
int** allocateMatrix(int rows, int cols) {
	int **m, i;
	CALLOCATE(m, rows)
		for (i = 0; i < rows; i++)
			CALLOCATE(m[i], cols)
			return m;
}
```

```c
void dealocateMatrix(int** m, int rows, int cols) {
	for (int i = 0; i < rows; free(m[i++]));
	free(m);
}
```

```c
int** readMatrix(int* rows, int* cols) {
	int **m, i, j;
	printf("Unesite dimenzije matrice, a zatim njene elemente po vrstama\n");
	scanf("%d%d", rows, cols);
	m = allocateMatrix(*rows, *cols);
	for (i = 0; i < *rows; i++)
		for (j = 0; j < *cols; j++) scanf("%d", &m[i][j]);
	return m;
}
```

```c
int** rotateMatrix90(int** m, int rows, int cols) {
	int rowsNew = cols;
	int colsNew = rows;
	int** mNew = 0;
	mNew = allocateMatrix(rowsNew, colsNew);
	for (int i = 0; i < rows; i++)
		for (int j = 0; j < cols; j++)
			mNew[j][rows - i - 1] = m[i][j];
	dealocateMatrix(m, rows, cols);
	return mNew;
}
```

```c
void writeMatrix(int** m, int rows, int cols) {
	printf("\n");
	for (int i = 0; i < rows; i++) {
		for (int j = 0; j < cols; printf("%d ", m[i][j++]));
		printf("\n");
	}
}
```

```c
int main(int argc, char **argv) {
	int **m, rows, cols;
	m = readMatrix(&rows, &cols);
	m = rotateMatrix90(m, rows, cols);
	writeMatrix(m, cols, rows);
	dealocateMatrix(m, cols, rows);
	return 0;
}
```

Rešenje 2. Zadatka

```c
#include <stdio.h>
#include <stdlib.h>
typedef struct kandidat {
    int sifra;
    double poeni;
    struct kandidat *sled;
} Kandidat;
```

```c
Kandidat* citaj_prijemni(FILE *prijemni) {
    Kandidat *prvi = NULL, *novi, *posl = NULL;
    int sifra, mat, fiz, ret;
    while ((ret = fscanf(prijemni, "%d %d %d", &sifra, &mat, &fiz)) > 0) {
        novi = malloc(sizeof(Kandidat)); if (!novi) return 2;
        novi->sifra = sifra; novi->poeni = mat > fiz ? mat : fiz; novi->sled = NULL;
        if (!prvi) prvi = novi; else posl->sled = novi;
        posl = novi;
    }
    return prvi;
}
void dodaj_ocene(FILE *ocene, Kandidat *lista) {
    int sifra, ret, i;
    double o;
    while ((ret = fscanf(ocene, "%d", &sifra)) > 0) {
        Kandidat *tek = lista;
        while(tek){
               if (tek->sifra == sifra) {
                      for (i = 0; i < 4; i++) {
                             fscanf(ocene, "%lf", &o);
                             tek->poeni += 2*o;
                      }
                      break;
               } tek = tek->sled;
        }
    }
}
```

```c
void dealociraj(Kandidat* prvi) {
    Kandidat *stari;
    while (prvi) { stari = prvi; prvi = prvi->sled; free(stari); }
}
```

```c
int main() {
    FILE *prijemni, *ocene;
    Kandidat *lista, *tek;
    int mikisif, mikirang = 1;
    double mikip;
    prijemni = fopen("prijemni.txt", "r"); ocene = fopen("ocene.txt", "r");
    if (!prijemni || !ocene) { printf("Greska pri radu sa datotekom!\n"); return 1; }
    lista = citaj_prijemni((prijemni)); dodaj_ocene(ocene, lista);
```

```c
        printf("Unesite sifru: ");
    scanf("%d", &mikisif);
    tek = lista;
    while (tek) {
        if (tek->sifra == mikisif) { mikip = tek->poeni; break; }
        tek = tek->sled;
    }
    tek = lista;
    while (tek) {
        if (tek->poeni > mikip) mikirang++;
        tek = tek->sled;
    }
    if (mikirang <= 400) printf("budzet!");
    else if (mikirang <= 494) printf("samofinansiranje!");
    else printf("nedovoljno poena!");
```

```c
        dealociraj(lista); fclose(prijemni); fclose(ocene); return 0;
}
```