SPA 1 Test 2. školska 2016/17

Ime i prezime Broj indeksa

1. (0.5) Dat je program *zadatak.c :*

```
#include <stdio.h>
int main(int brArg,char *arg[])
{
	if (brArg == 3) printf("Dobar broj argumenata \n");
	else printf("Nedovoljan broj argumenata");
}
```

Ovaj program je kompajliran sledećom naredbom : *gcc –o test zadatak.c .* Šta će biti ispisano na izlazu ako se program pokrene sa `./test prvi drugi 1` ?

Nedovoljan broj argumenata

2. (1) Šta je rezultat sledećeg programa?

```
#include <stdio.h>
#include <string.h>
void fja(char **s) {
 while (*s) {
	printf("%c\n",*(++(*s++)));
 }
}
void main() {
 char *str[3] = { "februar", "mart", "maj"};
 fja(str);
}
```

e
a
a

3. (0.5) Koja od sledećih tvrđenja su tačna za programski jezik C?

a) Efekat poziva malloc(n*sizeof(int)) nije identičan efektu poziva calloc(n, sizeof(int)).
b) Prilikom prosleđivanja niza pri pozivu funkcije čiji je prototip int f(int a[]); pravi se nova kopija niza.
c) Ukoliko su date deklaracije int a[10] = {1, 2, 3, 4}, *b = a+9; vrednost izraza b[-8]-*(a+2) je 1.

4. (1) Šta je rezultat izvršavanje sledećeg programa?

```
#include<stdio.h>
#include<stdlib.h>
typedef struct telephone
{
	char *name;
	int number;
}TELEPHONE;

int main()
{
	TELEPHONE index;
	scanf("%d",&index.number);
	getchar();
	scanf("%s",index.name);

	printf("Telephone number: %d\n", index.number);
	printf("Name: %s\n", index.name);
	return 0;
}
```

Program neće raditi jer nije alociran potreban memorijski prostor za smestanje vrednosti promenljive name

5. (1.5) Neka je data promenljiva `int **a`. Rezervisati potreban memorijski prostor za smeštanje date matrice.

```
a = (int **)malloc(3 * sizeof(int *));
for(i = 0; i < 3; i++)    a[i] = (int *)malloc(4 * sizeof(int));
```

1	2	4	5
10	2	1	6
22	3	0	5

6. (1.5) Na ulazu je zadat niz `13 28 2 13 30 25 4 28 20 24.` Koristeći Selection-sort ispisati korake u sređivanju niza.

```
2  28  13  13  30  25  4  28  20  24
2  4  28  13  30  25  13  28  20  24
2  4  13  28  30  25  13  28  20  24
2  4  13  13  30  28  25  28  20  24
2  4  13  13  20  30  28  28  25  24
2  4  13  13  20  24  30  28  28  25
2  4  13  13  20  24  25  30  28  28
2  4  13  13  20  24  25  28  30  28
2  4  13  13  20  24  25  28  28  30
```

7. (1.5) U programskom jeziku C napisati funkciju `int strcmp(char *s, char *t)` koja poredi stringove s i t i vraća negativnu vrednost, nulu ili pozitivnu vrednost ako je **s** leksikografski manje od, jednako ili veće od **t**. Nije dozvoljeno koristiti funkcije za rad sa stringovima.

```
int strcmp(char *s, char *t){
 int i;
 for(i=0; s[i]==t[i]; i++)
       if(s[i]==’\0’)
       return 0;
 return s[i] - t[i];
}
```

ILI

```
int strcmp( char *s, char *t){
 for( ; *s==*t; s++,t++ )
       if(*s==’\0’)
       return 0;
 return *s - *t;
}
```

8. Data je struktura

```
typedef struct djak {
      char ime[10];
      char oznaka;
      union {
            char opisna;
            int vrednost;
            } OCENA;
}DJAK;
```

a) (1) Napisati funkciju `Ucitaj` koja čita iz prosleđenog fajla informacije o đaku i vraća **pokazivač** na učitanog đaka (svaki đak ima 3 informaciju u fajlu zapisane jedna ispod druge). Ukoliko đak ide u osnovnu školu oznaka “O”, unosi se opisna ocena (neko od velikih slova engleskog alfabeta), a ukoliko ide u srednju školu, oznaka “S”, unosi se vrednost ocene od 1-5. Ukoliko učitavanje nije dobro proslo, odnosno nisu učitane odgovarajuće vrednosti funkcija vraća **null**.

b) (1) Napisati funkciju `Uporedivi` koja za prosleđena dva pokazivača na Djaka proverava da li su uporedivi (dva Djaka su uporediva ako imaju istu oznaku) i ako jesu vraća pokazivač na boljeg. Bolji je onaj djak koji ima veću ocenu. Ukoliko đaci nisu upordivi vraća **null**.

c) (1.5) U glavnom delu programa dinamički alocirati niz od 4 đaka. Iz fajla Ulaz.txt koristeći funkciju `Ucitaj` učitati podatke o đacima. Koristeću funkciju `Uporedivi` štampati ime najboljeg đaka osnovne i najboljeg đaka srednje škole.

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

typedef struct djak
{
	char ime[10];
	char oznaka;
	union
	{
		char opisna;
		int vrednost;
	} OCENA;
}DJAK;
//ne moraju da se rade sve provere
DJAK *Ucitaj(FILE *f)
{
	char pom[1];
	DJAK *djak = (DJAK *)malloc(sizeof(DJAK));
	if (djak == NULL)
		return NULL;

	int count = fscanf(f, "%s %s", djak->ime, pom);
	if (count != 2)
		return NULL;
	djak->oznaka = pom[0];

	if (djak->oznaka == 'O')
	{
		count = fscanf(f, "%s", pom);
		djak->OCENA.opisna = pom[0];
	}
	else
		count = fscanf(f, "%d", &djak->OCENA.vrednost);

	if (count != 1)
		return NULL;

	return djak;
}

// svejedno koja ocena vam je najveca
DJAK *Uporedivi(DJAK *prvi, DJAK *drugi)
{
	if (prvi->oznaka != drugi->oznaka)
		return NULL;

	int rez = prvi->OCENA.vrednost - drugi->OCENA.vrednost;

	if (prvi->oznaka == 'O')
	{
		if (rez > 0)
			return drugi;
		return prvi;
	}
	else
	{
		if (rez > 0)
			return prvi;
		return drugi;
	}
}

#define BROJ_DJAKA 4
int main()
{
```

```
	DJAK **nizDjaka = (DJAK **)malloc(sizeof(DJAK *) * BROJ_DJAKA);
	FILE *f = fopen("Ulaz.txt", "r");
	int i;

	for (i = 0; i < BROJ_DJAKA; i++)
	{
		nizDjaka[i] = Ucitaj(f);
		if (nizDjaka[i] == NULL)
		{
			fprintf(stderr, "Nije ucitan djak %d.\n", i);
			exit(1);
		}
	}

	DJAK *najboljiO = NULL, *najboljiS = NULL;

	for (i = 0; i < BROJ_DJAKA; i++)
	{
		if ((najboljiO != NULL) && (najboljiS != NULL))
			break;
		if (nizDjaka[i]->oznaka == 'O')
			najboljiO = nizDjaka[i];
		else
			najboljiS = nizDjaka[i];
	}

	for (i = 0; i < BROJ_DJAKA; i++)
	{
		DJAK *temp;
		if (najboljiO != NULL)
		{
			temp = Uporedivi(najboljiO, nizDjaka[i]);
			if (temp != NULL)
				najboljiO = temp;
		}
		if (najboljiS != NULL)
		{
			temp = Uporedivi(najboljiS, nizDjaka[i]);
			if (temp != NULL)
				najboljiS = temp;
		}
	}

	printf("Najbolji djak osnovne skole je: %s\n", (najboljiO == NULL) ? "nedefinisan"
: najboljiO->ime);
	printf("Najbolji djak srednje skole je: %s\n", (najboljiS == NULL) ? "nedefinisan"
: najboljiS->ime);
	return 0;
}
```