Strukture podataka i algoritmi 1

Test I (maksimalno 13 poena)

Ime i prezime	Indeks	Broj poena

1. Napisati rezultate sledećih kodova i dati odgovore na pitanja:

a)

```
int i = 4;
float d = 3.4;
i += d;
d -= i;
i += d;
printf("%d, %f", i, d);
```

3, 3.6

b)

```
int a = 3, b = 2;
int *p = &a,*q = &b;
*q = *p;
p = q;
(*q)--;
p++;
printf("%d, %d", a, b);
```

3, 2

c)

```
int x = 0, y = 1;
printf("%d,%d\n",  x++ && --x ? x-- + y : y++ + x, x);
```

2, 1

d)

```
#define DOUBLE (double)
#define MAX(x,y) x>y?x:y

int main() {
    int a = 3;
    int b = 10 / DOUBLE(a) + 0.8;
    printf("%d %d\n", b, 2+MAX(a,b));
    return 0;
}
```

4, 3

2. Dati odgovore na sledeća pitanja:

a) Koja Linux komanda može da se koristi za promenu imena fajla? Napisati primer upotrebe ove komande.
Komanda: mv Primer: mv staro_ime.txt novo_ime.txt

b) Sta je problem u sledecem kodu? Napisati ispod kako kod može da se napise da bi se potencijalni problem izbegao?

```
p = realloc(p, 10 * sizeof(int));
if (p == NULL)
{
        printf("Greska!\n");
        free(p);
}
```

Greska: Problem je moguce curenje memorije u slucaju neuspele realokacije. Jer se gubi prvobitna vrednost na koju je pokazivao pokazivac p a memorija ostaje zarobljena jer se sa free(p), gde je p=NULL, nista ne oslobadja. Korekcija:

```
int *temp = realloc(p, 10 * sizeof(int));
if (temp == NULL)
{
        printf("Greska!\n");
        free(p);
}
plse p = temp;
```

3. Objasniti šta radi sledeća funkcija i koji rezultat vraća ako se pozove: sa fun(0xB5)? Rezultat dati u heksadekadnom ili binarnom formatu.

```
unsigned char fun(unsigned char x)
{
    unsigned char y = 0;
    while (x)
    {
        y = y << 1 | (x & 1);
        x >>= 1;
    }
    return y;

}
```

Objašnjenje:

Funkcija vraca broj cija se binarna reprezentacija dobija obrtanjem binarne reprezentacije prosledjenog broja

Rezultat :**0b10101101 (0xAD)**

4. Napisati rezultat izvršavanja sledećeg koda:

```
#include <stdio.h>
int *func(int *a, int n, int t) {
    for (int i = 0; i < n; i++) {
        if (*(a + i) == t) return a + i;
        *(a+i) = 0;
    }
    return NULL;
}
```

```
int main() {
    int a[] = {5, 3, 8, 5, 9};
    int *p = func(&a[1], 4, 5);
    if (p != NULL) printf("%ld %d\n", p - a, *p);
    for (int i = 0; i < 5; printf("%d ", a[i]), i++);
    return 0;
}
```

3 5
5 0 0 5 9

5. Napisati rezultat izvršavanja sledećeg koda:

```
#include <stdio.h>
int a = -1;
void func(int i)
{
    !i ? : printf("%d ", --a);
    static int a = 1;
    i ? : printf("%d ", a++);
}
int main()
{
    int a = 0;
    for (int i=0; i<3; i++)
    {
        func(i++);
    }
}
```

1, -2

6. Napisati funkciju koja za dva cela nenegativna broja `a` i `b` i proverava da li se broj `b` može dobiti od broja `a` pomeranjem bitova ulevo ili udesno za neki broj pozicija.

```
void func(unsigned int a, unsigned int b)
{
    for(int i=0;i<=32;i++)
    {
        if((a<<i)==b || (a>>i)==b)
        {
            printf("Moze");
            return;
        }
    }
    printf("Ne moze");
}
```

7. Napisati rekurzivnu funkciju koja za dati ceo broj n ispisuje broj u binarnom sistemu. Funkciji nije dozvoljeno prosleđivanje dodatnih parametara osim celog broja n.

```
void printBin(unsigned int n) {

    if (n) printBin(n >> 1);

    printf("%d", n & 1);

}
```

*ako je neko predvideo i ispis negativnih brojeva, jos bolje.

8. Napisati program koji kreira, kopira, štampa i briše iz memorije niz realnih brojeva koristeći sledeće funkcije:

- Napisati funkciju **`Kreiraj`** koja prihvata cele brojeve `n` i `x` i dinamički kreira niz od `n` elemenata realnog tipa. Prvi element niza ima vrednost `x`, a svaki naredni element je za `0.2` manji od prethodnog.
- Napisati funkciju **`Kopiraj`** koja prihvata niz realnih brojeva i njegovu dužinu, a zatim dinamički kreira novi niz iste dužine. Element novog niza na poziciji `i` definiše se kao zbir levog i desnog suseda elementa na poziciji `i` u originalnom nizu. Smatra se da levi sused prvog elementa i desni sused poslednjeg elementa imaju vrednost `0`.
- Napisati funkciju **`Zavrsi`** koja ispisuje elemente novog niza, a zatim oslobađa svu dinamički alociranu memoriju.

Primer:
Ulaz 6, 4

Izlaz: 3.8 7.6 7.2 6.8 6.4 3.2

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

float* Kreiraj(int n, int x)
{
    float *niz = (float*) malloc(n * sizeof(float));
    if (niz == NULL) {
        printf("Greska pri alokaciji memorije!\n");
        exit(1);
    }

    niz[0] = x;
    for (int i = 1; i < n; i++) {
        niz[i] = niz[i - 1] - 0.2;
    }

    return niz;
}

float* Kopiraj(float *niz, int n)
{
    float *novi = (float*) malloc(n * sizeof(float));
    if (novi == NULL) {
        printf("Greska pri alokaciji memorije!\n");
        exit(1);
    }

    for (int i = 0; i < n; i++) {
        float levi  = (i == 0)     ? 0 : niz[i - 1];
        float desni = (i == n - 1) ? 0 : niz[i + 1];
        novi[i] = levi + desni;
    }

    return novi;
}

void Zavrsi(float *niz, float *novi, int n)
{
    for (int i = 0; i < n; i++) {
        printf("%.2f ", novi[i]);
    }
```

```
    printf("\n");

    free(niz);
    free(novi);
}

int main(void)
{
    int n, x;

    printf("Unesite broj elemenata n: ");
    scanf("%d", &n);
    printf("Unesite pocetnu vrednost x: ");
    scanf("%d", &x);

    float *niz  = Kreiraj(n, x);
    float *novi = Kopiraj(niz, n);

    Zavrsi(niz, novi, n);

    return 0;
}
```