

## ZADACI

### Programska petlja while, kratki i dugi oblik naredbe

#### 1. zadatak

Što će ispisati sljedeći programski odsječak?

```
int i = 1, j = 8;
while (i < j) {
    printf("%d ", i + j);
    i ++;
    j --;
}
```

#### 2. zadatak

Napišite programski odsječak, koristeći *while* petlju, koji izračunava sumu prvih 10 parnih prirodnih brojeva!

### Programska petlja do-while, kratki i dugi oblik naredbe

#### 1. zadatak

Što će ispisati sljedeći programski odsječak?

```
int i = 1, j = 1;
do
{
    printf("%d %d\n", i ,j);
    i += j;
    j = j * 2 % 3;
} while (j + i < 5 || i < 10);
```

#### 2. zadatak

Napišite programski odsječak, koristeći dvije *do-while* petlje, koji ispisuje sva velika slova engleske abecede na način da se svako slovo ispiše onoliko puta koliko iznosi njegov redni broj u abecedi.

Početak ispisa treba izgledati ovako:

```
A
BB
CCC
DDDD
EEEE
....
```

### Programska petlja for

#### 1. zadatak

Što će ispisati sljedeći program?

```
#include <stdio.h>
int main() {
```

```

int i, n, suma;
float arit_sred;

suma = 0;
n = 10;
for(i = 0; i < n; i ++ ) {
    suma += i;
}
arit_sred = (float) suma / n;
printf("Aritmeticka sredina je %f\n", arit_sred);
return 0;
}

```

## 2. zadatak

Što će ispisati sljedeći program?

```

#include <stdio.h>
int main() {
    float m = 2, i;

    for(i = 0; i <= m; i += 0.5 ) {
        printf("%f\n", i);
    }
    return 0;
}

```

## **Naredba switch**

### 1. zadatak

Što će se na zaslonu ispisati sljedećim programom?

```

int main () {
    char c;
    c = '1' + 1;
    switch (c) {
        case '1': printf ("%d", c);
        case '2': printf ("%d", c);
        case '3': printf ("%d", c); break;
        default: printf ("default");
    }
    return 0;
}

```

### 2. zadatak

Napišite program kojim ćete učitati redni broj dana u tjednu i ispisati naziv učitanoj dana. U zadatku koristiti naredbu switch.

## Korištenje goto

### 1. zadatak

Izvođenje programskog odsječka:

```
while(i < 10) {
    if ((a-i)%4 == 0) goto lab1;
    sum += i;
    lab1:
    i++;
}
```

ostalo bi nepromijenjeno ako bi se naredbu `goto lab1;` zamijenilo s

- a) naredbom `continue;`
- b) naredbom `break;`
- c) naredbom `printf ("\n");`
- d) naredbom `stop;`
- e) niti jedan od ponuđenih odgovora nije točan

### 2. zadatak

Što će se ispisati sljedećim programskim odsječkom ?

```
int main() {
    int i=0;
    ponovo:
    printf("%d ", ++i );
    if(i++ <= 5 ) goto ponovo;

    return 0;
}
```

### 3. zadatak

Izvođenje programskog odsječka:

```
while(i < 10) {
    i++;
    if ((a-i)%4 == 0) goto lab1;
    sum += i;
    lab1:
    i
}
```

ostalo bi nepromijenjeno ako bi se naredbu `goto lab1;` zamijenilo s

- a) naredbom `continue;`
- b) naredbom `break;`
- c) naredbom `printf ("\n");`
- d) naredbom `stop;`
- e) niti jedan od ponuđenih odgovora nije točan

## break, continue

### 1. zadatak

Što će ispisati sljedeći programski odsječak?

```
int sum = 0, i, j, gotovo = 0;
for( i = 1; i <= 5 && !gotovo; i++){
    for( j = 1; j <= 5 ; j++){
        if (j%4 == 0){
            gotovo = 1;
            continue;
        }
        sum += i+j;
    }
}
```

### 2. zadatak

Što će se ispisati nakon obavljanja programskog odsječka:

```
int i, j;
for (i = 1; i <= 3; i++)
    for (j = 1; j <= 4; j++) {
        if (j%2) continue;
        printf(" %d ", j);
        if (j!=i) break;
    }
```

## Korištenje bitovnih operatora

### 1. zadatak

Koja je vrijednost varijable a nakon izvođenja sljedećeg programskog odsječka:

```
signed char a=5;
a=~a;
```

### 2. zadatak

Koja je vrijednost u varijabli c nakon izvođenja sljedećeg programskog odsječka :

```
int a=6, b=12, c;
c = (a & b) + (a | b) + (a ^ b);
```

### 3. zadatak

Što će ispisati sljedeći program:

```
#include <stdio.h>
int main(){
    char a,b,c;
    b = 1;
    a = b << 2 < 2;
```

```

    c = a ^ b;
    printf("a=%d, b=%d, c=%d",a,b,c);
    return 0;
}

```

## Degenerirane programske petlje (beskonačne, one koje se ne izvršavaju)

### 1. zadatak

Što će se ispisati nakon izvođenja sljedećeg programa:

```

int main () {
    int i;
    i = 0;
    do {
        ++i;
        if (i%2 == 1){
            i *= 2;
        } else {
            i /= 2;
        }
        printf("%d",i);
    } while (i = 1);
    return 0;
}

```

### 2. zadatak

Što će se ispisati izvođenjem sljedećeg programa:

```

int main () {
    int i, j;
    for (i=0; i==i+1; ++i){
        for (j=0; j<2; j++){
            printf("*");
        }
    }
    return 0;}

```

## Kombinacije više tipova petlji

### 1. zadatak

Što će se ispisati nakon izvođenja sljedećeg programa?

```

int main()
{
    int i = 0;
    int suma=0;

    while(i < 10)
    {

```

```

        for(i = 0; i < 10; i++)
        {
            suma += 1;
        }
        i++;
    }
    printf("Suma = %d", suma);
    return 0;
}

```

## 2. zadatak

Što će se ispisati nakon izvođenja sljedećeg programa?

```

int main()
{
    int i = 0;
    int j = 0;
    int suma=0;

    while(i < 10)
    {
        for(j = 0; j < 10; j++)
        {
            suma += 1;
        }
        i++;
    }
    printf("Suma = %d", suma);
    return 0;
}

```

## **Kombinacije selekcija i programskih petlji**

### 1. zadatak

Što radi sljedeći programski odsječak?

```

int main()
{
    int i = 2;
    int ucitaniBroj = 0;
    int suma = 0;

    scanf("%d", &ucitaniBroj);
    while(i < ucitaniBroj)
    {
        if(!(i%2)) suma += i;
        i++;
    }
}

```

```

    printf("%d", suma);
    return 0;
}

```

## 2. zadatak

U ovome je zadatku programski odsječak iz 1. zadatka napisan na malo drukčiji način (pomoću do – while petlje). Za negativne vrijednosti varijable ucitaniBroj će se, za razliku od prethodnog primjera, izvršiti jedan prolaz kroz do-while petlju. Odgovoriti koliku vrijednost poprima varijabla suma za ucitaniBroj=-4 te komentirati rezultat u svjetlu uvodnog teksta.

```

int main()
{
    int i = 2;
    int ucitaniBroj = 0;
    int suma = 0;

    scanf("%d", &ucitaniBroj);
    do
    {
        if(!(i%2)) suma += i;
        i++;
    } while(i < ucitaniBroj);
    printf("%d", suma);
    return 0;
}

```

## **Definicija jednodimenzionalnih polja, dodjeljivanje početnih vrijednosti (bez znakovnih polja)**

### 1. zadatak

Koje su vrijednosti pohranjene u elementima polja ako je polje definirano i inicijalizirano na sljedeći način:

```
int p[5]={1};
```

### 2. zadatak

Koje su vrijednosti pohranjene u elementima polja ako je polje definirano na sljedeći način:

```
int p[5];
```

## **Korištenje jednodimenzionalnih polja (pristupanje članovima polja, indeksni izrazi)**

### 1. Zadatak

Što će se ispisati sljedećim programskim odsječkom?

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

```
int main()
{
```

```

int polje [10] = {7, 6, 3, 4, 10, 9, 1, 5, 2, 8}, i;
i = 0;
do
{
    if(i%2 && polje[i]%2) printf("%d", polje[i]);
    i++;
} while(i<10);
}

```

## 2. Zadatak

Programer kojega smo predstavili na prethodnoj provjeri znanja na računalu i koji nam je zadao glavobolju svojom brzopletošću se vratio – izradio je programski odsječak kojim je treba ispisati cijele brojeve uzlazno pohranjene u polju silaznim redoslijedom.

```

#include <stdio.h>

int main()
{
    int polje [10] = {1, 2, 3, 4, 5, 6, 7, 8, 9, 10}, i, j, k;
    i = 0;
    for(i = 0; i<10; i++)
    {
        j = 10-i;
        printf("%d ", polje[j]);
    }
}

```

Iako je program na prvo pogled dobar, postoji pogreška koja je vidljiva po pokretanju programa:

```
-858993460 10 9 8 7 6 5 4 3 2
```

Vidljivo je da će programer trebati pomoć FER-ovaca: pomognimo mu i pronađimo gdje je pogriješio!

## **Jednodimenzionalna znakovna polja, dodjeljivanje početnih vrijednosti nizu znakova**

### 1. Zadatak

Koje od sljedećih naredbi nisu ispravne:

- 1) char p[]={ 'A', 65 , 'C' };
- 2) char p[2]={ 'A', 0 , 0 };
- 3) char p[4]={ 'A', 0 , 0 };
- 4) char p[10]={ 1, 2, 3 };
- 5) char p[2]={ "AB", "CD" };
- 6) char p[4]={ "AB" , "CD" };

## 2. Zadatak

Ukoliko je polje p definirano kao:

```
char p[3];
```

koje od sljedećih naredbi nisu ispravne:

- 1) p['A'] = 'C';
- 2) p['3'] = 'C';
- 3) p['2'] = 'C';
- 4) p[1] = 'C';
- 5) p[3] = 67;
- 6) p[2] = 32;

## **Algoritmi s jednodimenzionalnim numeričkim poljima**

### 1. zadatak

Napisati odsječak kôda koji ispisuje indekse onih elemenata polja od n elemenata koji nisu djeljivi sa 2.

### 2. zadatak

Napisati odsječak kôda koji računa umnožak elemenata polja od n elemenata.

## **Algoritmi s jednodimenzionalnim znakovnim poljima**

### 1. zadatak

Napisati program koji pri ispisu "okrene" prvu polovicu nekog polja znakova, a drugu polovicu ispiše u poretku kako su znakovi i pohranjeni u polju. Npr. za polje "ABCDEFGH I" treba ispisati EDCBAFGH I . Testirati rad programa za paran i neparan broj elemenata polja.

### 2. zadatak

Što se ispiše sljedećim programom?

```
#include <stdio.h>
int main(){
    char polje[]="qwertzuiop";
    char tmp;
    int i;
    for(i=0;i<5;i++){
        if(polje[i]<polje[9-i]){
            tmp=polje[i];
            polje[i]=polje[9-i];
            polje[9-i]=tmp;
        }
    }
}
```

```

    for(i=0;i<10;i++){
        printf("%c",polje[i]);
    }
    return 0;
}

```

## **Definicija dvodimenzionalnih i višedimenzionalnih polja i dodjeljivanje početnih vrijednosti dvodimenzionalnim poljima**

### 1. zadatak

Kako izgleda drugi stupac matrice definirane na sljedeći na način:

```
int m[4][4]={{1,2,3},{4,5,6}};
```

### 2. zadatak

Što je neispravno u sljedećoj definiciji?

```
int m[4][0][4]={1,2,3,4,5,6};
```

## **Jednostavniji algoritmi s dvodimenzionalnim poljima**

### 1. zadatak

Napisati program koji će učitati matricu dimenzija 8\*8 i ispisati sumu elemenata koji se ne nalaze na "obodu" (obod definiramo kao uniju elemenata prvog retka, prvog stupca, zadnjeg retka i zadnjeg stupca) matrice.

### 2. zadatak

Što će se ispisati na zaslonu računala sljedećim programskim odsječkom?

```

int m[2][3] = {1, 2, 3, 4, 5, 6};
int i, j, suma = 0;
for (i = 0; i < 2; i++)
    for (j = 0; j <= i; j++)
        suma += m[i][j];
printf ("%d", suma);

```

## **Zauzeće memorije varijablama i poljima (sizeof, ručno brojanje, procjena)**

### 1. zadatak

Koliko je bajtova rezervirano u memoriji sljedećom definicijom:

```
char p[4][3][2];
```

### 2. zadatak

Koliko će bajtova biti utrošeno za pohranu u memoriji polja a?

```

int a[5], i;
for (i = 1; i <= 5; i++ )
    a[i] = (float) i / 2;

```

## Učitavanje polja i ispis polja (samo jednostavni formati kao npr. %5d, %15.7f, %s, %c)

### 1. Zadatak

Neka je  $p$  kvadratna matrica dimenzija  $N \times N$ . Što radi sljedeći programski odsječak?

```
for(j=0; j<N; j++) {  
    for(i=0; i<N; i++)  
        printf("%d ",p[i][j]);  
    printf("\n");  
}
```

### 2. Zadatak

Napisati programski odsječak kojim se učitavaju elementi na glavnoj dijagonali matrice deklarirane kao `float a[M][M]`?

## RJEŠENJA ZADATAKA

### Programska petlja while, kratki i dugi oblik naredbe

#### 1. zadatak

9 9 9 9

#### 2. zadatak

```
int suma = 0, i = 1;
while (i <= 10) {
    suma += 2*i;
    i++;
}
```

### Programska petlja do-while, kratki i dugi oblik naredbe

#### 1. Zadatak

1 1  
2 2  
4 1  
5 2  
7 1  
8 2

#### 2. Zadatak

```
char c='A';int i;
do
{
    i=0;
    do
    {
        printf("%c",c);
    }while(i++ < c-'A');
    printf("\n");
}while(c++ != 'Z');
```

### Programska petlja for

#### 1. zadatak

4.500000

#### 2. zadatak

0.000000  
0.500000  
1.000000  
1.500000  
2.000000

### Naredba switch

#### 1. zadatak

5050

## 2. zadatak

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

```
int main ()
{
    int dan;
    printf ("Unesite redni broj dana");
    scanf ("%d", &dan);

    switch (dan){
        case(1): printf ("Ponedjeljak"); break;
        case(2): printf ("Utorak"); break;
        case(3): printf ("Srijeda"); break;
        case(4): printf ("Cetvrtak"); break;
        case(5): printf ("Petak"); break;
        case(6): printf ("Subota"); break;
        case(7): printf ("Nedjelja"); break;
        default: printf ("Neispravno unesen dan.");
    }
    return 0;
}
```

## **Korištenje goto**

### 1. zadatak

e) niti jedan od ponuđenih odgovora nije točan

### 2. zadatak

1 3 5 7

### 3. zadatak

a) naredbom continue;

## **break, continue**

### 1. zadatak

15

### 2. zadatak

2 2 4 2

## **Korištenje bitovnih operatora**

### 1. zadatak

-6

### 2. zadatak

28

### 3. zadatak

a=0, b=1, c=1

## **Degenerirane programske petlje (beskonačne, one koje se ne izvršavaju)**

### 1. zadatak

Petlja će se obaviti beskonačno zato jer se prilikom ispitivanja uvjeta petlje:

```
while ( i = 1 )
```

koristi naredba za pridruživanje (=), a ne uspoređivanje (==) te se uvjet uvijek izračunava kao 1 (istina).

### 2. zadatak

Ništa, zato jer je u uvjetu petlje  $i == i + 1$  a taj izraz nije nikad istinit pa se petlja neće obaviti niti jednom.

## **Kombinacije više tipova petlji**

### 1. zadatak

Ispisat će se 10, a ne 100 kao što bi se moglo inicijalno očekivati. Naime, zbog korištenja varijable i i u vanjskoj i u unutarnjoj petlji, vanjska petlja će se izvršiti samo jednom. Za ispravan rad bi trebalo varijablu i u unutarnjoj petlji zamijeniti s novom varijablom (npr j)

### 2. zadatak

Ispisat će se 100. Za razliku od 1. zadatka pazilo se kod korištenja varijabli – svaka je petlja dobila "svoju" varijablu čime je riješen "problem" iz prethodnog zadatka, ali i olakšana čitljivost programa (sada je i osobi koja prije nije vidjela program sve mnogo razumljivije).

## **Kombinacije selekcija i programskih petlji**

### 1. zadatak

Programski odsječak sumira sve parne brojeve između nule i broja ucitaniBroj te ispisuje dobiveni rezultat – npr za ucitaniBroj=5 ispisuje 6.

### 2. zadatak

Programski odsječak radi istu stvar kao i programski odsječak u prvome zadatku, ali zbog korištenja do-while petlje uvijek ima jedan prolazak kroz nju. Baš će zbog toga, ukoliko se unese negativan parni broj u varijablu ucitaniBroj (npr. -4), doći do neželjenih rezultata – obaviti će se jedan prolaz kroz do-while petlju i ispisati broj različit od nule kao vrijednost sume (2). To se ne bi smjelo dogoditi budući da se traži suma parnih brojeva između 2 i -4 (donja granica je veća od gornje, pa takvi brojevi ne postoje).

Iz ovoga je primjera vidljivo da se petlje s ispitivanjem uvjeta na početku i petlje s ispitivanjem uvjeta na kraju trebaju koristiti s oprezom i ovisno o tome što u konačnici želimo postići.

## **Definicija jednodimenzionalnih polja, dodjeljivanje početnih vrijednosti (bez znakovnih polja)**

### 1. zadatak

1 0 0 0 0

## 2. zadatak

nepoznate vrijednosti (ne možemo točno reći koje)

### **Korištenje jednodimenzionalnih polja (pristupanje članovima polja, indeksni izrazi)**

#### 1. Zadatak

95

Kratko objašnjenje: Programskim se odsječkom ispisuje sadržaj polja na način da se ispisuje samo element koji se nalazi na neparnoj poziciji u polju (indeks polja za taj element je neparan broj) i koji istovremeno ima neparnu vrijednost. Pažnja: razlikovati indekse elemenata (tj. poziciju unutar polja) i vrijednost elemenata; da bi saznali vrijednost nekog elemenata polja nužno moramo znati njegovu poziciju u polju, tj. indeks.

#### 2. Zadatak

Programer je načinio jednu od klasičnih grešaka (koja je dosta teško uočljiva i s kojom se programeri hrvaju dan za danom), a ta je da je koristio nedozvoljene indekse polja: krenuo je od  $j=10$  do  $j=1$ . Naravno, indeksi polja idu od 0 do 9 te je zbog toga njegov program dao "čudan" prvi broj – otišao je u memoriju "iza" elementa s indeksom 9 i pročitao "smeće" kao element s indeksom 10 (nešto što se posve slučajno našlo u memoriji u tome trenu).

Da bi program ispravno radio potrebno je načiniti sljedeći ispravak:

```
j = 10-i-1;
```

Alternativno (ponešto i elegantnije) može se koristiti i indeksni izraz koji u sebi sadrži varijablu unutar for petlje:

```
for(i = 0; i<10; i++)
{
    printf("%d ", polje[10-i-1]);
}
```

### **Jednodimenzionalna znakovna polja, dodjeljivanje početnih vrijednosti nizu znakova**

#### 1. Zadatak

Rješenje: 2 5 6

#### 2. Zadatak

Rješenje: 1 2 3 5

### **Algoritmi s jednodimenzionalnim numeričkim poljima**

#### 1. zadatak

```
int suma = 0, i, n;
//...
for (i = 0; i < n; i++)
    if (a[i] % 2) printf(„%d“, i);
```

## 2. zadatak

```
int umnozак = 1, i, n;  
//...  
for (i = 0; i < n; i++)  
    umnozак *= a[i];
```

## **Algoritmi s jednodimenzionalnim znakovnim poljima**

### 1. zadatak

```
#include <stdio.h>  
#define velpolja 8  
int main(){  
    char polje[velpolja+1]="ABCDEFGH";  
    int i=0;  
    for(i=0;i<velpolja;i++){  
        if(i<=(velpolja-1)/2)  
            printf("%c",polje[(velpolja-1)/2-i]);  
        else  
            printf("%c",polje[i]);  
    }  
    return 0;  
}
```

### 2. zadatak

qwiuztreop

## **Definicija dvodimenzionalnih i višedimenzionalnih polja i dodjeljivanje početnih vrijednosti dvodimenzionalnim poljima**

### 1. zadatak

2  
5  
0  
0

### 2. zadatak

Ni jedna dimenzija kod višedimenzionalnih polja ne smije biti manja od 1!

## **Jednostavniji algoritmi s dvodimenzionalnim poljima**

### 1. zadatak

```
#include <stdio.h>  
#define BR_RED 8  
#define BR_STUP 8  
  
int main() {
```

```

int mat[BR_RED][BR_STUP];
int i, j, suma = 0;
for(i=0; i<BR_RED; i++){
    for(j=0; j<BR_STUP; j++){
        scanf("%d", &mat[i][j]);
        if (i!=0 && j!=0 && i!=BR_RED-1 && j!=BR_STUP-1){
            suma += mat[i][j];
        }
    }
}
printf ("%d", suma);
return 0;
}

```

## 2. zadatak

10

### **Zauzeće memorije varijablama i poljima (sizeof, ručno brojanje, procjena)**

#### 1. zadatak

24

#### 2. zadatak

20

### **Učitavanje polja i ispis polja (samo jednostavni formati kao npr. %5d, %15.7f, %s, %c)**

#### 1. Zadatak

Rješenje: ispisuje matricu koja nastaje transponiranjem matrice p.

#### 2. Zadatak

Rješenje: for (i = 0; i < M; i++) scanf("%f", &a[i][i]);