

ZADACI

Definicija funkcije

1. Koje su ispravne definicije funkcije `definici ja` ako funkcija ne vraća vrijednost i ne prima parametar:

a)

```
void definicija(){  
    return;  
}
```

b)

```
void definicija(){  
}
```

c)

```
void definicija{  
    return;  
}
```

d)

```
void definicija(void){  
    return;  
}
```

e)

```
void definicija(void){  
}
```

f)

```
void definicija{  
}
```

g)

```
void definicija[]{  
    return;  
}
```

h)

```
void definicija[void]{  
    return;  
}
```

i)

```
definicija(){  
    return;  
}
```

Naredba return

Kojeg je tipa povratna vrijednost u funkciji `pv`?

```
int pv(float a){  
    float b;  
    b=2*a;  
    return b;  
}
```

Void funkcije i funkcije bez argumenata

Što će ispisati sljedeći program?

```
#include <stdio.h>

void f(int a, int b) {
    a++;
    b++;
    printf("%d %d", a, b);
}

int main() {
    int a=3, b=4;
    f(3,4);
    printf("\n%d %d", a, b);
    return 0;
}
```

Prijenos kopija vrijednosti (bez polja)

Što će se ispisati obavljanjem sljedećeg programa?

```
int funkcija (int x, int y) {
    x = x + y;
    return x * y;
}

int main () {
    int x = 2, y = 3;
    printf ("%d, %d, %d", x, y, funkcija (x, y));
    return 0;
}
```

Prijenos referencija-adresa (bez polja)

Što će se ispisati sljedećim programom ?

```
void f(int x, int *y) {
    x %= 2;
    *y *= x;
}

int main () {
    int a=8, b=10;
    printf ("%d %d", a, b);
    f (a, &b);
    printf (" -> %d %d", a, b);
    return 0;
}
```

Jednodimenzionalna polja kao argumenti funkcije (rad s pokazivačima)

Pretpostavimo da funkcija suma računa sumu članova polja duljine N.

Koji od navedenih je pravilan poziv funkcije iz glavnog programa ako želimo izračunati sumu članova polja podaci?

```
float suma(float *polje, int n){
    (...)
}
int main(){
    float podaci[]={1, 1.4, 2, 3.5};
    float s;

    //poziv funkcije..

    return 0;
}
```

- 1) s=suma(*podaci[0],4);
- 2) s=suma(*podaci,4);
- 3) s=suma(&podaci,4);
- 4) s=suma(&podaci[0],4);
- 5) s=suma(podaci[0],4);

Smještajni razredi (postojanost, područje važenja varijabli) Samo elementarni pojmovi, po mogućnosti bez register i external!

Ukoliko vrijednost lokalne varijable treba biti zapamćena po izlasku i ponovnom povratku u funkciju kojem smještajnom razredu treba pripadati ta varijabla?

Dvodimenzionalna polja kao argumenti funkcije (rad s indeksnim izrazima)

Deklarirana je funkcija koja prima dvodimenzionalno polje kao argument:

```
int func( int p[], int m, int n, int maxstup )
```

Ukoliko je varijabli a potrebno pridružiti element polja s indeksom retka 7 i indeksom stupca 5, upotrijebit ćemo sljedeću naredbu.

- 1) a = p[5 * maxstup + 7];
- 2) a = p[7][5];
- 3) a = p[7 * maxstup + 5];
- 4) a = p[5][7];
- 5) a = p[7*5];

Dvodimenzionalna polje kao argument funkcije (rad s pokazivačima)

Ukoliko funkcija f treba izračunati sumu svih elemenata u matrici koju naredbu treba umetnuti na mjesto označeno s ### ?

```

int f(int *p, int m, int n, int maxstup){
    int i,j,suma=0;
    for(i=0; i<m ; i++){
        for(j=0; j<n; j++){
            suma += *p;
            p++;
        }
        ###
    }
    return suma;
}

```

Jednodimenzionalna polja kao argumenti funkcije (rad s indeksnim izrazima)

Napraviti funkciju koja provjerava jesu li elementi nekog cjelobrojnog polja fibonnacijevi brojevi. Funkcija mora imati prototip:

```
int provjeraFib(int [], int);
```

Formalni i stvarni argumenti (izrazi, redoslijed, tipovi pri pozivu funkcije)

Prototip funkcije je: void f2 (int x, float y, char z);

Koji su od ponuđenih poziva funkcije **ispravni**?

1. f2('A', "Abc", 'B');
2. void f2('A', 15.01, 'B');
3. f2('A', 15.01, 'B');
4. f2(23, 0.1f, 65);
5. int i; i = f('A', 56.f, 'z');

Prototipovi funkcija, organizacija složenijih programa

Koji su od navedenih prototipova funkcije traziMax **ispravni**?

```

int traziMax(int p[], int duljina) {
    int i = 0, max = p[0];
    for (i = 1; i < duljina; i++) {
        if (p[i] > max) {
            max = p[i];
        }
    }
    return max;
}

```

1. int traziMax(int *p, int duljina);
2. int traziMax(int p, int duljina);
3. void traziMax(int *p, int duljina);
4. int traziMax(int *p, int duljina, int max);
5. int traziMax(int p[], int duljina);

RJEŠENJA

Definicija funkcije

a, b, d, e

Naredba return

Povratna vrijednost je tipa `int`.

Void funkcije i funkcije bez argumenata

4 5

3 4

Prijenos kopija vrijednosti (bez polja)

2, 3, 15

Prijenos referencija-adresa (bez polja)

8 10 -> 8 0

Jednodimenzionalna polja kao argumenti funkcije (rad s pokazivačima)

4

Smještajni razredi (postojanost, područje važenja varijabli) Samo elementarni pojmovi, po mogućnosti bez register i external!

Odgovor: `static`

Dvodimenzionalna polja kao argumenti funkcije (rad s indeksnim izrazima)

3

Dvodimenzionalna polje kao argument funkcije (rad s pokazivačima)

`p += maxstup - n;`

Formalni i stvarni argumenti (izrazi, redoslijed, tipovi pri pozivu funkcije)

Rješenje: 3, 4

Prototipovi funkcija, organizacija složenijih programa

Rješenje: 1, 5

Jednodimenzionalna polja kao argumenti funkcije (rad s indeksnim izrazima)

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

```
#define N 10
```

```
int provjeraFib(int [], int);
```

```
void ispisPolja(int [], int);
```

```
int main()
```

```
{
```

```
    int poljeFib[N] = {1, 1, 2, 3, 5, 8, 13, 21, 34, 55};
```

```
    int poljeNeFib[N] = {1, 1, 2, 3, 5, 8, 13, 22, 34, 55};
```

```
    printf("Polje: ");
```

```

        ispisPolja(poljeFib, N);
        if(provjeraFib(poljeFib, N)) printf(" JE polje fibinaccijevih
brojeva\n");
        else printf(" NIJE polje fibinaccijevih brojeva\n");

        printf("Polje: ");
        ispisPolja(poljeNeFib, N);
        if(provjeraFib(poljeNeFib, N)) printf(" JE polje fibinaccijevih
brojeva\n");
        else printf(" NIJE polje fibinaccijevih brojeva\n");

        return 0;
}

void ispisPolja(int polje[], int n)
{
    int i;
    for(i=0; i<n; i++)
    {
        printf("%d ", polje[i]);
    }
}

int provjeraFib(int polje [], int n)
{
    int i;
    if(polje[0] != 1 || polje[1] != 1) return 0;
    for(i=2; i<n; i++)
    {
        if(polje[i] != polje[i-1] + polje[i-2])
        {
            return 0;
        }
    }
    return 1;
}

```

ZADACI

Definicija funkcije

1. zadatak

Definirati funkciju koja ne prima parametre, a uvijek vraća vrijednost 10 tipa `int`.

2. zadatak

Što je neispravno u definiciji funkcije `nacetvrtu`?

```
int nacetvrtu(int a){
    int a;
    return a*a*a*a;
}
```

Naredba `return`

1. zadatak

Koju vrijednost će vratiti funkcija ako ju pozovemo sa parametrom `2.7`?

```
pv(2.7);
```

```
int pv(float a){
    float b;
    b=2*a;
    return;
}
```

2. zadatak

Koju vrijednost će vratiti funkcija ako ju pozovemo sa parametrom `2.7`?

```
pv(2.7);
```

```
int pv(float a){
    float b;
    b=2*a;
    return b;
}
```

Void funkcije i funkcije bez argumenata

Što će ispisati sljedeći programi?

<u>1. zadatak</u>	<u>2. zadatak</u>
<pre>#include <stdio.h> void f(int a) { a*=a; printf("%d", a); } int main() { int a=3; f(a); printf("\n%d", a); return 0; }</pre>	<pre>#include <stdio.h> void f(int *a) { (*a)*=*a; printf("%d", *a); } int main() { int a=3; f(&a); printf("\n%d", a); return 0; }</pre>

Prijenos kopija vrijednosti (bez polja)

1. zadatak

Što će se ispisati obavljanjem sljedećeg programa?

```
char funkcija (int x, int y) {  
    return (char) x + y;  
}  
  
int main () {  
    int x = 40, y = 35;  
    printf ("%d, %d, %c", x, y, funkcija (x, y));  
    return 0;  
}
```

2. zadatak

Što će se ispisati izvođenjem sljedećeg programskog odsječka:

```
float funkcija (float f) {  
    if (f<0) return 2*f;  
    return f;  
}  
  
int main() {  
    int i=3, j=-3;  
    printf("%f %f",funkcija(i), funkcija(j));  
    return 0;  
}
```

Prijenos referencija-adresa (bez polja)

1. zadatak

Što će se ispisati sljedećim programom?

```
void f(int x, int *y) {  
    x = 4;  
    *y = x;  
}  
  
int main () {  
    int a=2, b=3;  
    f (a, &b);  
    printf ("%d %d\n", a, b);  
    return 0;  
}
```

2. zadatak

Što će ispisati na ekran sljedeći program?

```
void radi(char *x){
    char y = '1';
    *x = y;
}

int main(){
    char y = '3';
    radi(&y);
    printf("%c", y);
    return 0;
}
```

Jednodimenzionalna polja kao argumenti funkcije (rad s pokazivačima)

1. zadatak

Napisati funkciju ***negativni*** koja broji koliko elemenata u jednodimenzionalnom polju ima negativnu vrijednost. Napisati i poziv funkcije iz glavnog programa ?

```
#include <stdio.h>
int main () {
    int polje [25];
    int br_negativnih, i, n;
    printf ("Unesi broj elemenata i vrijednosti članova polja ");
    scanf ("%d", &n);
    for (i=0; i<n; i++) scanf ("%d", &polje[i]);
    .../* prebroji negativne brojeve*/
    return 0;
}
```

2. zadatak

Koji od sljedećih odgovora predstavlja ispravan prototip funkcije koja treba izračunati i vratiti najveći član u nizu?

- 1) void traziMax (long maxClan, int *niz, int brClan);
- 2) void traziMax (long *maxClan, int *niz, int brClan);
- 3) void traziMax (long maxClan, int niz, int brClan);
- 4) void traziMax (long *maxClan, int *niz[], int brClan);
- 5) void traziMax (long *maxClan, int niz, int brClan);

Smještajni razredi (postojanost, područje važenja varijabli) Samo elementarni pojmovi, po mogućnosti bez register i external!

Što će se ispisati u drugom pozivu funkcije f?

```
void f() {
    int a=3;
    static int b=2;
    b*=2;
    a+=b;
    printf("%d %d\n", b,a);
}
```

```
}
```

Dvodimenzionalna polje kao argument funkcije (rad s pokazivačima)

Ukoliko funkcija f treba izračunati sumu svih elemenata u matrici koju naredbu treba umetnuti na mjesto označeno s ### ?

```
int f(int *p, int m, int n, int maxstup){
    int i,j,suma=0;
    int *temp=p;
    for(i=0; i<m ; i++){
        for(j=0; j<n; j++){
            ###
        }
    }
    return suma;
}
```

- 1) suma += *temp + i * maxstup + j;
- 2) suma += temp + i * maxstup + j;
- 3) suma += *(temp[i * maxstup + j]);
- 4) suma += *(temp + i * maxstup + j);
- 5) suma += temp[i][j]

Jednodimenzionalna polja kao argumenti funkcije (rad s indeksnim izrazima)

Ukoliko se unutar funkcije u varijablu rez treba pridružiti umnožak elementa s indeksom retka 2 i indeksom stupca 4 iz matrice mat1 te elementa s indeksom retka 3 i indeksom stupca 5 iz matrice mat2 koje od sljedeći naredbi možemo navesti unutar sljedeće funkcije?

```
int f(int *mat1, int m, int n, int maxstup){
    int mat2[20][30];
    int *p = &mat2[0][0];
    ...
}
```

- 1) rez = mat1[2][4] * mat2[3][5];
- 2) rez = mat1[2*maxstup+4] * p[3*maxstup+5];
- 3) rez = mat1[2*maxstup+4] * mat2[3][5];
- 4) rez = mat1[2*maxstup+4] * p[3*30+5];
- 5) rez = mat1[2][4] * p[3*maxstup+5];
- 6) rez = mat1[2*maxstup+4] * p[3*20+5];

RJEŠENJA

Definicija funkcije

1. zadatak

```
int funkcija()  
{  
    return 10;  
}
```

2. zadatak

U funkciji se redefinira formalni parametar a, što nije dozvoljeno!

Naredba return

1. zadatak

Funkcija će uvijek vratiti nedefiniranu vrijednost, bez obzira na vrijednost ulaznog parametra!

2. zadatak

5

Void funkcije i funkcije bez argumenata

1. zadatak

9

3

2. zadatak

9

9

Prijenos kopija vrijednosti (bez polja)

1. zadatak

40, 35, K

2. zadatak

3.200000 -6.400000

Prijenos referencija-adresa (bez polja)

1. zadatak

2 4

2. zadatak

1

Jednodimenzionalna polja kao argumenti funkcije (rad s pokazivačima)

1. zadatak

```
int negativni (int *pom, int n) {  
    int i, br=0;  
    for (i=0; i<n; i++) if (pom[i]<0) br++;  
    return br;  
}
```

```
br_negativnih = negativni(polje, n);
```

2. zadatak

2

Smještajni razredi (postojanost, područje važenja varijabli) Samo elementarni pojmovi, po mogućnosti bez register i external!

Odgovor: 8 11

Dvodimenzionalna polje kao argument funkcije (rad s pokazivačima)

Odgovor: 4

Jednodimenzionalna polja kao argumenti funkcije (rad s indeksnim izrazima)

Odgovor: 3, 4