Programozás I.

6.

Függvények, memória verem

Feladat

- **5.7 Feladat**
  - Készítsen egy tízelemű, koordinátákat (x, y) tároló tömböt
  - Olvasson be 10 koordinátát
  - Jelenítse meg a koordinátákat
  - Rendezze a tömböt x komponens szerint növekvő sorrendbe
  - Jelenítse meg a tömböt
  - Rendezze a tömböt y komponens szerint növekvő sorrendbe
  - Jelenítse meg a tömböt

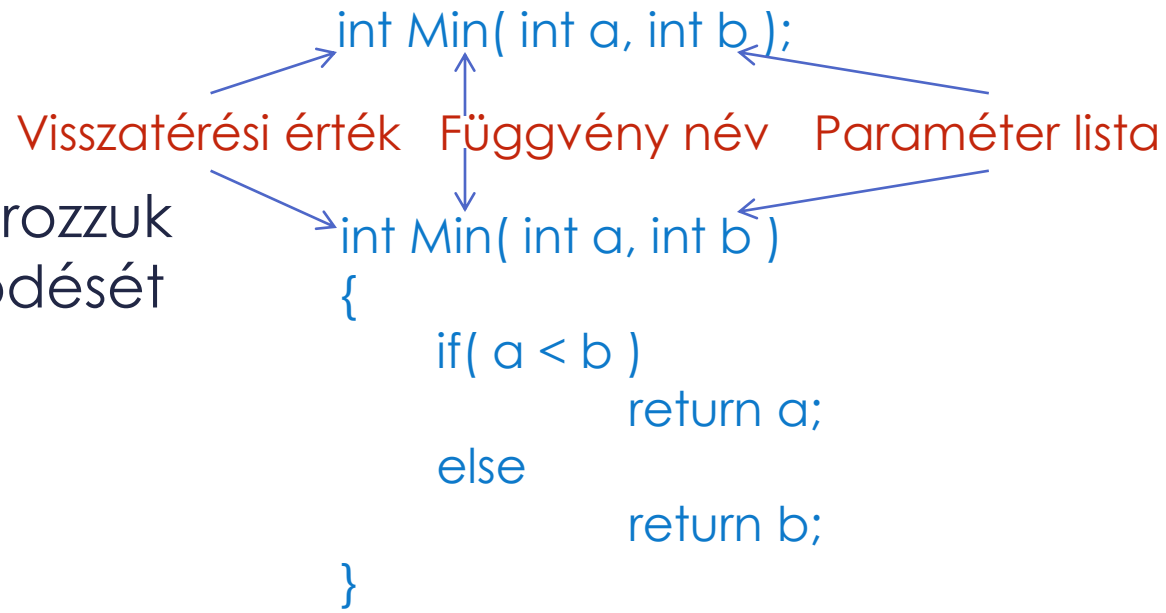
Függvények

Működése

- Függvényekbe a programkód bizonyos gyakran előforduló kódrészleteit rendezzük
- A függvények működésükhöz kaphatnak paramétereket
- Működésük eredményeképp visszaadhatnak egy értéket
- Ezekre a kódrészletekre a későbbiekben egyetlen paranccsal tudunk hivatkozni

Szintaktika

- Deklaráció
  - Minden névről meg kell mondani, hogy mire szeretnénk használni
- Definíció
  - Meghatározzuk a működését



Példa – tömb kiírása

```
void printArray(Coordinate coords[], int size)
{
        int i;
        for( i=0; i < size; i++ )
        {
                printf("( %d ; %d )", coords[i].x,
                                        coords[i].y);
        }
}
```

Paraméterek

- Változók: érték szerint **másolódnak**

```
int add(int a, int b){ a++; return a+b; }
int a = 5;
int b = 6;
int c = add( a, b);
```

Mennyi **a** értéke?

- Tömbök: ?

Paraméterek

- Tömbök esetén nem másolódik le a tartalom, így a függvényben végzett módosítások a paraméterben átadott tömbre is hatással lesznek.

```
void inc(int a[], int index){ a[index]++; }
int a[3] = { 0, 1, 2 };
inc(a, 0);
```

Mennyi **a** elemeinek értéke?

Feladat

- **6.5 Feladat**
  - Írjon függvényt, amely egy tömb elemeit írja ki a képernyőre
  - Készítsen egy tízelemű, koordinátákat (x, y) tároló tömböt
  - Olvasson be 10 koordinátát
  - Jelenítse meg a koordinátákat
  - Rendezze a tömböt x komponens szerint növekvő sorrendbe
  - Jelenítse meg a tömböt
  - Rendezze a tömböt y komponens szerint növekvő sorrendbe
  - Jelenítse meg a tömböt

Miért jó

- Segítségükkel a programunkat jobban tagolt, átláthatóbb formában készíthetjük el
- Jobban struktúrált program készíthető. Egy módosítás elvégzéséhez nem kell több helyen belenyúlni a kódba
- Lehetőségünk van külső könyvtárak használatára, mint például az “stdio.h” (printf, scanf)

Gyakorló feladatok

- **6.1 Feladat**
  - Írjon függvényt, amely két számot vár paraméterül és eredményül visszaadja a két szám összegét
  - Írjon programot, amely két számot olvas be mindaddig, míg azok összege nem osztható 7-tel. A feladat megoldásához használja fel az előzőleg megírt függvényt.
- **6.2 Feladat**
  - Írjon függvényt, amely két számot vár paraméterül és eredményül 1-et ad vissza, ha az első szám nagyobb a másodiknál, -1-et, ha a második nagyobb az elsőnél, 0-át, ha egyenlők
  - Olvasson be egy hételemű tömböt
  - Írassa ki a képernyőre azon szomszédos elemeket, amelyekre igaz, hogy az első nagyobb a másodiknál. A feladat megoldásához használja fel az előzőleg megírt függvényt.

Rekurzív függvények

- Rekurzív függvénynek nevezzük azokat, amelyek tartalmaznak hivatkozást önmagukra

```
int osszeg( int n )
{
        if( n == 1 )                    // leállási feltétel
                return 1;
        else
                return osszeg( n-1 ) + n;
}
```

Gyakorló feladatok

- **6.8 Feladat**
  - Írjon rekurzív függvényt, amely kiszámolja egy szám faktoriálisát
  - Olvasson be számot mindaddig, amíg az nem nulla. Írja ki a képernyőre a szám faktoriálisának értékét
- **6.9 Feladat**
  - Írjon rekurzív függvényt, amely visszaadja az n-edik fibonacci számot. A nulladik értéke 0, az elsőé 1, a továbbiaké pedig az előző kettő összege.
  - Olvasson be számot mindaddig, amíg az nem nulla. Írja ki a képernyőre az annyiadik fibonacci számot.

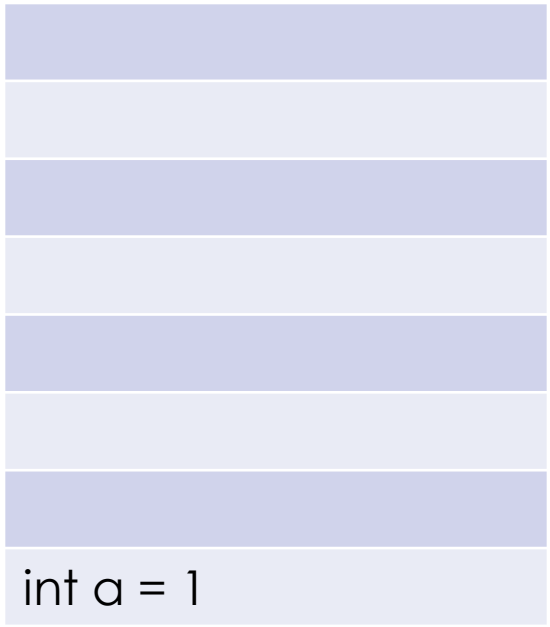
Memória verem

Mi történik a háttérben


```
int add( int p1, int p2)
{
        return p1 + p2;
}

int main()
{
        int a = 1;
        int b = 2;
        int c = add(a,b);
        return 0;
}
```

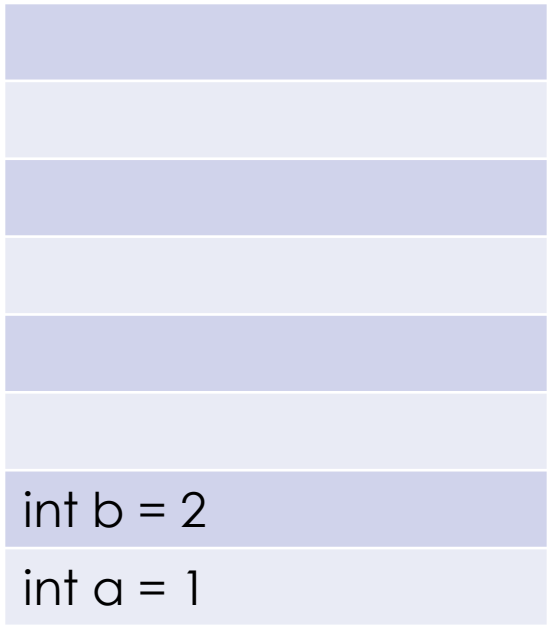
Mi történik a háttérben




```
int add( int p1, int p2)
{
        return p1 + p2;
}

int main()
{
        int a = 1;
        int b = 2;
        int c = add(a,b);
        return 0;
}
```

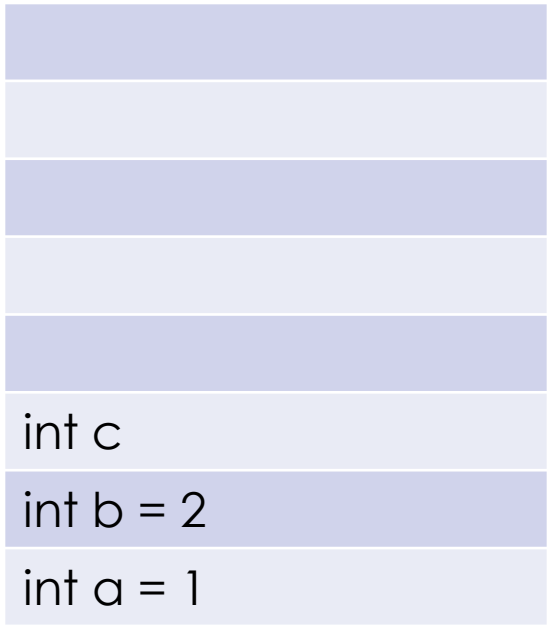
Mi történik a háttérben




```
int add( int p1, int p2)
{
        return p1 + p2;
}

int main()
{
        int a = 1;
        int b = 2;
        int c = add(a,b);
        return 0;
}
```

Mi történik a háttérben




```
int add( int p1, int p2)
{
        return p1 + p2;
}

int main()
{
        int a = 1;
        int b = 2;
        int c = add(a,b);
        return 0;
}
```

Mi történik a háttérben

Visszatérési cím
int p1 = 1
int p2 = 2
Visszatérési érték
int c
int b = 2
int a = 1

```
int add( int p1, int p2)
{
        return p1 + p2;
}

int main()
{
        int a = 1;
        int b = 2;
        int c = add(a,b);
        return 0;
}
```

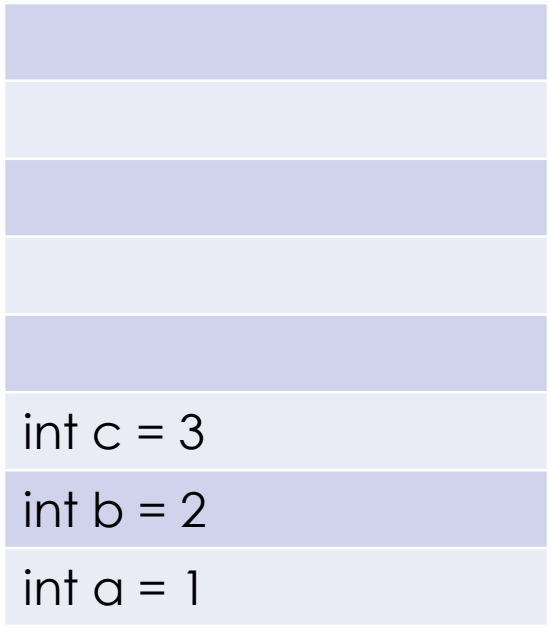
Mi történik a háttérben

Visszatérési cím
int p1 = 1
int p2 = 2
Visszatérési érték = 3
int c
int b = 2
int a = 1

```
int add( int p1, int p2)
{
        return p1 + p2;
}

int main()
{
        int a = 1;
        int b = 2;
        int c = add(a,b);
        return 0;
}
```

Mi történik a háttérben




```
int add( int p1, int p2)
{
        return p1 + p2;
}

int main()
{
        int a = 1;
        int b = 2;
        int c = add(a,b);
        return 0;
}
```