Másolásra épülő algoritmusok

Tartalomjegyzék

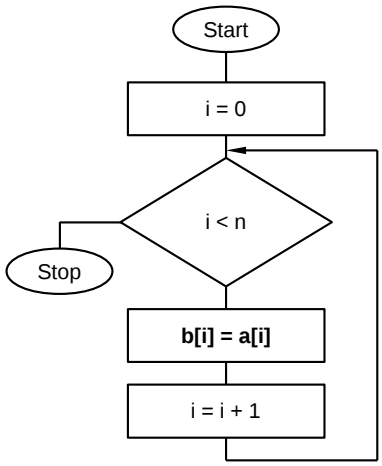
Másolás....2
Másolás és módosítás....3
Másolás és módosítás plusz....4
Tömbelemek módosítása....5
Kiválogatás....6
Szétválogat....7
Unió....8
Metszet....9
Összefuttatás....10
Összefuttatás ismétlődés nélkül....11

Készítette: Gál Tamás

Másolás

Mondatszerű leírás:

```
ciklus i = 0 .. n - 1
    b[i] = a[i]
ciklus vége
```



C# kód

```
int[] a = {101, 7, 8, 11, 255, 321};
int   n = a.Length; //a tömb mérete
int[] b = new int[n];

for(int i=0; i<n; i++) {
       b[i] = a[i];
}

//A c tömbbe töltött elemek száma: j
for (int i=0;i<n;i++)
{Console.Write(b[i]+", ");}
```

Java esetén:

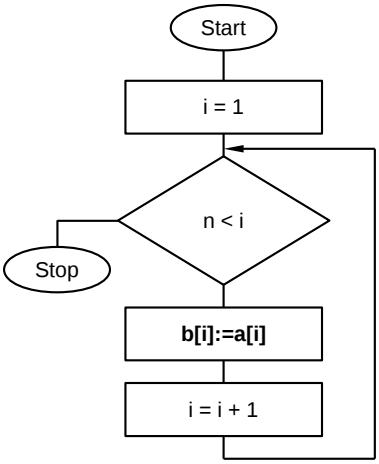
- **Length** helyett **length**
- **Console.Write** helyett **System.out.print**

Pascal esetén:

- A tömbindex rendszerint 1-től indul
- Kilépés a FOR ciklusból, ha a ciklusváltozó > mint a végérték

---

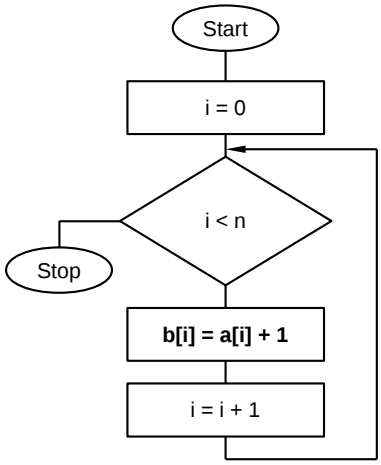
```
program Masol;
const n=20; {a tömb elemeinek a száma}
var a, b: array[1..n] of integer;
    i: integer;
begin
 randomize;
 for i:=1 to n do a[i]:=random(200);

 for i:=1 to n do begin
   b[i]:=a[i];
 end;

 for i:=1 to n do  write(a[i], ', ');writeln();
 for i:=1 to n do  write(b[i], ', ');
end.
```



Másolás és módosítás

Mondatszerű leírás:

```
ciklus i = 0 .. n - 1
    b[i] = a[i] + 1
ciklus vége
```



C# kód

```
int[] a = {101, 7, 8, 11, 255, 321};
int   n = a.Length; //a tömb mérete
int[] b = new int[n];

for(int i=0; i<n; i++) {
       b[i] = a[i] + 1;
}

//A c tömbbe töltött elemek száma: j
for (int i=0;i<n;i++)
{Console.Write(b[i]+", ");}
```

Java esetén:

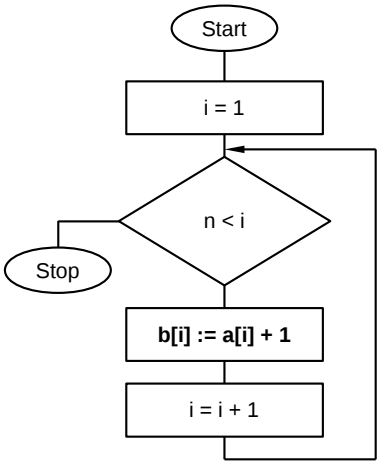
- **Length** helyett **length**
- **Console.Write** helyett **System.out.print**

Pascal esetén:

- A tömbindex rendszerint 1-től indul
- Kilépés a FOR ciklusból, ha a ciklusváltozó > mint a végérték

----

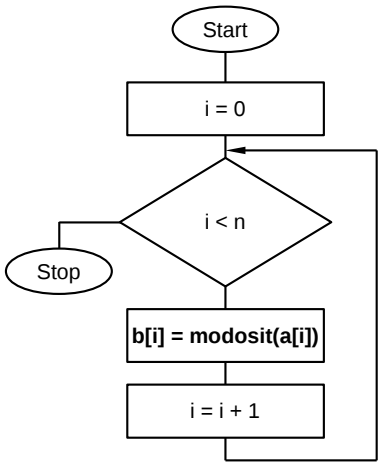
```
program MasolModosit;
const n=20; {a tömb elemeinek a száma}
var a, b: array[1..n] of integer;
    i: integer;
begin
 randomize;
 for i:=1 to n do a[i]:=random(200);

 for i:=1 to n do begin
   b[i]:=a[i]+1;
 end;

 for i:=1 to n do  write(a[i], ', ');writeln();
 for i:=1 to n do  write(b[i], ', ');
end.
```



Másolás és módosítás plusz

Mondatszerű leírás:

```
ciklus i = 0 .. n - 1
    b[i] = a[i] + 1
ciklus vége
```



C# kód

```
int[] a = {101, 7, 8, 11, 255, 321};
int   n = a.Length; //a tömb mérete
int[] b = new int[n];

for(int i=0; i<n; i++) {
       b[i] = modosit(a[i]);
}

//A c tömbbe töltött elemek száma: j
for (int i=0;i<n;i++)
{Console.Write(b[i]+", ");}
```

Java esetén:

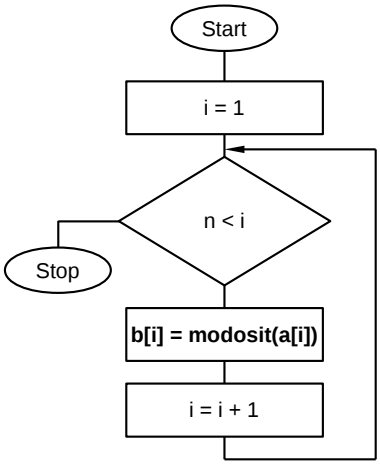
- **Length** helyett **length**
- **Console.Write** helyett **System.out.print**

Pascal esetén:

- A tömbindex rendszerint 1-től indul
- Kilépés a FOR ciklusból, ha a ciklusváltozó > mint a végérték

----

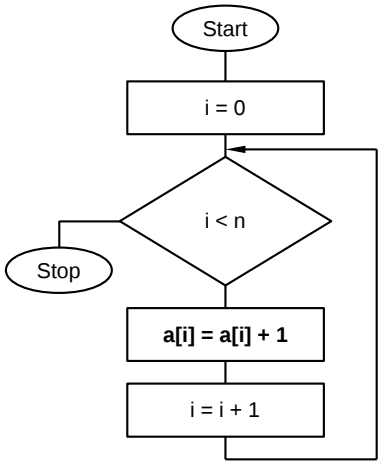
```
program MasolModositPlusz;
const n=20; {a tömb elemeinek a száma}
var a, b: array[1..n] of integer;
    i: integer;

function modosit(i:integer): integer;
begin
    modosit := i+1;
end;

begin
 randomize;
 for i:=1 to n do a[i]:=random(200);

 for i:=1 to n do begin
   b[i]:=a[i]+1;
 end;

 for i:=1 to n do  write(a[i], ', ');writeln();
 for i:=1 to n do  write(b[i], ', ');
end.
```



Tömbelemek módosítása

Mondatszerű leírás:

```
ciklus i = 0 .. n - 1
    a[i] = a[i] + 1
ciklus vége
```



C# kód

```
int[] a = {101, 7, 8, 11, 255, 321};
int   n = a.Length; //a tömb mérete

for(int i=0; i<n; i++) {
       a[i] = a[i] + 1;
}

//A c tömbbe töltött elemek száma: j
for (int i=0;i<n;i++)
{Console.Write(a[i]+", ");}
```

Java esetén:

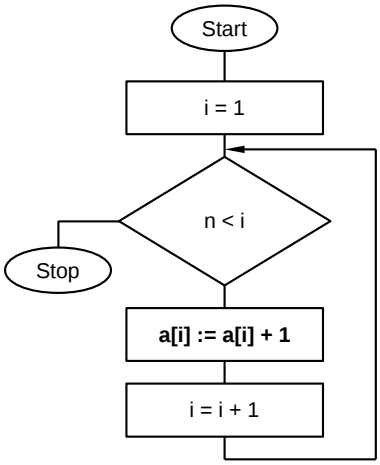
- **Length** helyett **length**
- **Console.Write** helyett **System.out.print**

Pascal esetén:

- A tömbindex rendszerint 1-től indul
- Kilépés a FOR ciklusból, ha a ciklusváltozó > mint a végérték

----

```
program TombModosit;
const n=20; {a tömb elemeinek a száma}
var a: array[1..n] of integer;
    i: integer;
begin
 randomize;
 for i:=1 to n do a[i]:=random(200);
 for i:=1 to n do  write(a[i], ', ');writeln();

 for i:=1 to n do begin
   a[i]:=a[i]+1;
 end;

 for i:=1 to n do  write(a[i], ', ');
end.
```



Kiválogatás

Mondatszerű leírás:

```
j = 0
ciklus i = 0 .. n - 1
    //Az 100-nál nagyobb számokat válogatjuk
  ha a[i] > 100
    b[j] = a[i]
    j = j + 1
  ha vége
ciklus vége
```

C# kód

```
int[] a = {101, 7, 8, 11, 255, 321};
int   n = a.Length; //a tömb mérete
int[] b = new int[n];

int j = 0;
for(int i=0; i<n; i++) {
      if(a[i] > 100)
      {
            b[j] = a[i];
            j++;
      }
}

//A b tömbbe töltött elemek száma: j
for (int i=0;i<j;i++)
{Console.Write(b[i]+", ");}
```

Java esetén:

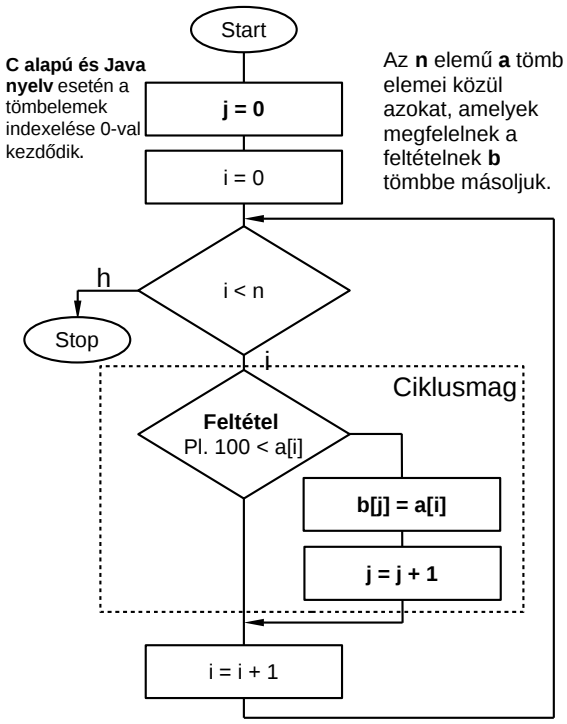
- **Length** helyett **length**
- **Console.Write** helyett **System.out.print**

Pascal esetén:

- A tömbindex rendszerint 1-től indul
- Kilépés a FOR ciklusból, ha a ciklusváltozó > mint a végérték

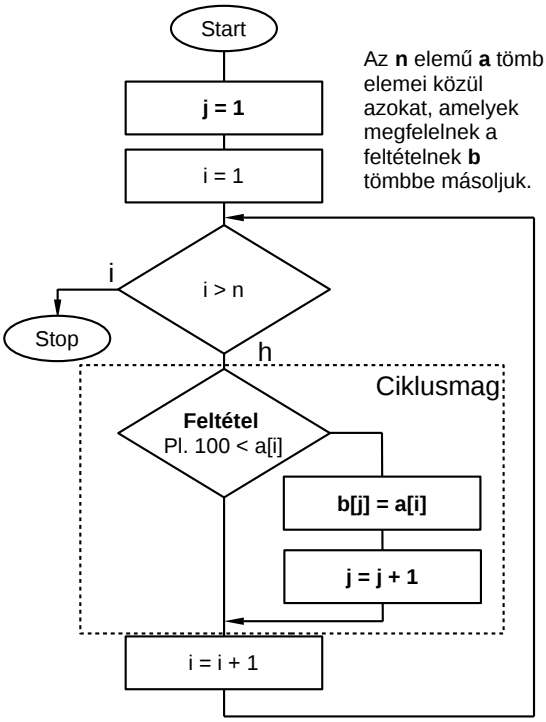
---

```
program Kivalogat;
const n=20; {a tömb elemeinek a száma}
var a, b: array[1..n] of integer;
    i, j: integer;
begin
 randomize;
 for i:=1 to n do a[i]:=random(200);
 j:=1;
 for i:=1 to n do begin
    if a[i]>100 then begin
        b[j]:=a[i];
        j:=j+1;
    end;
 end;
 for i:=1 to n do  write(a[i], ', ');
 writeln();
 for i:=1 to j-1 do  write(b[i], ', ');
end.
```



C alapú és Java nyelv esetén a tömbelemek indexelése 0-val kezdődik.

Az **n** elemű **a** tömb elemei közül azokat, amelyek megfelelnek a feltételnek **b** tömbbe másoljuk.



Az **n** elemű **a** tömb elemei közül azokat, amelyek megfelelnek a feltételnek **b** tömbbe másoljuk.

Szétválogat

Mondatszerű leírás:

```
j = 0
k = 0
ciklus i = 0 .. n-1
       ha a[i] > 100 akkor
              b[j] = a[i]
              j = j + 1
       különben
              c[k] = a[i]
              k = k + 1
       ha vége
ciklus vége
```

C# kód

```
int[] a = {109, 7, 3, 5, 344, 2, 6, 101};
int   n = a.Length;
int[] b = new int[n];
int[] c = new int[n];

int j = 0;
int k = 0;
for(int i=0; i<n; i++) {
       if(a[i] > 100)
       {
              b[j] = a[i];
              j++;
       }
       else
       {
              c[k] = a[i];
              k++;
       }
}

for (int i=0;i<j;i++) {Console.Write(b[i]+", ");}
Console.WriteLine();
for (int i=0;i<k;i++) {Console.Write(c[i]+", ");}
```

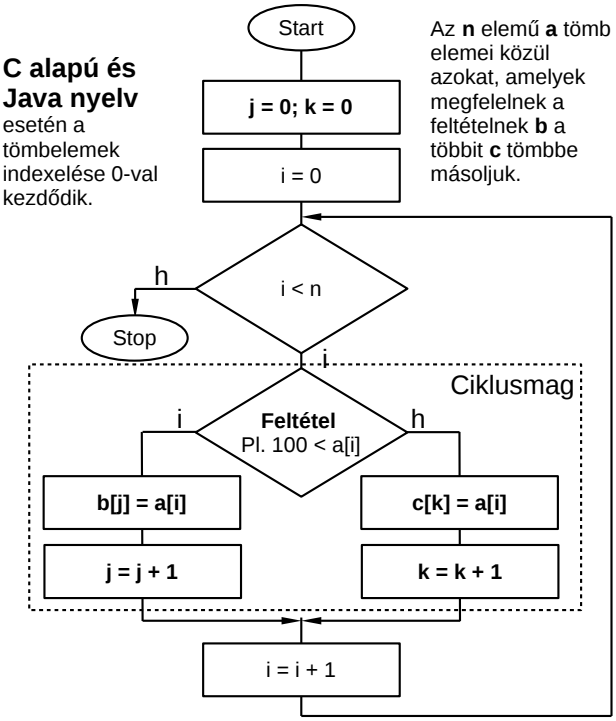
Java esetén:

- Length helyett **length**

- Console.Write helyett **System.out.print**

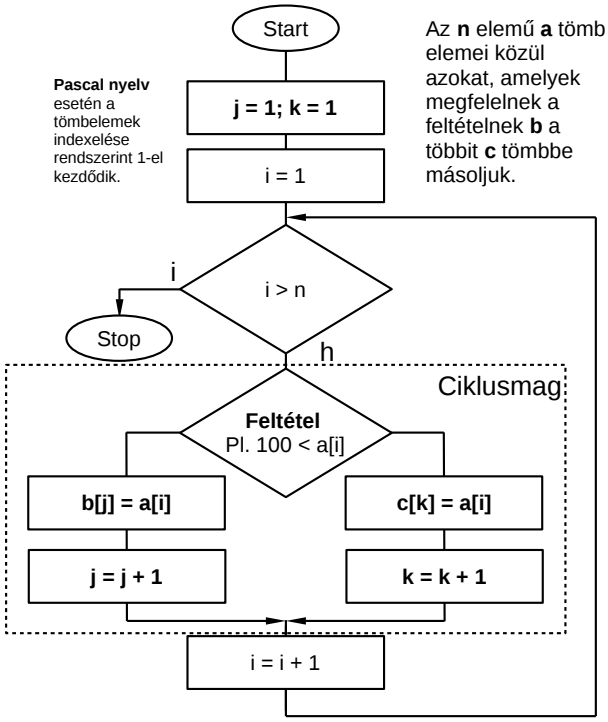
```
program Szetvalogat;
const n=20; {a tömb elemeinek a száma}
var a, b, c: array[1..n] of integer;
    i, j, k: integer;
begin
 randomize;
 for i:=1 to n do a[i]:=random(200);
 j:=1; k:=1;
 for i:=1 to n do begin
    if a[i]>100 then begin
        b[j]:=a[i];
        j:=j+1;
    end else begin
        c[k]:=a[i];
        k:=k+1;
    end;
 end;
 for i:=1 to n   do  write(a[i], ', ');
writeln();
 for i:=1 to j-1 do  write(b[i], ', ');
writeln();
 for i:=1 to k-1 do  write(c[i], ', ');
end.
```

C alapú és Java nyelv esetén a tömbelemek indexelése 0-val kezdődik.



Az **n** elemű **a** tömb elemei közül azokat, amelyek megfelelnek a feltételnek **b** a többit **c** tömbbe másoljuk.

Pascal nyelv esetén a tömbelemek indexelése rendszerint 1-el kezdődik.



Az **n** elemű **a** tömb elemei közül azokat, amelyek megfelelnek a feltételnek **b** a többit **c** tömbbe másoljuk.

Unió

Mondatszerű leírás:
n = a tömb mérete
m = b tömb mérete

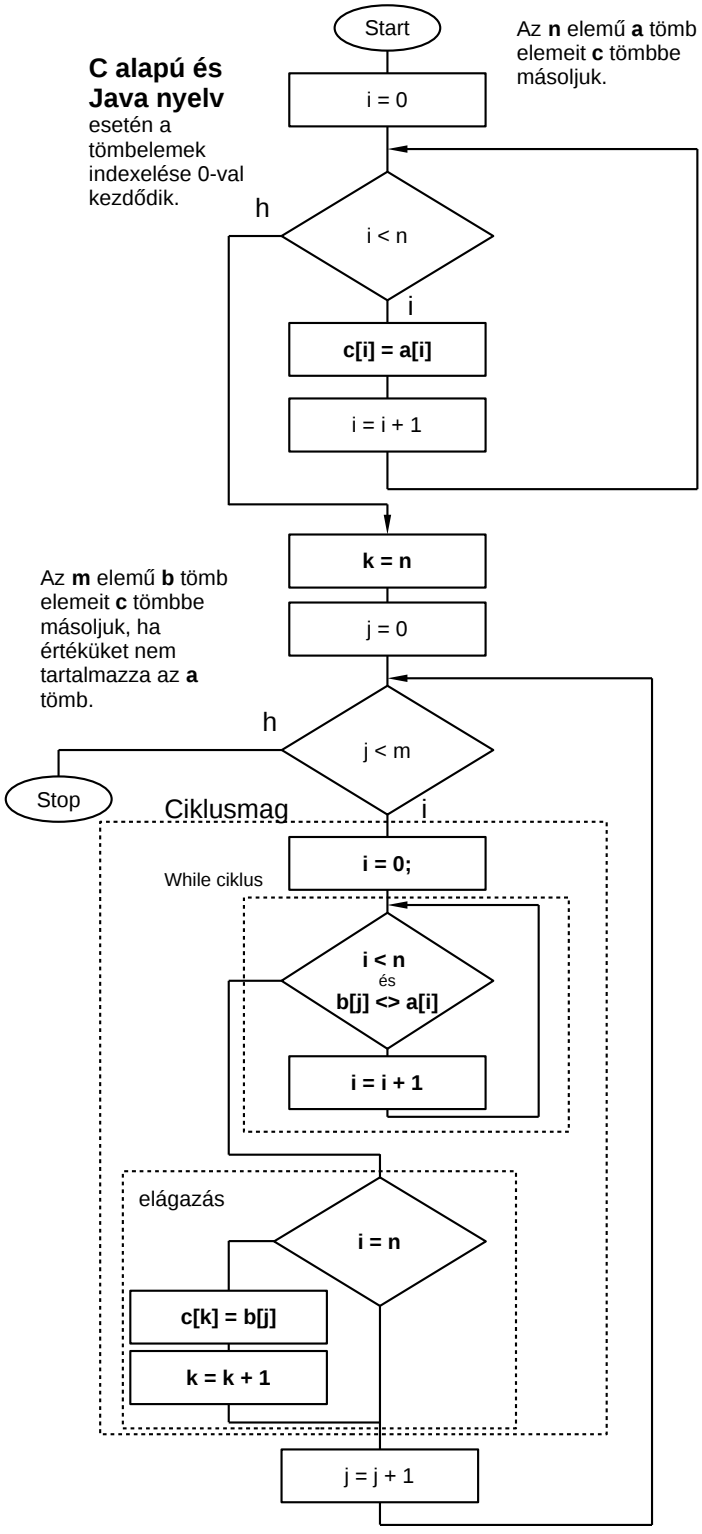
```
// a tömb elemeit átmásoljuk c-be
ciklus i = 0 .. n-1
  c[i] = a[i]
ciklus vége

k = n

// végig lépkedünk a b tömbön
ciklus j = 0 .. m-1

  // ellenőrizzük, hogy b[j] értéke
     megtalálható-e már c-ben
  i = 0
  ciklus amíg i<n és b[j]<>a[i]
    i = i + 1
  ciklus vége

  // ha b[j] értéke még nincs c-ben,
    akkor betesszük (Ekkor i=n)
  ha i=n akkor
    c[k] = b[j]
    k = k + 1
  ha vége
ciklus vége
```

C# kód

```
int[] a = {3, 5, 8, 4};
int[] b = {2, 3, 7, 9};

int n=a.Length, m=b.Length;
int   o = n + m;
int[] c = new int[o];

int i, j, k;

//Unió tétel
for(i=0; i<n; i++) {c[i] = a[i];}

k=n;
for(j=0; j<m; j++)
{
       i = 0;
       while(i<n && b[j] != a[i]) {i++;}

       if(i == n)
       {
              c[k] = b[j];
              k++;
       }
}
o = k; //A c tömbbe töltött elemek száma
for(i=0; i<o; i++)
{Console.Write(c[i]+", ");}
```

Java esetén:
- Length helyett **length**
- Console.Write helyett **System.out.print**

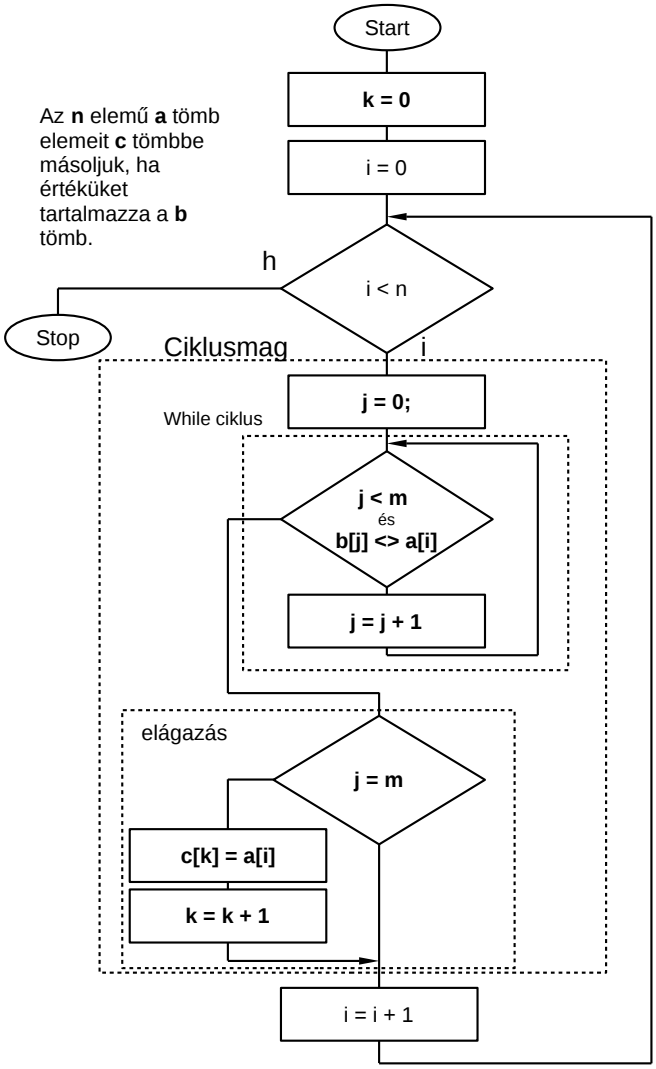
```
for (i=0;i<o;i++)
{System.out.print(c[i]+", ");}
```



Metszet

Mondatszerű leírás:

```
n = a tömb mérete
m = b tömb mérete

k = 0
// végig lépkedünk az a tömbön
ciklus i = 0 .. n-1
      j = 0
      //  ellenőrizzük, hogy a[i] értéke
            megtalálható-e b-ben
      ciklus amíg j<m és b[j]<>a[i]
            j = j + 1
      ciklus vége

   // ha a[i] értéke megtalálható b-ben,
      akkor betesszük a c tömbbe
      ha j<m akkor
            c[k] = a[i]
            k = k + 1
      ha vége
ciklus vége
```



C# kód

```
int[] a = {5, 9, 3, 4, 7 };
int[] b = {6, 5, 7, 8, 15, 20 };
int   n = a.Length; //a tömb mérete
int   m = b.Length; //b tömb mérete
int   o = n + m;
int[] c = new int[o];
int j;

int k = 0;
for (int i=0; i<n; i++)
{
      j = 0;
      while(j<m && b[j] != a[i]) {j++;}
      if (j<m)
      {
            c[k] = a[i];
            k++;
      }
}
o = k; //A c tömbbe töltött elemek száma

for(int i=0; i<o; i++) {Console.Write(c[i]+", ");}
```

Java esetén:
- Length helyett **length**
- Console.Write helyett **System.out.print**

```
for (int i=0;i<o;i++) {System.out.print(c[i]+", ");}
```

Összefuttatás

Mondatszerű leírás:

*// Az **n** elemű **a** és a **m** elemű **b** rendezett tömbök elemeit a **c** tömbbe másoljuk a sorrend megtartásával*

```
adb=0;
bdb=0;
cdb=0;
```

*// az **a** és a **b** tömb elemei közül mindig a következő legkisebb értékűt illesztjük a **c** tömb végére*

```
ciklus amíg adb<n és bdb<m
  Ha (a[adb]<b[bdb]) akkor
    c[cdb] = a[adb];
    adb++;
  egyébként
    c[cdb] = b[bdb];
    bdb++;
  ha vége
  cdb++;
ciklus vége
```

// Ha már csak az **a** tömbben maradtak értékek, akkor azokat másoljuk a **c** -be

```
ciklus amíg adb<n
  c[cdb] = a[adb];
  adb++;
  cdb++;
ciklus vége
```

// Ha már csak a b tömbben maradtak értékek, akkor azokat másoljuk a **c** -be

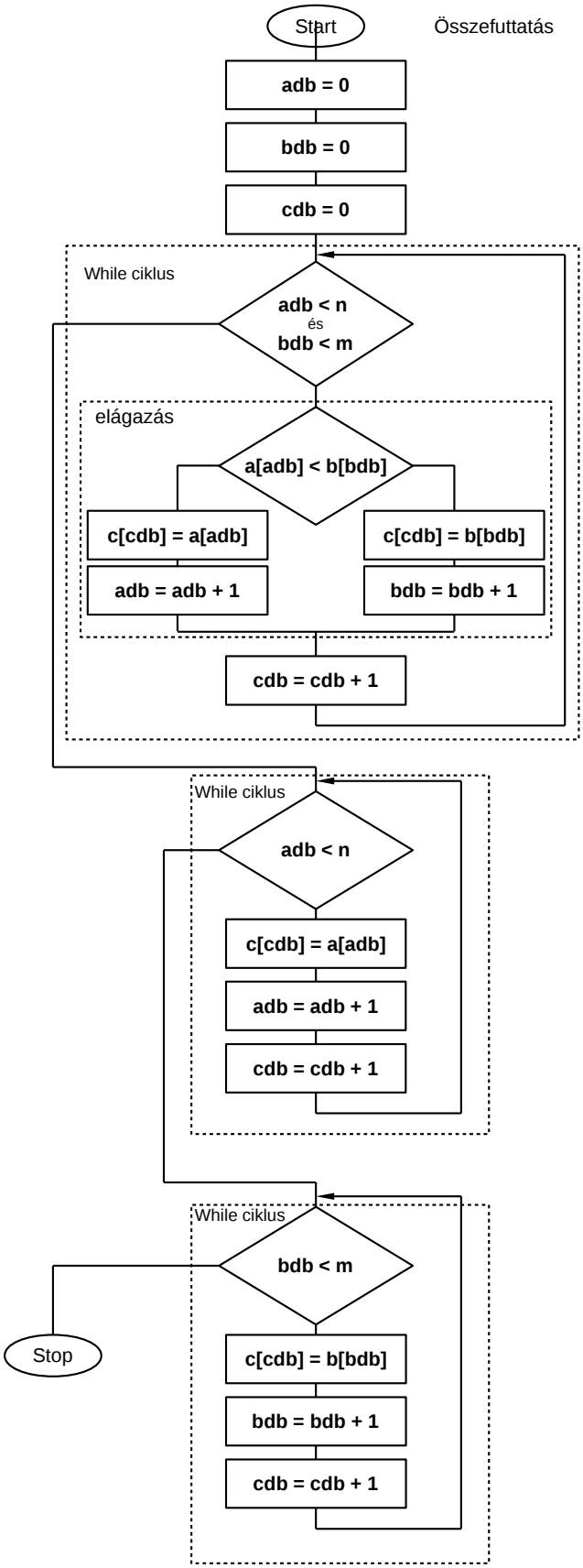
```
ciklus amíg bdb<m
  c[cdb] = b[bdb];
  bdb++;
  cdb++;
ciklus vége
```

C# kód

```
int[] a = {1, 7, 8, 11};
int[] b = {2, 5, 8, 9};
int   n = a.Length; //a tömb mérete
int   m = b.Length; //b tömb mérete
int   o = n + m;
int[] c = new int[o];

int adb=0;
int bdb=0;
int cdb=0;
while (adb<n && bdb<m) {
       if (a[adb]<b[bdb])
       {
              c[cdb] = a[adb];
              adb++;
       } else {
              c[cdb] = b[bdb];
              bdb++;
       }
       cdb++;
}
while (adb<n) {
       c[cdb] = a[adb];
       adb++; cdb++;
}
while (bdb<m) {
       c[cdb] = b[bdb];
       bdb++; cdb++;
}
// Az eredmény megjelenítése:
for (int i=0;i<o;i++)
{Console.Write(c[i]+", ");}
```

Java esetén:

- **Length** helyett **length**
- **Console.Write** helyett **System.out.print**



Összefuttatás ismétlődés nélkül

Az előző összefuttatásnál, ha egy érték mindkét tömbben szerepel akkor kétszer lesz beillesztve a c tömbbe.

Ennek elkerülésére a következő elemek értékének egyenlőségét is vizsgálni kell. Ha azonosak, akkor mindkét tömbmutatót növeljük 1-el.

```
while (adb<a.Length && bdb<b.Length)
{
	if (a[adb]==b[bdb])
	{
		c[cdb] = a[adb];
		adb++;
		bdb++;
	}
	else if (a[adb]<b[bdb])
	{
		c[cdb] = a[adb];
		adb++;
	}
	else
	{
		c[cdb] = b[bdb];
		bdb++;
	}
	cdb++;
}
```