

Programozás I.

3. előadás

Tömbök a C#-ban

Metódusok C#-ban

Egyszerű programozási tételek

Sergyán Szabolcs

`sergyan.szabolcs@nik.uni-obuda.hu`

Óbudai Egyetem

Neumann János Informatikai Kar

Szoftvertchnológia Intézet

2013. szeptember 23.



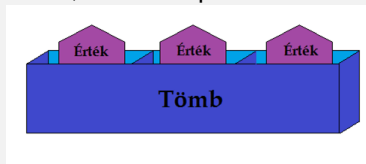
- 1 Tömbök a C#-ban
- 2 Metódusok C#-ban
- 3 Programozási tételek
- 4 Egyszerű programozási tételek
 - Sorozatszámítás
 - Eldöntés
 - Kiválasztás
 - Lineáris keresés
 - Megszámlálás
 - Maximumkiválasztás



- 1 Tömbök a C#-ban
- 2 Metódusok C#-ban
- 3 Programozási tételek
- 4 Egyszerű programozási tételek
 - Sorozatszámítás
 - Eldöntés
 - Kiválasztás
 - Lineáris keresés
 - Megszámlálás
 - Maximumkiválasztás



- Több, azonos típusú változó együttes kezelését teszi lehetővé

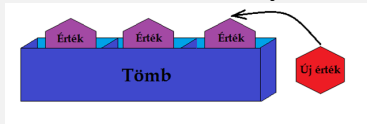


- `int a1; int a2; int a3; int a4; ...` helyett egyetlen "`int tömb`" típusú változó használata egy névvel
- Elemei a tömbön belüli sorszámukkal (index) érhetőek el (kezdő index: C#-ban 0, pseudokódban: 1)
- A tömbben tárolt típus és a tömb mérete nem módosítható → szigorú adatszerkezet, cserében nagyon gyors

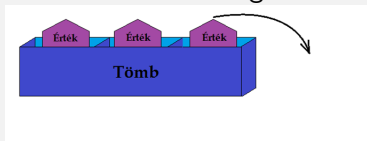


Tömbbel végezhető tevékenységek

- Deklaráció: a tömb nevének és elemei típusának megadása
- Tömblétrehozás: a tömb méretének meghatározása
- Értékadás: érték elhelyezése egy tömbelemen



- Érték lekérdezése: egy tömbelem tartalmának kiolvasása. Az érték a kiolvasás után is megmarad a tömbelemen



Tömbbel végezhető tevékenységek C#-ban

- 1 Deklaráció: `int[] tomb;`
- 2 Tömblétrehozás: `tomb = new int[10];`
- 3 Értékadás: `tomb[5] = 25;` vagy `tomb[5] = 6 * 2 - 29;`
- 4 Érték lekérdezése: `5 * 10 - tomb[5] + 2`

A deklaráció és a tömblétrehozás össze is vonható:

```
int[] tomb = new int[10];
```



Tömbelem elérése (indexelés)

- A tömb egy adott eleméhez a tömb neve után szögletes zárójelek között megadott sorszámmal (index) férhetünk hozzá:
`tömbnév[index]`
- Az index csak nemnegatív egész szám lehet
- A tömb első elemének indexe: 0
- A tömb utolsó elemének indexe: elemszám - 1
- Kisebb, vagy nagyobb index megadása futási hibát okoz.

	0.	1.	2.	3.	4.
tomb	28	3	17	11	50

↑
tomb[3]

tomb = **new** int[5];



Tömb hosszának (elemei számának) lekérdezése

- Általános formátum: `tömbnév.Length`
- A tömbben lévő elemek számát adja meg

	0.	1.	2.	3.	4.
tomb	28	3	17	11	50

`tomb = new int[5];`

`tomb.Length`



Tömb inicializálása

- A tömb deklarációja, létrehozása és elemeinek megadása egy utasításban is elvégezhető
- Formátuma: `típus[] tömbnév = {elem1, elem2, ..., elemN};`
- Példák:
`double[] valosak = {2.0, -3.5, 8.2, -1234.56};`
`bool[] logikai = {true, false, false, true, true};`



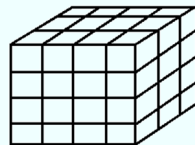
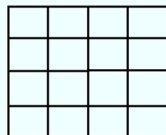
Tömbbel végezhető tevékenységek C#-ban

- A tevékenységek sorrendje:
 - 1 Deklaráció
 - 2 Tömblétrehozás
 - 3 Értékadás egy tömbelemnek, vagy egy tömbelem értékének lekérdezése
- A fentiek közül akármelyik tevékenységet szeretnénk is végrehajtani, előbb a sorrendben őt megelőzőt kell elvégezni
- A tömbelemeknek nem kötelező értéket adni az érték lekérdezése előtt
- Értékadás hiányában a tömbelem a típus alapértékét veszi fel



Többdimenziós tömbök

- 2 dimenziós tömb
 - sorok és oszlopok
 - elem elérése 2 indexszel
- 3 dimenziós tömbök
 - sorok, oszlopok, lapok
 - elem elérése 3 indexszel
- N dimenziós tömbök
 - 0., 1., ..., N . dimenzió
 - elem elérése N indexszel



Többdimenziós tömbök – Deklaráció

- Általános formátum: `típus[vesszők] tömbnév;`
- A szögletes zárójelbe dimenziószám - 1 darab vesszőt kell tenni
- Példák:
 - `int[,] matrix;`
 - `bool[, ,] haromdimenziostomb;`
 - `double[, , ,] otdimenziostomb;`



Többsdimenziós tömbök – Tömblétrehozás

- Általános formátum:
`tömbnév = new típus[elemszám1, ..., elemszámN];`
- Az egyes dimenziók elemszámait vesszőkkel elválasztva kell megadni
- A deklaráció és a tömblétrehozás itt is összevonható
- Példák
 - `matrix = new int[3, 5];`
 - `haromdimenziostomb = new bool[4, 2, 5];`
 - `int[,] t = new int[3, 3, 3];`
 - `int[,] egeszmatrix = {{1, 2, 3, 4}, {5, 6, 7, 8}, {9, 0, 1, 2}};`



Többdimenziós tömbök – Tömbelem elérése (indexelés)

- A szögletes zárójelek közé a tömbelem minden egyes dimenzióján belüli sorszámait kell vesszőkkel elválasztva megadni:
`tömbnév[index1, index2, ..., indexN]`
- Az indexekre vonatkozó szabályok ugyanazok, mint az egydimenziós tömbnél
- Pontosan annyi indexet kell megadni, ahány dimenziós a tömb

	0.	1.	2.	3.	4.
0.	28	3	17	11	50
1.	22	14	38	20	1

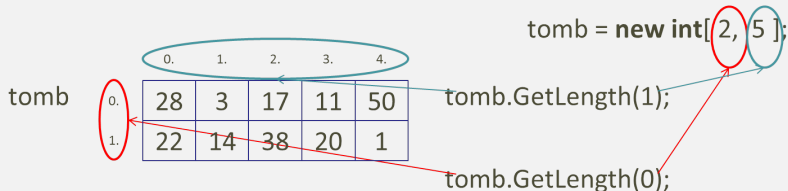
tomb = **new int**[2, 5];

tomb[1, 3]



Többszimenziós tömbök – Tömb méretének lekérdezése

- Elemek számának lekérdezése:
 - Összes tömbben lévő elem darabszáma:
`tömbnév.Length`
 - Egy adott dimenzió elemszáma (sorok száma, oszlopok száma, ...):
`tömbnév.GetLength(dimenziósorszám)`



Tömb bejárása – for utasítás

for(inicializátor; feltétel; iterátor)

- Az inicializátor és az iterátor tetszőleges utasítás lehet
- Működése:
 - Belépéskor egyszer végrehajtódik az inicializátor
 - Minden ciklusmenetben kiértékelődik a feltétel
 - Amennyiben a feltétel igaz, a ciklusmagban lévő utasítások egyszer végrehajtnak
 - A ciklusmag végeztével végrehajtódik az iterátor és ismét kiértékelődik a feltétel
 - A ciklus akkor ér véget, amikor a feltétel hamissá válik, ellenkező esetben újabb ciklusmenet következik
- Általában az inicializátor egy számlálót állít be, az iterátor pedig ezt a számlálót növeli vagy csökkenti
 - Legtöbbször akkor használjuk, ha előre ismert számú alkalommal szeretnénk végrehajtani egy utasítást



A for utasítás (példa)

```
//Számmátrix
```

```
//Ez a külső ciklus fut végig az összes soron
```

```
for (int i = 0; i < 100; i += 10)
```

```
{
```

```
    //Ez a belső ciklus fut végig egy soron belül
```

```
    //az összes oszlopon
```

```
    for (int j = i; j < i + 10; j++)
```

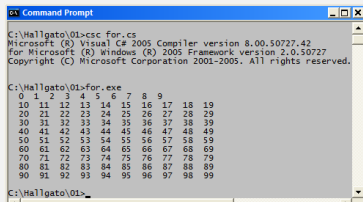
```
    {
```

```
        Console.Write(" {0}", j);
```

```
    }
```

```
    Console.WriteLine();
```

```
}
```



```
Command Prompt
C:\Hallgato\01>csc for.cs
Microsoft (R) Visual C# 2005 Compiler version 8.00.50727.42
for Microsoft (R) Windows (R) 2005 Framework version 2.0.50727
Copyright (C) Microsoft Corporation 2001-2005. All rights reserved.

C:\Hallgato\01>for.exe
0 1 2 3 4 5 6 7 8 9
10 11 12 13 14 15 16 17 18 19
20 21 22 23 24 25 26 27 28 29
30 31 32 33 34 35 36 37 38 39
40 41 42 43 44 45 46 47 48 49
50 51 52 53 54 55 56 57 58 59
60 61 62 63 64 65 66 67 68 69
70 71 72 73 74 75 76 77 78 79
80 81 82 83 84 85 86 87 88 89
90 91 92 93 94 95 96 97 98 99

C:\Hallgato\01>
```

Tömb bejárása – foreach utasítás

foreach (típus változó in gyűjtemény)

- Lehetővé teszi egy utasítás végrehajtását egy adott gyűjtemény összes elemére
 - A "gyűjtemény" pontos fogalmát később részletesen tárgyaljuk
 - A tömbök gyűjtemények, tehát a foreach utasítás tömbökkel is használható
- Működése:
 - Belépéskor létrejön egy típus típusú változó ("iterációs változó")
 - Ez a változó csak az utasításon belül használható
 - Az utasítás annyiszor hajtódik végre, ahány elemet tartalmaz a gyűjtemény
 - Az iterációs változó minden egyes végrehajtásnál felveszi a gyűjtemény soron következő elemének értékét
- Az iterációs változó az utasításban nem módosítható
 - Erre a célra a for utasítás használható



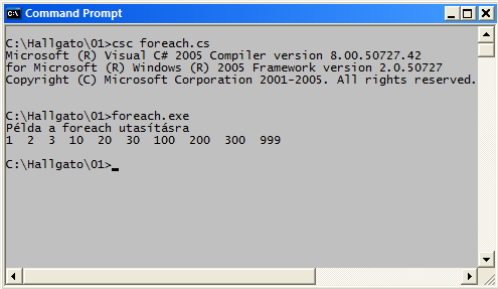
A foreach utasítás (példa)

```
int[] tesztömb = {1, 2, 3, 10, 20, 30, 100, 200, 300, 999};
```

```
Console.WriteLine("Példa a foreach utasításra");
```

```
foreach (int tömbérték in tesztömb)
{
    Console.Write("{0} ", tömbérték);
}
```

```
Console.WriteLine();
```



```
C:\Hallgato\01>csc foreach.cs
Microsoft (R) Visual C# 2005 Compiler version 8.00.50727.42
for Microsoft (R) Windows (R) 2005 Framework version 2.0.50727
Copyright (C) Microsoft Corporation 2001-2005. All rights reserved.

C:\Hallgato\01>foreach.exe
Példa a foreach utasításra
1 2 3 10 20 30 100 200 300 999

C:\Hallgato\01>
```

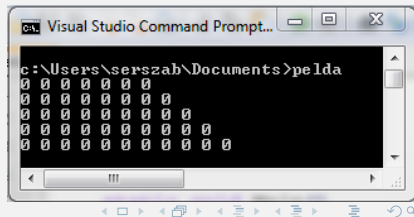
Fűrészfogas tömb

- A fűrészfogas tömb tömbök tömbje
- A belső (azaz tartalmazott) tömbök mérete különböző lehet
- A belső tömbök méretének felső határát minden egyes belső tömb külön-külön meg kell adni



Fűrészfogas tömb

```
int[] [] myJagArray = new int[5] [];  
  
for (int i = 0; i < myJagArray.Length; i++)  
    myJagArray[i] = new int[i+7];  
  
for (int i = 0; i < 5; i++)  
{  
    for (int j = 0; j < myJagArray[i].Length; j++)  
        Console.Write(myJagArray[i][j] + " ");  
    Console.WriteLine();  
}
```



```
Visual Studio Command Prompt...  
c:\Users\serszab\Documents>pelda  
0 0 0 0 0 0 0  
0 0 0 0 0 0 0 0  
0 0 0 0 0 0 0 0 0  
0 0 0 0 0 0 0 0 0 0  
0 0 0 0 0 0 0 0 0 0 0  
0 0 0 0 0 0 0 0 0 0 0 0
```

- 1 Tömbök a C#-ban
- 2 Metódusok C#-ban**
- 3 Programozási tételek
- 4 Egyszerű programozási tételek
 - Sorozatszámítás
 - Eldöntés
 - Kiválasztás
 - Lineáris keresés
 - Megszámlálás
 - Maximumkiválasztás



- A metódus egy kódblokk, amely utasítások sorozatát tartalmazza
- A program azáltal futtatja ezt a kódblokkot, hogy **meghívja** a metódust és megszabja a szükséges paramétereit
- C#-ban minden futtatandó utasítás egy metódusban helyezkedik el
 - Eddig programjainkat a `Main()` metódusba írtuk
- A többször használt kódrészeket írjuk metódusba
 - "Copy-Paste" helyett
 - Célszerű a hosszú metódusok feldarabolása az egyszerűbb értelmezés céljából



Szintaktika

```
visszatérési_típus metódusnév(paraméterek)  
{metódustörzs}
```



Metódusok típusa

- A típus a visszaadott érték típusa vagy void, ha nem adunk vissza semmit
- Egy metódusnak legfeljebb egy visszatérési értéke van
 - de az lehet tömb is
- A visszatérési érték típus előtt állhatnak különféle módosítók
 - Pl. static, abstract, override, new, illetve láthatóságot jelző kulcsszavak



- A metódushoz tartozó utasítások, amelyek használhatják a metódusnak átadott paramétereket
- A metódus visszatérési értékét a **return** kulcsszó után adjuk meg. Ennek hatására a metódusból azonnal visszatérünk a hívóhoz, akkor is, ha még vannak további utasítások a **return** után.
- Ha a metódus több ágon is véget érhet, akkor minden ág végére kell **return**
- Visszatérési érték nélküli (**void**) metódusnál – ha a program mindig a metódustörzs fizikai végénél fejeződik be – a **return** utasítás elhagyható



- Egy blokkban deklarált változók csak a deklarálástól kezdve a blokk végéig elérhetők
 - Következmény: az egyik metódusban deklarált x változó nem ugyanaz, mint a másik metódusbeli x változó
- A hívó környezet változói nem érhetők el a metódusban
 - Ezért szükséges a paraméter átadás és a visszatérési érték
 - Közös adat használható: globális változók, de használatuk nem javasolt



Téglalap területének számítása

```
static int terület(int a, int b) //paraméterek átadása
{
    return a * b;
}

static void Main()
{
    int egyikOldal = 5;
    int másikOldal = 7;
    Console.WriteLine("A téglalap területe: "
        + terület(egyikOldal, másikOldal));
}
```



- 1 Tömbök a C#-ban
- 2 Metódusok C#-ban
- 3 Programozási tételek**
- 4 Egyszerű programozási tételek
 - Sorozatszámítás
 - Eldöntés
 - Kiválasztás
 - Lineáris keresés
 - Megszámlálás
 - Maximumkiválasztás



- A programozási tételek jól megválasztott, egyszerű feladatok megoldásai
 - Segítségükkel a gyakorlatban szükséges feladatok jelentős része megoldható
 - Helyességük egyszerűen bizonyítható
 - Használatuk célszerű, hiszen (mások által is) jól áttekinthető kódot eredményeznek
- Egy lehetséges csoportosításuk
 - Egy sorozathoz egy értéket rendelő feladatok
 - Egy sorozathoz egy sorozatot rendelő feladatok
 - Egy sorozathoz több sorozatot rendelő feladatok
 - Több sorozathoz egy sorozatot rendelő feladatok
- Feldolgozandó intervallum alapján megkülönböztetünk
 - Rögzített intervallumos programozási tételeket
 - Feltételig tartó programozási tételeket (ezeket a változatokat nem tárgyaljuk)



- 1 Tömbök a C#-ban
- 2 Metódusok C#-ban
- 3 Programozási tételek
- 4 Egyszerű programozási tételek**
 - Sorozatszámítás
 - Eldöntés
 - Kiválasztás
 - Lineáris keresés
 - Megszámlálás
 - Maximumkiválasztás



- 1 Tömbök a C#-ban
- 2 Metódusok C#-ban
- 3 Programozási tételek
- 4 Egyszerű programozási tételek**
 - **Sorozatszámítás**
 - Eldöntés
 - Kiválasztás
 - Lineáris keresés
 - Megszámlálás
 - Maximumkiválasztás



Típusfeladatok

- 1 Egy osztály N darab tanulójának osztályzata alapján adjuk meg az osztály átlagát!
- 2 Egy M elemű betűsorozat betűit fűzzük össze egyetlen szöveg típusú változóba!
- 3 Készítsünk algoritmust, amely egy autóversenyző körönkénti ideje alapján meghatározza a versenyző egy kör megtételéhez szükséges átlagidejét!
- 4 A Balaton mentén K darab madarász végzett megfigyeléseket. Mindegyik megadta, hogy milyen madarakat látott. Készítsünk algoritmust, amely a megfigyelések alapján megadja a Balatonon előforduló madárfajokat!
- 5 Adjuk meg az első N darab pozitív egész szám szorzatát!



Sorozatszámítás

Bemenet

A: Feldolgozandó tömb
N: Tömb elemeinek száma

Kimenet

R: Művelet eredménye

Pszeudokód

Eljárás Sorozatszámítás(A , N , R)

$R \leftarrow R_0$

Ciklus $i \leftarrow 1$ -től N -ig

$R \leftarrow R$ művelet $A[i]$

Ciklus vége

Eljárás vége



Megjegyzések

- Adatok sorozatához egy értéket rendelő függvényt helyettesít
- Minden olyan esetben használható, ha ezt a függvényt felbonthatjuk értékpárokon kiszámított függvények sorozatára
- Az induláskor használt nullértéket értelemszerűen a kérdéses függvény (esetleg a feladat) alapján kell megválasztani

	összegzés	faktoriális	elemek uniója
R_0	0	1	{ }
művelet	$R \leftarrow R + A[i]$	$R \leftarrow R * A[i]$	$R \leftarrow R \cup A[i]$



Sorozat elemeinek összegzése

Eljárás Sorozatszámítás(A , N , R)

$R \leftarrow 0$

Ciklus $i \leftarrow 1$ -től N -ig

$R \leftarrow R + A[i]$

Ciklus vége

Eljárás vége

R

1	2	3	5	8	13
---	---	---	---	---	----



Sorozat elemeinek összegzése

Eljárás Sorozatszámítás(A, N, R)

$R \leftarrow 0$

Ciklus $i \leftarrow 1$ -től N -ig

$R \leftarrow R + A[i]$

Ciklus vége

Eljárás vége

$R = 0$

1	2	3	5	8	13
---	---	---	---	---	----



Sorozat elemeinek összegzése

Eljárás Sorozatszámítás(A, N, R)

$R \leftarrow 0$

Ciklus $i \leftarrow 1$ -től N -ig

$R \leftarrow R + A[i]$

Ciklus vége

Eljárás vége

$R = 0$

1	2	3	5	8	13
---	---	---	---	---	----

$i = 1$



Sorozat elemeinek összegzése

Eljárás Sorozatszámítás(A , N , R)

$R \leftarrow 0$

Ciklus $i \leftarrow 1$ -től N -ig

$R \leftarrow R + A[i]$

Ciklus vége

Eljárás vége

$R = 1$

1	2	3	5	8	13
---	---	---	---	---	----

$i = 1$



Sorozat elemeinek összegzése

Eljárás Sorozatszámítás(A, N, R)

$R \leftarrow 0$

Ciklus $i \leftarrow 1$ -től N -ig

$R \leftarrow R + A[i]$

Ciklus vége

Eljárás vége

$R = 1$

1	2	3	5	8	13
---	---	---	---	---	----

\uparrow
 $i = 2$



Sorozat elemeinek összegzése

Eljárás Sorozatszámítás(A, N, R)

$R \leftarrow 0$

Ciklus $i \leftarrow 1$ -től N -ig

$R \leftarrow R + A[i]$

Ciklus vége

Eljárás vége

$R = 3$

1	2	3	5	8	13
---	---	---	---	---	----

$i = 2$



Sorozat elemeinek összegzése

Eljárás Sorozatszámítás(A , N , R)

$R \leftarrow 0$

Ciklus $i \leftarrow 1$ -től N -ig

$R \leftarrow R + A[i]$

Ciklus vége

Eljárás vége

$R = 3$

1	2	3	5	8	13
---	---	---	---	---	----

\uparrow
 $i = 3$



Sorozat elemeinek összegzése

Eljárás Sorozatszámítás(A, N, R)

$R \leftarrow 0$

Ciklus $i \leftarrow 1$ -től N -ig

$R \leftarrow R + A[i]$

Ciklus vége

Eljárás vége

$R = 6$

1	2	3	5	8	13
---	---	---	---	---	----

$i = 3$



Sorozat elemeinek összegzése

Eljárás Sorozatszámítás(A , N , R)

$R \leftarrow 0$

Ciklus $i \leftarrow 1$ -től N -ig

$R \leftarrow R + A[i]$

Ciklus vége

Eljárás vége

$R = 6$

1	2	3	5	8	13
---	---	---	---	---	----

$i = 4$



Sorozat elemeinek összegzése

Eljárás Sorozatszámítás(A , N , R)

$R \leftarrow 0$

Ciklus $i \leftarrow 1$ -től N -ig

$R \leftarrow R + A[i]$

Ciklus vége

Eljárás vége

$R = 11$

1	2	3	5	8	13
---	---	---	---	---	----

$i = 4$



Sorozat elemeinek összegzése

Eljárás Sorozatszámítás(A, N, R)

$R \leftarrow 0$

Ciklus $i \leftarrow 1$ -től N -ig

$R \leftarrow R + A[i]$

Ciklus vége

Eljárás vége

$R = 11$

1	2	3	5	8	13
---	---	---	---	---	----

$i = 5$



Sorozat elemeinek összegzése

Eljárás Sorozatszámítás(A , N , R)

$R \leftarrow 0$

Ciklus $i \leftarrow 1$ -től N -ig

$R \leftarrow R + A[i]$

Ciklus vége

Eljárás vége

$R = 19$

1	2	3	5	8	13
---	---	---	---	---	----

$i = 5$



Sorozat elemeinek összegzése

Eljárás Sorozatszámítás(A, N, R)

$R \leftarrow 0$

Ciklus $i \leftarrow 1$ -től N -ig

$R \leftarrow R + A[i]$

Ciklus vége

Eljárás vége

$R = 19$

1	2	3	5	8	13
---	---	---	---	---	----

$i = 6$



Sorozat elemeinek összegzése

Eljárás Sorozatszámítás(A, N, R)

$R \leftarrow 0$

Ciklus $i \leftarrow 1$ -től N -ig

$R \leftarrow R + A[i]$

Ciklus vége

Eljárás vége

$R = 32$

1	2	3	5	8	13
---	---	---	---	---	----

$i = 6$



Eljárás Sorozatszámítás(A , N , R)

$R \leftarrow R_0$

1 értékadás

Ciklus $i \leftarrow 1$ -től N -ig

$1 + N$ értékadás és $N + 1$ összeg.

$R \leftarrow R$ művelet $A[i]$

N értékadás

Ciklus vége

Eljárás vége

A futási idő minden esetben:

$$T(N) = 1 + 2(N + 1) + N = 3N + 3 = O(N)$$



Összegzés

Bemenet: X – számokat tartalmazó tömb

N – tömb elemszáma

Kimenet: S – tömb elemeinek összege

Eljárás $\text{Összegzés}(N, X, S)$

$S \leftarrow 0$

Ciklus $i \leftarrow 1$ -től N -ig

$S \leftarrow S + X[i]$

Ciklus vége

Eljárás vége



- 1 Tömbök a C#-ban
- 2 Metódusok C#-ban
- 3 Programozási tételek
- 4 Egyszerű programozási tételek**
 - Sorozatszámítás
 - Eldöntés**
 - Kiválasztás
 - Lineáris keresés
 - Megszámlálás
 - Maximumkiválasztás



Típusfeladatok

- 1 Döntsük el egy szóról a hónapnevek sorozata alapján, hogy egy hónap neve-e!
- 2 Döntsük el egy tanuló év végi osztályzata alapján, hogy kitűnő tanuló-e!
- 3 Júniusban minden nap délben megmértük, hogy a Balaton Siófoknál hány fokos. Döntsük el a mérések alapján, hogy a víz hőfoka folyamatosan emelkedett-e!
- 4 Döntsük el egy számról, hogy prímszám-e!



Bemenet

A : Feldolgozandó tömb
 N : Tömb elemeinek száma
 T : Tulajdonság függvény

Kimenet

VAN : Logikai változó

Pszeudokód

Eljárás Eldöntés(A , N , T , VAN)

$i \leftarrow 1$

Ciklus amíg ($i \leq N$) és $\neg T(A[i])$

$i \leftarrow i + 1$

Ciklus vége

$VAN \leftarrow (i \leq N)$

Eljárás vége

Döntsük el, hogy egy sorozatban van-e páros szám!

Eljárás Eldöntés(A , N , T , VAN)

$i \leftarrow 1$

Ciklus amíg ($i \leq N$) és $\neg T(A[i])$

$i \leftarrow i + 1$

Ciklus vége

$VAN \leftarrow (i \leq N)$

Eljárás vége

1	3	5	7	8	13
---	---	---	---	---	----



Döntsük el, hogy egy sorozatban van-e páros szám!

Eljárás Eldöntés(A , N , T , VAN)

$i \leftarrow 1$

Ciklus amíg ($i \leq N$) és $\neg T(A[i])$

$i \leftarrow i + 1$

Ciklus vége

$VAN \leftarrow (i \leq N)$

Eljárás vége

1	3	5	7	8	13
---	---	---	---	---	----

$i = 1$



Döntsük el, hogy egy sorozatban van-e páros szám!

Eljárás Eldöntés(A , N , T , VAN)

$i \leftarrow 1$

Ciklus amíg ($i \leq N$) és $\neg T(A[i])$

$i \leftarrow i + 1$

Ciklus vége

$VAN \leftarrow (i \leq N)$

Eljárás vége

1	3	5	7	8	13
---	---	---	---	---	----

$i = 1$



Döntsük el, hogy egy sorozatban van-e páros szám!

Eljárás Eldöntés(A , N , T , VAN)

$i \leftarrow 1$

Ciklus amíg ($i \leq N$) és $\neg T(A[i])$

$i \leftarrow i + 1$

Ciklus vége

$VAN \leftarrow (i \leq N)$

Eljárás vége

1	3	5	7	8	13
---	---	---	---	---	----

$i = 1$



Döntsük el, hogy egy sorozatban van-e páros szám!

Eljárás Eldöntés(A , N , T , VAN)

$i \leftarrow 1$

Ciklus amíg ($i \leq N$) és $\neg T(A[i])$

$i \leftarrow i + 1$

Ciklus vége

$VAN \leftarrow (i \leq N)$

Eljárás vége

1	3	5	7	8	13
---	---	---	---	---	----

$i = 1$



Döntsük el, hogy egy sorozatban van-e páros szám!

Eljárás Eldöntés(A , N , T , VAN)

$i \leftarrow 1$

Ciklus amíg ($i \leq N$) és $\neg T(A[i])$

$i \leftarrow i + 1$

Ciklus vége

$VAN \leftarrow (i \leq N)$

Eljárás vége

1	3	5	7	8	13
---	---	---	---	---	----

$i = 2$



Döntsük el, hogy egy sorozatban van-e páros szám!

Eljárás Eldöntés(A , N , T , VAN)

$i \leftarrow 1$

Ciklus amíg ($i \leq N$) és $\neg T(A[i])$

$i \leftarrow i + 1$

Ciklus vége

$VAN \leftarrow (i \leq N)$

Eljárás vége

1	3	5	7	8	13
---	---	---	---	---	----

$i = 2$



Döntsük el, hogy egy sorozatban van-e páros szám!

Eljárás Eldöntés(A , N , T , VAN)

$i \leftarrow 1$

Ciklus amíg ($i \leq N$) és $\neg T(A[i])$

$i \leftarrow i + 1$

Ciklus vége

$VAN \leftarrow (i \leq N)$

Eljárás vége

1	3	5	7	8	13
---	---	---	---	---	----

$i = 2$



Döntsük el, hogy egy sorozatban van-e páros szám!

Eljárás Eldöntés(A , N , T , VAN)

$i \leftarrow 1$

Ciklus amíg ($i \leq N$) és $\neg T(A[i])$

$i \leftarrow i + 1$

Ciklus vége

$VAN \leftarrow (i \leq N)$

Eljárás vége

1	3	5	7	8	13
---	---	---	---	---	----

$i = 2$



Döntsük el, hogy egy sorozatban van-e páros szám!

Eljárás Eldöntés(A , N , T , VAN)

$i \leftarrow 1$

Ciklus amíg ($i \leq N$) és $\neg T(A[i])$

$i \leftarrow i + 1$

Ciklus vége

$VAN \leftarrow (i \leq N)$

Eljárás vége

1	3	5	7	8	13
---	---	---	---	---	----

$i = 3$



Döntsük el, hogy egy sorozatban van-e páros szám!

Eljárás Eldöntés(A , N , T , VAN)

$i \leftarrow 1$

Ciklus amíg ($i \leq N$) és $\neg T(A[i])$

$i \leftarrow i + 1$

Ciklus vége

$VAN \leftarrow (i \leq N)$

Eljárás vége

1	3	5	7	8	13
---	---	---	---	---	----

$i = 3$



Döntsük el, hogy egy sorozatban van-e páros szám!

Eljárás Eldöntés(A , N , T , VAN)

$i \leftarrow 1$

Ciklus amíg ($i \leq N$) és $\neg T(A[i])$

$i \leftarrow i + 1$

Ciklus vége

$VAN \leftarrow (i \leq N)$

Eljárás vége

1	3	5	7	8	13
---	---	---	---	---	----

$i = 3$



Döntsük el, hogy egy sorozatban van-e páros szám!

Eljárás Eldöntés(A , N , T , VAN)

$i \leftarrow 1$

Ciklus amíg ($i \leq N$) és $\neg T(A[i])$

$i \leftarrow i + 1$

Ciklus vége

$VAN \leftarrow (i \leq N)$

Eljárás vége

1	3	5	7	8	13
---	---	---	---	---	----

$i = 3$



Döntsük el, hogy egy sorozatban van-e páros szám!

Eljárás Eldöntés(A , N , T , VAN)

$i \leftarrow 1$

Ciklus amíg ($i \leq N$) és $\neg T(A[i])$

$i \leftarrow i + 1$

Ciklus vége

$VAN \leftarrow (i \leq N)$

Eljárás vége

1	3	5	7	8	13
---	---	---	---	---	----

$i = 4$



Döntsük el, hogy egy sorozatban van-e páros szám!

Eljárás Eldöntés(A , N , T , VAN)

$i \leftarrow 1$

Ciklus amíg ($i \leq N$) és $\neg T(A[i])$

$i \leftarrow i + 1$

Ciklus vége

$VAN \leftarrow (i \leq N)$

Eljárás vége

1	3	5	7	8	13
---	---	---	---	---	----

$i = 4$



Döntsük el, hogy egy sorozatban van-e páros szám!

Eljárás Eldöntés(A , N , T , VAN)

$i \leftarrow 1$

Ciklus amíg ($i \leq N$) és $\neg T(A[i])$

$i \leftarrow i + 1$

Ciklus vége

$VAN \leftarrow (i \leq N)$

Eljárás vége

1	3	5	7	8	13
---	---	---	---	---	----

$i = 4$



Döntsük el, hogy egy sorozatban van-e páros szám!

Eljárás Eldöntés(A , N , T , VAN)

$i \leftarrow 1$

Ciklus amíg ($i \leq N$) és $\neg T(A[i])$

$i \leftarrow i + 1$

Ciklus vége

$VAN \leftarrow (i \leq N)$

Eljárás vége

1	3	5	7	8	13
---	---	---	---	---	----

$i = 4$



Döntsük el, hogy egy sorozatban van-e páros szám!

Eljárás Eldöntés(A , N , T , VAN)

$i \leftarrow 1$

Ciklus amíg $(i \leq N)$ és $\neg T(A[i])$

$i \leftarrow i + 1$

Ciklus vége

$VAN \leftarrow (i \leq N)$

Eljárás vége

1	3	5	7	8	13
---	---	---	---	---	----

$i = 5$



Döntsük el, hogy egy sorozatban van-e páros szám!

Eljárás Eldöntés(A , N , T , VAN)

$i \leftarrow 1$

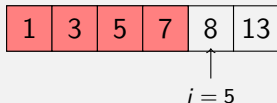
Ciklus amíg ($i \leq N$) és $\neg T(A[i])$

$i \leftarrow i + 1$

Ciklus vége

$VAN \leftarrow (i \leq N)$

Eljárás vége



Döntsük el, hogy egy sorozatban van-e páros szám!

Eljárás Eldöntés(A , N , T , VAN)

$i \leftarrow 1$

Ciklus amíg ($i \leq N$) és $\neg T(A[i])$

$i \leftarrow i + 1$

Ciklus vége

$VAN \leftarrow (i \leq N)$

Eljárás vége

1	3	5	7	8	13
---	---	---	---	---	----

$i = 5$



Döntsük el, hogy egy sorozatban van-e páros szám!

Eljárás Eldöntés(A , N , T , VAN)

$i \leftarrow 1$

Ciklus amíg ($i \leq N$) és $\neg T(A[i])$

$i \leftarrow i + 1$

Ciklus vége

$VAN \leftarrow (i \leq N)$

Eljárás vége

1	3	5	7	8	13
---	---	---	---	---	----

$i = 5$



Döntsük el, hogy egy sorozatban van-e páros szám!

Eljárás Eldöntés(A , N , T , VAN)

$i \leftarrow 1$

Ciklus amíg ($i \leq N$) és $\neg T(A[i])$

$i \leftarrow i + 1$

Ciklus vége

$VAN \leftarrow (i \leq N)$

Eljárás vége

1	3	5	7	8	13
---	---	---	---	---	----

$VAN \leftarrow \text{true}$



Megjegyzések

- A T tulajdonság helyes megválasztásával a tétel sokféle szituációban alkalmazható
- A "minden elem T tulajdonságú" feladatot egyszerűen visszavezethetjük az eldöntésre a T tulajdonság tagadásával
- A sorozatszámításnál megismert módszerrel ellentétben ez az algoritmus az első T tulajdonságú elem megtalálása után már nem folytatja a keresést



Eldöntés: minden elem T tulajdonságú-e?

Eljárás Eldöntés(A , N , T , VAN)

$i \leftarrow 1$

Ciklus amíg ($i \leq N$) és $T(A[i])$

$i \leftarrow i + 1$

Ciklus vége

$VAN \leftarrow (i > N)$

Eljárás vége



Legrosszabb eset

Legrosszabb esetnek azt tekinthetjük, amikor nincs a tömbben T tulajdonságú elem, hiszen ilyenkor az egész tömböt be kell járnunk.

Eljárás Eldöntés(A, N, T, VAN)

$i \leftarrow 1$

1 értékadás

Ciklus amíg $(i \leq N)$ és $\neg T(A[i])$

$N + 1$ összeh.

N összeh.

$i \leftarrow i + 1$

N értékadás

Ciklus vége

$VAN \leftarrow (i \leq N)$

1 értékadás és 1 összeh.

Eljárás vége

$$T(N) = 1 + N + 1 + N + N + 2 = 3N + 4 = O(N)$$



Legjobb eset

Ez az eset akkor áll elő, ha rögtön az első elem T tulajdonságú

$$T(N) = 1 + 1 + 1 + 0 + 2 = 5 = O(1)$$

Átlagos eset

$$T(N) = 1 + \frac{N}{2} + \frac{N}{2} + \frac{N}{2} + 2 = \frac{3N}{2} + 3 = O\left(\frac{N}{2}\right) = O(N)$$



Prímteszt

Bemenet: N – pozitív egész szám (legalább 2)

Kimenet: $PRIM$ – logikai változó; pontosan akkor igaz, ha N prímszám

Eljárás Eldöntés(N , $PRIM$)

$i \leftarrow 2$

Ciklus amíg ($i \leq N - 1$) és $\neg(i$ osztója N -nek)

$i \leftarrow i + 1$

Ciklus vége

$PRIM \leftarrow (i > N - 1)$

Eljárás vége



Monoton növekedés

Bemenet: X – feldolgozandó tömb

N – X elemszáma

Kimenet: *MONOTON* – logikai változó; pontosan akkor igaz,
ha X elemei monoton növekedők

Eljárás Eldöntés($N, X, MONOTON$)

$i \leftarrow 1$

Ciklus amíg ($i \leq N - 1$) és ($X[i] \leq X[i + 1]$)

$i \leftarrow i + 1$

Ciklus vége

$MONOTON \leftarrow (i > N - 1)$

Eljárás vége

- 1 Tömbök a C#-ban
- 2 Metódusok C#-ban
- 3 Programozási tételek
- 4 Egyszerű programozási tételek**
 - Sorozatszámítás
 - Eldöntés
 - Kiválasztás**
 - Lineáris keresés
 - Megszámlálás
 - Maximumkiválasztás



Típusfeladatok

- 1 Ismerjük egy hónap nevét. A hónapnevek sorozata alapján mondjuk meg a hónap sorszámát!
- 2 Adjuk meg egy egynél nagyobb természetes szám legkisebb, 1-től különböző osztóját!
- 3 A naptárban található névnapok alapján adjuk meg a legjobb barátunk névnapját!



Bemenet

A: Feldolgozandó tömb
N: Tömb elemeinek száma
T: Tulajdonság függvény

Kimenet

SORSZ: Első *T* tulajdonságú
elem indexe

Pszudokód

Eljárás Kiválasztás(*A*, *N*, *T*, *SORSZ*)

$i \leftarrow 1$

Ciklus amíg $\neg T(A[i])$

$i \leftarrow i + 1$

Ciklus vége

$SORSZ \leftarrow i$

Eljárás vége

Megjegyzések

- Az eldöntéssel ellentétben ez visszaadja az első T tulajdonságú elem sorszámát
- A tétel feltételezi, hogy **biztosan van legalább egy T tulajdonságú elem**
- Sorszám helyett visszaadhatjuk az elem értékét is, de célszerűbb a sorszám használata (ez alapján az elem is egyszerűen meghatározható)
- Az algoritmus működése és futási ideje hasonló az eldöntéshez



Legjobb barát névnapja

Bemenet: X – keresztneveket és a hozzájuk tartozó névnapokat tartalmazó tömb

N – X elemeinek száma

$BARAT$ – legjobb barátunk keresztneve

Kimenet: NAP – legjobb barátunk névnapja

Eljárás Kiválasztás($N, X, BARAT, NAP$)

$i \leftarrow 1$

Ciklus amíg ($X[i].nev \neq BARAT$)

$i \leftarrow i + 1$

Ciklus vége

$NAP \leftarrow X[i].nevnap$

Eljárás vége

- 1 Tömbök a C#-ban
- 2 Metódusok C#-ban
- 3 Programozási tételek
- 4 Egyszerű programozási tételek**
 - Sorozatszámítás
 - Eldöntés
 - Kiválasztás
 - Lineáris keresés**
 - Megszámlálás
 - Maximumkiválasztás



Típusfeladatok

- 1 Ismerjük egy üzlet egy havi forgalmát: minden napra megadjuk, hogy mennyi volt a bevétel és mennyi a kiadás. Adjunk meg egy olyan napot – ha van –, amelyik nem volt nyereséges!
- 2 A Budapest-Nagykanizsa vasúti menetrend alapján két adott állomáshoz adjunk meg egy olyan vonatot, amellyel el lehet jutni átszállás nélkül az egyikről a másikra!
- 3 Egy tetszőleges (nem 1) természetes számnak adjuk meg egy osztóját, ami nem az 1 és nem is önmaga!



Lineáris keresés

Bemenet

A : Feldolgozandó tömb
 N : Tömb elemeinek száma
 T : Tulajdonság függvény

Kimenet

VAN : Logikai változó
 $SORSZ$: Első T tulajdonságú elem indexe

Pszeudokód

Eljárás Keresés(A , N , T , VAN , $SORSZ$)

$i \leftarrow 1$

Ciklus amíg $(i \leq N)$ és $\neg T(A[i])$

$i \leftarrow i + 1$

Ciklus vége

$VAN \leftarrow (i \leq N)$

Ha VAN **akkor**

$SORSZ \leftarrow i$

Elágazás vége

Eljárás vége

Megjegyzések

- Tekinthető az eldöntés és a kiválasztás tétel ötvözetének is: választ ad arra, hogy van-e T tulajdonságú elem a sorozatban, és ha van, akkor visszaadja a sorszámát is.
- Értelemszerűen nem feltételezi, hogy biztosan van T tulajdonságú elem a sorozatban. Ha nincs, akkor a *VAN* értéke hamis, ilyenkor a *SORSZ* változó nem kap értéket.
- Az algoritmus működése és futási ideje hasonló az eldöntéshez



Nem nyereséges nap

Bemenet: K – kiadások tömbje

B – bevételek tömbje

N – K és B elemeinek száma

Kimenet: VAN – logikai változó; pontosan akkor igaz,
ha van nem nyereséges nap

$SORSZ$ – a nem nyereséges nap (ha van ilyen!) sorszáma

Eljárás Keresés(K , B , N , VAN , $SORSZ$)

$i \leftarrow 1$

Ciklus amíg ($i \leq N$) és ($B[i] - K[i] > 0$)

$i \leftarrow i + 1$

Ciklus vége

$VAN \leftarrow (i \leq N)$

Ha VAN **akkor**

$SORSZ \leftarrow i$

Elágazás vége

Eljárás vége

- 1 Tömbök a C#-ban
- 2 Metódusok C#-ban
- 3 Programozási tételek
- 4 Egyszerű programozási tételek**
 - Sorozatszámítás
 - Eldöntés
 - Kiválasztás
 - Lineáris keresés
 - Megszámlálás**
 - Maximumkiválasztás



Típusfeladatok

- 1 Családok létszáma, illetve jövedelme alapján állapítsuk meg, hogy hány család él a létminimum alatt!
- 2 Egy futóverseny időeredményei alapján határozzuk meg, hogy a versenyzők hány százaléka teljesítette az olimpiai induláshoz szükséges szintet!
- 3 Adjuk meg egy szöveg magánhangzóinak számát!



Megszámlálás

Bemenet

A : Feldolgozandó tömb
 N : Tömb elemeinek száma
 T : Tulajdonság függvény

Kimenet

DB : T tulajdonságú elemek száma

Pszeudokód

Eljárás Megszámlálás(A , N , T , DB)

$DB \leftarrow 0$

Ciklus $i \leftarrow 1$ -től N -ig

Ha $T(A[i])$ akkor

$DB \leftarrow DB + 1$

Elágazás vége

Ciklus vége

Eljárás vége

Páros számok megszámolása

Eljárás Megszámlálás(A, N, T, DB)

$DB \leftarrow 0$

Ciklus $i \leftarrow 1$ -től N -ig

Ha $T(A[i])$ akkor

$DB \leftarrow DB + 1$

Elágazás vége

Ciklus vége

Eljárás vége

2	7	4	10	3	6
---	---	---	----	---	---



Páros számok megszámolása

Eljárás Megszámlálás(A, N, T, DB)

$DB \leftarrow 0$

Ciklus $i \leftarrow 1$ -től N -ig

Ha $T(A[i])$ akkor

$DB \leftarrow DB + 1$

Elágazás vége

Ciklus vége

Eljárás vége

2	7	4	10	3	6
---	---	---	----	---	---

$DB \leftarrow 0$



Páros számok megszámlálása

Eljárás Megszámlálás(A, N, T, DB)

$DB \leftarrow 0$

Ciklus $i \leftarrow 1$ -től N -ig

Ha $T(A[i])$ akkor

$DB \leftarrow DB + 1$

Elágazás vége

Ciklus vége

Eljárás vége

2	7	4	10	3	6
---	---	---	----	---	---

$DB \leftarrow 0$

$i = 1$



Páros számok megszámlálása

Eljárás Megszámlálás(A, N, T, DB)

$DB \leftarrow 0$

Ciklus $i \leftarrow 1$ -től N -ig

Ha $T(A[i])$ akkor

$DB \leftarrow DB + 1$

Elágazás vége

Ciklus vége

Eljárás vége

2	7	4	10	3	6
---	---	---	----	---	---

$DB \leftarrow 0$

$i = 1$



Páros számok megszámlálása

Eljárás Megszámlálás(A , N , T , DB)

$DB \leftarrow 0$

Ciklus $i \leftarrow 1$ -től N -ig

Ha $T(A[i])$ akkor

$DB \leftarrow DB + 1$

Elágazás vége

Ciklus vége

Eljárás vége

2	7	4	10	3	6
---	---	---	----	---	---

$DB \leftarrow 0$

$i = 1$



Páros számok megszámlálása

Eljárás Megszámlálás(A, N, T, DB)

$DB \leftarrow 0$

Ciklus $i \leftarrow 1$ -től N -ig

Ha $T(A[i])$ akkor

$DB \leftarrow DB + 1$

Elágazás vége

Ciklus vége

Eljárás vége

2	7	4	10	3	6
---	---	---	----	---	---

$DB \leftarrow 1$

$i = 1$



Páros számok megszámolása

Eljárás Megszámlálás(A , N , T , DB)

$DB \leftarrow 0$

Ciklus $i \leftarrow 1$ -től N -ig

Ha $T(A[i])$ akkor

$DB \leftarrow DB + 1$

Elágazás vége

Ciklus vége

Eljárás vége

2	7	4	10	3	6
---	---	---	----	---	---

$i = 2$

$DB \leftarrow 1$



Páros számok megszámolása

Eljárás Megszámlálás(A , N , T , DB)

$DB \leftarrow 0$

Ciklus $i \leftarrow 1$ -től N -ig

Ha $T(A[i])$ akkor

$DB \leftarrow DB + 1$

Elágazás vége

Ciklus vége

Eljárás vége

2	7	4	10	3	6
---	---	---	----	---	---

$i = 2$

$DB \leftarrow 1$



Páros számok megszámolása

Eljárás Megszámlálás(A, N, T, DB)

$DB \leftarrow 0$

Ciklus $i \leftarrow 1$ -től N -ig

Ha $T(A[i])$ akkor

$DB \leftarrow DB + 1$

Elágazás vége

Ciklus vége

Eljárás vége

2	7	4	10	3	6
---	---	---	----	---	---

$i = 2$

$DB \leftarrow 1$



Páros számok megszámolása

Eljárás Megszámlálás(A, N, T, DB)

$DB \leftarrow 0$

Ciklus $i \leftarrow 1$ -től N -ig

Ha $T(A[i])$ akkor

$DB \leftarrow DB + 1$

Elágazás vége

Ciklus vége

Eljárás vége

2	7	4	10	3	6
---	---	---	----	---	---

$i = 3$

$DB \leftarrow 1$



Páros számok megszámolása

Eljárás Megszámlálás(A, N, T, DB)

$DB \leftarrow 0$

Ciklus $i \leftarrow 1$ -től N -ig

Ha $T(A[i])$ akkor

$DB \leftarrow DB + 1$

Elágazás vége

Ciklus vége

Eljárás vége

2	7	4	10	3	6
---	---	---	----	---	---

$i = 3$

$DB \leftarrow 1$



Páros számok megszámolása

Eljárás Megszámlálás(A, N, T, DB)

$DB \leftarrow 0$

Ciklus $i \leftarrow 1$ -től N -ig

Ha $T(A[i])$ akkor

$DB \leftarrow DB + 1$

Elágazás vége

Ciklus vége

Eljárás vége

2	7	4	10	3	6
---	---	---	----	---	---

$i = 3$

$DB \leftarrow 1$



Páros számok megszámolása

Eljárás Megszámlálás(A, N, T, DB)

$DB \leftarrow 0$

Ciklus $i \leftarrow 1$ -től N -ig

Ha $T(A[i])$ akkor

$DB \leftarrow DB + 1$

Elágazás vége

Ciklus vége

Eljárás vége

2	7	4	10	3	6
---	---	---	----	---	---

$i = 3$

$DB \leftarrow 2$



Páros számok megszámolása

Eljárás Megszámlálás(A, N, T, DB)

$DB \leftarrow 0$

Ciklus $i \leftarrow 1$ -től N -ig

Ha $T(A[i])$ akkor

$DB \leftarrow DB + 1$

Elágazás vége

Ciklus vége

Eljárás vége

2	7	4	10	3	6
---	---	---	----	---	---

$i = 4$

$DB \leftarrow 2$



Páros számok megszámolása

Eljárás Megszámlálás(A, N, T, DB)

$DB \leftarrow 0$

Ciklus $i \leftarrow 1$ -től N -ig

Ha $T(A[i])$ akkor

$DB \leftarrow DB + 1$

Elágazás vége

Ciklus vége

Eljárás vége

2	7	4	10	3	6
---	---	---	----	---	---

$i = 4$

$DB \leftarrow 2$



Páros számok megszámolása

Eljárás Megszámlálás(A, N, T, DB)

$DB \leftarrow 0$

Ciklus $i \leftarrow 1$ -től N -ig

Ha $T(A[i])$ akkor

$DB \leftarrow DB + 1$

Elágazás vége

Ciklus vége

Eljárás vége

2	7	4	10	3	6
---	---	---	----	---	---

$i = 4$

$DB \leftarrow 2$



Páros számok megszámolása

Eljárás Megszámlálás(A, N, T, DB)

$DB \leftarrow 0$

Ciklus $i \leftarrow 1$ -től N -ig

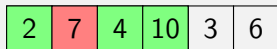
Ha $T(A[i])$ akkor

$DB \leftarrow DB + 1$

Elágazás vége

Ciklus vége

Eljárás vége



$i = 4$

$DB \leftarrow 3$



Páros számok megszámolása

Eljárás Megszámlálás(A, N, T, DB)

$DB \leftarrow 0$

Ciklus $i \leftarrow 1$ -től N -ig

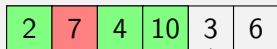
Ha $T(A[i])$ akkor

$DB \leftarrow DB + 1$

Elágazás vége

Ciklus vége

Eljárás vége



$i = 5$

$DB \leftarrow 3$



Páros számok megszámolása

Eljárás Megszámlálás(A, N, T, DB)

$DB \leftarrow 0$

Ciklus $i \leftarrow 1$ -től N -ig

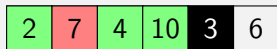
Ha $T(A[i])$ akkor

$DB \leftarrow DB + 1$

Elágazás vége

Ciklus vége

Eljárás vége



$DB \leftarrow 3$



Páros számok megszámolása

Eljárás Megszámlálás(A, N, T, DB)

$DB \leftarrow 0$

Ciklus $i \leftarrow 1$ -től N -ig

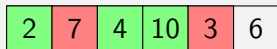
Ha $T(A[i])$ akkor

$DB \leftarrow DB + 1$

Elágazás vége

Ciklus vége

Eljárás vége



$i = 5$

$DB \leftarrow 3$



Páros számok megszámolása

Eljárás Megszámlálás(A, N, T, DB)

$DB \leftarrow 0$

Ciklus $i \leftarrow 1$ -től N -ig

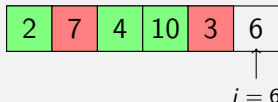
Ha $T(A[i])$ akkor

$DB \leftarrow DB + 1$

Elágazás vége

Ciklus vége

Eljárás vége



$DB \leftarrow 3$



Páros számok megszámolása

Eljárás Megszámlálás(A, N, T, DB)

$DB \leftarrow 0$

Ciklus $i \leftarrow 1$ -től N -ig

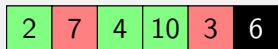
Ha $T(A[i])$ akkor

$DB \leftarrow DB + 1$

Elágazás vége

Ciklus vége

Eljárás vége



$i = 6$

$DB \leftarrow 3$



Páros számok megszámolása

Eljárás Megszámlálás(A, N, T, DB)

$DB \leftarrow 0$

Ciklus $i \leftarrow 1$ -től N -ig

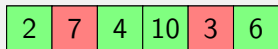
Ha $T(A[i])$ akkor

$DB \leftarrow DB + 1$

Elágazás vége

Ciklus vége

Eljárás vége



$i = 6$

$DB \leftarrow 3$



Páros számok megszámolása

Eljárás Megszámlálás(A, N, T, DB)

$DB \leftarrow 0$

Ciklus $i \leftarrow 1$ -től N -ig

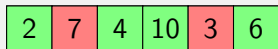
Ha $T(A[i])$ akkor

$DB \leftarrow DB + 1$

Elágazás vége

Ciklus vége

Eljárás vége



$DB \leftarrow 4$

$i = 6$



Megjegyzések

- Amennyiben nincs T tulajdonságú elem a sorozatban, akkor értelemszerűen 0 kerül a DB változóba
- Valójában egy sorozatszámítás, amely minden T tulajdonságú elem esetén 1-et hozzáad a DB értékéhez



Legrosszab eset

Futási idő szempontjából a legrosszabb esetnek azt tekinthetjük, ha minden elem T tulajdonságú, mert ilyenkor minden esetben DB -t is növelnünk kell

Eljárás Megszámolás(A, N, T, DB)

$DB \leftarrow 0$

1 értékadás

Ciklus $i \leftarrow 1$ -től N -ig

$1 + N$ értékadás és $N + 1$ összeh.

Ha $T(A[i])$ **akkor**

N összeh.

$DB \leftarrow DB + 1$

N értékadás

Elágazás vége

Ciklus vége

Eljárás vége

$$T(N) = 1 + 2(N + 1) + N + N = 4N + 3 = O(N)$$

Legjobb eset

Legjobb esetnek azt tekinthetjük, amikor egyetlen T tulajdonságú elem sincs a tömbben

$$T(N) = 1 + 2(N + 1) + N + 0 = 3N + 3 = O(N)$$

Átlagos eset

$$T(N) = 1 + 2(N + 1) + N + \frac{N}{2} = \frac{7N}{2} + 3 = O(N)$$



Olimpiai indulási szintet teljesítő futók százalékos aránya

Bemenet: ID – Futók időeredményeit tartalmazó tömb

N – ID elemeinek száma

$SZINT$ – Olimpiai induláshoz szükséges felső időkorlát

Kimenet: $SZAZ$ – ID elemei közül a $SZINT$ értéket meg nem haladók százalékos aránya

Eljárás Megszámlálás(N , ID , $SZINT$, $SZAZ$)

$DB \leftarrow 0$

Ciklus $i \leftarrow 1$ -től N -ig

Ha $(ID[i] \leq SZINT)$ **akkor**

$DB \leftarrow DB + 1$

Elágazás vége

Ciklus vége

$SZAZ \leftarrow Kerekít(100 * DB / N)$

Eljárás vége

- 1 Tömbök a C#-ban
- 2 Metódusok C#-ban
- 3 Programozási tételek
- 4 Egyszerű programozási tételek**
 - Sorozatszámítás
 - Eldöntés
 - Kiválasztás
 - Lineáris keresés
 - Megszámlálás
 - **Maximumkiválasztás**



Típusfeladatok

- 1 Egy kórházban megmérték minden beteg lázát. Adjuk meg, hogy ki a leglázasabb!
- 2 Egy család havi bevételei és kiadásai alapján adjuk meg, hogy melyik hónapban tudtak a legtöbbet megtakarítani!
- 3 Egy osztály tanulóinak nevei alapján adjuk meg a névsorban legelső tanulót!



Maximumkiválasztás

Bemenet

A: Feldolgozandó tömb
N: Tömb elemeinek száma

Kimenet

MAX: Maximális elem indexe

Pszeudokód

Eljárás Maximumiválasztás(A , N , MAX)

$MAX \leftarrow 1$

Ciklus $i \leftarrow 2$ -től N -ig

Ha $A[i] > A[MAX]$ **akkor**

$MAX \leftarrow i$

Elágazás vége

Ciklus vége

Eljárás vége



Eljárás Maximumiválasztás(A , N , MAX)

$MAX \leftarrow 1$

Ciklus $i \leftarrow 2$ -től N -ig

Ha $A[i] > A[MAX]$ **akkor**

$MAX \leftarrow i$

Elágazás vége

Ciklus vége

Eljárás vége

2	7	4	10	13	6
---	---	---	----	----	---



Maximumkiválasztás

Eljárás Maximumiválasztás(A , N , MAX)

$MAX \leftarrow 1$

Ciklus $i \leftarrow 2$ -től N -ig

Ha $A[i] > A[MAX]$ **akkor**

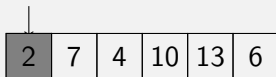
$MAX \leftarrow i$

Elágazás vége

Ciklus vége

Eljárás vége

$MAX = 1$



2	7	4	10	13	6
---	---	---	----	----	---



Maximumkiválasztás

Eljárás Maximumkiválasztás(A , N , MAX)

$MAX \leftarrow 1$

Ciklus $i \leftarrow 2$ -től N -ig

Ha $A[i] > A[MAX]$ **akkor**

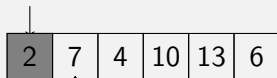
$MAX \leftarrow i$

Elágazás vége

Ciklus vége

Eljárás vége

$MAX = 1$



$i = 2$



Maximumkiválasztás

Eljárás Maximumiválasztás(A , N , MAX)

$MAX \leftarrow 1$

Ciklus $i \leftarrow 2$ -től N -ig

Ha $A[i] > A[MAX]$ **akkor**

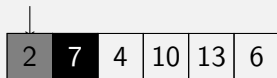
$MAX \leftarrow i$

Elágazás vége

Ciklus vége

Eljárás vége

$MAX = 1$



$i = 2$



Maximumkiválasztás

Eljárás Maximumiválasztás(A , N , MAX)

$MAX \leftarrow 1$

Ciklus $i \leftarrow 2$ -től N -ig

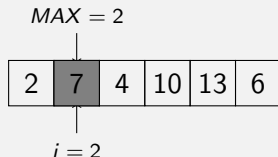
Ha $A[i] > A[MAX]$ **akkor**

$MAX \leftarrow i$

Elágazás vége

Ciklus vége

Eljárás vége



Maximumkiválasztás

Eljárás Maximumiválasztás(A , N , MAX)

$MAX \leftarrow 1$

Ciklus $i \leftarrow 2$ -től N -ig

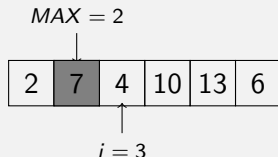
Ha $A[i] > A[MAX]$ **akkor**

$MAX \leftarrow i$

Elágazás vége

Ciklus vége

Eljárás vége



Maximumkiválasztás

Eljárás Maximumiválasztás(A , N , MAX)

$MAX \leftarrow 1$

Ciklus $i \leftarrow 2$ -től N -ig

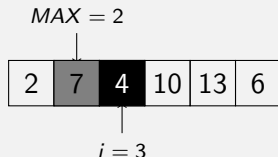
Ha $A[i] > A[MAX]$ **akkor**

$MAX \leftarrow i$

Elágazás vége

Ciklus vége

Eljárás vége



Maximumkiválasztás

Eljárás Maximumiválasztás(A , N , MAX)

$MAX \leftarrow 1$

Ciklus $i \leftarrow 2$ -től N -ig

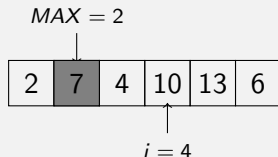
Ha $A[i] > A[MAX]$ **akkor**

$MAX \leftarrow i$

Elágazás vége

Ciklus vége

Eljárás vége



Maximumkiválasztás

Eljárás Maximumkiválasztás(A , N , MAX)

$MAX \leftarrow 1$

Ciklus $i \leftarrow 2$ -től N -ig

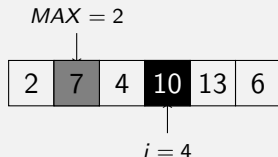
Ha $A[i] > A[MAX]$ **akkor**

$MAX \leftarrow i$

Elágazás vége

Ciklus vége

Eljárás vége



Maximumkiválasztás

Eljárás Maximumkiválasztás(A , N , MAX)

$MAX \leftarrow 1$

Ciklus $i \leftarrow 2$ -től N -ig

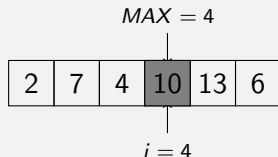
Ha $A[i] > A[MAX]$ **akkor**

$MAX \leftarrow i$

Elágazás vége

Ciklus vége

Eljárás vége



Maximumkiválasztás

Eljárás Maximumkiválasztás(A , N , MAX)

$MAX \leftarrow 1$

Ciklus $i \leftarrow 2$ -től N -ig

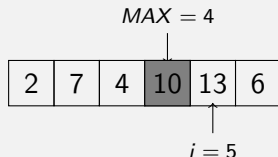
Ha $A[i] > A[MAX]$ **akkor**

$MAX \leftarrow i$

Elágazás vége

Ciklus vége

Eljárás vége



Maximumkiválasztás

Eljárás Maximumiválasztás(A , N , MAX)

$MAX \leftarrow 1$

Ciklus $i \leftarrow 2$ -től N -ig

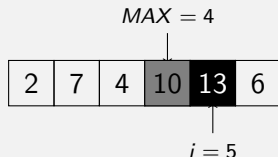
Ha $A[i] > A[MAX]$ **akkor**

$MAX \leftarrow i$

Elágazás vége

Ciklus vége

Eljárás vége



Maximumkiválasztás

Eljárás Maximumiválasztás(A , N , MAX)

$MAX \leftarrow 1$

Ciklus $i \leftarrow 2$ -től N -ig

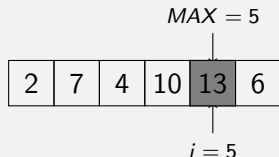
Ha $A[i] > A[MAX]$ **akkor**

$MAX \leftarrow i$

Elágazás vége

Ciklus vége

Eljárás vége



Maximumkiválasztás

Eljárás Maximumkiválasztás(A , N , MAX)

$MAX \leftarrow 1$

Ciklus $i \leftarrow 2$ -től N -ig

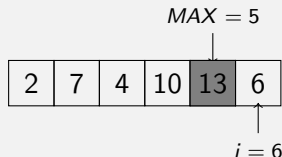
Ha $A[i] > A[MAX]$ **akkor**

$MAX \leftarrow i$

Elágazás vége

Ciklus vége

Eljárás vége



Maximumkiválasztás

Eljárás Maximumiválasztás(A , N , MAX)

$MAX \leftarrow 1$

Ciklus $i \leftarrow 2$ -től N -ig

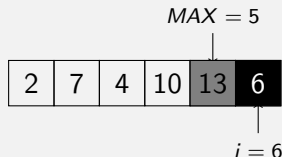
Ha $A[i] > A[MAX]$ **akkor**

$MAX \leftarrow i$

Elágazás vége

Ciklus vége

Eljárás vége



Megjegyzések

- Reláció megfordításával értelemszerűen minimumkiválasztás lesz a tétel célja
- Sorszám helyett visszaadhatjuk az elem értékét is, de célszerűbb a sorszám használata (ez alapján az elem is azonnal meghatározható)
- Feltételezzük, hogy legalább egy elem létezik a listában
- Több maximális elem esetén az elsőt adja vissza



Legrosszabb eset

Futási idő szempontjából a legrosszabb eset, ha növekvő módon rendezett a sorozat, mert így minden esetben módosítani kell MAX értékét.

Eljárás Maximumiválasztás(A, N, MAX)

$MAX \leftarrow 1$

1 értékadás

Ciklus $i \leftarrow 2\text{-től } N\text{-ig}$

N értékadás és N összeg.

Ha $A[i] > A[MAX]$ **akkor**

$N - 1$ összeg.

$MAX \leftarrow i$

$N - 1$ értékadás

Elágazás vége

Ciklus vége

Eljárás vége

$$T(N) = 1 + 2N + N - 1 + N - 1 = 4N - 1 = O(N)$$

Legjobb eset

Ha az első elem a legnagyobb, akkor *MAX* értékét soha nem kell módosítani

$$T(N) = 1 + 2N + N - 1 + 0 = 3N = O(N)$$

Átlagos eset

$$T(N) = 1 + 2N + N - 1 + \frac{N}{2} = \frac{7N}{2} = O(N)$$



Névsorban legelső tanuló

Bemenet: X – feldolgozandó tömb

N – X elemeinek száma

Kimenet: MIN – X (első) legkisebb elemének indexe

NEV – X – legkisebb elemének értéke

Eljárás Minimumkiválasztás(X , N , MIN , NEV)

$MIN \leftarrow 1$

Ciklus $i \leftarrow 2$ -től N -ig

Ha $X[MIN] > X[i]$ **akkor**

$MIN \leftarrow i$

Elágazás vége

Ciklus vége

$NEV \leftarrow X[MIN]$

Eljárás vége

Kirándulás során 100 méterenként feljegyeztük a tengerszint feletti magasság értékeket.

- 1 Volt a kirándulás során olyan szakasz, amikor sík terepen haladtunk?
- 2 Hány olyan száz méteres szakasz volt, amikor emelkedően haladtunk?
- 3 Mekkora volt a kirándulás során a teljes szintemelkedés? (És a szintcsökkenés?)
- 4 Hányszor 100 méteres volt az a leghosszabb szakasz, ahol folyamatosan emelkedően haladtunk?
- 5 Mekkora volt a 100 méter alatt megtett legnagyobb szintemelkedés?



- Korábbi évek OOP diásorai
- Szlávi Péter, Zsakó László: Módszeres programozás: Programozási tételek (Mikrológia 19). ELTE TTK, 2002
- Andrew Troelsen: A C# 2008 és a .NET 3.5. Szak Kiadó, 2009

