

# **„Programozás” komplex beadandó feladat**

*Készítette: Soós Csaba  
Neptun-azonosító: AZXX1Z  
E-mail: azxx1z@inf.elte.hu*

*Kurzuskód: IT-18PROGEG  
Gyakorlatvezető neve: Pluhár Zsuzsa*

***2025. január 12.***

## Tartalom

Felhasználói dokumentáció.....	3
Feladat.....	3
Futási környezet .....	3
Használat.....	3
A program indítása .....	3
A program használata billentyűzetről való bevitel esetén.....	3
A program használata fájlból való bevitel esetén.....	3
A program kimenete .....	4
Minta bemenet és kimenet .....	4
Hibalehetőségek .....	4
Fejlesztői dokumentáció .....	6
Feladat.....	6
Tervezés .....	6
Specifikáció.....	6
Visszavezetés .....	6
Algoritmus .....	6
Fejlesztői környezet .....	7
Forráskód .....	7
Megoldás.....	7
Függvénystruktúra .....	7
A kód .....	8
A kód (magas szintű függvényekkel).....	12
Tesztelés .....	16
Automatikus tesztek (Bíró) .....	16
Automatikus tesztek (Bíró, magas szintű függvényekkel).....	17
Érvényes tesztesetek .....	17
Érvénytelen tesztesetek.....	18
Fejlesztési lehetőségek.....	19

# Felhasználói dokumentáció

## Feladat

### Nagy változású települések

A meteorológiai intézet az ország  $N$  településére adott  $M$  napos időjárás előrejelzést, az adott településen az adott napra várt legmagasabb hőmérsékletet.

Készíts programot, amely megadja azokat a településeket, ahol a hőmérséklet egyik napról a következőre legalább 10 fokot változik!

## Futási környezet

IBM PC, exe futtatására alkalmas, 64-bites operációs rendszer (pl. Windows 11). Nem igényel egeret.

## Használat

### A program indítása

A program az

nagy\_valtozasu\_telepulesek\bin\Debug\net8.0\nagy\_valtozasu\_telepulesek.exe néven található a tömörített állományban.

### A program használata billentyűzetről való bevétel esetén

Az nagy\_valtozasu\_telepulesek.exe fájl elindításával a program az adatokat a **billentyűzetről** olvassa be a következő sorrendben:

#	Adat	Magyarázat
1.	Települések száma ( $N$ )	$1 \leq N \leq 1000$
2.	Napok száma ( $M$ )	$1 \leq M \leq 1000$
3.	1. településen az 1. napon jósolt hőmérséklet	$-50 \leq H_{i,j} \leq 50$ innentől
4.	1. településen az 2. napon jósolt hőmérséklet	
...	...	
	1. településen az $m$ . napon jósolt hőmérséklet	
	2. településen az 1. napon jósolt hőmérséklet	
	...	
	$n$ . településen az $m$ . napon jósolt hőmérséklet	

### A program használata fájlból való bevétel esetén

Lehetőségünk van az adatokat **fájlban** is megadni. Ekkor a programot *parancssorban* a következőképpen kell indítani, feltételezve, hogy a bemeneti fájlok mellette helyezkednek el:

```
nagy_valtozasu_telepulesek.exe < bel.txt
```

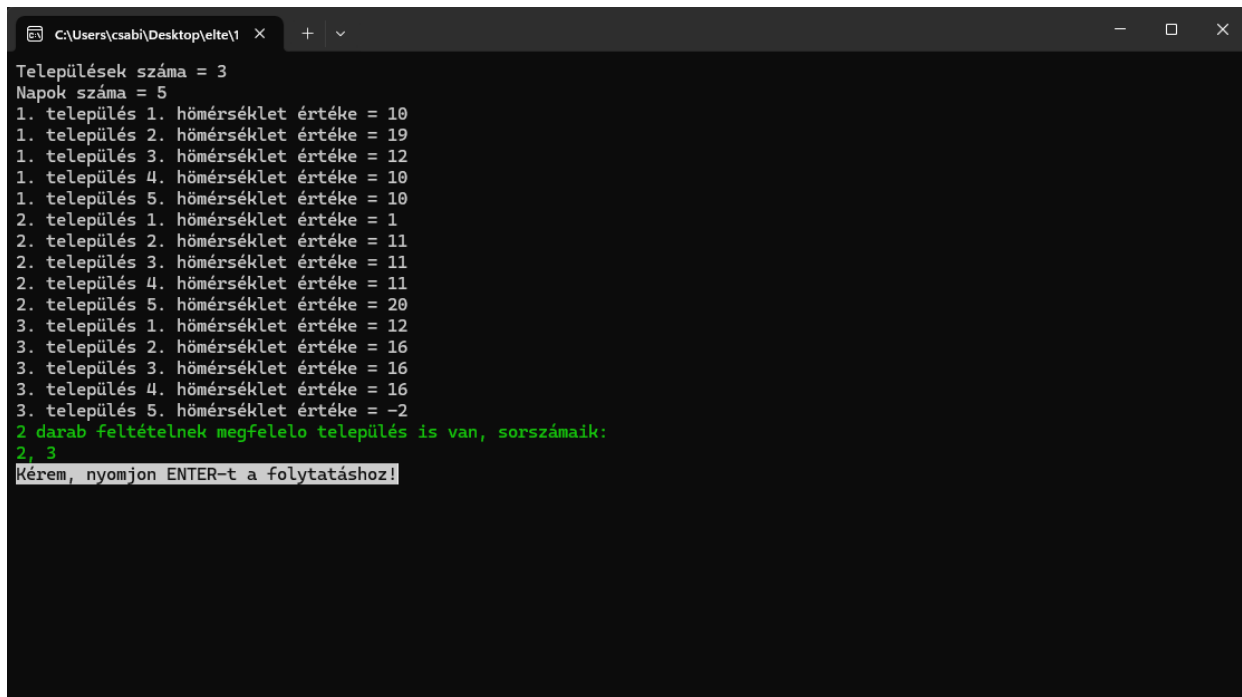
A fájl felépítésének a következő formai követelményei vannak. A standard bemenet első sorában a települések száma ( $N$ ) és a napok száma ( $M$ ) van. Az ezt követő  $N$  sorban az egyes napokra jósolt  $M$  hőmérséklet értéke található. Például:

```
3 3
10 19 12 10 10
1 11 11 11 20
12 16 16 16 -2
```

### *A program kimenete*

A program kiírja azon települések T számát kell kiírni, ahol a hőmérséklet egyik napról a következőre legalább 10 fokot változik. Ezt követi ezen települések sorszáma.

### *Minta bemenet és kimenet*

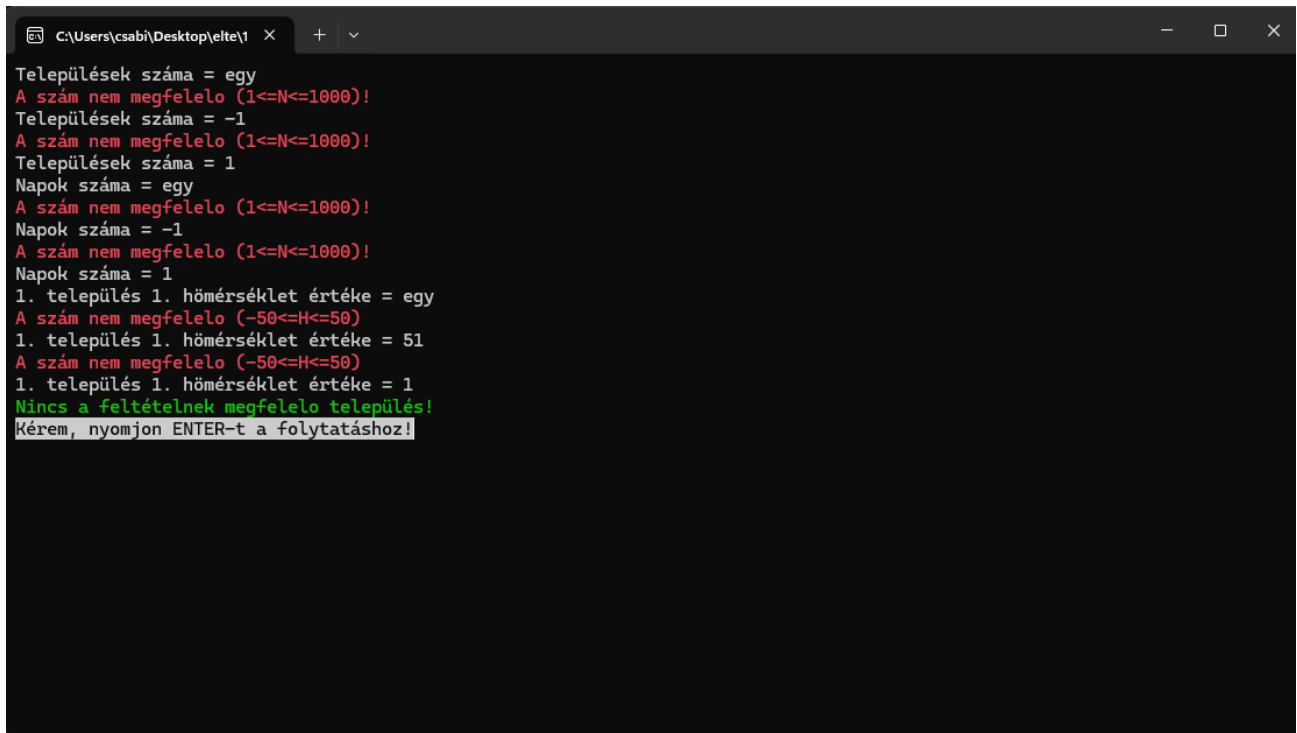


```
C:\Users\csabi\Desktop\elte\1 >
Települések száma = 3
Napok száma = 5
1. település 1. hőmérséklet értéke = 10
1. település 2. hőmérséklet értéke = 19
1. település 3. hőmérséklet értéke = 12
1. település 4. hőmérséklet értéke = 10
1. település 5. hőmérséklet értéke = 10
2. település 1. hőmérséklet értéke = 1
2. település 2. hőmérséklet értéke = 11
2. település 3. hőmérséklet értéke = 11
2. település 4. hőmérséklet értéke = 11
2. település 5. hőmérséklet értéke = 20
3. település 1. hőmérséklet értéke = 12
3. település 2. hőmérséklet értéke = 16
3. település 3. hőmérséklet értéke = 16
3. település 4. hőmérséklet értéke = 16
3. település 5. hőmérséklet értéke = -2
2 darab feltételnek megfelelő település is van, sorszámaik:
2, 3
Kérem, nyomjon ENTER-t a folytatáshoz!
```

### *Hibalehetőségek*

Az egyes bemeneti adatokat a fenti mintának megfelelően kell megadni. Hiba, ha bármelyik megadandó adat nem természetes szám. Hiba esetén a program azzal jelzi a hibát, hogy újra kérdezi azt.

### *Minta futás hibás bemeneti adatok esetén:*



```
C:\Users\csabi\Desktop\elte\1 >
Települések száma = egy
A szám nem megfelelő (1<=N<=1000)!
Települések száma = -1
A szám nem megfelelő (1<=N<=1000)!
Települések száma = 1
Napok száma = egy
A szám nem megfelelő (1<=N<=1000)!
Napok száma = -1
A szám nem megfelelő (1<=N<=1000)!
Napok száma = 1
1. település 1. hőmérséklet értéke = egy
A szám nem megfelelő (-50<=H<=50)
1. település 1. hőmérséklet értéke = 51
A szám nem megfelelő (-50<=H<=50)
1. település 1. hőmérséklet értéke = 1
Nincs a feltételnek megfelelő település!
Kérem, nyomjon ENTER-t a folytatáshoz!
```

# Fejlesztői dokumentáció

## Feladat

### Nagy változású települések

A meteorológiai intézet az ország  $N$  településére adott  $M$  napos időjárás előrejelzést, az adott településen az adott napra várt legmagasabb hőmérsékletet.

Készíts programot, amely megadja azokat a településeket, ahol a hőmérséklet egyik napról a következőre legalább 10 fokot változik!

## Tervezés

### Specifikáció

Be:  $n \in \mathbb{N}$ ,  $m \in \mathbb{N}$ ,  $\text{homerseklet} \in \mathbb{Z}[1..n, 1..m]$

Ki:  $db \in \mathbb{N}$ ,  $\text{sorszamok} \in \mathbb{N}[1..db]$

Ef:  $1 \leq n$  és  $n \leq 1000$  és  $1 \leq m$  és  $m \leq 1000$  és  $\forall i \in [1..n]: (\forall j \in [1..m]: (-50 \leq \text{homerseklet}[i,j]$  és  $\text{homerseklet}[i,j] \leq 50))$

Uf:  $(db, \text{sorszamok}) = \text{KIVÁLOGAT}(i=1..n, \text{VAN}(j=2..m, \text{homerseklet}[i,j-1] - \text{homerseklet}[i,j] \geq 10$  vagy  $\text{homerseklet}[i,j] - \text{homerseklet}[i,j-1] \geq 10), i)$

### Visszavezetés

*Kiválogatás*

$db \sim db$

$y \sim \text{sorszamok}$

$e..u \sim 1..n$

$T(i) \sim \text{VAN}(j=2..m, \text{homerseklet}[i,j-1] - \text{homerseklet}[i,j] \geq 10$  vagy  $\text{homerseklet}[i,j] - \text{homerseklet}[i,j-1] \geq 10)$

$f(i) \sim i$

*Eldöntés (van)*

$e..u \sim 2..m$

$T(i) \sim \text{homerseklet}[i,j-1] - \text{homerseklet}[i,j] \geq 10$  vagy  $\text{homerseklet}[i,j] - \text{homerseklet}[i,j-1] \geq 10$

### Algoritmus

db:=0	
i=1..n	
j:=2	
j<=m és nem (homerseklet[i,j-1]-homerseklet[i,j]>=10 vagy homerseklet[i,j]-homerseklet[i,j-1]>=10)	
j:=j+1	
j<=m	
db:=db+1	-
sorszamok[db]:=i	

## Fejlesztői környezet

IBM PC, exe futtatására alkalmas operációs rendszer (pl. Windows 11 Home). Visual Studio 2022 (Version 17.2.3) fejlesztői környezet.

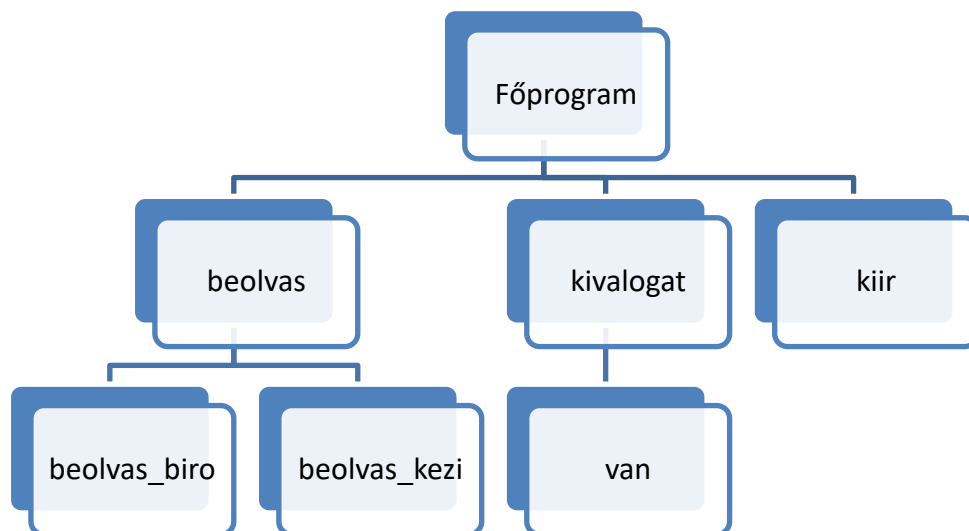
## Forráskód

A teljes fejlesztői anyag –kicsomagolás után– a nagy\_valtozasu\_telepulesek nevű könyvtárban található meg. A fejlesztés során használt könyvtár-struktúra:

Állomány	Magyarázat
nagy_valtozasu_telepulesek\bin\Debug\net8.0\nagy_valtozasu_telepulesek.exe	futtatható kód (a futtatáshoz szükséges fájlokkal)
nagy_valtozasu_telepulesek\obj\	mappa fordításhoz szükséges kódokkal
nagy_valtozasu_telepulesek\Program.cs	C# forráskód
nagy_valtozasu_telepulesek\teszt1.txt	teszt-bemeneti fájl <sub>1</sub>
nagy_valtozasu_telepulesek\teszt2.txt	teszt-bemeneti fájl <sub>2</sub>
nagy_valtozasu_telepulesek\teszt3.txt	teszt-bemeneti fájl <sub>3</sub>
nagy_valtozasu_telepulesek\teszt4.txt	teszt-bemeneti fájl <sub>4</sub>
nagy_valtozasu_telepulesek\teszt5.txt	teszt-bemeneti fájl <sub>5</sub>
nagy_valtozasu_telepulesek\doksi\Programozás komplex beadandó fázis2.docx	dokumentációk (ez a fájl)

## Megoldás

### Függvénystruktúra



## *A kód*

A Program.cs fájl tartalma:

```
using System;
using System.Collections.Generic;
namespace nagy_valtozasu_telepulesek
{
    class Program
    {
        static int[,] beolvas()
        {
            if (Console.IsInputRedirected)
            {
                return beolvas_biro();
            }
            else
            {
                return beolvas_kezi();
            }
        }
        static int[,] beolvas_biro()
        {
            string[] sortomb = Console.ReadLine().Split(' ');
            int n = int.Parse(sortomb[0]);
            int m = int.Parse(sortomb[1]);
            int[,] homerseklet = new int[n, m];

            for (int i = 0; i < n; i++)
            {
                sortomb = Console.ReadLine().Split(' ');
                for (int j = 0; j < m; j++)
                {
                    homerseklet[i, j] = int.Parse(sortomb[j]);
                }
            }
            return homerseklet;
        }
    }
}
```



```

}
static int[,] beolvas_kezi()
{
    int n, m;
    bool jo;
    do
    {
        Console.ResetColor();
        Console.Write("Települések száma = ");
        jo = int.TryParse(Console.ReadLine(), out n) && 1<=n && n<=1000;
        if (!jo)
        {
            Console.ForegroundColor = ConsoleColor.Red;
            Console.WriteLine("A szám nem megfelelő (1<=N<=1000)!");
        }
    } while (!jo);
    do
    {
        Console.ResetColor();
        Console.Write("Napok száma = ");
        jo = int.TryParse(Console.ReadLine(), out m) && 1 <= m && m <= 1000;
        if (!jo)
        {
            Console.ForegroundColor = ConsoleColor.Red;
            Console.WriteLine("A szám nem megfelelő (1<=N<=1000)!");
        }
    } while (!jo);

    int[,] homerseklet = new int[n,m];
    for (int i = 0; i < n; i++)
    {
        for (int j = 0; j < m; j++)
        {
            do
            {

```

```

        Console.ResetColor();
        Console.Write($"{i + 1}. település {j + 1}. hőmérséklet értéke = ");
        jo = int.TryParse(Console.ReadLine(), out homerseklet[i, j]) && -50 <=
homerseklet[i, j] && homerseklet[i, j] <= 50;
        if (!jo)
        {
            Console.ForegroundColor = ConsoleColor.Red;
            Console.WriteLine("A szám nem megfelelő (-50<=H<=50)");
        }
    } while (!jo);
}
}
return homerseklet;
}

```

```

static void kiir(int db, List<int> sorszamok)
{
    if (Console.IsOutputRedirected)
    {
        Console.Write($"{db} ");
        Console.Write(String.Join(' ', sorszamok));
    }
    else
    {
        Console.ForegroundColor = ConsoleColor.Green;
        if (db == 0)
        {
            Console.WriteLine("Nincs a feltételnek megfelelő település!");
        }
        else
        {
            Console.WriteLine("{0} darab feltételnek megfelelő település is van,
sorszámaik: ", db);
            Console.WriteLine(String.Join(", ", sorszamok));
        }
        Console.ForegroundColor = ConsoleColor.Black;
    }
}

```

```

        Console.BackgroundColor = ConsoleColor.Gray;
        Console.Write("Kérem, nyomjon ENTER-t a folytatáshoz!");
        Console.ResetColor();
        Console.ReadLine();
    }
}

static (int db, List<int> sorszamok) megold(int[,] homerseklet)
{
    int db = 0;
    List<int> sorszamok = new List<int>();
    for (int i = 0; i < homerseklet.GetLength(0); i++)
    {
        int j = 1;
        while (j < homerseklet.GetLength(1) && Math.Abs(homerseklet[i, j - 1] -
homerseklet[i, j]) < 10)
        {
            j += 1;
        }
        if (j < homerseklet.GetLength(1))
        {
            db += 1;
            sorszamok.Add(i + 1);
        }
    }
    return (db, sorszamok);
}

static void Main(string[] args)
{
    int[,] homerseklet;
    int db = 0;
    List<int> sorszamok = new List<int>();

    homerseklet = beolvas();

    (db, sorszamok) = megold(homerseklet);
}

```

```

        kiir(db, sorszamok);
    }
}

```

### *A kód (magas szintű függvényekkel)*

A Program.cs fájl tartalma:

```

using System;
using System.Collections.Generic;
using System.Linq;
using System.Numerics;
using System.Text;
using System.Threading.Tasks;
using magas_szintű_mintamegvalósítások;

```

```

namespace nagy_valtozasu_telepulesek_magas

```

```

{

```

```

    internal class Program

```

```

    {

```

```

        static void Main(string[] args)

```

```

        {

```

```

            int[,] homerseklet;

```

```

            int db;

```

```

            int[] sorszamok;

```

```

            homerseklet = beolvas();

```

```

            int n = homerseklet.GetLength(0);

```

```

            int m = homerseklet.GetLength(1);

```

```

            sorszamok = Mintak.Kivalogat(0, n - 1, i => Mintak.Van(1, m - 1, j =>
(Math.Abs(homerseklet[i, j - 1] - homerseklet[i, j]) >= 10)), i => i + 1);

```

```

            db = sorszamok.Length;

```

```

            kiir(db, sorszamok);

```

```

    }
    static void kiir(int db, int[] sorszamok)
    {
        if (Console.IsOutputRedirected)
        {
            Console.Write($"{db} ");
            Console.Write(String.Join(' ', sorszamok));
        }
        else
        {
            Console.ForegroundColor = ConsoleColor.Green;
            if (db == 0)
            {
                Console.WriteLine("Nincs a feltételnek megfelelő település!");
            }
            else
            {
                Console.WriteLine("{0} darab feltételnek megfelelő település is van, sorszámaik: ", db);
                Console.WriteLine(String.Join(", ", sorszamok));
            }
            Console.ForegroundColor = ConsoleColor.Black;
            Console.BackgroundColor = ConsoleColor.Gray;
            Console.Write("Kérem, nyomjon ENTER-t a folytatáshoz!");
            Console.ResetColor();
            Console.ReadLine();
        }
    }
}

static int[,] beolvas()
{
    if (Console.IsInputRedirected)
    {
        return beolvas_biro();
    }
}

```

```

else
{
    return beolvas_kezi();
}
}

static int[,] beolvas_biro()
{
    string[] sortomb = Console.ReadLine().Split(' ');
    int n = int.Parse(sortomb[0]);
    int m = int.Parse(sortomb[1]);
    int[,] homerseklet = new int[n, m];

    for (int i = 0; i < n; i++)
    {
        sortomb = Console.ReadLine().Split(' ');
        for (int j = 0; j < m; j++)
        {
            homerseklet[i, j] = int.Parse(sortomb[j]);
        }
    }
    return homerseklet;
}

static int[,] beolvas_kezi()
{
    int n, m;
    bool jo;
    do
    {
        Console.ResetColor();
        Console.Write("Települések száma = ");
        jo = int.TryParse(Console.ReadLine(), out n) && 1 <= n && n <= 1000;
        if (!jo)
        {
            Console.ForegroundColor = ConsoleColor.Red;
            Console.WriteLine("A szám nem megfelelő (1<=N<=1000)!");
        }
    }

```

```

    }
} while (!jo);
do
{
    Console.ResetColor();
    Console.Write("Napok száma = ");
    jo = int.TryParse(Console.ReadLine(), out m) && 1 <= m && m <= 1000;
    if (!jo)
    {
        Console.ForegroundColor = ConsoleColor.Red;
        Console.WriteLine("A szám nem megfelelő (1<=N<=1000)!");
    }
} while (!jo);

int[,] homerseklet = new int[n, m];
for (int i = 0; i < n; i++)
{
    for (int j = 0; j < m; j++)
    {
        do
        {
            Console.ResetColor();
            Console.Write($"{i + 1}. település {j + 1}. hőmérséklet értéke = ");
            jo = int.TryParse(Console.ReadLine(), out homerseklet[i, j]) && -50 <=
homerseklet[i, j] && homerseklet[i, j] <= 50;
            if (!jo)
            {
                Console.ForegroundColor = ConsoleColor.Red;
                Console.WriteLine("A szám nem megfelelő (-50<=H<=50)");
            }
        } while (!jo);
    }
}
return homerseklet;
}

```

}

}

## Tesztelés

### *Automatikus tesztek (Bíró)*

**Összpont: 100/**

Teszt#	Pont	Verdikt	futási idő
1.1	4/4	Helyes	0.032 sec
2.1	4/4	Helyes	0.034 sec
3.1	4/4	Helyes	0.034 sec
4.1	4/4	Helyes	0.034 sec
5.1	4/4	Helyes	0.034 sec
6.1	5/5	Helyes	0.036 sec
7.1	5/5	Helyes	0.034 sec
8.1	5/5	Helyes	0.033 sec
9.1	5/5	Helyes	0.034 sec
10.1	5/5	Helyes	0.034 sec
11.1	5/5	Helyes	0.034 sec
12.1	5/5	Helyes	0.036 sec
13.1	5/5	Helyes	0.036 sec
14.1	5/5	Helyes	0.035 sec
15.1	5/5	Helyes	0.037 sec
16.1	6/6	Helyes	0.073 sec
17.1	6/6	Helyes	0.069 sec
18.1	6/6	Helyes	0.062 sec
19.1	6/6	Helyes	0.137 sec
20.1	6/6	Helyes	0.332 sec

Beadva: 2025-01-12 00:04:09.0



## Automatikus tesztek (Bíró, magas szintű függvényekkel)

**Összpont: 100/**

Teszt#	Pont	Verdikt...	futási idő
1.1	4/4	Helyes	0.035 sec
2.1	4/4	Helyes	0.035 sec
3.1	4/4	Helyes	0.036 sec
4.1	4/4	Helyes	0.035 sec
5.1	4/4	Helyes	0.035 sec
6.1	5/5	Helyes	0.036 sec
7.1	5/5	Helyes	0.036 sec
8.1	5/5	Helyes	0.034 sec
9.1	5/5	Helyes	0.035 sec
10.1	5/5	Helyes	0.035 sec
11.1	5/5	Helyes	0.036 sec
12.1	5/5	Helyes	0.044 sec
13.1	5/5	Helyes	0.044 sec
14.1	5/5	Helyes	0.037 sec
15.1	5/5	Helyes	0.038 sec
16.1	6/6	Helyes	0.060 sec
17.1	6/6	Helyes	0.070 sec
18.1	6/6	Helyes	0.063 sec
19.1	6/6	Helyes	0.122 sec
20.1	6/6	Helyes	0.316 sec

Beadva: 2025-01-12 00:47:43.0

## Érvényes tesztesetek

### 1. teszteset: *teszt1.txt*

Bemenet
3 4 10 20 30 40 5 15 25 35 -10 0 10 20
Kimenet
3 1 2 3

2. *teszteset: test2.txt*

Bemenet
2 5 -10 -20 -30 -40 -50 50 40 30 20 10
Kimenet
2 1 2

3. *teszteset: be3.txt*

Bemenet
4 6 0 0 0 0 0 10 10 10 10 10 10 -50 -40 -30 -20 -10 0 5 15 5 15 5 15
Kimenet
2 3 4

4. *teszteset: be4.txt*

Bemenet
1 3 -5 10 -5
Kimenet
1 1

5. *teszteset: be5.txt*

Bemenet
5 7 10 10 10 10 10 10 10 -10 -10 -10 -10 -10 -10 -10 50 40 30 20 10 0 -10 15 15 25 15 25 15 25 0 10 20 30 40 50 50
Kimenet
3 3 4 5

*Érvénytelen tesztesetek*

Billentyűzetes bevétel esetén

## 6. *teszteset*

Bemenet – szöveges adat
N = 11tizenegy
Kimenet
Újrakérdezés: N =

## 7. *teszteset*

Bemenet – Nem megfelelő szám
N = -1
Kimenet
Újrakérdezés: N =

...

## 8. *teszteset*

...

## Fejlesztési lehetőségek

1. Többszöri futtatás megszervezése
2. Helységek és madárfajok nevének megadása
3. Grafikus visszajelzés a számolás lépéseiről

([Github repo](#))