„Programozási alapismeretek”
beadandó feladat

Készítette: Éles Júlia
Neptun-azonosító: GMGKGL
E-mail: juliaeles6@gmail.com

Kurzuskód: IT-13PA1EG
Gyakorlatvezető neve: Nikházy László

2023. január 12.

Tartalom

Felhasználói dokumentáció

Feladat

A települések legalább felében melegedő napok.

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 napokat, amikor legalább a települések felében melegedés várható az előző naphoz képest!

Futási környezet

IBM PC, exe futtatására alkalmas, 32-bites operációs rendszer (pl. Windows 7). Nem igényel egeret.

Használat

A program indítása

A program a `GMGKGL\bin\Release\GMGKGL.exe` néven található a tömörített állományban.

A program használata billentyűzetről való bevitel esetén

A `GMGKGL.exe` fájl elindításával a program az adatokat a **billentyűzet**ről olvassa be a következő sorrendben:

#	Adat	Magyarázat
1.	*Települések száma (N) és napok száma (M)*	1 és 1000 közötti egész számok
2.	*1. településre várható maximum hőmérsékletek (M db)*	-50 és 50 közötti egész számok innentől
3.	*2. településre várható maximum hőmérsékletek (M db)*	
4.	*3. településre várható maximum hőmérsékletek (M db)*	
...	*...*	
	i. településre várható maximum hőmérsékletek (M db)	
	...	
	N. településre várható maximum hőmérsékletek (M db)	

A program használata fájlból való bevitel esetén

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

```
GMGKGL.exe < be1.txt
```

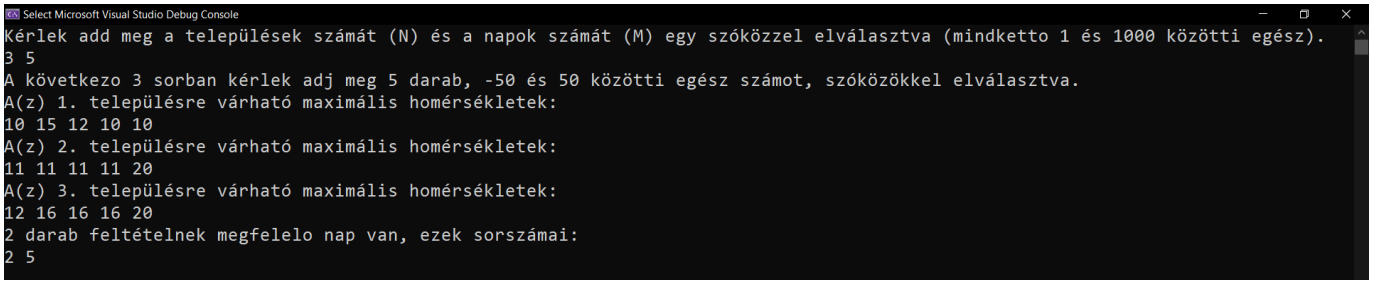
A fájl felépítésének a következő formai követelményei vannak. A fájl első sorában a települések száma (N) és a napok száma (M) van. A következő N sor mindegyikében M darabszám szerepel, közülük az i-edik sorban a j-edik szám az i-edik településen a j-edik napra várt maximum hőmérséklet áll. Például:

3 5
10 15 12 10 10
11 11 11 11 20
12 16 16 16 20

A program kimenete

A program kiírja azoknak a napoknak a darabszámát és a sorszámait, amely napokon legalább a települések felében melegedés várható az előző naphoz képest.

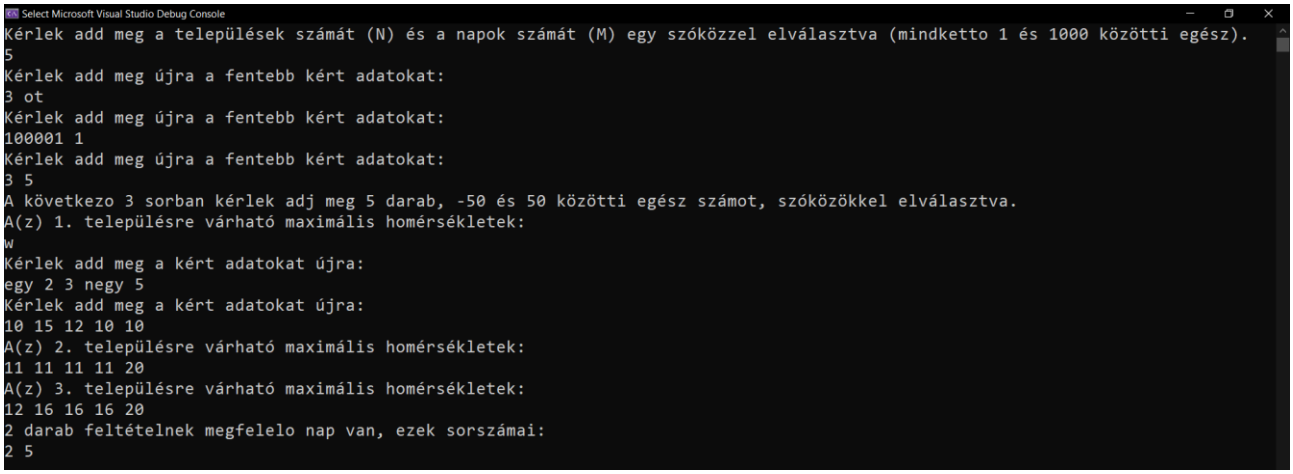
Minta bemenet és kimenet



Hibalehetőségek

Az egyes bemeneti adatokat a fenti mintának megfelelően kell megadni. Hiba, ha bármelyik megadandó adat nem természetes szám, vagy nem a megadott keretek közé esik, vagy egy sorban kevesebb/több adatot adunk meg, mint kellene. Hiba esetén a program azzal jelzi a hibát, hogy újra kérdezi az adatot.

Mintafutás hibás bemeneti adatok esetén:



Fejlesztői dokumentáció

Feladat

A települések legalább felében melegedő napok.

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 napokat, amikor legalább a települések felében melegedés várható az előző naphoz képest!

Tervezés

Specifikáció

Bemenet: $N, M \in \mathbb{N}, Homerseklet_{1..N,1..M} \in \mathbb{N}^{N \times M}$

Kimenet: $db \in \mathbb{N}, Napok_{1..db} \in \mathbb{N}^{db}$

Előfeltétel: $1 \leq N, M \leq 1000$ és $-50 \leq Homerseklet_{i,j} \leq 50, \forall i \in \{1..N\},\ j \in \{1..M\}$

Utófeltétel: $(db, Napok) = Kiválogat^{M}_{\substack{j=2 \\ jo(j,Homerseklet,N)}} j$ és

$jo: (\mathbb{N} \times \mathbb{N}^{N \times M} \times \mathbb{N}) \to \mathbb{L}$ és

$jo(j, Homerseklet, N) = \sum^{N}_{\substack{i=1 \\ Homerseklet_{i,j} > Homerseklet_{i,j-1}}} 1 \geq N/2$

Visszavezetés

Kiválogatás

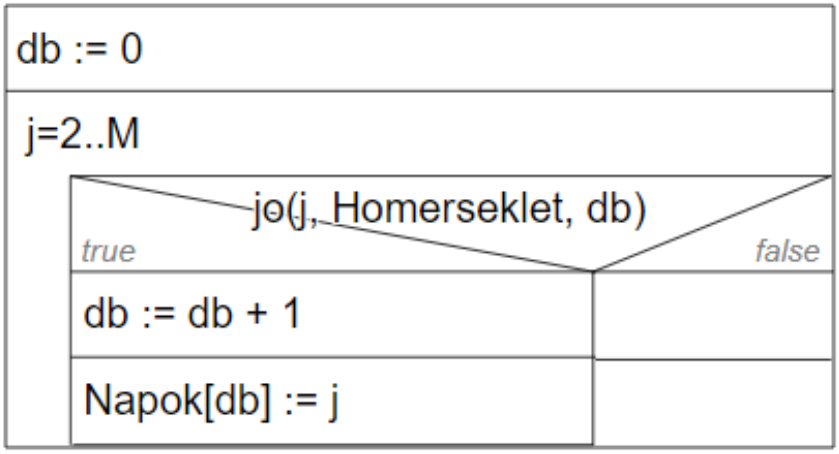
$y \sim Napok$

$T(x_i) \sim jo(j, Homerseklet, N)$

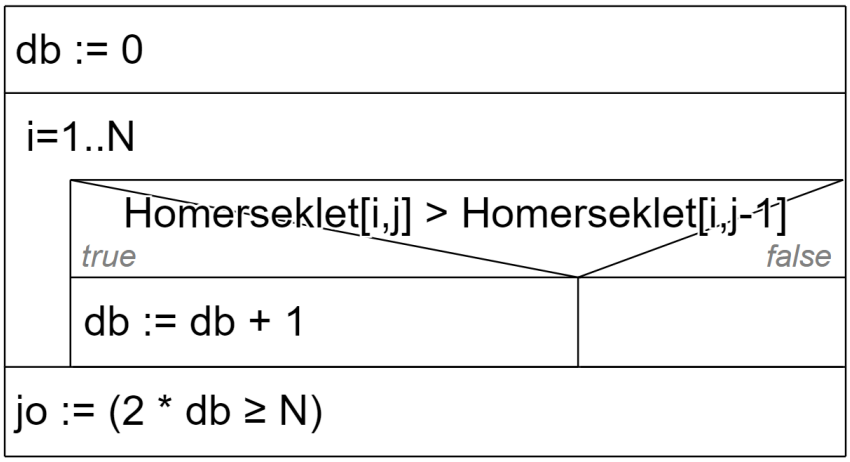
Megszámlálás

$T(x_i) \sim Homerseklet_{i,j} > Homerseklet_{i,j-1}$

Algoritmus



jo(j: Egész, Homerseklet: Tömb(1..N, 1..M: Egész), N: Egész): Logikai



Fejlesztői környezet

IBM PC, exe futtatására alkalmas operációs rendszer (pl. Windows 10 Pro). Visual Studio 2022 (Version 17.2.3) fejlesztői környezet.

Forráskód

A teljes fejlesztői anyag –kicsomagolás után– a `GMGKGL` nevű könyvtárban található meg. A fejlesztés során használt könyvtár-struktúra:

Állomány	**Magyarázat**
`GMGKGL\bin\Release\netcoreapp3.1\GMGKGL.exe`	futtatható kód (a futtatáshoz szükséges fájlokkal)
`GMGKGL\obj\`	mappa fordításhoz szükséges kódokkal
`GMGKGL\Program.cs`	C# forráskód
`GMGKGL\teszt1.txt`	teszt-bemeneti fájl$_1$
`GMGKGL\teszt2.txt`	teszt-bemeneti fájl$_2$
`GMGKGL\teszt3.txt`	teszt-bemeneti fájl$_3$
`GMGKGL\teszt4.txt`	teszt-bemeneti fájl$_4$
`GMGKGL\teszt5.txt`	teszt-bemeneti fájl$_5$
`GMGKGL\doksi\GMGKGL.docx`	dokumentációk (ez a fájl)

Megoldás

Programparaméterek

Változók

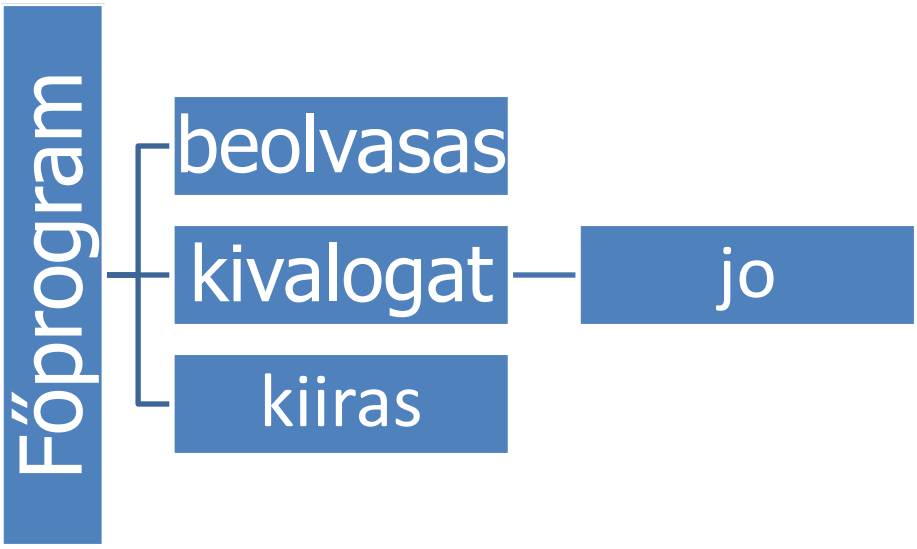
```
Homerseklet   : Tömb(1..N,1..M:Egész)
N, M, db      : Egész
Napok         : Tömb(1..db: Egész)
```

Programfelépítés

A program által használt modulok (és helyük):

`Program.cs` – program, a forráskönyvtárban
`GMGKGL.sln` – program 'megoldás fájl', a forráskönyvtárban
`GMGKGL.csproj` – program 'projekt fájl', a forráskönyvtárban

Függvénystruktúra



A kód

A `Program.cs` fájl tartalma:

```
using System;
using System.Collections.Generic;

namespace Nagybeadando {
    internal class Program {
        static void beolvasas(out int N, out int M, out int[,] Homerseklet) {
            bool jo = false;
            string[] be;
            N = M = 0;

            Console.Error.WriteLine("Kérlek add meg a települések számát (N) és a napok
számát (M) egy szóközzel elválasztva (mindkettő 1 és 1000 közötti egész).");

            while (!jo) {
                be = Console.ReadLine().Split(' ',
StringSplitOptions.RemoveEmptyEntries);
                if(be.Length == 2) {
                    jo = int.TryParse(be[0], out N) && int.TryParse(be[1], out M) && (1
<= Math.Min(N, M)) && (Math.Max(N, M) <= 1000);
                }

                if (!jo) {
                    Console.Error.WriteLine("Kérlek add meg újra a fentebb kért
adatokat:");
```

```
                }
            }


            Homerseklet = new int[N + 1, M + 1];

            Console.Error.WriteLine($"A következő {N} sorban kérlek adj meg {M} darab, -50
és 50 közötti egész számot, szóközökkel elválasztva.");

            for (int i = 1; i <= N; ++i) {
                jo = false;
                Console.Error.WriteLine($"A(z) {i}. településre várható maximális
hőmérsékletek: ");
                while (!jo) {
                    be = Console.ReadLine().Split(' ',
StringSplitOptions.RemoveEmptyEntries);
                    if(be.Length == M) {
                        bool jjo = true;
                        for (int j = 1; j <= M; ++j) {
                            jjo = jjo && int.TryParse(be[j - 1], out Homerseklet[i, j])
&& (-50 <= Homerseklet[i, j]) && (Homerseklet[i, j] <= 50);
                        }

                        jo = jjo;
                    }

                    if (!jo) {
                        Console.Error.WriteLine("Kérlek add meg a kért adatokat újra: ");
                    }
                }
            }
        }
        static bool jo(int j, int[,] Homerseklet, int N) {
            int db = 0;
            for(int i = 1; i <= N; ++i) {
                if (Homerseklet[i, j] > Homerseklet[i, j - 1]) {
                    ++db;
                }
            }

            return (2 * db >= N);
        }

        static void kivalogat(out int db, out List<int> Napok, int[,] Homerseklet, int N,
int M) {
            db = 0;
            Napok = new List<int>();
            for (int j = 2; j <= M; ++j) {
                if (jo(j, Homerseklet, N)) {
                    ++db;
                    Napok.Add(j);
                }
            }
        }

        static void kiiras(int db, List<int> Napok) {
            Console.Write(db + " ");
            Console.Error.WriteLine("darab feltételnek megfelelő nap van, ezek
sorszámai:");
            Console.WriteLine(string.Join(' ', Napok));
        }

        static void Main(string[] args) {
```

```
            int N, M;
            int[,] Homerseklet;

            beolvasas(out N, out M, out Homerseklet);

            int db;
            List<int> Napok;

            kivalogat(out db, out Napok, Homerseklet, N, M);
            kiiras(db, Napok);
        }
    }
}
```

Tesztelés

Érvényes tesztesetek

1. teszteset: be1.txt

Bemenet – *egy település, egy nap*
1 1 -50
Kimenet
0

2. teszteset: be2.txt

Bemenet – *egy település, több nap*
1 5 -50 0 0 12 25
Kimenet
3 2 4 5

3. teszteset: be3.txt

Bemenet – *több település, egy nap*
3 1 25 13 -8
Kimenet
0

4. teszteset: be4.txt

Bemenet – több település, több nap
3 5 10 15 12 10 10 11 11 11 11 20 12 16 16 16 20
Kimenet
2 2 5

5. teszteset: be5.txt

Bemenet – *egy település, több nap, egyforma hőmérsékletek*
1 5 13 13 13 13 13
Kimenet
0

Érvénytelen tesztesetek

Billentyűzetes bevitel esetén

6. teszteset

Bemenet – *szöveges adat*
11 tizenegy
Kimenet
Kérlek add meg a kért adatokat újra:

7. teszteset

Bemenet – *túl kevés adat*
3
Kimenet
Kérlek add meg a kért adatokat újra:

8. teszteset

Bemenet – *túl sok adat*
3 5 6
Kimenet
Kérlek add meg a kért adatokat újra:

Fejlesztési lehetőségek

1. Többszöri futtatás megszervezése
2. Települések nevének megadása
3. Valós számok használatának engedélyezése hőmérsékletek esetén