**„Programozás”  
beadandó feladat**

Készítette: *Jeney Zoltán*Neptun-azonosító: *DB7ZTC*E-mail:jeneyzoltan2@gmail.com

Kurzuskód: IT-18PROGEGGyakorlatvezető neve: Németh Kristóf

2024. január 12.

Tartalom

[Felhasználói dokumentáció 3](#_Toc155108363)

[Feladat 3](#_Toc155108364)

[Futási környezet 3](#_Toc155108365)

[Használat 3](#_Toc155108366)

[A program indítása 3](#_Toc155108367)

[A program használata billentyűzetről való bevitel esetén 3](#_Toc155108368)

[A program használata fájlból való bevitel esetén 3](#_Toc155108369)

[A program kimenete 4](#_Toc155108370)

[Minta bemenet és kimenet 4](#_Toc155108371)

[Hibalehetőségek 4](#_Toc155108372)

[Fejlesztői dokumentáció 5](#_Toc155108373)

[Feladat 5](#_Toc155108374)

[Tervezés 5](#_Toc155108375)

[Specifikáció 5](#_Toc155108376)

[Visszavezetés 5](#_Toc155108377)

[Algoritmus 6](#_Toc155108378)

[Fejlesztői környezet 6](#_Toc155108379)

[Forráskód 7](#_Toc155108380)

[Megoldás 7](#_Toc155108381)

[Függvénystruktúra 7](#_Toc155108382)

[A kód 7](#_Toc155108383)

[Tesztelés 10](#_Toc155108384)

[Érvényes tesztesetek 10](#_Toc155108385)

[Érvénytelen tesztesetek 11](#_Toc155108386)

[Fejlesztési lehetőségek 11](#_Toc155108387)

2. Felhasználói dokumentáció
   1. Feladat
   2. Legkisebb ingadozású települések
   3. 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 a legkisebb hőmérséklet ingadozású településeket!
   4. Futási környezet

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

* 1. Használat
     1. A program indítása

A program az DB7ZTC\bin\Debug\net6.0\ConsoleApp1 .exe néven található a tömörített állományban.

* + 1. A program használata billentyűzetről való bevitel esetén

Az ConsoleApp1.exe fájl elindításával a program az adatokat a **billentyűzet**ről olvassa be a következő sorrendben, szóközzel elválasztva, illetve az első két adat és minden M. hőmérséklet adat után soremeléssel elválasztva:

|  |  |  |
| --- | --- | --- |
| # | Adat | Magyarázat |
| **1.** | *Települések száma (N)* | *(1≤N≤1000)* |
| **2.** | *Napok száma (M)* | *(1≤M≤1000)* |
| **3.** | *1. településen az 1. nap várható legmagasabb hőmérséklete* | *(-50≤Hi,j≤50) innentől* |
| **4.** | *1. településen a 2. nap várható legmagasabb hőmérséklete* |  |
| **…** | *…* |  |
|  | *1. településen az M. nap várható legmagasabb hőmérséklete* |  |
|  | *2. településen az 1. nap várható legmagasabb hőmérséklete* |  |
|  | *…* |  |
|  | *N. településen azM. nap várható legmagasabb hőmérséklete* |  |

* + 1. A program kimenete

A program kiírja azon települések T számát, ahol a maximális és a minimális előrejelzés különbsége a lehető legkisebb volt, ezt követően pedig ezen települések sorszámait íratja ki, növekvő sorrendben.

* + 1. Minta bemenet és kimenet

A képen szöveg, képernyőkép látható

Automatikusan generált leírás

* + 1. Hibalehetőségek

Az egyes bemeneti adatokat a fenti mintának megfelelően kell megadni. Hiba, ha bármelyik megadandó adat nem a megadott intervallumba tartozik, illetve nem szám típusú. Hiba esetén a program azzal jelzi a hibát, hogy hibaüzenetet ír a kimenetre, a hibát tartalmazó sor után.

* + - 1. Mintafutás hibás bemeneti adatok esetén:

A képen szöveg, képernyőkép látható

Automatikusan generált leírás

1. Fejlesztői dokumentáció
   1. Feladat

**Legkisebb ingadozású 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 a legkisebb hőmérséklet ingadozású településeket!

* 1. Tervezés
     1. Specifikáció

Be: N∈Z, M∈Z, elorejelzes∈Z[1..N,1..M]

Ki: db, y∈[1..db]

Fv: soringadozas: Z, Z, Z[] ->Z, soringadozas(i, M, Matr)=ABS(MAX(j=1..M,Matr[i,j])-MIN(j=1..M,Matr[i,j]))

Fv: soringadozasmin: Z, Z, Z[]->Z soringadozasmin(N,M,Matr)=MIN(i=1..N,soringadozas(i,M,Matr))

Fv: T: Z->L, T(i)=soringadozas(i)=vsoringadozasmin

Sa: vsoringadozasmin∈Z

vsoringadozasmin:=soringadozasmin(N,M,elorejelzes)

Ef: N∈[1..1000], M∈[1..1000], ∀i,j∈[1..1000]:elorejelzes[i,j]∈[-50..50]

Uf: (db, y) = KIVÁLOGAT(i=1..N, T(i), i)

* + 1. Visszavezetés

Kivalogatás:

e..u ~ 1..N

db, y ~ db, y

T(i) ~ T(i)

f(i) ~ i

Minimumkiválasztás (külső):

e..u ~ 1..N

minért ~minert

minind ~ -

f(i) ~ soringadozas(i,M,Matr)

Minimum kiválasztás (belső)

e..u ~ 1..M

minert ~ minert

minind ~ -

f(j) ~ Matr[i,j]

Maximum kiválasztás (belső)

e..u ~ 1..M

maxert ~ maxert

maxert ~ -

f(j) ~ Matr[i,j]

* + 1. Algoritmus

A képen szöveg, képernyőkép, szoftver látható

Automatikusan generált leírás

* 1. Fejlesztői környezet

IBM PC, exe futtatására alkalmas operációs rendszer (pl. Windows 11 Home). Visual Studio 2022 (Version 17.6.2) fejlesztői környezet.

* 1. Forráskód

A teljes fejlesztői anyag –kicsomagolás után– az DB7ZTC nevű könyvtárban található meg. A fej­lesztés során használt könyvtár-struktúra:

|  |  |
| --- | --- |
| Állomány | Magyarázat |
| DB7ZTC\bin\Debug\net6.0\ConsoleApp1.exe | futtatható kód (a futtatáshoz szükséges fájlokkal) |
| DB7ZTC\obj\ | mappa fordításhoz szükséges kódokkal |
|  |  |
| DB7ZTC\Program.cs | C# forráskód |
|  |  |
| DB7ZTC\be1.txt | teszt-bemeneti fájl1 |
| DB7ZTC\be2.txt | teszt-bemeneti fájl2 |
| DB7ZTC\be3.txt | teszt-bemeneti fájl3 |
| DB7ZTC\be4.txt | teszt-bemeneti fájl4 |
| DB7ZTC\be5.txt | teszt-bemeneti fájl5 |
|  |  |
| DB7ZTC\dokumentacio\DB7ZTC.docx | dokumentáció (ez a fájl) |

* 1. Megoldás
     1. Függvénystruktúra
     2. A kód

A Program.cs fájl tartalma:

/\*

Készítette: Jeney Zoltán

Neptun: DB7ZTC

E-mail: jeneyzoltan2@gmail.com

Feladat: Legkisebb ingadozású települések

\*/

using System;

using System.Collections.Generic;

namespace ConsoleApp1

{

internal class Program

{

#region Maximum-, minimumkiválasztás

#region Max:Intervallum

public static (int maxind, H maxert) Max<H>(int e, int u, Func<int, H> f, Func<H, H, bool> isGreater = null)

{

bool \_isGreater(H a, H b)

{

if (a is IComparable<H> aa)

{

return aa.CompareTo(b) > 0;

}

else

{

return isGreater != null ? isGreater(a, b) : false;

}

}

H maxert = f(e);

int maxind = e;

for (int i = e + 1; i <= u; i++)

{

if (\_isGreater(f(i), maxert))

{

maxert = f(i);

maxind = i;

}

}

return (maxind, maxert);

}

#endregion

#region Max:Tömb transzformátumfüggvénnyel

public static (int maxind, G maxert) Max<H, G>(H[] arr, Func<H, G> f, Func<G, G, bool> isGreater = null)

{

return Max(0, arr.Length - 1, i => f(arr[i]), isGreater);

}

#endregion

#region Max:Tömb

public static (int maxind, H maxert) Max<H>(H[] arr, Func<H, H, bool> isGreater = null)

{

return Max(arr, p => p, isGreater);

}

#endregion

#region Min:Intervallum

public static (int minind, H minert) Min<H>(int e, int u, Func<int, H> f, Func<H, H, bool> isSmaller = null)

{

bool \_isSmaller(H a, H b)

{

if (a is IComparable<H> aa)

{

return aa.CompareTo(b) < 0;

}

else

{

return isSmaller != null ? isSmaller(a, b) : false;

}

}

H minert = f(e);

int minind = e;

for (int i = e + 1; i <= u; i++)

{

if (\_isSmaller(f(i), minert))

{

minert = f(i);

minind = i;

}

}

return (minind, minert);

}

#endregion

#region Min:Tömb transzformátumfüggvénnyel

public static (int minind, G minert) Min<H, G>(H[] arr, Func<H, G> f, Func<G, G, bool> isGreater = null)

{

return Min(0, arr.Length - 1, i => f(arr[i]), isGreater);

}

#endregion

#region Min:Tömb

public static (int minind, H minert) Min<H>(H[] arr, Func<H, H, bool> isGreater = null)

{

return Min(arr, p => p, isGreater);

}

#endregion

#endregion

public static int soringadozas(int i, int M, int[,] Matr)

{

int[] vec = new int[M];

for (int j = 0; j < M; j++)

{

vec[j] = Matr[i, j];

}

int maxind, maxert, minert, minind;

(maxind, maxert) = Max(vec, (a, b) => a > b);

(minind, minert) = Min(vec, (a, b) => a < b);

return (Math.Abs(minert - maxert));

}

public static int soringadozasmin(int N, int M, int[,] Matr)

{

int seged;

int minert = soringadozas(0, M, Matr);

for (int i = 0; i < N; i++)

{

seged = soringadozas(i, M, Matr);

if (minert > seged)

{

minert = seged;

}

}

return (minert);

}

static void Main(string[] args)

{

//bemeneti változó deklarálása

int[,] elorejelzes;

//kimenet deklarálása

List<int> y = new List<int>();

elorejelzes = bekeres();

y = kivalogatas(elorejelzes);

kiiratas(y);

}

static void kiiratas(List<int> y)

{

Console.Write(y.Count);

Console.Write(' ');

for (int i = 0; i < y.Count; i++)

{

Console.Write((y[i] + 1) + " ");

}

}

static List<int> kivalogatas(int[,] matrix)

{

List<int> y = new List<int>();

int vsoringadozasmin = soringadozasmin(matrix.GetLength(0), matrix.GetLength(1), matrix);

bool T(int k)

{

return vsoringadozasmin == soringadozas(k, matrix.GetLength(1), matrix);

}

//Kivalogatas

for (int i = 0; i < matrix.GetLength(0); i++)

{

if (T(i))

{

y.Add(i);

}

}

return y;

}

static int[,] bekeres()

{

{

if (Console.IsInputRedirected)

{

return bekeres\_biro();

}

else

{

return bekeres\_kezi();

}

}

}

public static int[,] bekeres\_biro()

{

string S = Console.ReadLine();

int N;

int M;

M = int.Parse(S.Split(' ')[1]);

N = int.Parse(S.Split(' ')[0]);

string[] sorbaszamok = new string[M];

int[,] elorejelzes = new int[N, M];

for (int i = 0; i < N; i++)

{

sorbaszamok = (Console.ReadLine().Split(' '));

for (int j = 0; j < M; j++)

{

elorejelzes[i, j] = int.Parse(sorbaszamok[j]);

}

}

return elorejelzes;

}

static int[,] bekeres\_kezi() {

Console.WriteLine("Irja be szokozzel elvalasztva, hany telepules es hany nap van!");

string S = Console.ReadLine();

int N;

int M;

while (!int.TryParse(S.Split(' ')[0], out N) || !int.TryParse(S.Split(' ')[1], out M))

{

Console.WriteLine("Hibas (nem szamertekkel biro) adat(ok), irja be ujra oket: ");

S = Console.ReadLine();

}

while (N < 1 || N > 1000)

{

Console.WriteLine("Hibas telepules ertek (tartomanyon(1..1000) kivul eso adat), irja be ujra: ");

N = int.Parse(Console.ReadLine());

}

while (M < 1 || M > 1000)

{

Console.WriteLine("Hibas nap ertek (tartomanyon(1..1000) kivul eso adat), irja be ujra: ");

M = int.Parse(Console.ReadLine());

}

Console.WriteLine("Irja be sorrol sorra a telepulesek napi homersekleteit:");

string[] sorbaszamok = new string[M];

int[,] elorejelzes = new int[N, M];

for (int i = 0; i < N; i++)

{

sorbaszamok = (Console.ReadLine().Split(' '));

for (int j = 0; j < M; j++)

{

elorejelzes[i, j] = int.Parse(sorbaszamok[j]);

if (elorejelzes[i, j] < -50 || elorejelzes[i, j] > 50)

{

Console.WriteLine("Hibas adat");

}

}

}

return elorejelzes;

}

}

}

* 1. Tesztelés
     1. Érvényes tesztesetek
        1. teszteset: be1.txt

|  |
| --- |
| Bemenet – *egy telepules, egy nap* |
| 1 1  1 |
| Kimenet |
| 1 1 |

* + - 1. teszteset: be2.txt

|  |
| --- |
| Bemenet – *2 telepules, 2 nap* |
| 2 2  -47 44  -20 44 |
| Kimenet |
| 1 2 |

* + - 1. teszteset: be3.txt

|  |
| --- |
| Bemenet – *3 telepules, ugyanolyan adatok* |
| 3 5  -47 44 43 -29 48  -47 44 43 -29 48  -47 44 43 -29 48 |
|  |
| Kimenet |
| 3 1 2 3 |

* + - 1. teszteset: be4.txt

|  |
| --- |
| Bemenet – *3 telepules, utolso a megoldas* |
| 3 5  -47 44 43 -29 48  -47 44 43 -29 48  -27 44 43 -29 48 |
|  |
| Kimenet |
| 1 3 |

* + - 1. teszteset: be5.txt

|  |
| --- |
| Bemenet – *…* |
| 22 31 |
| -8 -27 26 -31 4 20 -31 -6 48 10 -44 -4 4 -24 -48 11 37 48 -49 -49 -25 -44 42 44 49 -40 28 46 48 21 14  39 47 -46 -42 -21 -2 -18 -34 21 -10 -40 -21 -39 -11 -42 -39 37 -34 42 -48 -28 -1 -20 -37 -41 1 7 37 22 -12 6  45 -33 -36 -50 2 -36 46 47 -49 -9 -49 46 -9 -27 -6 -19 20 -4 -4 21 -35 -42 45 -1 -19 22 -24 -22 -50 -3 7  12 -1 11 -27 15 49 3 0 -10 28 -16 -21 -20 40 -42 10 -44 -50 -8 47 -14 13 -33 9 20 -12 11 -46 -19 -3 -15  -41 38 30 -7 47 -38 -12 -40 -4 23 10 16 -43 -31 -33 -49 19 3 14 -43 -26 -8 -20 2 2 32 -46 40 23 -17 29  49 -50 -7 12 -5 -6 26 -34 -21 -11 4 -9 31 6 -22 -29 -28 -49 19 -48 -41 -30 -44 -18 15 -6 -41 29 -13 -29 13  -1 -4 -27 -44 -43 -1 -11 36 -5 -12 24 26 9 41 18 -41 -16 -44 27 -28 -48 31 -37 -14 -8 -25 -33 32 -27 -2 -11  -50 14 24 -28 -48 23 -13 -19 -25 -21 -37 -35 -4 48 -9 -15 -9 -38 -11 -50 37 47 35 -15 41 45 32 30 -9 -44 16  6 2 14 -28 -21 9 16 41 -13 20 19 -3 -8 20 -33 -45 -29 -26 -6 -30 14 -37 -24 -32 -28 10 6 -13 -26 -46 -36  -19 -45 -33 0 0 13 -38 15 40 25 35 42 28 9 18 6 42 -25 1 -41 47 29 41 -37 -6 27 36 -35 20 13 -30  -44 36 33 -1 -2 12 11 23 49 28 -49 11 4 -19 -29 47 13 -39 -34 24 34 -48 28 -32 5 -26 -41 24 12 -36 -23  6 5 -45 -17 -26 35 33 33 46 -11 49 27 -8 -42 29 -12 7 -47 -50 -48 -18 22 34 27 -16 -19 34 12 -39 -5 -3  -12 28 30 -23 -24 -35 -8 33 -45 -16 -17 15 -20 -49 15 -42 -46 37 29 -17 41 -28 -13 -49 -34 32 11 0 -34 -32 -12  17 48 -20 -9 6 23 -4 -17 -35 9 40 -41 -2 41 -16 16 6 -50 18 29 -6 45 17 -43 -16 33 -35 -4 30 5 17  -3 -34 49 -21 -4 47 -27 18 15 -35 -21 -48 -42 42 15 -26 27 -9 12 1 14 -43 -20 -6 21 26 7 -22 15 -18 -16  -29 22 6 37 49 -2 34 -26 4 24 -14 0 -32 -37 -49 -26 -2 -18 47 27 -49 -41 39 29 -25 -6 19 18 -29 -44 -21  25 -50 -43 11 43 -11 -40 36 46 -47 -25 -6 -35 9 3 34 -11 24 4 -7 36 35 16 39 49 -49 -12 -44 16 -14 -44  41 22 3 24 7 -18 3 23 21 48 22 -9 7 -18 3 48 25 42 -25 14 -40 -32 -38 49 27 32 -25 8 -22 38 -31  -10 24 4 -2 6 1 27 41 5 22 38 -46 -40 19 15 17 -12 -3 47 -37 -9 30 -31 16 -33 34 38 -46 -43 40 -29  12 -14 11 -30 -6 16 -19 2 -48 40 37 10 22 -37 16 -21 -36 -49 -49 -47 -3 -21 -34 -29 20 -11 -8 30 -48 27 13  29 44 44 -8 -42 8 -39 -13 -18 -33 36 -36 1 -14 43 -43 -20 -41 43 21 -16 5 -17 48 -36 39 48 -3 46 -5 11  19 38 1 45 15 38 -16 41 -29 33 20 -21 -35 43 29 36 -11 4 -41 -14 -23 28 6 33 -28 -46 12 -13 45 -48 -39 |
| Kimenet |
| 1 9 |

* + 1. Érvénytelen tesztesetek

Billentyűzetes bevitel esetén

* + - 1. teszteset

|  |
| --- |
| Bemenet – *szöveges adat* |
| Tizenegy tizenketto |
| Kimenet |
| Hibas (nem szamertekkel biro) adat(ok), irja be ujra oket: |

* + - 1. teszteset

|  |
| --- |
| Bemenet – *Negatív szám* |
| -3 5 |
| Kimenet |
| Hibas telepules ertek (tartomanyon(1..1000) kivul eso adat), irja be ujra: |

…

* 1. Fejlesztési lehetőségek

1. Többszöri futtatás megszervezése
2. Települések és napok nevének megadása
3. Grafikus visszajelzés a számolás lépéseiről