**„Programozás”  
beadandó feladat**

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

Kurzuskód: IT-18PROGEGGyakorlatvezető neve:Németh Kristóf

2024. január 8.

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\Release\DB7ZTC.exe néven található a tömörített állományban.

* + 1. A program használata billentyűzetről való bevitel esetén

Az DB7ZTC.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. 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 |
|  |  |
| A1B2C3\teszt1.txt | teszt-bemeneti fájl1 |
| A1B2C3\teszt2.txt | teszt-bemeneti fájl2 |
| A1B2C3\teszt3.txt | teszt-bemeneti fájl3 |
| A1B2C3\teszt4.txt | teszt-bemeneti fájl4 |
| A1B2C3\teszt5.txt | teszt-bemeneti fájl5 |
|  |  |
| A1B2C3\doksi\A1B2C3.docx | dokumentációk (ez a fájl) |

* 1. Megoldás
     1. Függvénystruktúra
     2. A kód

A Program.cs fájl tartalma:

/\*

Készítette: ???

Neptun: ???

E-mail: ???

Feladat: Madármegfigyelés/ Helységek csupa máshol is előforduló madárfajjal

\*/

using System;

namespace beadando\_helysegek\_csupa\_mashol\_madarral {

internal class Program {

static void Main(string[] args) {

// deklarálás: bemenet

int[,] mad;

// deklarálás: kimenet

// statikus tömbbel dolgozunk, így szükség van a db-re is

int db;

int[] helyseg;

mad = beolvas();

(db, helyseg) = kivalogat(mad);

kiir(db, helyseg);

}

static int[,] beolvas() {

if (Console.IsInputRedirected) {

return beolvas\_biro();

}

else {

return beolvas\_kezi();

}

}

static int[,] beolvas\_biro() {

string[] sor = Console.ReadLine().Split(" ");

int n = int.Parse(sor[0]);

int m = int.Parse(sor[1]);

int[,] mad = new int[n, m];

for (int i = 0; i < n; i++) {

sor = Console.ReadLine().Split(" ");

for (int j = 0; j < m; j++) {

mad[i, j] = int.Parse(sor[j]);

}

}

return mad;

}

static int[,] beolvas\_kezi() {

int n, m;

bool jo;

do {

Console.ResetColor();

Console.Write("Helységek száma = ");

jo = int.TryParse(Console.ReadLine(), out n) && n >= 0;

if (!jo) {

Console.ForegroundColor = ConsoleColor.Red;

Console.WriteLine("Természetes szám kell!");

}

} while (!jo);

do {

Console.ResetColor();

Console.Write("Madárfajok száma = ");

jo = int.TryParse(Console.ReadLine(), out m) && m >= 0;

if (!jo) {

Console.ForegroundColor = ConsoleColor.Red;

Console.WriteLine("Természetes szám kell!");

}

} while (!jo);

int[,] mad = new int[n, m];

for (int i = 0; i < n; i++) {

for (int j = 0; j < m; j++) {

do {

Console.ResetColor();

Console.Write("{0}. helyseg {1}. madárfaj darabszáma = ", i + 1, j + 1);

jo = int.TryParse(Console.ReadLine(), out mad[i, j]) && mad[i, j] >= 0;

if (!jo) {

Console.ForegroundColor = ConsoleColor.Red;

Console.WriteLine("Természetes szám kell!");

}

} while (!jo);

}

}

return mad;

}

static (int db, int[] helyseg) kivalogat(int[,] mad) {

int n = mad.GetLength(0);

int[] helyseg = new int[n];

int db = 0;

for (int i = 1; i <= n; i++) {

if (vanmadar(i, mad) && jo(i, mad)) {

db = db + 1;

helyseg[db - 1] = i;

}

}

return (db, helyseg);

}

static bool vanmadar(int i, int[,] mad) {

int m = mad.GetLength(1);

int j = 1;

while (j <= m && !(mad[i - 1, j - 1] > 0)) {

j = j + 1;

}

bool van = j <= m;

return van;

}

static bool jo(int i, int[,] mad) {

int m = mad.GetLength(1);

int j = 1;

while (j <= m && (mad[i - 1, j - 1] == 0 || masholis(i, j, mad))) {

j = j + 1;

}

bool mind = j > m;

return mind;

}

static bool masholis(int i, int j, int[,] mad) {

int n = mad.GetLength(0);

int k = 1;

while (k <= n && !(i != k && mad[k - 1, j - 1] > 0)) {

k = k + 1;

}

bool van = k <= n;

return van;

}

static void kiir(int db, int[] helyseg) {

if (Console.IsOutputRedirected) {

Console.WriteLine(db);

for (int i = 0; i < db; i++) {

Console.Write("{0} ", helyseg[i]);

}

Console.WriteLine();

}

else {

Console.ForegroundColor = ConsoleColor.Green;

if (db == 0) {

Console.WriteLine("Nincs a feltételnek megfelelő helység!");

}

else {

Console.WriteLine("{0} darab feltételnek megfelelő helység is van, sorszámaik:", db);

for (int i = 0; i < db - 1; i++) {

Console.Write("{0}, ", helyseg[i]);

}

Console.WriteLine(helyseg[db - 1]);

}

Console.ForegroundColor = ConsoleColor.Black;

Console.BackgroundColor = ConsoleColor.Gray;

Console.WriteLine("Kérem, nyomjon ENTER-t a folytatáshoz!");

Console.ResetColor();

Console.ReadLine();

}

}

}

}

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

|  |
| --- |
| Bemenet – *nincs helyseg, nincs madarfaj* |
| 0 0 |
| Kimenet |
| 0 |

* + - 1. teszteset: be2.txt

|  |
| --- |
| Bemenet – *1 helység, 1 madárfaj, 1 darab* |
| 1 1  1 |
| Kimenet |
| 0 |

* + - 1. teszteset: be3.txt

|  |
| --- |
| Bemenet – *1 helység, 1 madárfaj, nincs madár* |
| … |
|  |
| Kimenet |
| … |

* + - 1. teszteset: be4.txt

|  |
| --- |
| Bemenet – *…* |
| … |
|  |
| Kimenet |
| … |

* + - 1. teszteset: be5.txt

|  |
| --- |
| Bemenet – *…* |
| … |
| … |
| Kimenet |
| … |

* + 1. Érvénytelen tesztesetek

Billentyűzetes bevitel esetén

* + - 1. teszteset

|  |
| --- |
| Bemenet – *szöveges adat* |
| N = 11tizenegy |
| Kimenet |
| Újrakérdezés: N = |

* + - 1. teszteset

|  |
| --- |
| Bemenet – *Negatív szám* |
| N = -1 |
| Kimenet |
| Újrakérdezés:  N= |

…

* + - 1. 8. teszteset

…

* 1. Fejlesztési lehetőségek

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