

„Programozás” beadandó feladat

*Készítette: ???
Neptun-azonosító: ???
E-mail: ???*

*Kurzuskód: IT-18PROGEG
Gyakorlatvezető neve: ???*

2024. január 2.

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.....	5
Feladat.....	5
Tervezés.....	5
Specifikáció.....	5
Visszavezetés.....	5
Algoritmus.....	6
Fejlesztői környezet.....	6
Forráskód.....	7
Megoldás.....	7
Függvénystruktúra.....	7
A kód.....	7
Tesztelés.....	10
Érvényes tesztesetek.....	10
Érvénytelen tesztesetek.....	11
Fejlesztési lehetőségek.....	11

Felhasználói dokumentáció

Feladat

Helységek csupa máshol is előforduló madárfajjal

Az ország néhány helységében madármegfigyelést végeztünk. Mindegyikben megadtuk, hogy milyen fajú madárból hányat láttunk.

Készíts programot, amely megadja azokat a helységeket, ahol csak olyan madarat láttunk, amelyik valahol máshol is előfordult!

Futási környezet

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

Használat

A program indítása

A program az A1B2C3\bin\Release\A1B2C3.exe néven található a tömörített állományban.

A program használata billentyűzetről való bevétel esetén

Az A1B2C3.exe fájl elindításával a program az adatokat a **billentyűzetről** olvassa be a következő sorrendben:

#	Adat	Magyarázat
1.	Helységek száma (<i>n</i>)	Nemnegatív szám
2.	Madárfajok száma (<i>m</i>)	Nemnegatív szám
3.	1. helységben az 1. madárfaj száma	Nemnegatív szám innentől
4.	1. helységben az 2. madárfaj száma	
...	...	
	1. helységben az <i>m</i> . madárfaj száma	
	2. helységben az 1. madárfaj száma	
	...	
	<i>n</i> . helységben az <i>m</i> . madárfaj száma	

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:

```
A1B2C3.exe < bel.txt
```

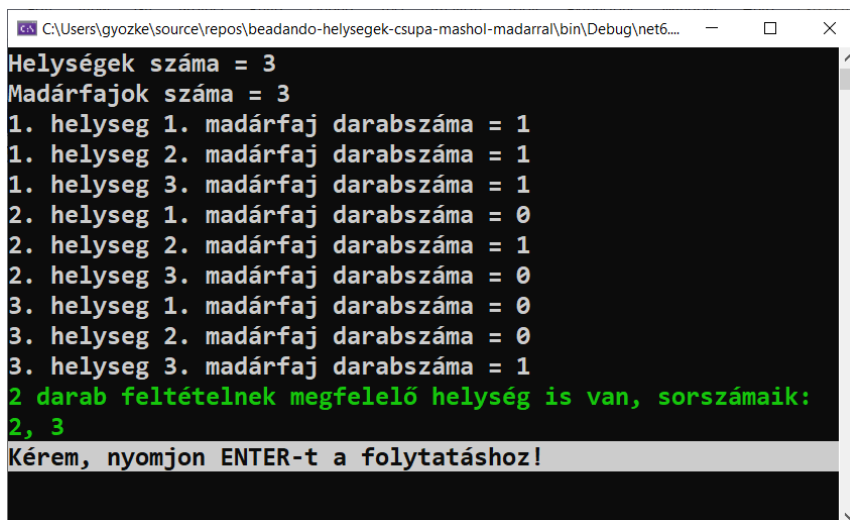
A fájl felépítésének a következő formai követelményei vannak. A fájl első sorában a helységek száma (*n*) és a madárfajok száma (*m*) van. A következő *n* sor mindegyikében *m* darabszám szerepel, közülük az *i*-edik sorban a *j*-edik szám az *i*-edik helységben a *j*-edik sorszámú fajból megfigyelt madarak száma. Például:

```
4 3
1 0 4
2 0 1
0 2 0
1 0 0
```

A program kimenete

A program kiírja azoknak a helységeknek a darabszámát és a sorszámaikat, ahol csak olyan madarat láttunk, amelyik valahol máshol is előfordul.

Minta bemenet és kimenet

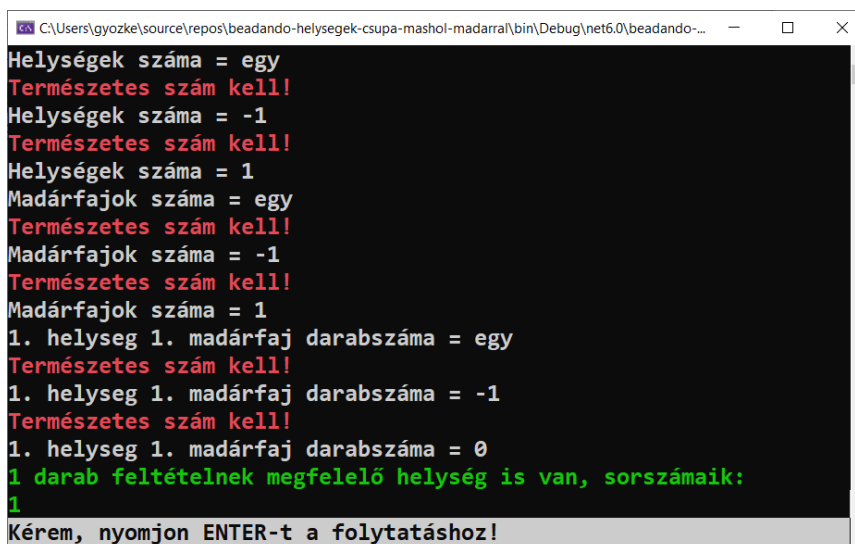


```
C:\Users\gyozke\source\repos\beadando-helysegek-csupa-mashol-madarra\bin\Debug\net6...
Helységek száma = 3
Madárfajok száma = 3
1. helyseg 1. madárfaj darabszáma = 1
1. helyseg 2. madárfaj darabszáma = 1
1. helyseg 3. madárfaj darabszáma = 1
2. helyseg 1. madárfaj darabszáma = 0
2. helyseg 2. madárfaj darabszáma = 1
2. helyseg 3. madárfaj darabszáma = 0
3. helyseg 1. madárfaj darabszáma = 0
3. helyseg 2. madárfaj darabszáma = 0
3. helyseg 3. madárfaj darabszáma = 1
2 darab feltételnek megfelelő helység 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\gyozke\source\repos\beadando-helysegek-csupa-mashol-madarra\bin\Debug\net6.0\beadando-...
Helységek száma = egy
Természetes szám kell!
Helységek száma = -1
Természetes szám kell!
Helységek száma = 1
Madárfajok száma = egy
Természetes szám kell!
Madárfajok száma = -1
Természetes szám kell!
Madárfajok száma = 1
1. helyseg 1. madárfaj darabszáma = egy
Természetes szám kell!
1. helyseg 1. madárfaj darabszáma = -1
Természetes szám kell!
1. helyseg 1. madárfaj darabszáma = 0
1 darab feltételnek megfelelő helység is van, sorszámaik:
1
Kérem, nyomjon ENTER-t a folytatáshoz!
```

Fejlesztői dokumentáció

Feladat

Helységek csupa máshol is előforduló madárfajjal

Az ország néhány helységében madármegfigyelést végeztünk. Mindegyikben megadtuk, hogy milyen fajú madárból hányat láttunk.

Készíts programot, amely megadja azokat a helységeket, ahol csak olyan madarat láttunk, amelyik valahol máshol is előfordult!

Tervezés

Specifikáció

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

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

Fv: $\text{vanmadár}: \mathbb{N} \rightarrow \mathbb{L}$,
 $\text{vanmadár}(i) = \text{VAN}(j=1..m, \text{mad}[i,j] > 0)$

Fv: $\text{másholis}: \mathbb{N} \times \mathbb{N} \rightarrow \mathbb{L}$,
 $\text{másholis}(i,j) = \text{VAN}(k=1..n, i \neq k \text{ és } \text{mad}[k,j] > 0)$

Fv: $\text{jó}: \mathbb{N} \rightarrow \mathbb{L}$,
 $\text{jó}(i) = \text{MIND}(j=1..m, \text{mad}[i,j] = 0 \text{ vagy } \text{másholis}(i,j))$

Ef: -

Uf: $(\text{db}, \text{helység}) = \text{KIVÁLOGAT}(i=1..n, \text{vanmadár}(i) \text{ és } \text{jó}(i), i)$

Visszavezetés

Kiválogatás

y	\sim	helység
$e..u$	\sim	$1..n$
$T(i)$	\sim	$\text{vanmadár}(i) \text{ és } \text{jó}(i)$
$f(i)$	\sim	i

Eldöntés (vanmadár)

i	\sim	j
$e..u$	\sim	$1..m$
$T(i)$	\sim	$\text{mad}[i,j] > 0$

(Optimista) eldöntés (jó)

i	\sim	j
$e..u$	\sim	$1..m$
$T(i)$	\sim	$\text{mad}[i,j] = 0 \text{ vagy } \text{másholis}(i,j)$

Eldöntés (másholis)

i	\sim	k
$e..u$	\sim	$1..n$
$T(i)$	\sim	$i \neq k \text{ és } \text{mad}[k,j] > 0$

Algoritmus

db:=0	
i=1..n	
vanmadár(i) és jó(i)	
true	false
db:=db+1	
helység[db]:=i	

vanmadár(i: Egész): Logikai
Vált j:Egész, van: Logikai

j:=1
j≤m és nem mad[i,j]>0
j:=j+1
van:=j≤m
vanmadár:=van

jó(i: Egész): Logikai
Vált j:Egész, mind: Logikai

j:=1
j≤m és (mad[i,j]=0 vagy másholis(i,j))
j:=j+1
mind:=j>m
jó:=mind

másholis(i,j:Egész): Logikai
Vált k:Egész, van: Logikai

k:=1
k≤n és nem(i≠k és mad[k,j]>0)
k:=k+1
van:=k≤n
másholis:=van

Fejlesztői környezet

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

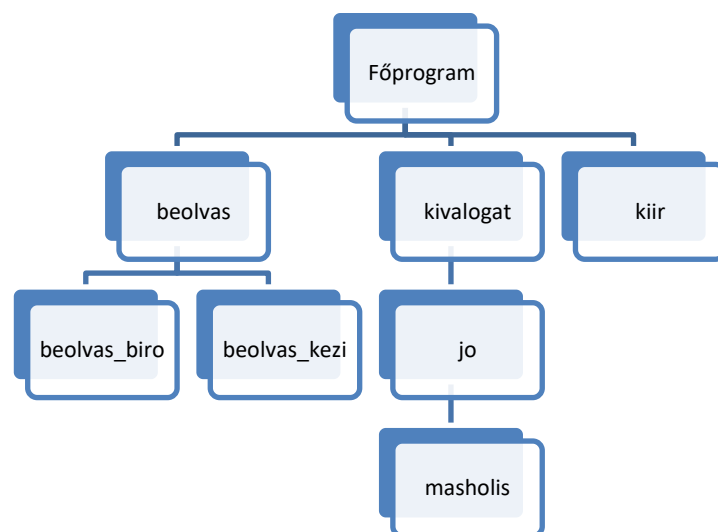
Forráskód

A teljes fejlesztői anyag –kicsomagolás után– az A1B2C3 nevű könyvtárban található meg. A fejlesztés során használt könyvtár-struktúra:

Állomány	Magyarázat
A1B2C3\bin\Release\netcoreapp3.1\A1B2C3.exe	futtatható kód (a futtatáshoz szükséges fájlokkal)
A1B2C3\obj\	mappa fordításhoz szükséges kódokkal
A1B2C3\Program.cs	C# forráskód
A1B2C3\teszt1.txt	teszt-bemeneti fájl ₁
A1B2C3\teszt2.txt	teszt-bemeneti fájl ₂
A1B2C3\teszt3.txt	teszt-bemeneti fájl ₃
A1B2C3\teszt4.txt	teszt-bemeneti fájl ₄
A1B2C3\teszt5.txt	teszt-bemeneti fájl ₅
A1B2C3\doksi\A1B2C3.docx	dokumentációk (ez a fájl)

Megoldás

Függvénystruktúra



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_madarra {
```

```

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.WriteLine("{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.WriteLine("{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,
sorszámuk:", 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();
}
}
}
}

```

Tesztelés

Érvényes tesztesetek

1. teszt eset: be1.txt

Bemenet – nincs helység, nincs madárfaj
0 0
Kimenet
0

2. teszt eset: be2.txt

Bemenet – 1 helység, 1 madárfaj, 1 darab
1 1
1
Kimenet
0

3. teszt eset: be3.txt

Bemenet – 1 helység, 1 madárfaj, nincs madár
...
Kimenet
...

4. *teszteset: be4.txt*

Bemenet – ...
...
Kimenet
...

5. *teszteset: be5.txt*

Bemenet – ...
...
...
Kimenet
...

Érvénytelen tesztesetek

Billentyűzetes bevétel esetén

6. *teszteset*

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

7. *teszteset*

Bemenet – Negatív 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