Programozási alapismeretek  
komplex beadandó feladat

Készítette: Csontos Dávid  
Neptun azonosító: Elaip4  
E-mail: csontos.david01@gmail.com

Kurzuskód: gyakorlat: 1 IK-19fszPAEG 1, előadás: ZDOF7J  
Gyakorlatvezető neve: Szalayné Tahi Zsuzsanna

**Tartalom**

[Felhasználói dokumentáció 3](#_Toc410323328)

[Feladat 3](#_Toc410323329)

[Futási környezet 3](#_Toc410323330)

[Használat 3](#_Toc410323331)

[A program indítása 3](#_Toc410323332)

[A program bemenete 3](#_Toc410323333)

[A program kimenete 4](#_Toc410323334)

[Minta bemenet és kimenet 4](#_Toc410323335)

[Hibalehetőségek 4](#_Toc410323336)

[Fejlesztői dokumentáció 5](#_Toc410323337)

[Feladat 5](#_Toc410323338)

[Specifikáció 5](#_Toc410323339)

[Fejlesztői környezet 5](#_Toc410323340)

[Forráskód 6](#_Toc410323341)

[Megoldás 6](#_Toc410323342)

[Programparaméterek 6](#_Toc410323343)

[Programfelépítés 7](#_Toc410323344)

[Függvénystruktúra 7](#_Toc410323345)

[Algoritmus 7](#_Toc410323346)

[A kód 9](#_Toc410323347)

[Tesztelés 12](#_Toc410323348)

[Érvényes tesztesetek 12](#_Toc410323349)

[Érvénytelen tesztesetek 14](#_Toc410323350)

[Fejlesztési lehetőségek 15](#_Toc410323351)

Felhasználói dokumentáció

Feladat

**Szempontok, amelyekben volt tökéletes kutya**

Egy N résztvevőjű kutyaszépségversenyen M különböző szempont szerint pontoznak minden kutyát. Minden szemponthoz adott egy maximális pontszám. Az összetett versenyből automatikusan kiesik az a kutya, amely valamelyik szempont szerint nem éri el a szempontonként megadott alsó ponthatárt – az adott szempontból sem értékelhető, más szempontokból viszont igen. A program megadja azokat a szempontokat, amelyekben volt tökéletes kutya!

Futási környezet

IBM PC, exe futtatására alkalmas, 32 vagy 64 bites operációs rendszer (pl. Windows 10). Konzol alkalmazás.

Használat

A program indítása

A program a Programozas\_komplex\_Beadando2021\_ELAIP4\bin\Debug\netcoreapp3.1\ Progkomplex\_Beadando.exe néven található a tömörített állományban. A Progkomplex\_Beadando.exe néven fájl kiválasztásával indítható.

A program bemenete

A program az adatokat a **billentyűzet**ről olvassa be a következő sorrendben:

|  |  |  |
| --- | --- | --- |
| # | Adat | Magyarázat |
| **1.** | *N* | A résztvevő kutyák száma (1≤N≤100). |
| **2.** | *M* | A verseny szempontjainak száma (1≤M≤100). |
| **3.** | *Maxpontszámok* | Az egyes szempontok maximális pontszámai, M mennyiségű adat beírása szükséges, szóközzel elválasztva. (1≤Maxpontszámok≤100). |
| **4.** | *MinPontszámok* | Az egyes szempontok minimális pontszámai, M mennyiségű adat beírása szükséges, szóközzel elválasztva. (1≤MinPontszámok≤MaxPontszámok). |
| **5.** | *KutyaMátrix* | A versenyben résztvevő kutyák és pontszámaik mátrixa. A *KutyaMátrix* hossza:M\*N |

A program kimenete

A program kiírja keresett szempontok számát és azok sorszámát növekvő sorrendben.

***Minta bemenet és kimenet***

A képen szöveg látható

Automatikusan generált leírás

Hibalehetőségek

A bemeneti egységeket a fenti mintának megfelelőlen kell megadni, az adatokat szóközzel kell tagolni. Ha a résztvevő kutyák száma, a verseny szempontjainak száma és a maximális pontszámok értéke nem esik 1 és 100 közé, akkor a program jelzi a hibát és új adatot kér. Valamint, ha a Minpontszámok nem esik az 1 és Maxpontszámok érték intervallum közé, akkor a program szintén hibát jelez és új adatot kér. Végül, ha kutyák számának és pontszámának mátrixa kevesebb vagy több mint N\*M, valamint ha a mátrix valamelyik eleme negatív, vagy több mint a Maxpontszámok azon indexedik eleme, akkor program szintén hibát jelzéssel és új adat kérésével reagál.

* + - 1. Mintafutás hibás bemeneti adatok esetén:

A képen szöveg látható

Automatikusan generált leírás

Fejlesztői dokumentáció

Feladat

**Szempontok, amelyekben volt tökéletes kutya**

Egy N résztvevőjű kutyaszépségversenyen M különböző szempont szerint pontoznak minden kutyát. Minden szemponthoz adott egy maximális pontszám. Az összetett versenyből automatikusan kiesik az a kutya, amely valamelyik szempont szerint nem éri el a szempontonként megadott alsó ponthatárt – az adott szempontból sem értékelhető, más szempontokból viszont igen. A program megadja azokat a szempontokat, amelyekben volt tökéletes kutya!

Specifikáció

**Bemenet:** N ∈ ℕ, M ∈ ℕ, ∈ , ∈ ,

**Kimenet:** SzempontokSzáma ∈ ℕ, ∈

**Előfeltétel:** 1≤N≤100, 1≤M≤100, 1≤Maxpontszámok≤100, 1≤MinPonstzámok≤MaxPontszámok, 0≤ *VersenyzóKutyákMátrix*<MaxPontszámok

**Utófeltétel:** ∀i(1≤i≤N): ∀j(1≤j≤M):

*VersenyzóKutyákMátrix[i,j]=MaxPontszámok[i]*

*i*

∀i(1≤i≤Szempontoksorzáma):

Szempontoksorzáma[i]!= Szempontoksorzáma[i-1]

SzempontokSzáma+=1

Fejlesztői környezet

IBM PC, A .csproj fájlban található. exe futtatására alkalmas operációs rendszer (pl. Windows 10) Visual Studio 2019 (Version 16.8.5) .Net fejlesztői környezet.

Forráskód

A fejlesztői anyag a Programozas\_komplex\_Beadando2021\_ELAIP4 nevű könyvtárban található. A könyvtár struktúrája az alábbi:

|  |  |
| --- | --- |
| **Állomány** | **Magyarázat** |
| Programozas\_komplex\_Beadando2021\_ELAIP4*\bin\Debug\netcoreapp3.1\* Progkomplex\_Beadando*.exe* | futtatható kód (a futtatáshoz szükséges fájlokkal) |
|  |  |
|  |  |
| Programozas\_komplex\_Beadando2021\_ELAIP4*\program.cs* | C# forráskód |
|  |  |
| Programozas\_komplex\_Beadando2021\_ELAIP4*\teszt1.txt* | teszt-bemenet1 |
| Programozas\_komplex\_Beadando2021\_ELAIP4*\teszt2.txt* | teszt-bemenet2 |
| Programozas\_komplex\_Beadando2021\_ELAIP4*\teszt3.txt* | teszt-bemenet3 |
| Programozas\_komplex\_Beadando2021\_ELAIP4*\teszt4.txt* | teszt-bemenet4 |
| Programozas\_komplex\_Beadando2021\_ELAIP4*\teszt5.txt* | teszt-bemenet5 |
| Programozas\_komplex\_Beadando2021\_ELAIP4*\hibasteszt1.txt* | hibasteszt-bemenet1 |
| Programozas\_komplex\_Beadando2021\_ELAIP4*\hibasteszt2.txt* | hibasteszt-bemenet2 |
| Programozas\_komplex\_Beadando2021\_ELAIP4*\hibasteszt3.txt* | hibasteszt-bemenet3 |
| Programozas\_komplex\_Beadando2021\_ELAIP4*\hibasteszt4.txt* | hibasteszt-bemenet4 |
| Programozas\_komplex\_Beadando2021\_ELAIP4*\hibasteszt5.txt* | hibasteszt-bemenet5 |
|  |  |
| Programozas\_komplex\_Beadando2021\_ELAIP4*\**Programozási\_alapismeretek\_komplex\_beadandó.docx* | dokumentáció |

Megoldás

Programparaméterek

* + - 1. Konstans

1. MaxN : **Egész**(100) [A versenyző kutyák maximális száma]  
   MaxM : **Egész**(100) [A verseny szempontok maximális száma]
2. MinN : **Egész**(1) [A versenyző kutyák minimális száma]  
   MinM : **Egész**(1) [A verseny szempontok minimális száma]

Típus

1. TMaximálisSzempontok = **Tömb**(1..MaxM:**Egész**)  
   TMinimálisSzempontok = **Tömb**(1..MaxM:**Egész**)

TMátrix= **Tömb**[1..MaxN,1..MaxM:Egész]

1. input= **Tömb**[1…M:**Egész**]
2. * + 1. Változó

N : **Egész**  
M : **Egész**  
MaxPontszámok : **TMaximálisSzempontok**

MinPontszámok : **TMinimálisSzempontok**

KutyaMátrix : **TMátrix**

SzempontokÉsSorszáma : **Tömb**[1..M:**Egész**]

DistinctSzempontokÉsSorszáma : **Tömb**[1..M:**Egész**]

Programfelépítés

A program által használt modulok (és helyük):

Program.cs   
Progkomplex\_Beadando.sln   
Progkomplex\_Beadando.csproj

* + 1. Függvénystruktúra

Főprogram

Kiírás

SzempontlistaFeltoltes

kutyaMatrixLetrehozas

TryParseMin

TryParseMaxok

Nvizsgalat

Mvizsgalat

* + 1. A teljes program algoritmusa

eredmeny: **Tömb**[1…M]

eredmeny2: **Tömb**[1…M]

sor≔ Tömb[1..M:Egész]

sorok≔ Tömb[1..M:Egész]

[főprogram:]

Nvizsgalat(elsosor, min, max)

Nvizsgalat(elsosor, min, max)

TryParseMaxok(input, M, maxPontszamok

TryParseMin(input2, M, Minpontszamok, Maxpontszamok)

kutyaMatrixLetrhezas(N, M, kutyaMatrix)

SzempontlistaFeltoltes(Maxpontszamok, kutyaMatrix)

kiírás(DistinctSzempontokÉsSorszáma, SzempontokSzama)

**Program vége.**

[alprogramok:]

Be: elsosor [elsosor∈[1..MaxN]Szöveg]

Be: N [N∈[1…MaxN]]

Be: M [N∈[1…MaxM]]

**Eljárás** Nvizsgalat (Tömb elsosor:Szöveg, **Változó** min:Egész, Változó max:Egész):

**Ciklus amíg** elsosor[0] <> Egész vagy N≤ min vagy max≤N

**Ciklus vége**

N:= elsosor[0]

**Eljárás vége**

**Eljárás** Nvizsgalat (Tömb elsosor:Egész, **Változó** min:Egész, Változó max:Egész):

**Ciklus amíg** elsosor[1] <> Egész vagy N≤ min vagy max≤N

M:= elsosor[1]

**Ciklus vége**

**Eljárás vége**

**Eljárás** TryParseMaxok(Tömb input:Egész, **Változó** M:Egész, Tömb MaxPontszámok: Egész)

**Változó** eredmeny≔ Tömb[1..M:Egész]

Be **Változó** sor≔ Tömb[1..M:Szöveg]

**Változó** i:Egész

**Ciklus** i≔1-től M-ig

**Ha** sor hossza <M vagy eredmeny[i] <> Egész vagy eredmeny [i]≤1 vagy eredmeny[i]> 100

**akkor** TryParseMaxok≔ hamis

**különben** Maxpontszámok≔eredmeny

**Elágazás** **vége**

**Ciklus** **vége**

**Eljárás vége**

**Eljárás** TryParseMin(Tömb input2:Egész, Változó M:Egész, Tömb MaxPontszámok: Egész, Tömb MinPontszámok: Egész)

Be **Változó** sorok≔ Tömb[1..M:Egész]

**Változó** eredmeny2≔ Tömb[1..M:Egész]

**Változó** i:Egész

**Ciklus** i≔1-től M-ig

**Ha** eredmeny2[i] <> Egész vagy eredmeny2[i]<1 vagy eredmeny2[i]≥ MaxPontszámok [i]

**akkor** TryParseMin≔ hamis

**különben** MinPontszámok≔eredmeny

**Elágazás vége**

**Ciklus vége**

**Eljárás vége**

**Eljárás** kutyaMatrixLetrehozas( **Változó** N:Egész, **Változó** M:Egész, kutyaMátrix Típus TMátrix)

**Változó** i,j:Egész

**Ciklus** i≔1-től N-ig

Ciklus j≔1-től M-ig

Be: kutyaMatrix[i,j]

**Ha** kutyaMatrix[i, j]>maxpontszamok[j] vagy kutyaMatrix[i, j] < minpontszamok[j]

**Akkor** Ki(”Hibás Bemenet ”

kutyaMatrixLetrehozas( **Változó** N:Egész, **Változó** M:Egész, kutyaMátrix Típus TMátrix)

**Elágazás vége**

**Ciklus vége**

**Ciklus vége**

**Eljárás vége**

**Eljárás** SzempontListaFeltoltes(Tömb maxpontszamok: Egész, kutyaMátrix **Típus** TMátrix, Tömb SzempontokÉsSorszama: Egész)

**Változó** i,j:Egész db≔0

**Ciklus** i≔1-től N-ig

**Ciklus** j≔1-től M-ig

**Ha** kutyaMatrix[j, i] = mapontszamok[i]

**akkor** db≔ db+1 SzempontokÉsSorszáma[db]= i

**Elágazás vége**

**Ciklus vége**

**Ciklus vége**

**Eljárás Vége**

**Eljárás** kiírás (Tömb DistinctSzempontokÉsSorszáma: Egész, Változó SzempontokSzáma: Egész)

Ki: SzempontokSzáma

Ki: DistinctSzempontokÉsSorszáma

**Eljárás vége**

**Program vége**

* + 1. A kód

A Program.cs fájl tartalma:

/// Név: Csontos Dávid

/// Neptun-kód: Elaip4

/// E-mail: Csontos.david01@gmail.com

using System;

using System.Collections.Generic;

using System.Linq;

namespace Programozasi\_alapismeretek\_Komplex\_Beadando

{

class Program

{

static void Main(string[] args)

{

// Versenyző kutyák és a versenyszempontok bekérése, egy sorban szóközzel elválasztva.

Console.WriteLine("Adja meg a kutyaszépség versenyen résztvevő kutyák számát, majd az értékelési szempontok számát egy sorban, szóközzel elválasztva!");

string[] elsosor = Console.ReadLine().Split(' ');

int N = 0;

int M = 0;

int min = 0;

int max = 100;

// Függvény, amely ellenőrzi, hogy a beírt adatokat átlehet-e int-té konvertálni, valamint, hogy 1 és 100 között vannak-e.

// Ha hibás valamelyik adat, akkor a program jelzi a felhasználónak és új adatot kér be.

N = Nvizsgalat(elsosor, min, max);

// Függvény, amely ellenőrzi, hogy a beírt adatokat át lehet-e int-té konvertálni, valamint, hogy 1 és 100 között vannak-e.

// Ha hibás valamelyik adat, akkor a program jelzi a felhasználónak és új adatot kér be.

M = Mvizsgalat(elsosor, min, max);

// A versenyszempontok maximális értékének bekérése.

Console.WriteLine("Adja meg az egyes versenyszempontok maximális értékét.");

int[] maxpontszamok = new int[M];

string uzenet = $"Adjon meg annyi számot, mint amennyi a versenyszempontok száma, {M}, az értékeket szóközzel kell elválasztani.\n";

Console.Write(uzenet);

// Függvény, amely ellenőrzi, hogy a beírt adatokat átlehet-e int-té konvertálni, valamint, hogy 1 és 100 között vannak-e.

// Ha hibás valamelyik adat, akkor a program jelzi a felhasználónak és új adatot kér be.

while (!TryParseMaxok(Console.ReadLine(), M, out maxpontszamok))

{

Console.WriteLine("Helytelen bemenet próbálja újra...");

}

// A versenyszempontok minimális értékének bekérése.

Console.WriteLine("Adja meg az egyes versenyszempontok minimális értékét.");

int[] minpontszamok = new int[M];

string uzenet2 = $"Adjon meg annyi számot, mint amennyi a versenyszempontok száma, azaz {M}, az értékeket szóközzel kell elválasztani.\nÜgyeljen arra, hogy az egyes szempontok maxpontszámánál ne adjon többet.\n";

Console.Write(uzenet2);

// Függvény, amely ellenőrzi, hogy a beírt adatokat át lehet-e int-té konvertálni, valaminthogy 1 és 100 között vannak-e.

// Ha hibás valamelyik adat, akkor a program jelzi a felhasználónak és új adatot kér be.

while (!TryParseMin(Console.ReadLine(), M, maxpontszamok, out minpontszamok))

{

Console.WriteLine("Helytelen bemenet próbálja újra...");

}

// Kutyák mátrixának és a szempontok listájának létrehozása.

int[,] kutyaMatrix;

kutyaMatrix = new int[N, M];

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

// Kutya mátrix adatainak bekérése.

Console.WriteLine("Adja meg a kutyák és a pontszámaok mátrixát, a sorok kutyákat, míg az oszlopok a pontszámokat jelölik");

kutyaMatrixLetrehozas(N, M, kutyaMatrix);

// A program végig megy a mátrix sorain és oszlopain, és ha a maxpontszámok tömb valamelyik indexe megegyezik a mátrix azon indexedik elemével,

// akkor azt az indexet hozzáadja a SzempontokSzamaÉsSorszáma listához. Mivel az index 0-tól indul, ezért hozzá kell adnunk plusz 1-et, hogy a tényleges sorszámot kapjuk.

SzempontlistaFeltoltes(maxpontszamok, kutyaMatrix, SzempontokSzamaÉsSorszáma);

// Mivel az előző ciklus kétszer adja hozzá ugyanazt az indexet, ezért a Distinct metódussal kiválasszuk a lista csak azon tagjait, amelyek különbözőek.

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

DistinctSzempontokSzamaÉsSorszáma = SzempontokSzamaÉsSorszáma.Distinct().ToList();

int SzempontokSzama = DistinctSzempontokSzamaÉsSorszáma.Count;

// A különböző versenyszempontok számának és a különböző versenyszempontok listájának tagjainak kiírása.

kiíras(DistinctSzempontokSzamaÉsSorszáma, SzempontokSzama);

}

static int Nvizsgalat(string[] elsosor, int min, int max)

{

int N;

while (!int.TryParse(elsosor[0], out N) || N <= min || max <= N)

{

Console.WriteLine("Az első adat helytelen, írja be csak az első adatot...");

elsosor[0] = Console.ReadLine();

}

return N;

}

static int Mvizsgalat(string[] elsosor, int min, int max)

{

int M;

while (!int.TryParse(elsosor[1], out M) || M <= min || max <= M)

{

Console.WriteLine("A második adat helytelen, írja be csak a második adatot...");

elsosor[1] = Console.ReadLine();

}

return M;

}

public static bool TryParseMaxok(string input, int M, out int[] maxpontszamok)

{

maxpontszamok = default;

string[] sor = input.Split(" ");

int[] eredmeny = new int[M];

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

{

if (sor.Length<M || !int.TryParse(sor[i], out eredmeny[i]) || (eredmeny[i] < 1 || eredmeny[i] > 100))

{

return false;

}

}

maxpontszamok = eredmeny;

return true;

}

public static bool TryParseMin(string input2, int M, int[] maxpontszamok, out int[] minpontszamok)

{

minpontszamok = default;

string[] sorok = input2.Split(" ");

int[] eredmeny2 = new int[M];

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

{

if (sorok.Length < M || !int.TryParse(sorok[i], out eredmeny2[i]) || (eredmeny2[i] < 1 || eredmeny2[i] ≥ maxpontszamok[i] ))

{

return false;

}

}

minpontszamok = eredmeny2;

return true;

}

static void kutyaMatrixLetrehozas(int N, int M, int[,] kutyaMatrix, int [] maxpontszamok, int[] minpontszamok)

{

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

{

string[] bemenetString = Console.ReadLine().Split(' ');

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

{

kutyaMatrix[i, j] = int.Parse(bemenetString[j]);

if (kutyaMatrix[i, j]>maxpontszamok[j] || kutyaMatrix[i, j] < minpontszamok[j])

{

Console.WriteLine("Helytelen bemenet próbáld újra...");

kutyaMatrixLetrehozas(N, M, kutyaMatrix, maxpontszamok, minpontszamok);

}

}

}

}

static void SzempontlistaFeltoltes(int[] maxpontszamok, int[,] kutyaMatrix, List<int> SzempontokSzamaÉsSorszáma)

{

for (int i = 0; i < kutyaMatrix.GetLength(1); i++)

{

for (int j = 0; j < kutyaMatrix.GetLength(0); j++)

{

if (kutyaMatrix[j, i] == maxpontszamok[i])

{

SzempontokSzamaÉsSorszáma.Add(i + 1);

}

}

}

}

static void kiíras(List<int> DistinctSzempontokSzamaÉsSorszáma, int SzempontokSzama)

{

Console.Write(SzempontokSzama + " ");

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

{

Console.Write(DistinctSzempontokSzamaÉsSorszáma[i] + " ");

}

}

}

}

Tesztelés

* + 1. Érvényes tesztesetek
       1. teszteset: teszt1.txt

|  |
| --- |
| Bemenet |
| N = 6  M= 8  MaxPontszámok = 9 9 9 9 9 9 9 9  MinPontszámok = 5 5 5 5 5 5 5 5  KutyaMatrix =  8 4 6 6 6 6 6 6  7 5 7 6 6 6 6 5  6 6 6 5 5 5 5 6  8 6 8 7 7 7 7 6  8 6 6 6 6 6 6 6  8 6 6 6 6 6 6 5 |
| Kimenet |
| 0 |

1. teszteset: teszt2.txt

|  |
| --- |
| Bemenet |
| N = 6  M= 8  MaxPontszámok = 9 9 9 9 9 9 9 9  MinPontszámok = 5 5 5 5 5 5 5 5  KutyaMatrix =  8 5 6 6 6 6 6 6  8 6 6 6 9 6 6 6  8 6 8 7 7 7 7 6  6 6 6 6 6 6 6 6  8 6 6 6 9 4 6 6  8 6 6 6 6 6 6 6 |
| Kimenet |
| 1 5 |

1. teszteset: teszt3.txt

|  |
| --- |
| Bemenet |
| N = 4  M= 5  MaxPontszámok = 10 7 9 12 15  MinPontszámok = 1 3 2 4 5  KutyaMatrix =  10 7 9 12 15  2 3 4 6 6  10 6 3 4 6  3 5 7 8 9 |
| Kimenet |
| 5 1 2 3 4 5 |

1. teszteset: teszt4.txt

|  |
| --- |
| Bemenet |
| N = 3  M = 3  MaxPontszámok = 100 100 100  MinPontszámok = 1 1 1  kutyaMatrix =  88 56 42  43 34 54  1 32 100 |
| Kimenet |
| 1 3 |

1. teszteset: teszt5.txt

|  |
| --- |
| Bemenet |
| N = 3  M = 3  MaxPontszámok = 10 10 10  MinPontszámok = 1 1 1  kutyaMatrix =  1 1 1  6 6 6  10 10 3 |
| Kimenet |
| 2 1 2 |

* + 1. Érvénytelen tesztesetek
       1. 6. teszteset: hibásteszt1

|  |
| --- |
| Bemenet – (Hiba N kisebb mint 1) |
| N = 0  … |
| Kimenet |
| Helytelen bemenet próbálja újra... |

* + - 1. 7. teszteset: hibásteszt2

|  |
| --- |
| Bemenet – (M nem Egész típusú) |
| N = 3  M = cydfa  … |
| Kimenet |
| Helytelen bemenet próbálja újra... |

* + - 1. 8. teszteset: hibásteszt3

|  |
| --- |
| Bemenet – (Hiba MaxPontszámok kevesebb mint M) |
| N = 6  M = 10  MaxPontszámok = 10 10 10 10 10 10 10 |
| Kimenet |
| Helytelen bemenet próbáld újra... |

* + - 1. 9. teszteset: hibásteszt4

|  |
| --- |
| Bemenet – (Hiba MinPontszámok egyik tagja kevesebb mint 1) |
| N = 5  M = 5  MaxPontszámok = 10 10 10 10 10 10  MinPontszámok = 0 2 5 7 3 1 |
| Kimenet |
| Helytelen bemenet próbáld újra... |

* + - 1. 10. teszteset: hibásteszt5

|  |
| --- |
| Bemenet – (Hiba KutyaMatrix egyik tagja több mint a MaxPontszámok azon indexedik tagja) |
| N = 3  M = 3  MaxPontszámok = 5 5 5  MinPontszámok = 2 2 2  KutyaMatrix =  3 3 3  3 3 3  3 3 6 |
| Kimenet |
| Helytelen bemenet próbáld újra... |

**Fejlesztési lehetőségek**

* + - * 1. A kutyák mátrixának beolvasásakor a szöveg típusú bemenetnél is új adatok kérése, nem pedig kilépés a programból.
        2. A program kimenetében megjelenhetne azon kutyák száma és sorszáma, ahol a minimális pontszámot értek el.
        3. A program összeadhatná a kutyák pontszámait és kiírhatná a versenyző kutyák sorrendjét.
        4. A program kiírhatná a versenyző kutyák pontszámainak átlagát.
        5. A program további adatokat kérhetne be, pl. kutyák fajtája, súlya mérete stb., amik alapján újabb statisztikai adatokat lehetne kiírni.