**„Programozás”  
komplex beadandó feladat**

Készítette: *Restye János Barnabás*Neptun-azonosító: *F8U9I2*E-mail: f8u9i2@inf.elte.hu

Kurzuskód: K81DNC-PROGGyakorlatvezető neve: *Klettner Péter Zsolt*

2025. január 12.

Tartalom

[Felhasználói dokumentáció 3](#_Toc186797335)

[Feladat 3](#_Toc186797336)

[Futási környezet 3](#_Toc186797337)

[Használat 3](#_Toc186797338)

[A program indítása 3](#_Toc186797339)

[A program használata billentyűzetről való bevitel esetén 3](#_Toc186797340)

[A program használata fájlból való bevitel esetén 3](#_Toc186797341)

[A program kimenete 4](#_Toc186797342)

[Minta bemenet és kimenet 4](#_Toc186797343)

[Hibalehetőségek 4](#_Toc186797344)

[Fejlesztői dokumentáció 5](#_Toc186797345)

[Feladat 5](#_Toc186797346)

[Tervezés 5](#_Toc186797347)

[Specifikáció 5](#_Toc186797348)

[Visszavezetés 5](#_Toc186797349)

[Algoritmus 6](#_Toc186797350)

[Fejlesztői környezet 6](#_Toc186797351)

[Forráskód 6](#_Toc186797352)

[Megoldás 7](#_Toc186797353)

[Függvénystruktúra 7](#_Toc186797354)

[A kód 7](#_Toc186797355)

[A kód (magas szintű függvényekkel) 10](#_Toc186797356)

[Tesztelés 11](#_Toc186797357)

[Automatikus tesztek (Bíró) 11](#_Toc186797358)

[Automatikus tesztek (Bíró, magas szintű függvényekkel) 12](#_Toc186797359)

[Érvényes tesztesetek 13](#_Toc186797360)

[Érvénytelen tesztesetek 13](#_Toc186797361)

[Fejlesztési lehetőségek 14](#_Toc186797362)

2. Felhasználói dokumentáció
   1. Feladat

**Település valamikor maximális hőmérséklettel**

* 1. A meteorológiai intézet az ország N településére adott M napos időjárás előrejelzést, az adott településen az adott napra várt legmagasabb hőmérsékletet.
  2. A program megadja azokat a településeket, amelyeken előfordul valamelyik napi előrejelzések maximuma.
  3. 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 telepules\_valamikor\_maximalis\_homerseklettel \szokásos\bin\Release\telepules\_valamikor\_maximalis\_homerseklettel.exe és a telepules\_valamikor\_maximalis\_homerseklettel \magas\_szintű\bin\Release\magas\_szintu\_ck.exe néven található a tömörített állományban.

* + 1. A program használata billentyűzetről való bevitel esetén

Az telepules\_valamikor\_maximalis\_homerseklettel.exe vagy a magas\_szintu\_ck.exe fájl elindításával a program az adatokat a **billentyűzet**ről olvassa be a következő sorrendben:

|  |  |  |
| --- | --- | --- |
| # | Adat | Magyarázat |
| **1.** | *Települések száma (n)* | 1 és 1000 közé eső szám |
| **2.** | *napok száma (m)* | 1 és 1000 közé eső szám |
| **3.** | *1. településben az 1. napon mért hőfok* | -50 és 50 közé eső szám |
| **4.** | *1. településben az 2. napon mért hőfok* | -50 és 50 közé eső szám |
| **…** | *…* | -50 és 50 közé eső szám |
|  | *1. településben az m. napon mért hőfok* | -50 és 50 közé eső szám |
|  | *2. településben az 1. napon mért hőfok* | -50 és 50 közé eső szám |
|  | *…* | -50 és 50 közé eső szám |
|  | *n. településben az m. napon mért hőfok* | -50 és 50 közé eső szám |

* + 1. 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:

telepules\_valamikor\_maximalis\_homerseklettel.exe < teszt1.txt

vagy

magas\_szintu\_ck.exe < teszt1.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 darab szám szerepel, közülük az i-edik sorban a j-edik szám az i-edik településen a j-edik sorszámú napon mért hőmérséklet száma. Például:

3 5

10 15 12 10 10

11 11 11 11 20

12 16 16 16 20

* + 1. A program kimenete

A program kiírja azoknak a településeknek a darabszámát és a sorszámait, amelyiken előfordul valamelyik napi előrejelzések maximuma.

* + 1. Minta bemenet és kimenet

A képen szöveg, képernyőkép, szoftver, Multimédiás szoftver látható

Automatikusan generált leírás

* + 1. 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.

* + - 1. Mintafutás hibás bemeneti adatok esetén:

A képen szöveg, képernyőkép, szoftver, Multimédiás szoftver látható

Automatikusan generált leírás

1. Fejlesztői dokumentáció
   1. Feladat

**Település valamikor maximális hőmérséklettel**

* 1. A meteorológiai intézet az ország N településére adott M napos időjárás előrejelzést, az adott településen az adott napra várt legmagasabb hőmérsékletet.
  2. A program megadja azokat a településeket, amelyeken előfordul valamelyik napi előrejelzések maximuma.
  3. Tervezés

* + 1. [Specifikáció](https://progalap.elte.hu/specifikacio/?data=H4sIAAAAAAAACo2RwUrDQBCGX%2BVnT4mObSZFDwMpVFARtR7UIqQ5pE1KV8xGklgs4sGb7%2BWb%2BCSyiWlr6aGH3dn9%2F51%2FvpB3Vb6kUz3T07jSuVGiTlOB%2Bfn6GhKypsxzW0PudAxxp5NF42Js7mJBFr81nhWutCCZNB3LtiFCtwtj7fOFYJIaky5iIxjiqI9rsgaAleGUeeEGo8HQeQrsJDs7LPOCniIEzTjX9pzNBKYfMGKTIEM%2FYKs%2BzAQONa%2BCm8Gjo22IOWhiHI0jsItEL5DhEExopSxPGilybWJL5SQTwtINri5H35%2FXtxeD%2B79EWgNrl6BdRapKy6pUEr6rJK5iJcoIejYpExzbOs8FYRsdskfgYwL7hPrsRbQ2mbC5%2FH%2Bm7ThZL9%2Br%2F0e91d8u8D17SSYC3x6WgtAn9CJFysRZqkQpUkVavj5XSviDdjH3djAT6hjaUvwdyjZSryXiloj3wuE%2FHF7hhBxtBvP%2BwdHHLxc92unuAgAA)

**Be:** n∈N, m∈N, ho∈N[1..n,1..m]

**Sa:** maxho∈N

**Ki:** db∈N, y∈N[1..] // n

**Fv:** bennevan: N -> L,

    bennevan(sor)=**VAN**(j=1..m, ho[sor,j] = maxho)

**Ef:** n>=1 and m >=1

**Uf:** (,maxho)=**MAX**(i=1..n\*m, ho[(i - 1) div m + 1, (i - 1) mod m + 1]) and

    (db, y)=**KIVÁLOGAT**(i=1..n, bennevan(i), i)

***Visszavezetés***

*Maximum kiválasztás*

maxert ~ helység

maxind ~ -

e..u ~ 1..n\*m

f(i) ~ ho[(i – 1) div m + 1, (i – 1) mod m + 1]

*Eldöntés (bennevan (sor))*

van ~ van

i ~ j

e..u ~ 1..m

T(i) ~ ho[sor,j] = maxho

*Kiválogatás*

db ~ db

i ~ i

e..u ~ 1..n

T(i) ~ bennevan(i)

y ~ y

* + 1. [Algoritmus](https://progalap.elte.hu/stuki/?data=H4sIAAAAAAAACrVY3VLiSBR%2BldSxakrXXiewQST%2BVMnoqKOijC5r4XDRSTokGLohCfhXPMxe7vU%2Bgi%2B21YE%2B6YxSC6vLDV2HzvfT5%2BTkhGcIPbDB3KhVrNpWxdqyrJJVK29ZQCAQYxafeGDzURQRSFjE3JR5eYQLjyVgP8PBaTyklbumXGd4GCAwoDHjaX5REHoe42D7NEoYgfRxwMCGhA1HjLsMCLhBGHkx4ydeAvYt3P1xY5ln9SEQqB07kdcrRdCZkDyuOLWNOaeuZA5zSlPWZzwFAil7SMGGPn0IhGHvGoG4LRGj1IEJAcZ65%2Bftsy%2FIh4H3e%2FxSi0xuNdtAoNTYNKtWdJ15xLji1DZqHjUlC3v0HGnQlM66pnd2chRfIwsGCiyobHl%2F7cbo5lvX%2FJ6ZQhxFpwEvY%2Bo%2BCKOMS3AvTEPBwYZwt7SxwYGAI7zHq5mek4KnCYHmqSg1RqMyKsBAQQGqXt5wlz6Nfm8cUiBwXmPNh3L1KfM%2Bji9a4urxFJkxUGAeltpW9fCmPpeZCzGQTnCjwtOu%2FBgnKFDKRxxFpwFrdFr9vE0X%2Bj%2FlzWGcszHlq%2BGaLNF4xOox5W5QSKGWpQzujS1oX5a1ygGWdZ4UTayGuuTN4znGulGSVJhjRYWBd1M93npOR7KFn7iTDLYlnTi4%2FJpuDo6QDgPvb0Wt6%2F3kgtcDIBCXh32nftcFAuLkIL6phjdZFeAWRa9do7nVRC3sdkx51ndpP0y2jZ5cZwd8fRO4R23mIiUGCpSa4mV9C%2FMp8t1BM3OIOIpOA17G4VstirO%2BIW1S7hk941OUbu8a%2Fbc6FlqUtewETd8PeV7LKlAUpEws778RfqMPqkW3Lr7v1zdHVp5hFfgoOud%2BeHQ4cFhGhzhYzzmwRqflfMG2EojbRMSk1zF2jeyhPr%2B5aAc6r7ngIUxIflxKMgaKnTBHXfYeCLv0STLhSSkmDBSYtAwtzJTdXz1j3dhJBpQbSfoYsd0f4IpIxLYRd53VcqVCDJMY5tr2D9gr7XyWG%2FekrM023%2BeNagtlYaAgC0e25Uuknp5eHgj%2FPisRxFF0GvAy096coWFdjg2%2FvHkboq8Jga1glMYXzEEVGCioQOXLm7448a9K3VZrOuI6lCbDr2E%2B4qpAcVIS9cvj3lHwr9MCbsQRNr%2FyY%2BSjQCkfcRSdBqzRaYWz4G09nc9l65QT%2Bmpo%2FGqU1gwvHBt9%2BUQmhgr1hTcNLVjgHVXgc9uElvB5bQIPdULydKojwEDhCDTU%2F%2FSS8n8cwUR76mNfVoF3P%2BjVwCcNjCmHyYRAksYjNxXdmPZlMT0Dp315%2BTnNGuiYxiF1IvnGCa2XP6NUPL38veOF473pZBTah92Xv5Knnc8yJucXIV51hSQQ960cSGZ5Gjxm1GPxTPmEIPk8YnshZu0l5hXJm2I0ZpyJExHPCNbsM9EN72i4iKre7CIylaX9MqZcAb2WrGVTl5yf1M%2BKO2RaTdk%2FAdP03sKK75umaQKBFfnty1t4xfddjDlmBVd%2Btqqav5k0W7mM%2BltV6EyrpU7du24sRtybATtlR47QK%2FJbATNXxVjVpWrlZyvPck1WlnDyD4tXcK7ruhmr67oKzi%2BrGPM9dx6cVrCvUOVn5nkhfN%2Bv1aa%2FmuaWW6vIFWPM8jb1E5lM%2FgGaGC1ZsREAAA%3D%3D)[[1]](#footnote-1)

A képen szöveg, képernyőkép, Betűtípus, szám látható

Automatikusan generált leírás

A képen szöveg, képernyőkép, Betűtípus, szám látható

Automatikusan generált leírás

A képen szöveg, képernyőkép, Betűtípus, szám látható

Automatikusan generált leírás

* 1. Fejlesztői környezet

IBM PC, exe futtatására alkalmas operációs rendszer (pl. Windows 11 Home). Visual Studio 2022 (Version 17.12.2) fejlesztői környezet.

* 1. Forráskód

A teljes fejlesztői anyag –kicsomagolás után– az telepules\_valamikor\_maximalis\_homerseklettel nevű könyvtárban található meg. A fej­lesztés során használt könyvtár-struktúra:

|  |  |
| --- | --- |
| Állomány | Magyarázat |
| telepules\_valamikor\_maximalis\_homerseklettel\szokásos\bin\Release\net8.0\telepules\_valamikor\_maximalis\_homerseklettel.exe | futtatható kód (a futtatáshoz szükséges fájlokkal) (szokásos) |
| telepules\_valamikor\_maximalis\_homerseklettel\magas\_szintű\bin\Release\net8.0\magas\_szintu\_ck.exe | futtatható kód (a futtatáshoz szükséges fájlokkal) (magas szintű) |
| telepules\_valamikor\_maximalis\_homerseklettel\szokásos\obj\ | mappa fordításhoz szükséges kódokkal (szokásos) |
| telepules\_valamikor\_maximalis\_homerseklettel\magas\_szintű\obj\ | mappa fordításhoz szükséges kódokkal (magas szintű) |
|  |  |
| telepules\_valamikor\_maximalis\_homerseklettel\szokásos\Program.cs | C# forráskód (szokásos) |
| telepules\_valamikor\_maximalis\_homerseklettel\magas\_szintű\Program.cs | C# forráskód (magas szintű) |
|  |  |
| telepules\_valamikor\_maximalis\_homerseklettel\szokásos\bin\Release\net8.0\teszt1.txt | teszt-bemeneti fájl1 (szokásos) |
| telepules\_valamikor\_maximalis\_homerseklettel\szokásos\bin\Release\net8.0\teszt2.txt | teszt-bemeneti fájl2 (szokásos) |
| telepules\_valamikor\_maximalis\_homerseklettel\szokásos\bin\Release\net8.0\teszt3.txt | teszt-bemeneti fájl3 (szokásos) |
| telepules\_valamikor\_maximalis\_homerseklettel\szokásos\bin\Release\net8.0\teszt4.txt | teszt-bemeneti fájl4 (szokásos) |
| telepules\_valamikor\_maximalis\_homerseklettel\szokásos\bin\Release\net8.0\teszt5.txt | teszt-bemeneti fájl5 (szokásos) |
|  |  |
| telepules\_valamikor\_maximalis\_homerseklettel\magas\_szintű\bin\Release\net8.0\teszt1.txt | teszt-bemeneti fájl1 (magas szintű) |
| telepules\_valamikor\_maximalis\_homerseklettel\magas\_szintű\bin\Release\net8.0\teszt2.txt | teszt-bemeneti fájl2 (magas szintű) |
| telepules\_valamikor\_maximalis\_homerseklettel\magas\_szintű\bin\Release\net8.0\teszt3.txt | teszt-bemeneti fájl3 (magas szintű) |
| telepules\_valamikor\_maximalis\_homerseklettel\magas\_szintű\bin\Release\net8.0\teszt4.txt | teszt-bemeneti fájl4 (magas szintű) |
| telepules\_valamikor\_maximalis\_homerseklettel\magas\_szintű\bin\Release\net8.0\teszt5.txt | teszt-bemeneti fájl5 (magas szintű) |
|  |  |
| telepules\_valamikor\_maximalis\_homerseklettel\szokásos\bin\Release\net8.0\rosszteszt1.txt | érvénytelen teszt-bemeneti fájl1 (szokásos) |
| telepules\_valamikor\_maximalis\_homerseklettel\szokásos\bin\Release\net8.0\rosszteszt2.txt | érvénytelen teszt-bemeneti fájl2 (szokásos) |
| telepules\_valamikor\_maximalis\_homerseklettel\szokásos\bin\Release\net8.0\rosszteszt3.txt | érvénytelen teszt-bemeneti fájl3 (szokásos) |
|  |  |
| telepules\_valamikor\_maximalis\_homerseklettel\magas\_szintű\bin\Release\net8.0\rosszteszt1.txt | érvénytelen teszt-bemeneti fájl1 (magas szintű) |
| telepules\_valamikor\_maximalis\_homerseklettel\magas\_szintű\bin\Release\net8.0\rosszteszt2.txt | érvénytelen teszt-bemeneti fájl2 (magas szintű) |
| telepules\_valamikor\_maximalis\_homerseklettel\magas\_szintű\bin\Release\net8.0\rosszteszt3.txt | érvénytelen teszt-bemeneti fájl3 (magas szintű) |
|  |  |
| telepules\_valamikor\_maximalis\_homerseklettel\doksi\telepules\_valamikor\_maximalis\_homerseklettel.docx | dokumentációk (ez a fájl) |

* 1. Megoldás
     1. Függvénystruktúra

A kód

A Program.cs fájl tartalma:

1. /\*
2. Készítette: Restye János Barnabás
3. Neptun: F8U9I2
4. E-mail: f8u9i2@inf.elte.hu
5. Feladat: települések, amelyeken előfordul valamelyik napi előrejelzések maximuma
6. \*/
8. using System;
10. namespace telepules\_valamikor\_maximalis\_homerseklettel {
11. internal class Program {
12. static void Main (string[] args) {
13. // deklaralas
14. int maxho, db;
15. int[,] ho;
16. int[] y;
18. // beolvasas
19. ho = beolvas();
21. // megoldas
22. maxho = max(ho);
23. (db, y) = kivalaszt(ho, maxho);
25. // kiiras
26. kiir(db, y);
27. }
29. static int[,] beolvas () {
30. if (Console.IsInputRedirected) {
31. return beolvas\_biro();
32. } else {
33. return beolvas\_kezi();
34. }
35. }
36. static int[,] beolvas\_biro () {
37. string[] line = Console.ReadLine().Split(' ');
38. int n = int.Parse(line[0]);
39. int m = int.Parse(line[1]);
41. int[,] ho = new int[n + 1, m + 1];
43. for (int i = 1; i <= n; ++i) {
44. line = Console.ReadLine().Split(' ');
45. for (int j = 1; j <= line.Length; ++j) {
46. int.TryParse(line[j - 1], out ho[i, j]);
47. }
48. }
49. return ho;
50. }
52. static int[,] beolvas\_kezi () {
53. int n, m;
54. bool jo;
55. do {
56. Console.ResetColor();
57. Console.Write("Települések száma = ");
58. jo = int.TryParse(Console.ReadLine(), out n) && n >= 1 && n <= 1000;
59. if (!jo) {
60. Console.ForegroundColor = ConsoleColor.Red;
61. Console.WriteLine("A települések számának 1 és 1000 közé kell esnie!");
62. }
63. } while (!jo);
64. do {
65. Console.ResetColor();
66. Console.Write("Napok száma = ");
67. jo = int.TryParse(Console.ReadLine(), out m) && m >= 1 && m <= 1000;
68. if (!jo) {
69. Console.ForegroundColor = ConsoleColor.Red;
70. Console.WriteLine("A napok számának 1 és 1000 közé kell esnie!");
71. }
72. } while (!jo);
74. int[,] ho = new int[n + 1, m + 1];
75. for (int i = 1; i <= n; ++i) {
76. for (int j = 1; j <= m; ++j) {
77. do {
78. Console.ResetColor();
79. Console.Write("{0}. település {1}. nap hőmérséklete = ", i, j);
80. jo = int.TryParse(Console.ReadLine(), out ho[i, j]) && ho[i, j] >= -50 && ho[i, j] <= 50;
81. if (!jo) {
82. Console.ForegroundColor = ConsoleColor.Red;
83. Console.WriteLine("A hőfoknak -50 és 50 közé kell esnie!");
84. }
85. } while (!jo);
86. }
87. }
88. return ho;
89. }
91. static int max (int[,] ho) {
92. int n = ho.GetLength(0) - 1;
93. int m = ho.GetLength(1) - 1;
95. int maxho = ho[1, 1];
96. for (int i = 1 + 1; i <= n \* m; ++i) {
97. if (maxho < ho[(i - 1) / m + 1, (i - 1) % m + 1]) {
98. maxho = ho[(i - 1) / m + 1, (i - 1) % m + 1];
99. }
100. }
101. return maxho;
102. }
104. static bool bennevan (int sor, int[,] ho, int maxho) {
105. int m = ho.GetLength(1) - 1;
107. bool van = false;
108. int j = 1;
109. while (!van && j <= m) {
110. if (ho[sor, j] == maxho) {
111. van = true;
112. } else {
113. ++j;
114. }
115. }
116. return van;
117. }
119. static (int db, int[] y) kivalaszt (int[,] ho, int maxho) {
120. int n = ho.GetLength(0) - 1;
121. int[] y = new int[n + 1];
123. int db = 0;
124. for (int i = 1; i <= n; ++i) {
125. if (bennevan(i, ho, maxho)) {
126. ++db;
127. y[db] = i;
128. }
129. }
130. return (db, y);
131. }
133. static void kiir (int db, int[] y) {
134. if (Console.IsInputRedirected) {
135. Console.Write("{0} ", db);
136. for (int i = 1; i <= db; ++i) {
137. Console.Write("{0} ", y[i]);
138. }
139. } else {
140. Console.ForegroundColor = ConsoleColor.Green;
141. if (db == 0) {
142. Console.WriteLine("Nincs a feltételnek megfelelő település!");
143. } else {
144. Console.WriteLine("{0} darab feltételnek megfelelő település is van, sorszámaik:", db);
145. for (int i = 1; i < db; ++i) {
146. Console.Write("{0}, ", y[i]);
147. }
148. Console.WriteLine(y[db]);
149. }
150. Console.ForegroundColor = ConsoleColor.Black;
151. Console.BackgroundColor = ConsoleColor.Gray;
152. Console.WriteLine("Kérem, nyomjon ENTER-t a folytatáshoz!");
153. Console.ResetColor();
154. Console.ReadLine();
155. }
156. }
157. }
158. }
     * 1. A kód (magas szintű függvényekkel)

A Program.cs fájl tartalma:

1. using System;
2. using System.Collections.Generic;
3. using System.Linq;
4. using System.Numerics;
5. using System.Text;
6. using System.Threading.Tasks;
8. namespace magas\_szintu\_ck {
9. public static class Mintak {
10. #region Ã–sszegzÃ©s
11. #region ÃltalÃ¡nos Ã¶sszegzÃ©s
12. public static G Szumma<H, G> (int e, int u, Func<int, H> f, G kezd, Func<G, H, int, G> add) {
13. G s = kezd;
14. for (int i = e; i <= u; i++) {
15. s = add(s, f(i), i);
16. }
17. return s;
18. }
19. public static G Szumma<H, G> (int e, int u, Func<int, H> f, G kezd, Func<G, H, G> add) {
20. return Szumma(e, u, f, kezd, (s, p, i) => add(s, p));
21. }
22. #endregion
23. #region Intervallum szÃ¡mok Ã¶sszegzÃ©sre
24. public static int Szumma (int e, int u, Func<int, int> f) {
25. return Szumma(e, u, f, 0, (s, p) => s + p);
26. }
27. public static double Szumma (int e, int u, Func<int, double> f) {
28. return Szumma(e, u, f, 0.0, (s, p) => s + p);
29. }
30. #endregion
31. #region TÃ¶mb
32. public static G Szumma<H, G> (H[] arr, G kezd, Func<G, H, G> add) {
33. return Szumma(0, arr.Length - 1, i => arr[i], kezd, (s, p, i) => add(s, p));
34. }
35. #endregion
36. #region TÃ¶mb szÃ¡mok Ã¶sszegzÃ©sre
37. public static int Szumma<H> (H[] arr, Func<H, int> f) {
38. return Szumma(arr, 0, (s, p) => s + f(p));
39. }
40. public static int Szumma (int[] arr) {
41. return Szumma(arr, p => p);
42. }
43. public static double Szumma<H> (H[] arr, Func<H, double> f) {
44. return Szumma(arr, 0.0, (s, p) => s + f(p));
45. }
46. public static double Szumma (double[] arr) {
47. return Szumma(arr, p => p);
48. }
49. #endregion
50. #endregion
52. #region MegszÃ¡molÃ¡s
53. #region Intervallum
54. public static int Darab (int e, int u, Func<int, bool> t) {
55. int db = 0;
56. for (int i = e; i <= u; i++) {
57. if (t(i)) {
58. db++;
59. }
60. }
61. return db;
62. }
63. #endregion
64. #region TÃ¶mb
65. public static int Darab<H> (H[] arr, Func<H, bool> t) {
66. return Darab(0, arr.Length - 1, i => t(arr[i]));
67. }
68. #endregion
69. #endregion
71. #region Maximum-, minimumkivÃ¡lasztÃ¡s
72. #region Max:Intervallum
73. public static (int maxind, H maxert) Max<H> (int e, int u, Func<int, H> f, Func<H, H, bool> isGreater = null) {
75. bool \_isGreater (H a, H b) {
76. if (a is IComparable<H> aa) {
77. return aa.CompareTo(b) > 0;
78. } else {
79. return isGreater != null ? isGreater(a, b) : false;
80. }
81. }
83. H maxert = f(e);
84. int maxind = e;
85. for (int i = e + 1; i <= u; i++) {
86. if (\_isGreater(f(i), maxert)) {
87. maxert = f(i);
88. maxind = i;
89. }
90. }
91. return (maxind, maxert);
92. }
93. #endregion
94. #region Max:TÃ¶mb transzformÃ¡tumfÃ¼ggvÃ©nnyel
95. public static (int maxind, G maxert) Max<H, G> (H[] arr, Func<H, G> f, Func<G, G, bool> isGreater = null) {
96. return Max(0, arr.Length - 1, i => f(arr[i]), isGreater);
97. }
98. #endregion
99. #region Max:TÃ¶mb
100. public static (int maxind, H maxert) Max<H> (H[] arr, Func<H, H, bool> isGreater = null) {
101. return Max(arr, p => p, isGreater);
102. }
103. #endregion
105. #region Min:Intervallum
106. public static (int minind, H minert) Min<H> (int e, int u, Func<int, H> f, Func<H, H, bool> isSmaller = null) {
108. bool \_isSmaller (H a, H b) {
109. if (a is IComparable<H> aa) {
110. return aa.CompareTo(b) < 0;
111. } else {
112. return isSmaller != null ? isSmaller(a, b) : false;
113. }
114. }
116. H minert = f(e);
117. int minind = e;
118. for (int i = e + 1; i <= u; i++) {
119. if (\_isSmaller(f(i), minert)) {
120. minert = f(i);
121. minind = i;
122. }
123. }
124. return (minind, minert);
125. }
127. #endregion
128. #region Min:TÃ¶mb transzformÃ¡tumfÃ¼ggvÃ©nnyel
129. public static (int minind, G minert) Min<H, G> (H[] arr, Func<H, G> f, Func<G, G, bool> isGreater = null) {
130. return Min(0, arr.Length - 1, i => f(arr[i]), isGreater);
131. }
132. #endregion
133. #region Min:TÃ¶mb
134. public static (int minind, H minert) Min<H> (H[] arr, Func<H, H, bool> isGreater = null) {
135. return Min(arr, p => p, isGreater);
136. }
137. #endregion
138. #endregion
140. #region FeltÃ©teles maximum-, minimumkeresÃ©s
141. #region FeltMax:Intervallum
142. public static (bool van, int maxind, H maxert) FeltMax<H> (int e, int u, Func<int, H> f, Func<int, bool> t, Func<H, H, bool> isGreater = null) {
144. bool \_isGreater (H a, H b) {
145. if (a is IComparable<H> aa) {
146. return aa.CompareTo(b) > 0;
147. } else {
148. return isGreater != null ? isGreater(a, b) : false;
149. }
150. }
152. bool van = false;
153. H maxert = f(e);
154. int maxind = e;
155. for (int i = e; i <= u; i++) {
156. if (van && t(i)) {
157. if (\_isGreater(f(i), maxert)) {
158. maxert = f(i);
159. maxind = i;
160. }
161. } else if (!van && t(i)) {
162. van = true;
163. maxert = f(i);
164. maxind = i;
165. }
166. }
167. return (van, maxind, maxert);
168. }
169. #endregion
170. #region FeltMax: TÃ¶mb transzformÃ¡tor fÃ¼ggvÃ©nnyel
171. public static (bool van, int maxind, G maxert) FeltMax<H, G> (H[] arr, Func<H, G> f, Func<H, bool> t, Func<G, G, bool> isGreater = null) {
172. return FeltMax(0, arr.Length - 1, i => f(arr[i]), i => t(arr[i]), isGreater);
173. }
174. #endregion
175. #region FeltMax:TÃ¶mb
176. public static (bool van, int maxind, H maxert) FeltMax<H> (H[] arr, Func<H, bool> t, Func<H, H, bool> isGreater = null) {
177. return FeltMax(arr, p => p, t, isGreater);
178. }
179. #endregion
181. #region FeltMin:Intervallum
182. public static (bool van, int minind, H minert) FeltMin<H> (int e, int u, Func<int, H> f, Func<int, bool> t, Func<H, H, bool> isSmaller = null) {
184. bool \_isSmaller (H a, H b) {
185. if (a is IComparable<H> aa) {
186. return aa.CompareTo(b) < 0;
187. } else {
188. return isSmaller != null ? isSmaller(a, b) : false;
189. }
190. }
192. bool van = false;
193. H minert = f(e);
194. int minind = e;
195. for (int i = e; i <= u; i++) {
196. if (van && t(i)) {
197. if (\_isSmaller(f(i), minert)) {
198. minert = f(i);
199. minind = i;
200. }
201. } else if (!van && t(i)) {
202. van = true;
203. minert = f(i);
204. minind = i;
205. }
206. }
207. return (van, minind, minert);
208. }
209. #endregion
210. #region FeltMin: TÃ¶mb transzformÃ¡tor fÃ¼ggvÃ©nnyel
211. public static (bool van, int minind, G minert) FeltMin<H, G> (H[] arr, Func<H, G> f, Func<H, bool> t, Func<G, G, bool> isSmaller = null) {
212. return FeltMin(0, arr.Length - 1, i => f(arr[i]), i => t(arr[i]), isSmaller);
213. }
214. #endregion
215. #region FeltMin:TÃ¶mb
216. public static (bool van, int minind, H minert) FeltMin<H> (H[] arr, Func<H, bool> t, Func<H, H, bool> isSmaller = null) {
217. return FeltMin(arr, p => p, t, isSmaller);
218. }
219. #endregion
220. #endregion
222. #region KeresÃ©s
223. #region Intervallum
224. public static (bool van, int ind) Keres (int e, int u, Func<int, bool> t) {
225. int ind = e;
226. while (ind <= u && !t(ind)) {
227. ind++;
228. }
229. bool van = ind <= u;
230. return (van, ind);
231. }
232. #endregion
233. #region TÃ¶mb
234. public static (bool van, int ind) Keres<H> (H[] arr, Func<H, bool> t) {
235. return Keres(0, arr.Length - 1, i => t(arr[i]));
236. }
237. #endregion
238. #region TÃ¶mb Ã©rtÃ©kkel
239. //public static (bool van, int ind, H ertek) Keres<H>(H[] arr, Func<H, int, bool> t) {
240. //  var result = Keres(0, arr.Length - 1, i => t(arr[i], i));
241. //  return (result.van, result.ind, arr[result.ind]);
242. //}
243. #endregion
244. #endregion
246. #region EldÃ¶ntÃ©s
247. #region Intervallum
248. public static bool Van (int e, int u, Func<int, bool> t) {
249. return Keres(e, u, t).van;
250. }
251. #endregion
252. #region TÃ¶mb
253. public static bool Van<H> (H[] arr, Func<H, bool> t) {
254. return Keres(arr, t).van;
255. }
256. #endregion
258. #region Mind:Intervallum
259. public static bool Mind (int e, int u, Func<int, bool> t) {
260. return !Keres(e, u, (i) => !t(i)).van;
261. }
262. #endregion
263. #region Mind:TÃ¶mb
264. public static bool Mind<H> (H[] arr, Func<H, bool> t) {
265. return !Keres(arr, e => !t(e)).van;
266. }
267. #endregion
268. #endregion
270. #region KivÃ¡lasztÃ¡s
271. #region Intervallum
272. public static int Kivalaszt (int e, Func<int, bool> t) {
273. return Keres(e, int.MaxValue, t).ind;
274. }
275. #endregion
276. #region TÃ¶mb
277. public static int Kivalaszt<H> (H[] arr, Func<H, bool> t) {
278. return Keres(arr, t).ind;
279. }
280. #endregion
281. #endregion
283. #region MÃ¡solÃ¡s
284. #region Intervallum
285. public static H[] Masol<H> (int e, int u, Func<int, H> f) {
286. return Szumma(e, u, f, new List<H>(), (s, p, i) => {
287. s.Add(p);
288. return s;
289. }).ToArray();
290. }
291. #endregion
292. #region TÃ¶mb
293. public static G[] Masol<H, G> (H[] arr, Func<H, G> f) {
294. return Masol(0, arr.Length - 1, i => f(arr[i]));
295. }
296. public static G[] Masol<H, G> (H[] arr, Func<H, int, G> f) {
297. return Masol(0, arr.Length - 1, i => f(arr[i], i));
298. }
299. #endregion
300. #endregion
302. #region KivÃ¡logatÃ¡s
303. #region Intervallum
304. public static H[] Kivalogat<H> (int e, int u, Func<int, bool> t, Func<int, H> f) {
305. return Szumma(e, u, f, new List<H>(), (s, p, i) => {
306. if (t(i)) {
307. s.Add(p);
308. }
309. return s;
310. }).ToArray();
311. }
312. #endregion
313. #region TÃ¶mb
314. public static G[] Kivalogat<H, G> (H[] arr, Func<H, bool> t, Func<H, int, G> f) {
315. return Kivalogat(0, arr.Length - 1, i => t(arr[i]), i => f(arr[i], i));
316. }
317. public static H[] Kivalogat<H> (H[] arr, Func<H, bool> t) {
318. return Kivalogat(arr, p => t(p), (p, i) => p);
319. }
320. #endregion
321. #endregion
323. }
325. internal class Program {
326. static void Main(string[] args) {
327. // deklaralas
328. int maxho, maxind, db;
329. int[,] ho;
330. int[] y;
332. // beolvasas
333. ho = beolvas();
335. // megoldas
336. int n = ho.GetLength(0) - 1;
337. int m = ho.GetLength(1) - 1;
339. (maxind, maxho) = Mintak.Max(1, n \* m, i => ho[(i - 1) / m + 1, (i - 1) % m + 1]);
340. y = Mintak.Kivalogat(1, n, i => bennevan(i), i => i);
341. db = y.Length;
343. bool bennevan (int i) {
344. return Mintak.Van(1, m, j => ho[i, j] == maxho);
345. }
347. // kiiras
348. kiir(db, y);
349. }
351. static int[,] beolvas () {
352. if (Console.IsInputRedirected) {
353. return beolvas\_biro();
354. } else {
355. return beolvas\_kezi();
356. }
357. }
359. static int[,] beolvas\_biro () {
360. string[] line = Console.ReadLine().Split(' ');
361. int n = int.Parse(line[0]);
362. int m = int.Parse(line[1]);
364. int[,] ho = new int[n + 1, m + 1];
366. for (int i = 1; i <= n; ++i) {
367. line = Console.ReadLine().Split(' ');
368. for (int j = 1; j <= line.Length; ++j) {
369. int.TryParse(line[j - 1], out ho[i, j]);
370. }
371. }
372. return ho;
373. }
375. static int[,] beolvas\_kezi () {
376. int n, m;
377. bool jo;
378. do {
379. Console.ResetColor();
380. Console.Write("TelepÃ¼lÃ©sek szÃ¡ma = ");
381. jo = int.TryParse(Console.ReadLine(), out n) && n >= 1 && n <= 1000;
382. if (!jo) {
383. Console.ForegroundColor = ConsoleColor.Red;
384. Console.WriteLine("A telepÃ¼lÃ©sek szÃ¡mÃ¡nak 1 Ã©s 1000 kÃ¶zÃ© kell esnie!");
385. }
386. } while (!jo);
387. do {
388. Console.ResetColor();
389. Console.Write("Napok szÃ¡ma = ");
390. jo = int.TryParse(Console.ReadLine(), out m) && m >= 1 && m <= 1000;
391. if (!jo) {
392. Console.ForegroundColor = ConsoleColor.Red;
393. Console.WriteLine("A napok szÃ¡mÃ¡nak 1 Ã©s 1000 kÃ¶zÃ© kell esnie!");
394. }
395. } while (!jo);
397. int[,] ho = new int[n + 1, m + 1];
398. for (int i = 1; i <= n; ++i) {
399. for (int j = 1; j <= m; ++j) {
400. do {
401. Console.ResetColor();
402. Console.Write("{0}. telepÃ¼lÃ©s {1}. nap hÅ‘mÃ©rsÃ©klete = ", i, j);
403. jo = int.TryParse(Console.ReadLine(), out ho[i, j]) && ho[i, j] >= -50 && ho[i, j] <= 50;
404. if (!jo) {
405. Console.ForegroundColor = ConsoleColor.Red;
406. Console.WriteLine("A hÅ‘foknak -5 Ã©s 5 kÃ¶zÃ© kell esnie!");
407. }
408. } while (!jo);
409. }
410. }
411. return ho;
412. }
414. static void kiir (int db, int[] y) {
415. if (Console.IsInputRedirected) {
416. Console.Write("{0} ", db);
417. for (int i = 0; i < db; ++i) {
418. Console.Write("{0} ", y[i]);
419. }
420. Console.WriteLine();
421. } else {
422. Console.ForegroundColor = ConsoleColor.Green;
423. if (db == 0) {
424. Console.WriteLine("Nincs a feltÃ©telnek megfelelÅ‘ telepÃ¼lÃ©s!");
425. } else {
426. Console.WriteLine("{0} darab feltÃ©telnek megfelelÅ‘ telepÃ¼lÃ©s is van, sorszÃ¡maik:", db);
427. for (int i = 0; i < db - 1; ++i) {
428. Console.Write("{0}, ", y[i]);
429. }
430. Console.WriteLine(y[db - 1]);
431. }
432. Console.ForegroundColor = ConsoleColor.Black;
433. Console.BackgroundColor = ConsoleColor.Gray;
434. Console.WriteLine("KÃ©rem, nyomjon ENTER-t a folytatÃ¡shoz!");
435. Console.ResetColor();
436. Console.ReadLine();

* 1. Tesztelés
     1. Automatikus tesztek (Bíró)

A képen szöveg, képernyőkép, szoftver, Multimédiás szoftver látható

Automatikusan generált leírás

* + 1. Automatikus tesztek (Bíró, magas szintű függvényekkel)

A képen szöveg, képernyőkép, szoftver, Multimédiás szoftver látható

Automatikusan generált leírás

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

|  |
| --- |
| Bemenet – *alap teszt* |
| 3 5  10 15 12 10 10  11 11 11 11 20  12 16 16 16 20 |
| Kimenet |
| 2 2 3 |

* + - 1. teszteset: be2.txt

|  |
| --- |
| Bemenet – *1 település, 1 nap* |
| 1 1  25 |
| Kimenet |
| 1 1 |

* + - 1. teszteset: be3.txt

|  |
| --- |
| Bemenet – *minden hőfok ugyanannyi (és negatív)* |
| 3 2  -10 -10  -10 -10  -10 -10 |
| Kimenet |
| 3 1 2 3 |

* + - 1. teszteset: be4.txt

|  |
| --- |
| Bemenet – *csak 1 település* |
| 1 3  1 2 3 |
| Kimenet |
| 1 1 |

* + - 1. teszteset: be5.txt

|  |
| --- |
| Bemenet – *csak 1 nap* |
| 3 1  1  2  3 |
| Kimenet |
| 1 3 |

* + 1. Érvénytelen tesztesetek

Billentyűzetes bevitel esetén

* + - 1. teszteset

|  |
| --- |
| Bemenet – *szöveges adat* |
| Települések száma = 11tizenegy |
| Kimenet |
| A települések számának 1 és 1000 közé kell esnie! N = |

* + - 1. teszteset

|  |
| --- |
| Bemenet – *Mátrix mérete negatív szám* |
| Települések száma = 1  Napok száma = -1 |
| Kimenet |
| A napok számának 1 és 1000 közé kell esnie!  Napok száma = |

8. teszteset

|  |
| --- |
| Bemenet – *Mátrix mérete túl nagy (>1000)* |
| Települések száma = 1001 |
| Kimenet |
| A települések számának 1 és 1000 közé kell esnie!  Települések száma = |

9. teszteset

|  |
| --- |
| Bemenet – *hőmérséklet intervallumon kívül (negatív)* |
| Települések száma = 1  Napok száma = 1  1. település 1. nap homérséklete = -51 |
| Kimenet |
| A hofoknak -50 és 50 közé kell esnie!  1. település 1. nap homérséklete = |

10. teszteset

|  |
| --- |
| Bemenet – *hőmérséklet intervallumon kívül (pozitív)* |
| Települések száma = 1  Napok száma = 1  1. település 1. nap homérséklete = 51 |
| Kimenet |
| A hofoknak -50 és 50 közé kell esnie!  1. település 1. nap homérséklete = |

* 1. Fejlesztési lehetőségek

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

1. a pirossal kiemelt részek az első fázisban szereplő algoritmus apró hibáinak javítását jelölik. Ezen kívül még a változók típusai is hozzá lettek adva a maximális pontosság kedvéért (ezeket nem lehet pirossal kiemelni). [↑](#footnote-ref-1)