Komplex beadandó (Ck) - Legtöbbször legmelegebb települések

Neptun kód:	**MNDJ3P**	Név:	**Bartók Patrik**
Beadás verziószáma:	1.		

Tisztelt Tanár Úr!

Tudom hogy hiányos a beadandó és ami megvan az sem működőképes, de sajnos ez még az előző beadandónál is sokkal nagyobb nehézséget okozott. Megpróbáltam fordítottan megcsinálni, és megírtam először a kódot, hátha az alapján könnyebben menne és megpróbáltam minden fellelhető segítséget felhasználni, de sajnos így is csak ez a hiányos és hibás verzió sikerült.

Csatolom a kódót is ami alapján megpróbáltam a specifikációt és az algoritmust megcsinálni.

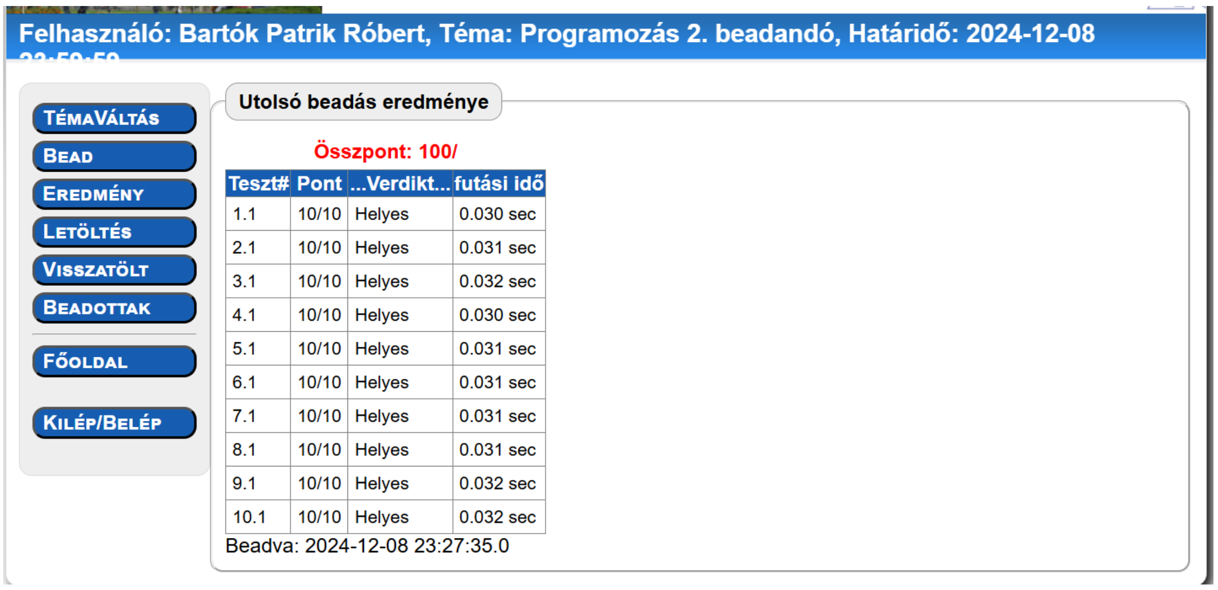
Nem tudom ez jó lehet-e egyáltalán kiindulópontnak, vagy teljesen újra kéne gondolnom az egészet.

Kód:

```
using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;


namespace legtobbszorMelegNapok_BartokPatrik_mndj3p_komplexBead

{

  internal class Program

  {

    static int settlementCount, dayCount;

    static int[,] tempForecast;

    static int[] maxTempDays;


    static void ReadData()

    {

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

      settlementCount = int.Parse(line[0]);

      dayCount = int.Parse(line[1]);

      tempForecast = new int[settlementCount, dayCount];

      for (int i = 0; i < settlementCount; i++)
```

```
        {
            line = Console.ReadLine().Split(' ');
            for (int j = 0; j < dayCount; j++)
            {
                tempForecast[i, j] = int.Parse(line[j]);
            }
        }
    }

    static void maxTempDay(int j)
    {
        List<int> maxTempDaysIndex = new List<int>();
        int maxTempDay = tempForecast[0, j];
        maxTempDaysIndex.Add(0);
        for (int i = 1; i < settlementCount; i++)
        {
            if (tempForecast[i, j] > maxTempDay)
            {
                maxTempDaysIndex.Clear();
                maxTempDay = tempForecast[i, j];
                maxTempDaysIndex.Add(i);
            }
            else if (tempForecast[i, j] == maxTempDay)
            {
                maxTempDaysIndex.Add(i);
            }
        }

        for (int i = 0; i < maxTempDaysIndex.Count(); i++)
        {
            maxTempDays[maxTempDaysIndex[i]]++;
        }
    }

    static void maxTempDayAll()
    {
        maxTempDays = new int[settlementCount];

        for (int j = 0; j < dayCount; j++)
        {
            maxTempDay(j);
        }
    }

    static void findMaxSettlement()
    {
        int max = maxTempDays[0];
        for (int i = 1; i < maxTempDays.Length; i++)
        {
            if (maxTempDays[i] > max)
            {
                max = maxTempDays[i];
            }
        }
        List<int> hottestSettlement = new List<int>();
        for (int i = 0; i < maxTempDays.Length; i++)
        {
            if (maxTempDays[i] == max)
            {
                hottestSettlement.Add(i + 1);
            }
        }
```

```
      PrintResult(hottestSettlement);
    }

    static void PrintResult(List<int> hottestSettlement)
    {
      int hottestSettlementLength = hottestSettlement.Count();
      Console.Write(hottestSettlementLength);
      foreach (int i in hottestSettlement)
      {
        Console.Write(" " + i);
      }
    }
    static void Main(string[] args)
    {
      ReadData();
      maxTempDayAll();
      findMaxSettlement();
    }
  }
}
```



Felhasználó: Bartók Patrik Róbert, Téma: Programozás 2. beadandó, Határidő: 2024-12-08

Utolsó beadás eredménye

Összpont: 100/

Teszt#	Pont	...Verdikt...	futási idő
1.1	10/10	Helyes	0.030 sec
2.1	10/10	Helyes	0.031 sec
3.1	10/10	Helyes	0.032 sec
4.1	10/10	Helyes	0.030 sec
5.1	10/10	Helyes	0.031 sec
6.1	10/10	Helyes	0.031 sec
7.1	10/10	Helyes	0.031 sec
8.1	10/10	Helyes	0.031 sec
9.1	10/10	Helyes	0.032 sec
10.1	10/10	Helyes	0.032 sec

Beadva: 2024-12-08 23:27:35.0

Feladat

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 azokat a településeket, amelyeken a legtöbbször fordul elő valamelyik napi előrejelzések maximuma!

Specifikáció

```
Be: settlementCount∈N,
    dayCount∈N,
    tempForecast∈Z[0..settlementCount-1, 0..dayCount-1]

Sa: maxTempDays∈N[0..settlementCount-1]

Ki: hottestSettlementLength∈N,
    hottestSettlement∈N[1..hottestSettlementLength]

Fv: maxTempForDay: N x Z[0..settlementCount-1,0..dayCount-1] x N -> N,
    maxTempForDay(j, tempForecast, settlementCount) = MAX(k=0..settle-
mentCount-1, tempForecast[k, j])

Fv: isMaxTempForDay: N x N x Z[0..settlementCount-1,0..dayCount-1] x N ->
L,
    isMaxTempForDay(i, j, tempForecast, settlementCount) =
(tempForecast[i, j] = maxTempForDay(j, tempForecast, settlementCount))

Fv: calculateMaxTempDays: Z[0..settlementCount-1,0..dayCount-1] x N x N -
> N[0..settlementCount-1],
calculateMaxTempDays(tempForecast, settlementCount, dayCount) =
    (forall i in [0..settlementCount-1]: (j ∈ [0..dayCount-1] or isMax-
TempForDay(i, j, tempForecast, settlementCount)))

Ef: 1 <= settlementCount <= 1000 és
    1 <= dayCount <= 1000 és
    ∀i∈[0..settlementCount-1]:(∀j∈[0..dayCount-1]:(-50 <=
tempForecast[i,j] <= 50))

Uf:
    maxTempDays = calculateMaxTempDays(tempForecast, settlementCount,
dayCount) és
    let
        max_overall = MAX(i ∈ [0..settlementCount-1], maxTempDays[i]),
        hottestSettlement = {i+1 ∈ [1..settlementCount] | maxTempDays[i]
= max_overall},
        hottestSettlementLength = |hottestSettlement|
    in
```

```
       ∀k∈[1..hottestSettlementLength]:(hottestSettlement[k-1] ∈
[1..settlementCount] és maxTempDays[(hottestSettlement[k-1]-1)] =
max_overall)
```

Visszavezetés

Algoritmus – **Nem engedte beilleszteni linknek a hossza miatt**

https://progalap.elte.hu/stuki/?data=H4sIA-
AAAAAAACq1YXVPiPBT%2BK0525h13tsPUilB4xwsUFVBx-
FRDR8SKkKa2UFtsgsDv8950Ec5Ku4FLEG-
zuH9Hme85Wc9DfyHVRGZs4ulg4sK1%2B08qaVL1klZCAveqNx3UHlcBlEBkpo-
QAmjjrKEkUMTVP6NbDaNevbFLX8WeGAw0BjHNGTqJc93HBqisouDhBqlz-
ccUlVFCXyc0JBQZiHh%2B4MQ0rDsJKj%2BhcfKSxBN3igz0Wh82zLDeRga6xiX%2FrHkSI-
ANV%2Br0ujqoOel4YqPlAbuPuYwOUgOHrSvq43TyLr9qCCNc6R7PqyQCI-
wPB1ou7w%2FqKB6wQZqHl3cOZ3aBEZqNry4kqj1hL0sETSa%2B8oel3UGhEMMzqiIUMGYnT-
GUBmN8Kx8PMKzNh2Nq3iePJnPaGGgXvXSu%2BvFKrJg%2BLrDE%2FuA3twM68hAg-
VuoXbn5O%2BEl2CWntlDzU-
lOSxct3BytBsP%2BdewjUkg0MX2Zz%2FdC5xrMWZSwQPy0ZobYlo1bsGqPWTVv4Vz7eY3Q0Po9
iSnDCnnzjRSQUukmSa%2B21M%2FKkHjp0lqs4zr4pXD5rJRPb6UgBUOKVev-
vrMWU1O5vQ3q97BPAkXQacBYnp54fCK4odHzmRyEqI%2F%2F4IJdLIKWn0UTEoB8589a7un-
rKQ57xi87Yrt91VMalIaUHfMjuvnVawKduVWTRbEbn7JY%2F3j50ayWcYBEUVroZ-
taevqpXBsCsRLRfPr%2F3xm6ADHEmnAWt0WiG-
spvPdvxLwoaz3%2Fhuw%2F%2FdU9fGKjCf0JMYh8VI50cIuSFYsgaAsDBVU6YIeZeW-
Chrp9r5wGFMfv%2B4NMIFSLyujOaNftD1AykhwMu%2FeZ7w%2B%2BcDlpFR9KSdBTxSINKVYo-
sOy1WW14PTy4rYjafBiOfh6SVhfowJ-
CiK9jDs8Sem2vpwigac%2FmwUOJpb%2B5GPgjk8gFH0mnAGp3W31u31vFGfaVla11fQU-
QWhsqF1A%2BG9E6kUHdRY4Q1k8L4SB27YEixwmC5xQRnz6aVweRUJAlwJJ0G%2FPUzyMz-
lPm4h%2FCja%2F77qMAJX%2BRAp-
VcIQKQ0pYVp0ton%2B09%2Fynvzn5%2FSEuWrFjwMu8eawUXX6thqzwZCSCKN59lQVa8lkNj18
FKkCHEmnAWt02qVi81S98FQ5eM6T46xKDbi2MJQqKQQMKSFaNLZIzf6LaAcLvx2dm52aOty-
kIe20uopkjXHFfXTvSe%2Fn8oomceCKpoCzXF3WjmRa-
NeWuaDhg3qpog5MLA8X15mVlmFfbEBjSG4J0I3sEhqTgJW1%2FLCI-
wvpknvc58po51aUjROU3nDnfmtX8eOrBQ4mlv7kY%2BCBT7mcSB%2FUwBa3RaFW146Ohbgq
%2BGufWnjZamdacNhGJhqCRI4WBICddQszRV6oTkDnDKuyP%2FV6mkDVFgS-
FHC3T57an5FVuVxfkZEagAHzlMF%2FPXeMjftLXCSX%2FhYA5Or%2BY268ElDShK4kT0Cl69env-
zsPogIDGrsLWZ5db8EQ4pu1o5fYs%2B%2B%2BmdvwUKJp725G%2FkgkMsHHEm-
nAWt0WhVt11vHnzeW-
lqN1jQVxWBgqA1I1GNI3PIW6cWN5EWM0YerLyXKO%2B3HwHS34dYHFE8KiQYx-
HPJi%2FUYhHNH3I%2BSHbe%2BGz0BuOfdwP%2BCdLhAwUR9GHwSvxoum9Wva-
ujltrFDs0RmUesIWxgqcSBJ9waBPD32irWTWODx%2BPPuHROjsz-
zzX2w0%2Bgte9eG0I%2F8xIMolh8I17m8wl9c13TNPl96Bv%2F7%2FJC%2Fea6BGx98wieXPFUN
A9NLJ4Ixa5dRM%2FL8jjBZDiIxTi1BO5bfYsv4%2F8lMCXSRosEyydXPDl5YlKLw%2FFP2R%2FgCC
FEsBJCJJxrSRt1HbIO-
TivLD6j8793njfBdt1Ra%2FmqaNikd8SdKad4p6BFZLP4Acas%2F3ssXAAA%3D

```
maxTempDay(int j)
    maxTempDay:= tempForecast[i,j]
    maxTempDaysIndex.Add(0)
    i=1..settlementCount
        tempForecast[i,j] > maxTempDay
        T:
            maxTempDaysIndex.Clear()
            maxTempDay:= tempForecast[i,j]
            maxTempDaysIndex.Add(i)
        F:
            tempForecast[i,j] = maxTempDay
            T:
                maxTempDaysIndex.Add(i)
            F:
                -
    i=0..maxTempDaysIndex.Count()
        maxTempDays[maxTempDaysIndex[i]]:=maxTempDays[maxTempDaysIndex[i]]+1
```

```
maxTempDayAll
    j=0..dayCound
        maxTempDay(j)
```

```
findMaxSettlement
    max:=maxTempDays[0]
    i=1..maxTempDays.Length
        maxTempDays[i] > max
        T:
            max = maxTempDays[i]
        F:
            -
    i=0..maxTempDays.Length
        maxTempDays[i] = max
        T:
            hottestSettlement.Add(i+1)
        F:
            -
```

```
Main
    maxTempDayAll()
    findMaxSettlement()
```