„Programozási alapismeretek”  
beadandó feladat

Készítette: *Szobonya Péter Csaba*Neptun-azonosító: *YKMM49*E-mail: [szobonya.peter@gmail.com](mailto:szobonya.peter@gmail.com)

Kurzuskód: IP-18PROGEGGyakorlatvezető neve: Horváth Győző

2020. január 5.

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 3](#_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 6](#_Toc410323344)

[Függvénystruktúra 6](#_Toc410323345)

[Algoritmus 7](#_Toc410323346)

[A kód 7](#_Toc410323347)

[Tesztelés 9](#_Toc410323348)

[Érvényes tesztesetek 9](#_Toc410323349)

[Érvénytelen tesztesetek 10](#_Toc410323350)

[Fejlesztési lehetőségek 10](#_Toc410323351)

2. Felhasználói dokumentáció
   1. 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 a legszélsőségesebb településeket, azaz azokat, ahol a legkisebb és a legnagyobb várt hőmérséklet eltérése maximális!

* 1. Futási környezet

IBM PC, exe futtatására alkalmas, 32-bites operációs rendszer (pl. Windows 7). Nem igényel egeret.

* 1. Használat
     1. A program indítása

A program az YKMM49\Release\komplex.exe néven található a tömörített állományban. A A1B2C3.exe fájl kiválasztásával indítható.

* + 1. A program bemenete

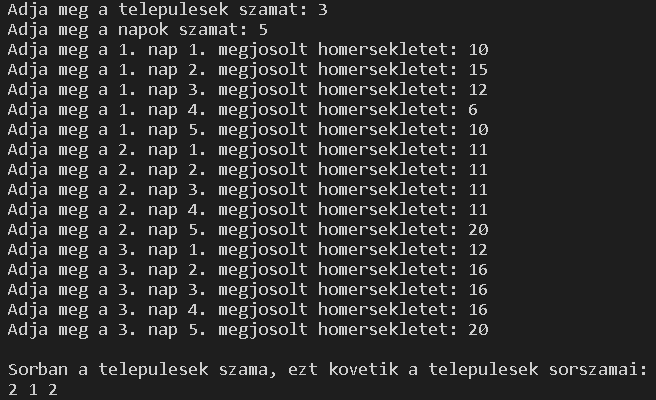
A program az adatokat a **billentyűzet**ről olvassa be a következő sorrendben:

|  |  |  |
| --- | --- | --- |
| # | Adat | Magyarázat |
| **1.** | *N* | A települések száma (1≤N≤1000). |
| **2.** | *M* | A napok száma (1≤M≤1000). |
| **3.** | *H(1..N, 1..M)* | A jósolt hőmérséklet(-50≤H(i,j) ≤50). Ahol (0≤i≤N) és (0≤j≤M) |

* + 1. A program kimenete

A standard kimenet első sorába a legszélsőségesebb települések T számát kell kiírni! Ezt kövesse ezen települések sorszáma, növekvő sorrendben!

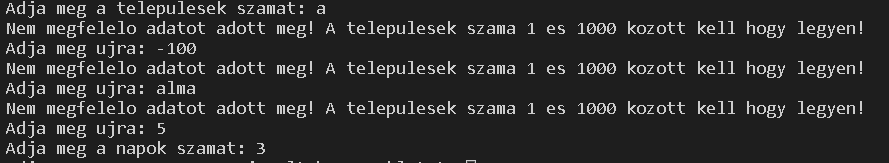
* + 1. Minta bemenet és kimenet



* + 1. Hibalehetőségek

Az egyes bemeneti adatokat a fenti mintának megfelelően kell megadni. Hiba, ha a napok vagy a megjósolt hőmérsékletek száma nem egész szám, vagy nem esik a **1..1 000** intervallumba; vagy valamely megjósolt érték nem szám, vagy nem esik a **-50..50** intervallumba. Hiba esetén a program azzal jelzi a hibát, hogy újra kérdezi az éppen megadott értéket.

* + - 1. Mintafutás hibás bemeneti adatok esetén:



1. Fejlesztői dokumentáció
   1. Feladat

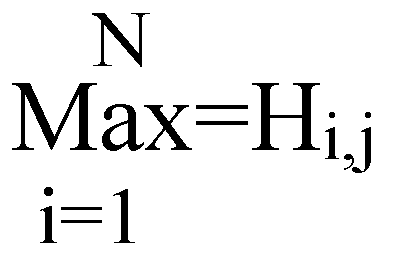
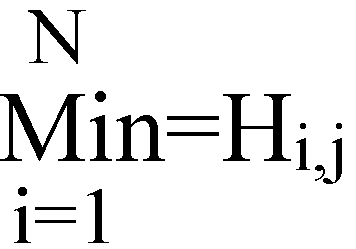
A meteorológiai intézet az ország N településére adott M napos időjárás előrejelzést, az adott településen az adott napra várt legmagasabb hőmérsékletet.

Készíts programot, amely megadja a legszélsőségesebb településeket, azaz azokat, ahol a legkisebb és a legnagyobb várt hőmérséklet eltérése maximális!

* 1. Specifikáció

**Bemenet**: N∈N, M∈N, HN,M∈ NNXM

**Kimenet**: Db∈N, maxkulDb∈NDb, Maxért∈N, MaxKülN∈NN

**Előfeltétel**: 1≤N≤1000, 1≤M≤1000, -50≤HN,M≤50

**Utófeltétel**: ∀i(1≤i≤N): (MaxKüli = - ) és



MaxÉrt= és

(Db,Y)=

* 1. Fejlesztői környezet

IBM PC, exe futtatására alkalmas operációs rendszer (pl. Windows 7). mingw32-g++.exe c++ fordítóprogram (v4.7), Code::Blocks (v13.12) fejlesztői környezet.

* 1. Forráskód

A teljes fejlesztői anyag –kicsomagolás után– az YKMM49 nevű könyvtárban található meg. A fej­lesztés során használt könyvtár-struktúra:

|  |  |
| --- | --- |
| Állomány | Magyarázat |
| YKMM49\Release\komplex.exe | futtatható kód |
| YKMM49\Release\komplex.o | félig lefordított kód |
|  |  |
| YKMM49\Forraskod\komplex.cpp | C++ forráskód |
|  |  |
| YKMM49\Tesztek\teszt1.txt | teszt-bemeneti fájl1 |
| YKMM49\Tesztek\teszt2.txt | teszt-bemeneti fájl2 |
| YKMM49\Tesztek\teszt3.txt | teszt-bemeneti fájl3 |
| YKMM49\Tesztek\teszt4.txt | teszt-bemeneti fájl4 |
| YKMM49\Tesztek\teszt5.txt | teszt-bemeneti fájl5 |
| … |  |
|  |  |
| YKMM49\Dokumentacio\dokumentacio.docx | dokumentációk (ez a fájl) |

* 1. Megoldás
     1. Programparaméterek
        1. Konstans

1. MaxN : **Egész**(1000) [a települések maximális száma]  
   MaxM : **Egész**(1000) [a napok maximális száma]
   * + 1. Típus
2. TH = **Mátrix**(1..MaxN:**Egész;** 1..MaxM:**Egész**)
3. TMaxkul = **Tömb**(1..Maxn:**Egész**)
   * + 1. Változó

N : **Egész**

M : **Egész**  
H : TH

maxkul : TMaxkul

* + 1. Programfelépítés

A program által használt modulok (és helyük):

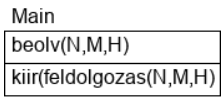
komplex.cpp – program, a forráskönyvtárban  
iostream – képernyő-, és billentyűkezelés, a C++ rendszer része

* + 1. Függvénystruktúra

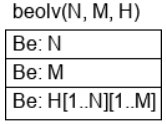


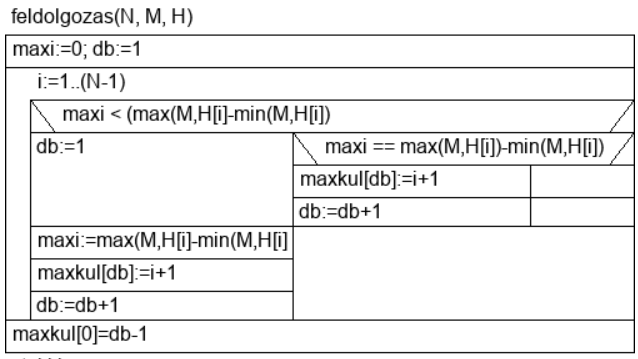
* + 1. A teljes program algoritmusa

Főprogram:

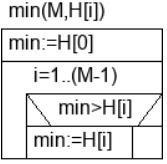


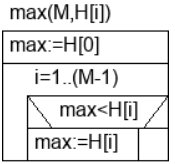
Alprogramok:

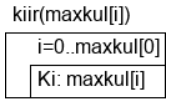
****

****

**Változó**  
 db:**Egész**  
 …

****

****

****

* + 1. A kód

A komplex.cpp fájl tartalma:

/\*

Keszitette: Szobonya Péter CSaba

Neptun: YKMM49

Email: peter.csaba.szobonya@gmail.com

Feladat: "Progalap beadando feladat" tema "Legszelsosegesebb telepulesek"

\*/

#include <iostream>

using namespace std;

const int MAXN = 1000;

const int MAXM = 1000;

int maxkul[MAXN];

int N;

int M;

int H[MAXN][MAXM];

void beolv();

int \*feldolgozas(int,int,int[][MAXM]);

int min(int, int[]);

int max(int, int[]);

void kiir(int[]);

int main(){

    beolv();

    kiir(feldolgozas(N,M,H));

    return 0;

}

void beolv(){

    bool joN,joM,joH;

    cerr << "Adja meg a telepulesek szamat: ";

    cin >> N;

    do{

        joN = (N>=1 && N<=1000 && !cin.fail());

        if(!joN){

            cin.clear();

            cin.ignore(1000,'\n');

            cerr << "Nem megfelelo adatot adott meg! A telepulesek szama 1 es 1000 kozott kell hogy legyen!\nAdja meg ujra: ";

            cin >> N;

            joN = (N>=1 && N<=1000 && !cin.fail());

        }

    }while(!joN);

    cerr << "Adja meg a napok szamat: ";

    cin >> M;

    do{

        joM = (M>=1 && M<=1000 && !cin.fail());

        if(!joM){

            cin.clear();

            cin.ignore(1000,'\n');

            cerr << "Nem megfelelo adatot adott meg! A josolot homersekletek szama 1 es 1000 kozott kell hogy legyen!\nAdja meg ujra: ";

            cin >> M;

            joM = (M>=1 && M<=1000 && !cin.fail());

        }

    }while(!joM);

    for(int i=0; i<N; ++i){

        for(int j=0; j<M; ++j){

            cerr << "Adja meg a " << i+1 << ". nap " << j+1 << ". megjosolt homersekletet: ";

            cin >> H[i][j];

            do{

                joH = (H[i][j]>=-50 && H[i][j]<=50 && !cin.fail());

                if(!joH){

                    cin.clear();

                    cin.ignore(1000,'\n');

                    cerr << "Nem megfelelo adatot adott meg! A josolt homerseklet -50 es 50 kozott kell hogy legyen!\nAdja meg ujra: ";

                    cin >> H[i][j];

                    joH = (H[i][j]>=-50 && H[i][j]<=50 && !cin.fail());

                }

            } while(!joH);

        }

    }

}

int \*feldolgozas(int N, int M, int H[][MAXM]){

    int maxi=0;

    int db=1;

    for(int i=0; i<N; ++i){

        if(maxi < max(M,H[i])-min(M,H[i])){

            db = 1;

            maxi = max(M,H[i])-min(M,H[i]);

            maxkul[db]=i+1;

            ++db;

        }else if(maxi == max(M,H[i])-min(M,H[i])){

            maxkul[db] = i+1;

            ++db;

        }

    }

    maxkul[0] = db-1;

    return maxkul;

}

int min(int M,int H[]){

    int min = H[0];

    for(int i=1; i<M; ++i){

        if(min > H[i]) min = x[i];

    }

    return min;

}

int max(int n,int x[]){

    int max = x[0];

    for(int i=1; i<n; ++i){

        if(max < x[i]) max = x[i];

    }

    return max;

}

void kiir(int maxkul[]){

    cerr << "\nSorban a telepulesek szama, ezt kovetik a telepulesek sorszamai:\n";

    for(int i=0; i<=maxkul[0]; i++){

        cout << maxkul[i] << " ";

    }

}

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

|  |
| --- |
| Bemenet |
| Adja meg a telepulesek szamat: 1  Adja meg a napok szamat: 1  Adja meg a 1. nap 1. megjosolt homersekletet: 1 |
| Kimenet |
| Sorban a telepulesek szama, ezt kovetik a telepulesek sorszamai:  1 1 |

* + - 1. teszteset: teszt2.txt

|  |
| --- |
| Bemenet |
| Adja meg a telepulesek szamat: 3  Adja meg a napok szamat: 5  Adja meg a 1. nap 1. megjosolt homersekletet: 10  Adja meg a 1. nap 2. megjosolt homersekletet: 15  Adja meg a 1. nap 3. megjosolt homersekletet: 12  Adja meg a 1. nap 4. megjosolt homersekletet: 6  Adja meg a 1. nap 5. megjosolt homersekletet: 10  Adja meg a 2. nap 1. megjosolt homersekletet: 11  Adja meg a 2. nap 2. megjosolt homersekletet: 11  Adja meg a 2. nap 3. megjosolt homersekletet: 11  Adja meg a 2. nap 4. megjosolt homersekletet: 11  Adja meg a 2. nap 5. megjosolt homersekletet: 20  Adja meg a 3. nap 1. megjosolt homersekletet: 12  Adja meg a 3. nap 2. megjosolt homersekletet: 16  Adja meg a 3. nap 3. megjosolt homersekletet: 16  Adja meg a 3. nap 4. megjosolt homersekletet: 16  Adja meg a 3. nap 5. megjosolt homersekletet: 20 |
| Kimenet |
| Sorban a telepulesek szama, ezt kovetik a telepulesek sorszamai:  2 1 2 |

* + - 1. teszteset: teszt3.txt

|  |
| --- |
| Bemenet |
| Adja meg a telepulesek szamat: 2  Adja meg a napok szamat: 2  Adja meg a 1. nap 1. megjosolt homersekletet: 1  Adja meg a 1. nap 2. megjosolt homersekletet: 10  Adja meg a 2. nap 1. megjosolt homersekletet: 1  Adja meg a 2. nap 2. megjosolt homersekletet: 2 |
| Kimenet |
| Sorban a telepulesek szama, ezt kovetik a telepulesek sorszamai:  1 1 |

* + - 1. teszteset: teszt4.txt

|  |
| --- |
| Bemenet |
| Adja meg a telepulesek szamat: 2  Adja meg a napok szamat: 1  Adja meg a 1. nap 1. megjosolt homersekletet: 1  Adja meg a 2. nap 1. megjosolt homersekletet: 8 |
| Kimenet |
| Sorban a telepulesek szama, ezt kovetik a telepulesek sorszamai:  2 1 2 |

* + - 1. teszteset: teszt5.txt

|  |
| --- |
| Bemenet |
| Adja meg a telepulesek szamat: 3  Adja meg a napok szamat: 3  Adja meg a 1. nap 1. megjosolt homersekletet: -40  Adja meg a 1. nap 2. megjosolt homersekletet: 30  Adja meg a 1. nap 3. megjosolt homersekletet: 7  Adja meg a 2. nap 1. megjosolt homersekletet: -35  Adja meg a 2. nap 2. megjosolt homersekletet: 27  Adja meg a 2. nap 3. megjosolt homersekletet: 39  Adja meg a 3. nap 1. megjosolt homersekletet: 20  Adja meg a 3. nap 2. megjosolt homersekletet: -12  Adja meg a 3. nap 3. megjosolt homersekletet: 33 |
| Kimenet |
| Sorban a telepulesek szama, ezt kovetik a telepulesek sorszamai:  1 2 |

* + 1. Érvénytelen tesztesetek
       1. teszteset

|  |
| --- |
| Bemenet – *Rossz bemenet* |
| Adja meg a telepulesek szamat: sok |
| Kimenet |
| Nem megfelelo adatot adott meg! A telepulesek szama 1 es 1000 kozott kell hogy legyen!  Adja meg ujra: |

* + - 1. teszteset

|  |
| --- |
| Bemenet – *Rossz magasság* |
| Adja meg a telepulesek szamat: -5 |
| Kimenet |
| Nem megfelelo adatot adott meg! A telepulesek szama 1 es 1000 kozott kell hogy legyen!  Adja meg ujra: |

* 1. Fejlesztési lehetőségek

1. Adatok –a felhasználó igénye szerint– akár fájlból is fogadása.
2. Hibás fájl-bemenetek felismerése, és a hiba helyének (sor sorszámának) kiírása.
3. Többszöri futtatás megszervezése
4. Grafikus megjelenítés.