„Programozási alapismeretek”  
beadandó feladat

Készítette: *Koller Dávid Dániel* **[[1]](#footnote-1)**Neptun-azonosító: *IDJJQ2*E-mail: david.daniel.koller@gmail.com

Kurzuskód: IP-18ePROGEGGyakorlatvezető neve: *Siegler Gábor*

2020. január 6.

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
   2. 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 azt a települést, ahol a legmagasabb hőmérséklet várható!
   3. 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 IDJJQ2\IDJJQ2.exe néven található a tömörített állományban. Az IDJJQ2.exe fájl kiválasztásával indítható.

* + 1. A program bemenete

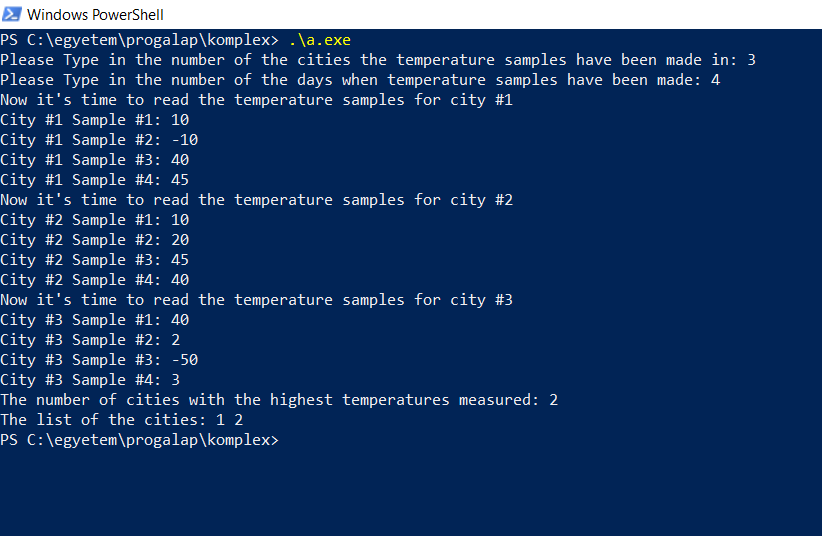
A program az adatokat a **billentyűzet**ről olvassa be a következő sorrendben:

|  |  |  |
| --- | --- | --- |
| # | Adat | Magyarázat |
| **1.** | N | A városok száma (1≤ *N* ≤1000). |
| **2.** | *M* | A városok száma (1≤ *M* ≤1000). |
| **3.** | *samples11* | Az első városban az első napon mért hőmérséklet |
| **4.** | *samples12* | Az első városban a második napon mért hőmérséklet |
| *...* | *samples21* | A második városban az első napon mért hőmérséklet |
| **...** | *…* |  |
| **N+M.** | *samplesnm* | Az N-edik városban az M-edik napon mért hőmérséklet |

* + 1. A program kimenete

A program kiírja azon települések számát, ahol a legmagasabb hőmérséklet várható, majd ezen települések sorszámát is.

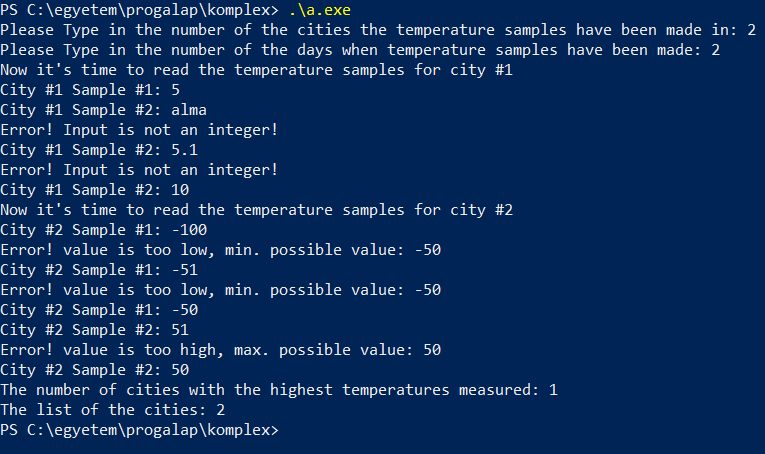
* + 1. Minta bemenet és kimenet



* + 1. Hibalehetőségek

Az egyes bemeneti adatokat a fenti mintának megfelelően kell megadni. Hiba, ha a napok, városok száma nem egész szám, vagy nem esik az **1..10 00** intervallumba; vagy valamely hőmérsékleti érték nem egész szám, vagy nem esik a **-50..50** intervallumba. Hiba esetén a program azzal jelzi a hibát, hogy újra kérdezi azt.

* + - 1. Mintafutás hibás bemeneti adatok esetén:



1. Fejlesztői dokumentáció
   1. Feladat
   2. 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 azt a települést, ahol a legmagasabb hőmérséklet várható!
   3. Specifikáció

**Bemenet**: N∈N, M∈N, H∈ZNxM

**Kimenet**: Db∈N, T∈ NN

**Előfeltétel**: N∈[1..1000] ∧ M∈[1..1000] ∧ ∀i∈[1..N]: ∀j∈[1..M]: Hij∈[-50..50]

**Utófeltétel**: Db=

∧

∀k∈[1..Db]: LegMelegebbVaros(Tk)

**Definíció**: LegMelegebbVaros: N→L  
LegMelegebbVaros(i):= ∃j∈[1..M]:Hij = LegNagyobbHőm(H)

LegNagyobbHőm: ZNxM→Z

∃x∈[1..N], y∈[1..M]: LegNagyobbHőm(H) = Hxy

∧

~~∃~~z∈[1..N], w∈[1..M]: LegNagyobbHőm(H) < Hzw

* 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 IDJJQ2 nevű könyvtárban található meg. A fej­lesztés során használt könyvtár-struktúra:

|  |  |
| --- | --- |
| Állomány | Magyarázat |
| IDJJQ2\bin\IDJJQ2.exe | futtatható kód |
| IDJJQ2\src\main.cpp | C++ forráskód |
| IDJJQ2\tests\test1.txt | teszt-bemeneti fájl1 |
| IDJJQ2\tests\test2.txt | teszt-bemeneti fájl2 |
| IDJJQ2\tests\test3.txt | teszt-bemeneti fájl3 |
| IDJJQ2\tests\test4.txt | teszt-bemeneti fájl4 |
| IDJJQ2\tests\test5.txt | teszt-bemeneti fájl5 |
| IDJJQ2\Docs\IDJJQ2.docx | dokumentáció (ez a fájl) |

* 1. Megoldás
     1. Programparaméterek
        1. Konstans

1. minCountOfCities : **Egész**(1) [a városok (N) minimális száma]
2. maxCountOfCities : **Egész**(1000) [a városok (N) maximális száma]
3. minCountOfDays : **Egész**(1) [a napok (M) minimális száma]
4. maxCountOfDays : **Egész**(1000) [a napok (M) maximális száma]
5. minSampeValue : **Egész**(-50) [a hőmér. mérések minimálisan felvehető értéke]
6. maxSampeValue : **Egész**(50) [a hőmér. mérések maximálisan felvehető értéke]
7. userFriendlyMode : **Logikai**(Igaz) [legyen-e visszacsatolás a felhasználó felé illetve input validálás]
8. verboseMode : **Logikai**(Igaz) [legyen-e üzenet a konzolon az eredményen kívűl]
9. skipInputValdation : **Logikai** (Hamis) [validáljuk-e a bemeneti értékeket]
   * + 1. Típus
10. samples = **Tömb**(1.. maxCountOfCities)(1.. maxCountOfDays):**Egész**  
    inputStruct = **Rekord**(countOfCities,countOfDays:**Egész,** samples)
11. maxTempCities = **Tömb**(1.. maxCountOfCities):**Egész**
12. outputStruct = **Rekord**(maxTempCitiesCount:**Egész,** maxTempCities)
    * + 1. Változó

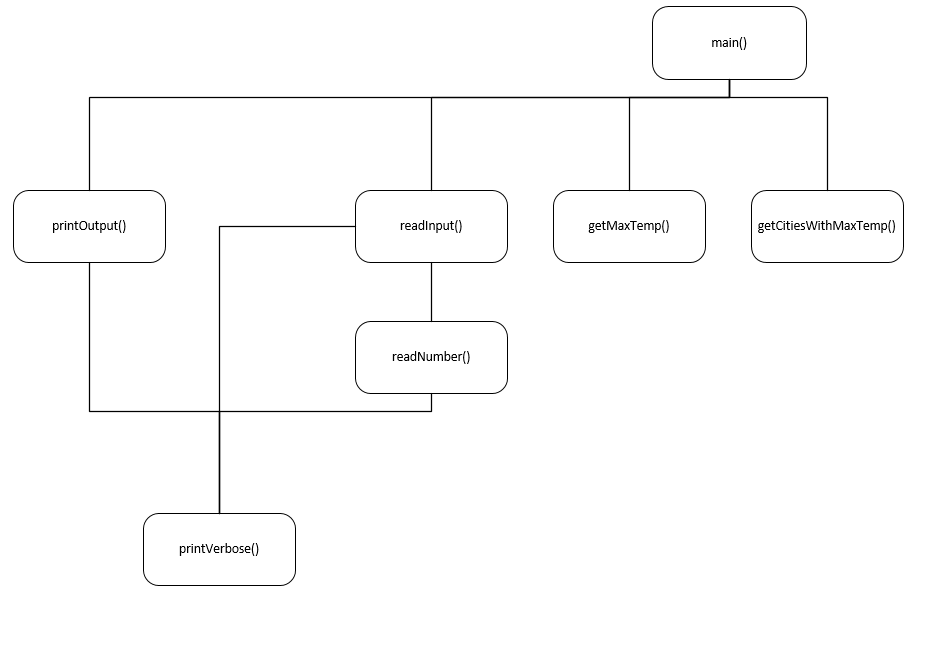
input : inputStruct  
outputStruct : outputStruct  
maxTempValue : **Egész**

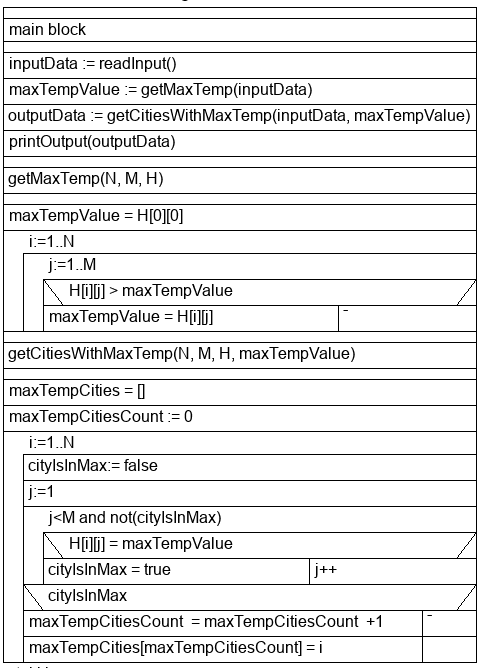
* + 1. Programfelépítés

A program által használt modulok (és helyük):

main.cpp – program, a forráskönyvtárban  
iostream – képernyő-, és billentyűkezelés, a C++ rendszer része  
limits – numerikus limiteket definiáló könyvtár

* + 1. Függvénystruktúra



* + 1. A program lényegi részének az algoritmusa
    2. 
    3. A kód

A forráskód egészét a dokumentáció mellett található main.cpp fájl tartalmazza.

/\*

Szerzo: Koller David Daniel

Neptun: IDJJQ2

E-mail: david.daniel.koller@gmail.com

Feladat: "Legmelegebb telepulesek"

\*/

#include <iostream>

#include <limits>

...

//structure that holds together everything the user will provide

struct inputStruct

{

short countOfCities;

short countOfDays;

signed char samples[maxCountOfCities][maxCountOfDays];

};

//structure that holds together everything that will be printed out as the solution

struct outputStruct

{

short maxTempCitiesCount;

short maxTempCities[maxCountOfCities];

};

//simple function that uses the verboseMode global variable to decided whether to print or not

void printVerbose(string text, bool endline=true)

{

...

}

//reads a number from console, does simple validation, unless overwritten by the skipInputValidation global parameter

short readNumber(short minValue, short maxValue, string repeatText = "")

{

short number;

...

return number;

}

//the method that grabs all the required data from the user and puts them into one record

inputStruct readInput()

{

inputStruct input;

...

return input;

}

//enumerates across every temperature sample and outputs the highest value seen

short getMaxTemp(inputStruct inputData)

{

short maxTempValue;

...

return maxTempValue;

}

//counts the cities where the max temp has been measured and also stores their indexes

outputStruct getCitiesWithMaxTemp(inputStruct inputData, short maxTempValue)

{

outputStruct result;

...

return result;

}

//prints the solution to the output, with some verbose text if verboseMode global var is set that way

void printOutput(outputStruct outputData)

{

...

}

//main method grabs the input, calculated the max. temperature value that has been measured

// then solves and prints out the solution

int main()

{

inputStruct inputData = readInput();

short maxTempValue = getMaxTemp(inputData);

outputStruct outputData = getCitiesWithMaxTemp(inputData, maxTempValue);

printOutput(outputData);

...

return 0;

}

* 1. Tesztelés
     1. Érvényes tesztesetek
        1. teszteset: test1.txt

|  |
| --- |
| Bemenet – *lehető legkevesebb, de valid adat* |
| N = 1  M = 1  Hőmérsékletek11 = 0 |
| Kimenet |
| 1 1 |

* + - 1. teszteset: test2.txt

|  |
| --- |
| Bemenet – *kicsi de valid mátrix két forró várossal* |
| N = 2  M = 2  Hőmérsékletek11 = 0  Hőmérsékletek12 = 30  Hőmérsékletek21 = 30  Hőmérsékletek22 = -10 |
| Kimenet |
| 2 1 2 |

* + - 1. teszteset: test3.txt

|  |
| --- |
| Bemenet – *kicsi de valid mátrix egy forró várossal* |
| N = 2  M = 2  Hőmérsékletek11 = 0  Hőmérsékletek12 = 10  Hőmérsékletek21 = 30  Hőmérsékletek22 = -10 |
| Kimenet |
| 1 2 |

* + - 1. teszteset: test4.txt

|  |
| --- |
| Bemenet – *kicsi de valid mátrix egy forró várossal, duplikált méréssel* |
| N = 2  M = 2  Hőmérsékletek11 = 40  Hőmérsékletek12 = 40  Hőmérsékletek21 = 30  Hőmérsékletek22 = -10 |
| Kimenet |
| 1 1 |

* + 1. Érvénytelen tesztesetek
       1. teszteset: test5.txt

|  |
| --- |
| Bemenet – *Nem szám* |
| N = 11tizenegy |
| Kimenet |
| Újrakérdezés: N = |

* + - 1. teszteset: test6.txt

|  |
| --- |
| Bemenet – *Túl nagy szám* |
| N = 1100 |
| Kimenet |
| Újrakérdezés:  N= |

* + - 1. teszteset: test7.txt

|  |
| --- |
| Bemenet – *Túl kicsi szám* |
| N = -51 |
| Kimenet |
| Újrakérdezés:  N= |

* 1. Fejlesztési lehetőségek

1. Adatok –a felhasználó igénye szerint– akár fájlból is fogadása.
2. A mérések mátrixának beolvasása alatt finomhangolt válaszok a felhasználó felé a hibák esetén
3. Hibás fájl-bemenetek felismerése, és a hiba helyének (sor sorszámának) kiírása.
4. Többszöri futtatás megszervezése
5. Lokalizáció (a program most angol nyelven fut)
6. Performancia javítás: Mivel sohasem kell visszanézni az előző hőmérsékleti értékeket a feladat megoldása közben, működő implementáció lenne, ha egyből a beolvasás közben keresnénk meg a maximális hőmérsékleti értéket és növelnénk a számlálót (meg a tömböt) ha ismétlődést találtunk.

Így a háromszoros iteráció helyett elég lenne egy és a mátrixot sem kell memóriában eltárolni.

Egyből az adott város/nap hőmérséklete beolvasásakor állítanánk vissza a számlálót és nulláznánk a tömböt, nem foglalkoznánk az eddigi hőmérsékletekkel soha sem, csak a max-ot tárolnánk le.

Ellenben így a kód nehezebben átlátható és bővíthető, ezért döntöttem a mostani implementáció mellett. Ha viszont ez nem elsődleges szempont, ellenben növelni kell a minták számát, ez egy opció lehet.

1. Értelemszerűen töltendők itt ki a szerzőre vonatkozó adatok.  
   **A lábjegyzet(ek) a végső dokumentációból törlendő(k)! Csak az Ön segítését szolgálja.** [↑](#footnote-ref-1)