„Programozási alapismeretek”  
beadandó feladat:  
„ProgAlap beadandó feladatok” téma 4. feladat

Készítette: *András Emese*Neptun-azonosító: *T6ESML*E-mail: emcsi7895@gmail.com

Kurzuskód: IP-08PAED

Gyakorlatvezető neve: *László Ildikó*

2015. november 21.

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)

[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)

[Fejlesztési lehetőségek 10](#_Toc410323351)

2. Felhasználói dokumentáció
   1. Feladat

Egy folyón N helyen mérik a vízállást, amit egy referenciamagassághoz képest centiméterben adnak meg. Elsőfokú árvízvédelmi készültséget kell elrendelni, ha a magasság meghaladja a 800 centimétert, másodfokút, ha meghaladja a 900 centimétert és harmadfokút, ha meghaladja az 10 métert. Folyószakasznak nevezzük a leghosszabb egymás mellett levő egyforma tulajdonságú mérésekből álló sorozatokat. Árvíznek nevezzük azt a szakaszt, ahol minden hely legalább elsőfokú készültségű.

* 1. Készíts programot, amely meghatározza a 2 legközelebbi árvízmentes folyószakaszt!
  2. 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 beadando\_4es\bin\Debug\beadando\_4es.exe néven található a tömörített állományban. A beadando\_4es.exe fájl kiválasztásával indítható.

* + 1. A program bemenete

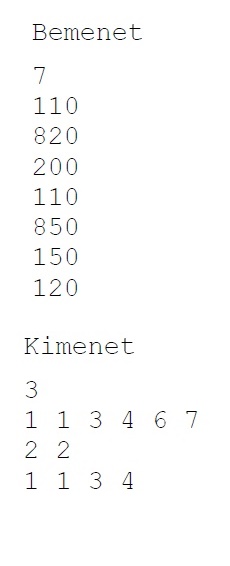
A program az adatokat a **billentyűzet**ről olvassa be a következő sorrendben:

|  |  |  |
| --- | --- | --- |
| # | Adat | Magyarázat |
| **1.** | *N* | A mérési pontok száma (1≤N≤10000). |
| **2.** | *Mérési eredmény1* | Az első mérési eredmény (0≤Mérési eredmény1≤3000). |
| **3.** | *Mérési eredmény 2* | A második mérési eredmény (0≤ Mérési eredmény2≤3000). |
| **...** | *…* |  |
| **N+1.** | *Mérési eredmény N* | Az N-edik mérési eredmény (0≤ Mérési eredményN≤3000). |

* + 1. A program kimenete

A kimenet első sorába az az árvízmentes folyószakaszok számát, a második sorba ezen folyószakaszok kezdetének és végének a sorszáma kerül. Ha nem lenne legalább két árvízmentes folyószakasz, akkor egyetlen nincs több sora a kimenetnek. A harmadik sorba az árvízmentes folyószakaszok távolsága kerül. A negyedik sorba íródik ki a két legközelebbi árvízmentes folyószakasz kezdő- és végsorszáma.

* + 1. Minta bemenet és kimenet



Fejlesztői dokumentáció

* 1. Feladat

Egy folyón N helyen mérik a vízállást, amit egy referenciamagassághoz képest centiméterben adnak meg. Elsőfokú árvízvédelmi készültséget kell elrendelni, ha a magasság meghaladja a 800 centimétert, másodfokút, ha meghaladja a 900 centimétert és harmadfokút, ha meghaladja az 10 métert. Folyószakasznak nevezzük a leghosszabb egymás mellett levő egyforma tulajdonságú mérésekből álló sorozatokat. Árvíznek nevezzük azt a szakaszt, ahol minden hely legalább elsőfokú készültségű.

* 1. Készíts programot, amely meghatározza a 2 legközelebbi árvízmentes folyószakaszt!
  2. Specifikáció

**Bemenet**: N∈N, Mérési eredmények∈N\*, Mérési eredmény > 800: N\* -> L

**Kimenet**: k∈N, min∈N

**Előfeltétel**: N=Darab(Mérési eredmények) ∧ N∈[1..10000] ∧ ∀i∈[1..N]: Mérési eredményeki∈[0..3000], N>0

**Utófeltétel**: k= ∧   
1≤Min≤N ∧   
∀i∈[1≤i≤N]: Mérési eredményekMin ≤ Mérési eredményeki

**Megjegyzés**: a „ha kevesebb, mint 2 ilyen van” kitételt 4 helyett 2 sor kiírásával jelzi a program.

* 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 A1B2C3 nevű könyvtárban található meg. A fej­lesztés során használt könyvtár-struktúra:

|  |  |  |  |
| --- | --- | --- | --- |
| Állomány | | Magyarázat | |
| beadando\_4es\bin\Debug\beadando\_4es.exe | | futtatható kód | |
| beadando\_4es\obj\Debug\main.o | | félig lefordított kód | |
|  | |  | |
| beadando\_4es\main.cpp | | C++ forráskód | |
|  | |  | |
| beadando\_4es\be1.txt | | teszt-bemeneti fájl1 | |
| beadando\_4es\be2.txt | | teszt-bemeneti fájl2 | |
| beadando\_4es\ki1.txt | | teszt-kimeneti fájl1 | |
| beadando\_4es\ki2.txt | | teszt-kimeneti fájl2 | |
|  | |  | |
| beadando\_4es\beadando\_4es.docx | | dokumentációk (ez a fájl) | |

* 1. Megoldás
     1. Programparaméterek
        1. Konstans

1. MaxN : **Egész**(10000) [a mérések maximális száma]
   * + 1. Típus
2. TAdatok = **Tömb**(1..MaxN:**Egész**)
   * + 1. Változó

N : **Egész**  
k : TAdatok

* + 1. Programfelépítés

A program által használt modulok (és helyük):

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

* + 1. Függvénystruktúra

Főprogram

negyedik

Beolvas

harmadik

* + 1. A teljes program algoritmusa

Főprogram:

Árvízmentes folyószakaszok

Beolvas(Adatok,N)

harmadik(seged,k,seged3,db3)

negyedik(min,seged3,db3)

Alprogramok:

,,,

negyedik(Változó min:Egész, Változó db3:Egész, magok:TMérési eredmények

harmadik(Változó k:Egész, Változó db3:Egész, magok:TMérési eredmények

,,,

Beolvas(Változó N:Egész, magok:TMérési eredmények

Be:N [1<N<MaxN+1

Be:magok[1..N] [0<=magok[1..N]<MaxMagasság+1

* + 1. A kód

A main.cpp fájl tartalma:

* 1. #include<iostream>
  2. using namespace std;
  3. const int MaxN = 10000;
  4. void Beolvas(int Adatok[],int&);
  5. int harmadik(int seged[],int,int seged3[],int&);
  6. int negyedik(int&,int seged3[],int);
  7. int main()
  8. {
  9. int Adatok[MaxN];
  10. int N;
  11. Beolvas(Adatok,N);
  12. int i = 0; //szakasz elejének indexe
  13. int j = 0; //szakasz végének indexe
  14. int seged[MaxN]; //második sor elemei
  15. int k = 0; //második sor elemeinek indexe
  16. while(i < N)
  17. {
  18. j = i + 1;
  19. if(Adatok[i] <= 800)
  20. {
  21. j = i;
  22. while(j < N && Adatok[j] <= 800) //j a szakasz vége, keressük h meddig kisebb 800-nál
  23. {
  24. j++;
  25. }
  26. seged[k] = i+1; //második sor k-adik elemének értéke, szakasz eleje
  27. k++; //következõ értéket (szakasz vége) új indexhez írja, ne írja felül
  28. seged[k] = j; //második sor k+1-edik elemének értéke, szakasz vége
  29. k++; //következõ szakasz elejét új indexhez írja
  30. }
  31. i = j; //a szakasz eleje (i) a következõ szakasz vége (j) lesz, onnantól vizsgáljuk tovább a magasságot, a kettõ közöttiekrõl tudjuk h folyószakasz értékei (ezért nem léptetjük simán)
  32. }
  33. //első sor
  34. cout << k/2 << endl; //elsõ sor: szakaszok eleje és vége is benne van a k-ban, k fele a szakaszok száma
  35. //második sor
  36. for(int i = 0; i < k; ++i)
  37. {
  38. cout << seged[i] << " "; //második sor: szakaszok elejét és végét kiírjuk
  39. }
  40. //harmadik sor
  41. if(k/2 > 1) //csak akkor írja ki ha min 2 szakasz van
  42. {
  43. cout << endl;
  44. int seged3[MaxN]; //harmadik sor elemei
  45. int db3 = 0; //haramdik sor elemeinek indexe
  46. harmadik(seged,k,seged3,db3);
  47. for(int i = 0; i < db3; ++i)
  48. {
  49. cout << seged3[i] << " ";
  50. }
  51. cout << endl;
  52. //negyedik sor
  53. int min = 0;
  54. negyedik(min,seged3,db3);
  55. cout << seged[2\*min] << " " << seged[2\*min+1] << " "; //elsõ szakasz: egy szakasz 2 indexet foglal, a páros (2\*min) az eleje (0-tól indul), a páratlan (2\*min+1) a vége
  56. cout << seged[2\*min+2] << " " << seged[2\*min+3]; //második szakasz: csak 2 szakaszt írunk ki
  57. }
  58. }
  59. void Beolvas(int Adatok[MaxN],int &N)
  60. {
  61. cin >> N;
  62. for(int i = 0; i < N; ++i)
  63. {
  64. cin >> Adatok[i];
  65. }
  66. }
  67. int harmadik(int seged[MaxN],int k,int seged3[MaxN],int& db3)
  68. {
  69. for(int i = 2; i < k; ++i) //3. értéktõl, 2. szakasz elejétõl vizsgáljuk
  70. {
  71. seged3[db3] = seged[i]-seged[i-1]; //harmadik sor: következõ szakasz elejébõl kivonjuk az elõzõ szakasz végét
  72. db3++; //következõ távolságot új indexhez írja
  73. i++; //még egyszer léptetjük h a következő szakasz elejétõl vizsgálja, különben az adott szakasz végét nézné csak egyszer léptetve
  74. }
  75. return db3;
  76. }
  77. int negyedik(int& min,int seged3[],int db3)
  78. {
  79. for(int i = 1; i < db3; ++i) //megvizsgáljuk az egymás melletti távolságokat
  80. {
  81. if(seged3[i] < seged3[min]) //ha pl a 2. távolság kisebb az 1.-nél (ha két legkisebb van, akkor csak az elsõt nézi, mert nem engedjük meg az egyenlõséget)
  82. {
  83. min = i; //akkor a 2. lesz az eddigi legkisebb
  84. }
  85. }
  86. return min;
  87. }
  88. Tesztelés
      1. Érvényes tesztesetek
         1. teszteset:

|  |
| --- |
| Bemenet – *több mint 2 árvízmentes folyószakasz* |
| N = 7  Mérési eredmény1 = 110  Mérési eredmény2 = 820  Mérési eredmény3 = 200  Mérési eredmény4 = 110  Mérési eredmény5 = 850  Mérési eredmény6 = 150  Mérési eredmény7 = 120 |
| Kimenet |
| 3  1 1 3 4 6 7  2 2  1 1 3 4 |

* + - 1. teszteset: be1.txt

|  |
| --- |
| Bemenet – *több mint 2 árvízmentes folyószakasz* |
| N = 30  Mérési eredmény1 = 814  Mérési eredmény2 = 1187  Mérési eredmény3 = 1289  Mérési eredmény4 = 1966  Mérési eredmény5 = 1273  Mérési eredmény6 = 1348  Mérési eredmény7 = 1524  Mérési eredmény8 = 895  Mérési eredmény9 = 1743  Mérési eredmény10 = 71  Mérési eredmény11 = 658  Mérési eredmény12 = 685  Mérési eredmény13 = 1229  Mérési eredmény14 = 1446  Mérési eredmény15 = 1168  Mérési eredmény16 = 917  Mérési eredmény17 = 1610  Mérési eredmény18 = 1302  Mérési eredmény19 = 1768  Mérési eredmény20 = 1600  Mérési eredmény21 = 1592  Mérési eredmény22 = 1738  Mérési eredmény23 = 1585  Mérési eredmény24 = 1846  Mérési eredmény25 = 1772  Mérési eredmény26 = 1475  Mérési eredmény27 = 858  Mérési eredmény28 = 913  Mérési eredmény29 = 1598  Mérési eredmény30 = 1765 |
| Kimenet |
| 1  10 12 |

* + - 1. teszteset: be2.txt

|  |
| --- |
| Bemenet – *több mint 2 árvízmentes folyószakasz* |
| N = 100 |
| Mérési eredmény1 = 586  Mérési eredmény2 = 523  Mérési eredmény3 = 899  Mérési eredmény4 = 837  Mérési eredmény5 = 1088  Mérési eredmény6 = 980  Mérési eredmény7 = 1120  Mérési eredmény8 = 1627  Mérési eredmény9 = 1891  Mérési eredmény10 = 1489  Mérési eredmény11 = 1880  Mérési eredmény12 = 653  Mérési eredmény13 = 491  Mérési eredmény14 = 210  Mérési eredmény15 = 283  Mérési eredmény16 = 365  Mérési eredmény17 = 1390  Mérési eredmény18 = 976  Mérési eredmény19 = 1359  Mérési eredmény20 = 889  Mérési eredmény21 = 1792  Mérési eredmény22 = 1791  Mérési eredmény23 = 1420  Mérési eredmény24 = 567  Mérési eredmény25 = 189  Mérési eredmény26 = 350  Mérési eredmény27 = 646  Mérési eredmény28 = 661  Mérési eredmény29 = 1111  Mérési eredmény30 = 1062  Mérési eredmény31 = 1295  Mérési eredmény32 = 49  Mérési eredmény33 = 102  Mérési eredmény34 = 1303  Mérési eredmény35 = 1560  Mérési eredmény36 = 114  Mérési eredmény37 = 625  Mérési eredmény38 = 24  Mérési eredmény39 = 269  Mérési eredmény40 = 1203  Mérési eredmény41 = 782  Mérési eredmény42 = 24  Mérési eredmény43 = 1028  Mérési eredmény44 = 1689  Mérési eredmény45 = 1053  Mérési eredmény46 = 1209  Mérési eredmény47 = 483  Mérési eredmény48 = 167  Mérési eredmény49 = 1618  Mérési eredmény50 = 1764  Mérési eredmény51 = 1765  Mérési eredmény52 = 1063  Mérési eredmény53 = 1595  Mérési eredmény54 = 1962  Mérési eredmény55 = 1614  Mérési eredmény56 = 1003  Mérési eredmény57 = 753  Mérési eredmény58 = 714  Mérési eredmény59 = 933  Mérési eredmény60 = 1772  Mérési eredmény61 = 342  Mérési eredmény62 = 181  Mérési eredmény63 = 707  Mérési eredmény64 = 107  Mérési eredmény65 = 1947  Mérési eredmény66 = 1670  Mérési eredmény67 = 1196  Mérési eredmény68 = 863  Mérési eredmény69 = 983  Mérési eredmény70 = 1632  Mérési eredmény71 = 1430  Mérési eredmény72 = 1538  Mérési eredmény73 = 1004  Mérési eredmény74 = 1398  Mérési eredmény75 = 464  Mérési eredmény76 = 610  Mérési eredmény77 = 1007  Mérési eredmény78 = 1980  Mérési eredmény79 = 928  Mérési eredmény80 = 1916  Mérési eredmény81 = 1672  Mérési eredmény82 = 1957  Mérési eredmény83 = 1772  Mérési eredmény84 = 338  Mérési eredmény85 = 45  Mérési eredmény86 = 641  Mérési eredmény87 = 681  Mérési eredmény88 = 1274  Mérési eredmény89 = 467  Mérési eredmény90 = 643  Mérési eredmény91 = 1653  Mérési eredmény92 = 1524  Mérési eredmény93 = 1791  Mérési eredmény94 = 39  Mérési eredmény95 = 800  Mérési eredmény96 = 801  Mérési eredmény97 = 1824  Mérési eredmény98 = 1781  Mérési eredmény99 = 1926  Mérési eredmény100 = 233 |
| Kimenet |
| 14  1 2 12 16 24 28 32 33 36 39 41 42 47 48 57 58 61 64 75 76 84 87 89 90 94 95 100 100  10 8 4 3 2 5 9 3 11 8 2 4 5  36 39 41 42 |

* + - 1. teszteset: be4.txt

|  |
| --- |
| Bemenet – *kevesebb, mint 2 árvízmentes folyószakasz* |
| N = 7  Mérési eredmény1 = 810  Mérési eredmény2 = 820  Mérési eredmény3 = 200  Mérési eredmény4 = 110  Mérési eredmény5 = 850  Mérési eredmény6 = 950  Mérési eredmény7 = 820 |
| Kimenet |
| 1  3 4 |

* + - 1. teszteset:

|  |
| --- |
| Bemenet – *nincs árvízmentes folyószakasz* |
| N = 7  Mérési eredmény1 = 810  Mérési eredmény2 = 820  Mérési eredmény3 = 900  Mérési eredmény4 = 810  Mérési eredmény5 = 850  Mérési eredmény6 = 950  Mérési eredmény7 = 820 |
| Kimenet |
| 0 |

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. A bemeneti sorozat grafikus megjelenítése, s az eredmény-szigetek elütő színű kijelzése.