

Programozási alapismeretek
beadandó feladat

Árvíz: 14.

Készítette: Prorok Ernest

Neptun-azonosító: FILSPA

E-mail: ernest.prorok@gmail.com

Kurzuskód: IP-08PAEG/5

Gyakorlatvezető neve: Menyhárt László Gábor

2015. december 6.

Tartalom

Felhasználói dokumentáció.....	3
Feladat.....	3
Környezet.....	3
Használat	3
A program indítása	3
A program bemenete	3
A program kimenete.....	4
Minta bemenet és kimenet	4
Hibalehetőségek:.....	5
Fejlesztői dokumentáció.....	6
Feladat.....	6
Specifikáció.....	6
Környezet.....	6
Forráskód.....	6
Megoldás	7
Tesztelés	16
Fejlesztési lehetőségek.....	17

Felhasználói dokumentáció

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 a 10 métert. Folyószakasznak nevezzük a leghosszabb egymás mellett levő egyforma tulajdonságú mérésekből álló sorozatokat. Árvíznek nevezzük azt a szakaszt, ahol minden hely legalább elsőfokú készültséggű. Készíts programot, amely meghatározza azokat az árvizeket, melyeken belül volt árvízvédelmi készültség csökkenés!

Sorszám	Feladat szövege
1. sor	A standard kimenet első sorába az árvizek K darabszámát kell írni (0, ha nincs ilyen, ebben az esetben nincsenek további sorok)!
2. sor	A második sorba ennek a K árvíznek a kezdete és vége kerüljön!
3. sor	A 3. sorba az árvízvédelmi készültség csökkenést tartalmazó árvizek M számát kell írni (0, ha nincs ilyen, ebben az esetben nincsenek további sorok)!
4. sor	Végül a 4. sor ennek az M árvíznek a kezdetét és végét tartalmazza!

Környezet

IBM PC, exe futtatására alkalmas operációs rendszer (pl. Windows XP). Nem igényel egeret.

Használat

A program indítása

A program a bead\bin\Debug\bead.exe néven található a tömörített állományban. A bead.exe fájl kiválasztásával indítható.

A program bemenete

A program az adatokat a billentyűzetről vagy fájlból olvassa be a következő sorrendben:

Sorindex	Adat	Magyarázat
1.	<i>bemód</i>	A beolvasás módja (fájlból olvasás esetén 2, billentyűzetről olvasás esetén 1)
2.	<i>maunál</i>	mért pontok száma, majd az értékek sorrendben cm-ben)

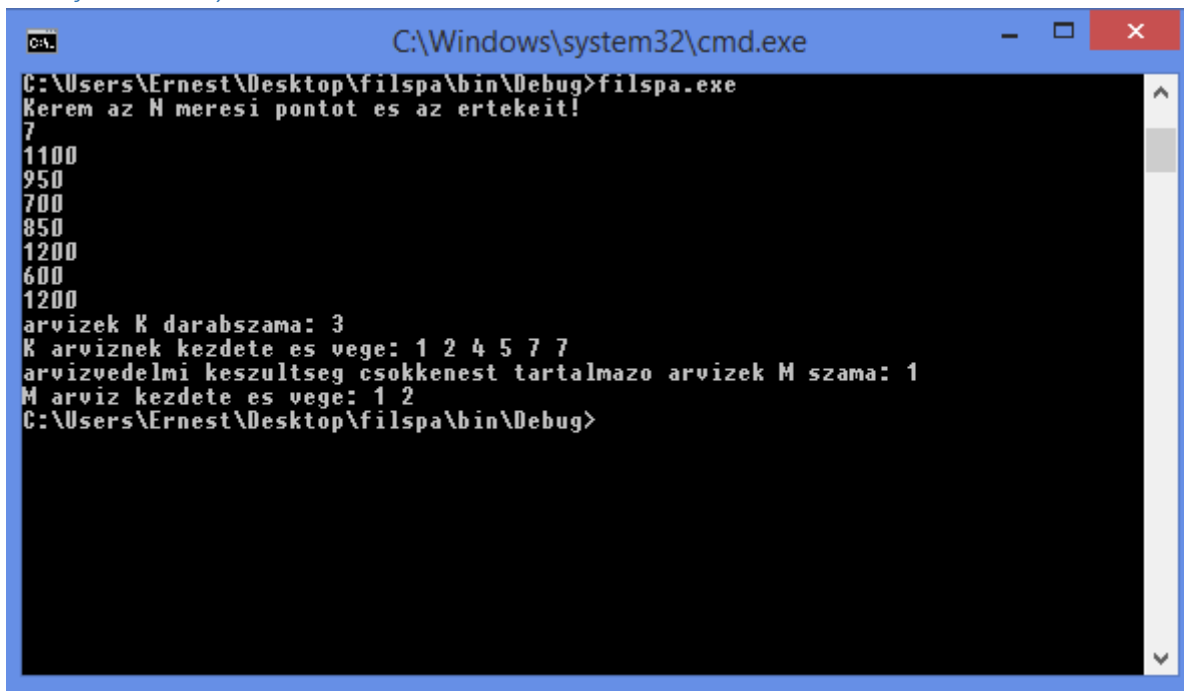
A standard bemenet első sorában a mérési pontok száma szerepel (1≤N≤10 000), a következő N sor mindegyike egy egész számot tartalmaz, a mérési eredményt (0≤A i ≤3000).

A program kimenete

A standard kimenet első sorába az árvizek K darabszámát kell írni (0, ha nincs ilyen, ebben az esetben nincsenek további sorok)! A második sorba ennek a K árvíznek a kezdete és vége kerüljön! A 3. sorba az árvízvédelmi készütség csökkenést tartalmazó árvizek M számát kell írni (0, ha nincs ilyen, ebben az esetben nincsenek további sorok)! Végül a 4. sor ennek az M árvíznek a kezdetét és végét tartalmazza!

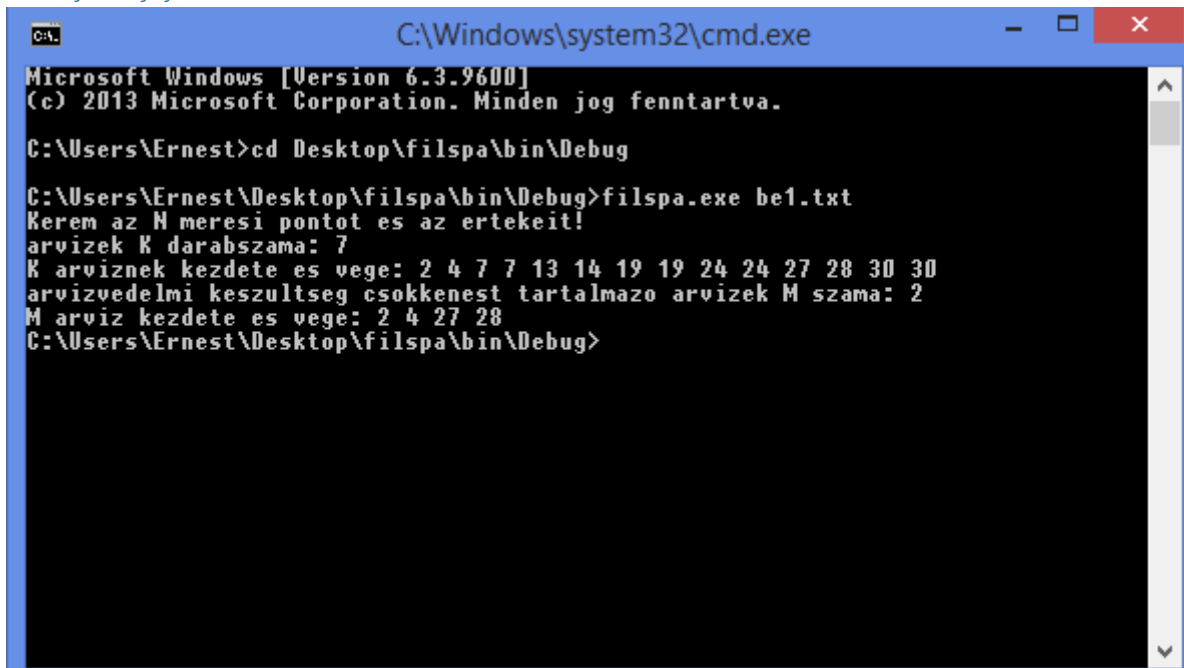
Minta bemenet és kimenet

Minta futás billentyűzetről való beolvasás esetén:



```
C:\Windows\system32\cmd.exe
C:\Users\Ernest\Desktop\filspa\bin\Debug>filspa.exe
Kerem az N meresi pontot es az ertekeit!
7
1100
950
700
850
1200
600
1200
arvizek K darabszama: 3
K arviznek kezdete es vege: 1 2 4 5 7 7
arvizvedelmi keszultseg csokkenest tartalmazo arvizek M szama: 1
M arviz kezdete es vege: 1 2
C:\Users\Ernest\Desktop\filspa\bin\Debug>
```

Minta futás fájlból való beolvasás esetén:



```
C:\Windows\system32\cmd.exe
Microsoft Windows [Version 6.3.9600]
(c) 2013 Microsoft Corporation. Minden jog fenntartva.

C:\Users\Ernest>cd Desktop\filspa\bin\Debug

C:\Users\Ernest\Desktop\filspa\bin\Debug>filspa.exe be1.txt
Kerem az N meresi pontot es az ertekeit!
arvizek K darabszama: 7
K arviznek kezdete es vege: 2 4 7 7 13 14 19 19 24 24 27 28 30 30
arvizvedelmi keszultseg csokkenest tartalmazo arvizek M szama: 2
M arviz kezdete es vege: 2 4 27 28
C:\Users\Ernest\Desktop\filspa\bin\Debug>
```

Hibalehetőségek:

Minta futás hibás bemeneti adatok esetén:

```
Kerem a mert helyek szamat!  
Kerem a mert helyek vizallasat sorrendben, cm-ben!  
@  
7  
1100s  
850d  
450a  
900s  
850b  
asd100  
500  
622  
2  
1 2 4 5  
2  
1 2 4 5  
Process returned 0 (0x0)    execution time : 53.685 s  
Press any key to continue.
```

Túlcsordulás:

[illegible]

Fejlesztői dokumentáció

Feladat

Egy folyón N helyen méri 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 a 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. Készíts programot, amely meghatározza azokat az árvizeket, melyeken belül volt árvízvédelmi készültség csökkenés!

Sorszám	Feladat szövege
1. sor	A standard kimenet első sorába az árvizek K darabszámát kell írni (0, ha nincs ilyen, ebben az esetben nincsenek további sorok)!
2. sor	A második sorba ennek a K árvíznek a kezdete és vége kerüljön!
3. sor	A 3. sorba az árvízvédelmi készültség csökkenést tartalmazó árvizek M számát kell írni (0, ha nincs ilyen, ebben az esetben nincsenek további sorok)!
4. sor	Végül a 4. sorba ennek az M árvíznek a kezdetét és végét tartalmazza!

Specifikáció

Bemenet: $N * MAXN \in \mathbb{Z}$, $meres \in MAXN^N$, $T:H \rightarrow L$

Előfeltétel: $N > 0$ és $\exists i(1 \leq i \leq N): T(meres[i])$ és $megB[i] \neq 0$ és $megD[i] \neq 0$

A feladat

Kimenet: $megA \in \mathbb{Z}$

Utófeltétel: $Db = \sum_{i=1}^N 1$

$meres[j] > 800 \ \&\& \ j < n$ és $[meres[i] > 800]$ és $db > 0$

B feladat

Kimenet: $megB \in \mathbb{Z}$

Utófeltétel: $1 \leq k \leq N$ és $Db = \sum_{i=1}^N 1$

$meres[i] > 800$ és $meres[j] > 800 \ \&\& \ j < n$ és $meres[i] > 800$ és $db > 0$

C feladat

Kimenet: $megC \in \mathbb{Z}$

Utófeltétel: $Db = \sum_{i=1}^N 1$

$i < n$ és $i < n \ \&\& \ meres[i] \leq 800$ és $i < n \ \&\& \ meres[i] > 800$ és $!kategValtas \ \&\& \ getArvizKat(meres[i]) < getArvizKat(meres[i-1])$ és $kategValtas$

D feladat

Kimenet: $megD \in \mathbb{Z}$

Utófeltétel: $1 \leq kezdInd \leq N$ és $Db = \sum_{i=1}^N 1$

$i < n$ és $i < n \ \&\& \ meres[i] \leq 800$ és $i < n \ \&\& \ meres[i] > 800$ és $!kategValtas \ \&\& \ getArvizKat(meres[i]) < getArvizKat(meres[i-1])$ és $kategValtas$

Környezet

IBM PC, exe futtatására alkalmas operációs rendszer (pl. Windows XP). C++ fordítóprogram (gcc v4.4.1), Code::Blocks fejlesztői környezet.

Forráskód

A teljes fejlesztői anyag a bead nevű könyvtárban található meg. A fejlesztés során használt könyvtár-struktúra:

Állomány	Magyarázat
<i>bead\bin\Debug\filspa.exe</i>	nyomkövethető állapotú futtatható kód
<i>bead\obj\Debug\main.o</i>	nyomkövethető állapotú, félig lefordított (object-) kód
<i>bead\filspa.cbp</i>	projekt fájl,
<i>bead\main.cpp</i>	C++ forrás
<i>bead\filspa.layout</i>	layout file
<i>bead\Debug\be1.txt</i>	tesztfájl#1
<i>bead\Debug\be2.txt</i>	tesztfájl#2
<i>bead\filspa.depend</i>	depend file
<i>bead\main.o</i>	nyomkövethető állapotú, félig lefordított (object-) kód
<i>bead\main.exe</i>	nyomkövethető állapotú futtatható kód
<i>bead\dokumentacio.pdf</i>	a dokumentáció

Megoldás

Programértékek

Konstans

MAXN:egész [a mért pontok max száma]

Típus

-

Változó

meres tömb[MAXN]

megB, MegD tömb[MAXN]

n egész

megA, megC egész

felA, felB, felC, felD tömb[meres]

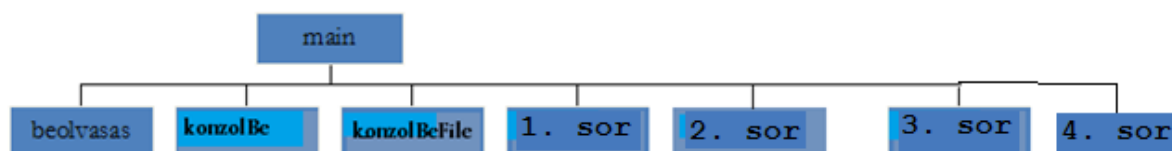
Programfelépítés

A program által használt modulok (és helyük):

bead – main.cpp

iostream, fstream, cstdlib – a C++ rendszer része.

Függvénystruktúra



Algoritmus

Az algoritmizálás szempontjából a részfeladatokat megoldó alprogramok érdekesek: Ezek algoritmusai a következők:

1. sor (felA): megszámlálás és eldöntés

Db:=0	
i=1..N	
meres[i]>800	
Db:=Db+1	

result:=0	
i=1..N	
Db>0	
result:=result+1	

j:=i	
meres[j]>800 && j<n	
j:j+1	-

2. sor (felB): kiválogatás és keresés

Db:=0	
i=1..N	
meres[i]>800	
Db:=Db+1	

k:=0	
i:=j	
i=1..N	
Db>0	
k:=k+1	-
meg[B]:=i	

3. sor (felC): megszámlálás és eldöntés

result:=0	
i=1..N	
kategvaltas	
result:=result+1	

j:=i	
i=1..N	
meres[j]>800 && j<N	
j:=j+1	

i:=0	
i<N és meres[i]<=800 vagy meres[i]>800	
i:=i+1	-

4. sor (felD): kiválogatás és keresés

i:=0	
i<N && meres[i]<=800	
i=i+1	
kategvaltas	
meg[D]:=i	-

k:=0	
i=1..N	
meres[i]>800	
k:=k+1	-
meg[D]:=kezdInd	

[A kód](#)

//Prorok Ernest

//FILSPA

//14. feladat

#include <iostream>

```
#include <fstream>

#include <cstdlib>


using namespace std;


const int MAXN = 10000;


void beKonz(int &n, int meres[]);
void beFile(char * filename, int &n, int meres[]);


int getArvizKat(const int meres);


int felA(const int n, const int meres[]);
void felB(const int n, const int meres[], int megB[]);
int felC(const int n, const int meres[]);
void felD(const int n, const int meres[], int megD[]);


int main(int argc, char * argv[])
{
    int n;
    int meres[MAXN];


    int megA;
    int megB[MAXN] = {0};
    int megC;
    int megD[MAXN] = {0};
    cout<<"Kerem az N meresi pontot es az ertekeit! "<<endl;


    if (argc > 1) beFile(argv[1], n, meres);
    else beKonz(n, meres);
```

```

megA = felA(n, meres);

felB(n, meres, megB);

megC = felC(n, meres);

felD(n, meres, megD);


cout << "arvizek K darabszama: " << megA << endl;

cout << "K arviznek kezdete es vege: ";

for (int i = 0; i < n; i++) {
    if (megB[i] != 0) cout << megB[i] << " ";
} cout << endl;


cout << "arvizvedelmi keszultseg csokkenest tartalmazo arvizek M szama: " << megC << endl;

cout << "M arviz kezdete es vege: ";

for (int i = 0; i < n; i++) {
    if (megD[i] != 0) cout << megD[i] << " ";
}


return 0;
}


void beKonz(int &n, int meres[]) {
    bool hiba;

    do {
        cin >> n;

        hiba = n < 0 || n > MAXN || cin.fail();

        if (hiba) {
            cin.clear();
            cin.ignore(1000, '\n');

```

```

    }
} while(hiba);

for (int i = 0; i < n; i++) {
    do {
        cin >> meres[i];

        hiba = meres[i] < 0 || meres[i] > 3000 || cin.fail();

        if (hiba) {
            cin.clear();
            cin.ignore(1000, '\n');
        }
    } while(hiba);
}
}

```

```

void beFile(char * filename, int &n, int meres[]) {
    ifstream file;
    file.open(filename);

    if (file.is_open()) {
        file >> n;

        for (int i = 0; i < n; i++) {
            file >> meres[i];
        }

        file.close();
    } else exit(1);
}

```

```
int getArvizKat(const int meres) {  
  
    if (meres <= 800) return 0;  
    else if (meres < 900) return 1;  
    else if (meres < 1000) return 2;  
  
    return 3;  
}
```

```
int felA(const int n, const int meres[]) {  
    int result = 0;  
    int j;  
    int db = 0;  
  
    for (int i = 0; i < n; i++) {  
        j = i;  
  
        while (meres[j] > 800 && j < n) {  
            if (meres[i] > 800) db++;  
  
            j++;  
        }  
  
        i = j;  
  
        if (db > 0) {  
            result++;  
            db = 0;  
        }  
    }
```

```

    }

    return result;
}

void felB(const int n, const int meres[], int megB[]) {
    int j;
    int k = 0;
    int db = 0;

    for (int i = 0; i < n; i++) {
        j = i;

        if (meres[i] > 800) {
            megB[k] = i+1;
            k++;
        }

        while (meres[j] > 800 && j < n) {
            if (meres[i] > 800) db++;

            j++;
        }

        i = j;

        if (db > 0) {
            megB[k] = i;
            k++;
            db = 0;
        }
    }
}

```

```
    }  
}
```

```
int felC(const int n, const int meres[]) {  
    int result = 0;  
    int i = 0;  
    bool kategValtas = false;  
  
    while (i < n) {  
        while (i < n && meres[i] <= 800) {  
            kategValtas = false;  
            i++;  
        }  
  
        while (i < n && meres[i] > 800) {  
            if (!kategValtas && getArvizKat(meres[i]) < getArvizKat(meres[i-1])) kategValtas = true;  
  
            i++;  
        }  
  
        if (kategValtas) result++;  
    }  
  
    return result;  
}
```

```
void felD(const int n, const int meres[], int megD[]) {  
    int i = 0;  
    int k = 0;  
    int kezdInd;  
    bool kategValtas = false;
```

```

while (i < n) {
    while (i < n && meres[i] <= 800) {
        kategValtas = false;
        i++;
    }

    kezdInd = i+1;

    while (i < n && meres[i] > 800) {

        if (!kategValtas && getArvizKat(meres[i]) < getArvizKat(meres[i-1])) {
            kategValtas = true;
        }

        i++;
    }

    if (kategValtas) {
        megD[k] = kezdInd;
        k++;

        megD[k] = i;
        k++;
    }
}

```

Tesztelés

Érvényes teszteset

Bemenet
N = 7

lehet akár egy sorba is	1100 950 700 850 1200 600 1200
Kimenet	
1	3
2	1 2 4 5 7 7
3	1
4	1 2

Érvénytelen teszteset

Bemenet	
N = 0	
Kimenet	
összes	0
sor	0

Fejlesztési lehetőségek

Amennyiben a felhasználó szeretné saját maga is létrehozhatna közvetlenül a programon keresztül állományokat, melynek adatait saját magunk adjuk meg a programon keresztül, illetve lehetőség lenne generált (random) adatokkal való munkára.