

EAF I Feladat dokumentáció 2.feladatcsoport 3. házi feladathoz

Feladat

Határozzuk meg egy egész számokat tartalmazó vektor legkisebb, k -val osztható értékét!

Specifikáció

$V = vect(Z, Z)$

$A = V \times N_0 \times Z \times Z \times L \times Z$
 $\quad \quad \quad v \quad \quad i \quad \quad n \quad \quad min \quad \quad l \quad \quad k$

$B = V \times N \times Z$
 $\quad \quad \quad v' \quad \quad n' \quad \quad k'$

$Q = (v = v' \wedge n = n' \wedge m \leq n + 1 \wedge k > 0)$

$R = (Q \wedge l = (i \in [v.lob, v.hib] : \beta(i)) \wedge l \rightarrow (ind \in [v.lob, v.hib] : \beta(ind) \wedge min = f(ind) \wedge$
 $\forall i \in [v.lob, v.hib] : \beta(i) \rightarrow (f(i) \geq min)))$

$\beta(i) = \text{maradék} = 0, \text{ ha } (v[i] \% k)$
 $\text{maradék} > 0, \text{ ha } !(v[i] \% k)$

Absztrakt program

l, i := hamis, v.lob			
i ≠ n			
$(v[i] \% k) == 0$	$(v[i] \% k) == 0 \wedge l$	$(v[i] \% k) == 0 \wedge !l$	
SKIP	$v[i] < min$	$min := v[i]$ $ind := i$ $l := igaz$	
	$min := v[i]$ $ind := i$		
i := i + 1			

Tesztelési terv

Tesztesetek a feladat alapján (fekete doboz tesztelés)

1. Bevitelek tesztelése
2. Fájlból felolvasás tesztelése, elsőnek a vect mérete kell, majd space-el elválasztva a számjegyek, ennek is helyességét ellenőrizve.

Tesztesetek a kód alapján (fehér doboz tesztelés)

1. Olyan számadatok (helyes és hibás) megadása, amellyel a beolvasó ellenőrző ágát letesztelhetjük.
2. Hibás adat esetén, új adat bekérése

Megoldás C++

```
// hazi_II_3.cpp : Defines the entry point for the console application.
// Határozzuk meg egy egész számokat tartalmazó vektor legkisebb, k-val
osztható értékét!
#include <fstream>
#include <string>
#include <conio.h>
#include <stdio.h>
#include <iostream>
using namespace std;

int getNum(bool end);           //Ellenőrzöm, hogy csak számot írjon be a flsz.
megadhatom, hogy többkaraktert, vagy csak let vihet fel! Ha több számjegy is
lehet, akkor 0-t megengedem, ha csak 1 számjegy, akkor 0 nincs megengedve!
int checkNum(string tmp, char kezd); //csak szám lehet! itt végzem el a
string vizsgálatát
int* getTMB(int &n, bool file, bool klav, string filename);
bool checkTMB(int v[], int n, int act);
int main(int argc, char* argv[])
{
    string con;
    do{
        string filename;
        bool file = false, klav = false;
        if(argv[ 1 ] != "" && argc >= 2)
        {
            cout << "A fajl neve := ";
            filename = argv[ 1 ];
            cout << argv[ 1 ] << endl;
            file = true;
        }
        else
        {
            do
            {
                string str;
                cout << "Tomb feltoltese billenyuzetrol, \nakarja? (
I/N ) := ";

                cin >> str;
                char * copySTR = _strdup( str.c_str() );
                str = _strupr( copySTR );
                if(str == "I")
                {
                    klav = true;
                }
                else if(str == "N")
                {
                    file = true;
                }
            }while(file == false && klav == false);
        }
        int n = 0;
        int* v = getTMB(n, file, klav, filename );// a vector feltoltese
        //Kiírjuk a tömb elemeit a szabványos kimenetre.
        cout << "A tomb hossza: " << n << endl;
        cout << "A tomb elemei: ";
        for (int i = 0; i < n - 1; ++i) cout << v[ i ] << ", ";
        cout << v[ n - 1 ] << endl;
        //Osztó bekérése
        cout << "Az osztó := ";
        int k = getNum(false); //bekerem az osztót!!!
    }
}
```

```

//Ellenőrizzük az előfeltételt
if (n < 1){
    cout << "A tomb ures!";
    exit(2);
}
//Minimum kiválasztás k osztó szerint..
int ind, min;
ind = 0; min = -1;
bool l = false;
for(int i = 0; i < n; ++i){
    if((v[ i ] % k) == 0 && l)
    {
        if(v[ i ] < min)
        {
            min = v[ i ];
            ind = i;
        }
    }
    else if((v[ i ] % k) == 0 && !l)
    {
        min = v[ i ];
        ind = i;
        l = true;
    }
}

//Kiíratás
if(l != false)
{
    cout << "A tomb egyik legkisebb, k-val osztható eleme: " << min
<< endl;
    cout << "Ez a " << (ind + 1) << ". elem." << endl;
}
else
{
    cout << "A tomben nem található k osztója!!!" << endl;
}
do
{
    cout << "Újbol, \nakarja? ( I/N ) := ";
    cin >> con;
    char * copySTR = _strdup( con.c_str() );
    con = _strupr( copySTR );
}while(!(con == "N") && !(con == "I"));
}while( !(con == "N"));

return 0;
}
//-----
int checkNum(string tmp, char kezd)
{
    for(int h = 0; h < (signed)tmp.length(); h++)
    {
        if(tmp[ h ] >= kezd && tmp[ h ] <= '9')
        {
            //hmm
        }
        else
        {
            cout << "hiba! tartalmaz más értéket is!" << endl;
            cout << "Adja meg újra a kért értéket := ";
            return -1;
        }
    }
    int n = atoi(tmp.c_str());
    return n;
}
//int checkNum(string tmp)

```

```

//-----
int* getTMB(int &n, bool file, bool klav, string filename)
{
    //Bekérjük billentyűzetről, ha nem file-ból akarja feltölteni a tmb-t
    //ciklusbol kell bekérni, hogy mit is akarok..
    if(klav == true)
    {
        cout << "A Tomb hossza := ";
        n = getNum(false); //nem lehet majd 0as a tomb!!

        int* v = new int[ n ];
        bool ell = false;
        for(int i = 0; i < n; i++)
        {
            if(ell == true)
            {
                i--;
            }
            cout << "Adja meg a(z) " << i << " helyiertekeket := ";
            v[ i ] = getNum(true);
            ell = checkTMB(v, i, v[ i ]);
        }
        return v;
    }
    //Bekérjük az állomány nevét a szabványos bemenetről.
    if(file == true)
    {
        if(filename == "")
        {
            cout << "A fajl neve := ";
            cin >> filename;
        }
        //Definiáljuk és megnyitjuk a fájlt
        ifstream x(filename.c_str());
        //Ha hiba van befejezzük a programot
        if (x.fail()){
            cout << "A megadott fájlt nem talalom!";
            exit(1);
        }
        //Beolvassuk/kiírjuk a tömb hosszát
        x >> n;
        //Létrehozunk egy n elemű tömböt és kitöltjük
        int* v = new int [ n ];
        for(int i = 0; i < n; ++i)
        {
            x >> v[ i ];
            bool ell = checkTMB(v, i, v[ i ]);
            if(ell == true)
            {
                cout << "Utkozik a(z) " << i << " helyen levo adattal! \nKerem ellenorizze
a parameter fajt!";
            }
        }
        return v;
    }
}
//int getTMB(ifstream &x)
//-----
int getNum(bool end)
{
    string str;
    int n = 0, ell = 1;
    do
    {
        cin >> str;
        if(end == true)
        {
            ell = 0;
            n = checkNum( str, '0' );
        }
        else if(end == false)
        {
            ell = 1;
            n = checkNum( str, '1' );
        }
    }while( !(ell <= n));
    return n;
}
//int getNum()
//-----
bool checkTMB(int v[], int n, int act)
{
    bool ell = false;
    for(int i = 0; i < n; i++)
    {
        if(v[ i ] == act)
        {
            ell = true;
            cout << "Utkozik a(z) " << i << " helyen levo adattal! Kerem adja meg ujra.";
        }
    }
    return ell;
}
//bool checkTMB(int v[], int n, int act)

```