

EAF I Feladat dokumentáció a 3. házi feladathoz

Feladat

Adott egy x valós, és egy n természetes szám. Számoljuk ki az x^n hatványt úgy, hogy a hatványozás műveletét nem engedjük használni!

Specifikáció

$$A = \mathbb{Z} \times \mathbb{N} \times \mathbb{Z} \times \mathbb{Z}$$

$$x \quad n \quad i \quad d$$

$$B = \mathbb{Z} \times \mathbb{Z}$$

$$x' \quad n'$$

$$Q = (n = n' \wedge x = x' \wedge n \neq 0)$$

$$R = (Q \wedge r = x^n)$$

Absztrakt program

i, d = 1, x	
i ≠ n	
	d := x * d
	i := i + 1

Tesztelési terv

Tesztesetek a feladat alapján (fekete doboz tesztelés)

1. x -re (-) érték bevitele, . (tizedes pont) bevitele, csak egyszer legyen lehetséges!
2. n -re csak természetes szám bevitele legyen lehetséges.

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.
3. Hatványozás ellenőrzése.
4. Eredmény ellenőrzése.

Megoldás C++

```
// hazi_3.cpp : Defines the entry point for the console application.
//
#include <iostream>
#include <string>
#include <conio.h>
using namespace std;

void getNum(int &n); //integer szám bekérése, csak 1-9 es közé eshet,
//természetes szám.
void getNum(long double &n); //valós szám bekérése.
//-----
void main()
{
    int n = 0;
    long double x = 0;
    getNum( x ); //valós szám bekérése
    getNum( n ); //természetes szám bekérése
    long double d = 0;
    d = x;
    for(int i = 1; i < n; i++)
    {
        //muvelet elvégzése
        d = x * (double)d;
    }

    cout << "\nErteke = ";
    cout << d;

    getch();
} //void main()
//-----
void getNum(int &n)
{
    bool ell = false;
    string str;
    do
    {
        cout << "Termeszetes tipus hossza := ";
        cin >> str;
        for(int h = 0; h < (signed)str.length(); h++)
        {
            if(str[ h ] >= '1' && str[ h ] <= '9')
            {
                //hmm
                ell = true;
            }
            else
            {
                cout << "hiba! tartalmaz mas karaktert is!" << endl;
                ell = false;
                break;
            }
        }
    }
    while( ell == false);
    n = atoi( str.c_str() );
} //void getNum(string str, int n)
//-----
```

```

//-----
void getNum(long double &n)
{
    bool ell = false;
    string str;
    do
    {
        cout << "Valos tipus hossza := ";
        cin >> str;
        int dot = 0;
        for(int h = 0; h < (signed)str.length(); h++)
        {
            if( (str[ h ] >= '0' && str[ h ] <= '9') ||
                ( (str[ h ] == '.' && dot <= 1) ||
                  (str[ h ] == '-' && h == 0)))
            {
                ell = true;
                if(str[ h ] == '.')
                {
                    dot++;
                }
            }
            else
            {
                cout << "hiba! tartalmaz mas karaktert is!" << endl;
                ell = false;
                break;
            }
        }
    }while( ell == false);
    n = atof( str.c_str() ); //lehetne szebb megoldas is, de közelítőnek
    elég jó
} //void getNum(string str, int n)

```