

KOMPLEX SZÁMOK KÉPZÉSE

Feladat:

Valósítsa meg a komplex számok típusát! Ábrázolja a komplex számokat az algebrai alakjukkal ($x+iy$)! Implementálja az összeadás és a szorzás műveleteit operátor felüldefiniálással! A főprogram ilyen komplex számokkal töltsön fel egy tömböt, amelynek ezután ki kell számolni a tömb összegét és szorzatát!

Specifikáció:

Bemenő adatok: $t = \text{Vect}(C)$

$$C := \{x \mid x = a + b * i : a, b \in R \wedge i := \sqrt{-1}\}$$

Eredmény adatok: $x, y \in C$

Előfeltétel: $t = t'$

Utófeltétel: $x = \sum_{j=t.lob}^{t.hib} t_j \wedge y = \prod_{j=t.lob}^{t.hib} t_j$

$R(x, +, *)$:

$$\forall x, y \in C : x = a + b * i \wedge y = c + d * i;$$

$$x + y = (a + c) + (b + d) * i = y + x;$$

$$x * y = (a * c - b * d) + (a * d + c * b) * i = y * x.$$

Vagyis definiáltunk egy komplex számokból álló gyűrűt, amin értelmeztük a valós számok halmazán megszokott összeadás és szorzás műveletet, ezzel kiküszöbölve az ehhez szükséges függvények bevezetését.

Megoldás:

$x, y, j := t.lov, t.lov, t.lob+1$
$j \leq t.hib$
$x := x + t_j$
$y := y * t_j$
$j := j + 1$

A feladat tehát megoldható egy összegzés és egy összegzésre visszavezetett szorzatképzés szekvenciájaként;

C++ Forráskód (Komplex.h):

```
#ifndef KOMPLEX_H
#define KOMPLEX_H
#include <iostream>
#include <string>

using namespace std;

class Komplex{
private:
    double a,b;
public:
    Komplex(){ a = 0; b = 0; }
    Komplex(double x, double y)
    { a = x; b = y; }
    void SetA(double x){ a = x; }
    void SetB(double x){ b = x; }
    friend double GetA(const Komplex x);
```

```

        friend double GetB(const Komplex x);

        friend Komplex operator+(const Komplex& x,const Komplex& y);
        friend Komplex operator*(const Komplex& x,const Komplex& y);
};

double GetA(const Komplex x)
{
    return x.a;
}

double GetB(const Komplex x)
{
    return x.b;
}

Komplex operator+(const Komplex& x, const Komplex& y)
{
    Komplex Eredm;
    Eredm.a = x.a + y.a;
    Eredm.b = x.b + y.b;
    return Eredm;
}

Komplex operator*(const Komplex& x, const Komplex& y)
{
    Komplex Eredm;
    Eredm.a = ( x.a * y.a ) - ( x.b * y.b );
    Eredm.b = ( x.a * y.b ) + ( y.a * x.b );
    return Eredm;
}

void GetNumber(const Komplex x)
{
    if( GetB(x)<0 )
    { cout << GetA(x) << GetB(x) << "i"; }
    else
    { cout << GetA(x) << "+" << GetB(x) << "i"; }
}

#endif

```

C++ Forráskód (main.cpp):

```

#include <iostream>
#include <string>
#include <fstream>
#include "Komplex.h"

using namespace std;

int main()
{
    //ALAPVALTOZOK
    string ANS,fname,ragain;
    Komplex* t;
    Komplex x,y;
    int n,a,b,j;
    //INTERFACE
    do{
        cout << "KOMPLEX SZAMOK KEPZESE" << endl;
        cout << "Feldolgozas:" << endl;
        do{
            cout << "Fajlbol [F]" << endl;
            cout << "Billentyurol [B]" << endl;
            cout << "Valasztas: "; cin >> ANS;
        }while(( ANS != "F" )&&( ANS != "f")&&( ANS != "B")&&( ANS != "b"));
        if(( ANS == "B" )||( ANS == "b" ))
        {
            cout << "Tomb elemszama: "; cin >> n;
            if( n<1 )
            {
                cout << "HIBA: Legalabb 1 elem kell!" << endl;
                exit(1);
            }
            t = new Komplex[n];
            for(int i=0; i<n; ++i)
            {

```

```

        cout << i+1 << ". szam valos resze: "; cin >> a;
        t[i].SetA(a);
        cout << i+1 << ". szam kepzetes resze: "; cin >> b;
        t[i].SetB(b);
    }
else
{
    cout << "Fajlnev: "; cin >> fname;
    ifstream inp(fname.c_str());
    if(inp.fail())
    {
        cout << "A megadott fajlt nem talalom!" << endl;
        exit(1);
    }
    inp >> n;
    t = new Komplex[n];
    for(int i=0; i<n; ++i)
    {
        inp >> a;
        t[i].SetA(a);
        inp >> b;
        t[i].SetB(b);
        GetNumber(t[i]); cout << endl;
    }
}
//OSSZEGZES-PRODUKTUM
x = t[0];
y = t[0];
j = 1;
while( j<n )
{
    x = x + t[j];
    y = y * t[j];
    j = j + 1;
}
//EREDMENYEK
cout << "EREDMENYEK:" << endl;
cout << "Tomb-osszeg: "; GetNumber(x); cout << endl;
cout << "Tomb-szorzat: "; GetNumber(y); cout << endl;
delete[] t;
cout << "Ujraindítás (I/N): "; cin >> ragain;
}while(( ragain != "N" )&&( ragain != "n" ));
return 0;
}

```

Tesztelési terv:

Tesztesek a feladat alapján (fekete doboz tesztelés):

- *Fájlból olvasás kipróbálása (létező és nem létező fájlal is);*
- *Újrafuttatás kipróbálása;*
- *Billentyűről olvasás kipróbálása;*
- *Tömb elemszám megadása ($n < 0$, $n = 0$ és $n > 0$ értékekre);*
- *Tömb feltöltése;*
- *Újrafuttatás ismételt kipróbálása;*

Tesztesek a kód alapján (fehér doboz tesztelés):

- *Minden, ami a fekete doboz tesztelésnél is kellett;*
- *Plusz: Tömb feltöltése szélsőséges értékekkel ($a + b * i \mid a \leq 0 \vee b \leq 0$);*