Laboratorul 6

Supraîncărcarea operatorilor este o metodă prin care au diferite implementări care depind de operanzii pe care îi folosesc.

1. **Supraîncărcarea operatorului binar + ca funcţie membră a clasei `Complex`.** Testaţi programul de mai jos. Clasa `Complex` dispune de două date membre de tip real, un contructor cu parametri care au valori implicite, funcţii getter si setter pentru datele membre şi un operator de adunare.

complex.h

```
#ifndef COMPLEX_H
#define COMPLEX_H
class Complex
{
  public:
    Complex(double=0, double=0);
    void setReal(double);
    double getReal();
    void setImaginar(double);
    double getImaginar();
    Complex operator+(Complex);
  private:
    double real;
    double imaginar;
};
#endif
```

complex.cpp

```
#include "complex.h"
Complex:: Complex(double r, double i)
{
    real = r;
    imaginar = i;
}
void Complex::setReal(double r)
{
    real = r;
}
double Complex::getReal()
{
    return real;
}
void Complex::setImaginar(double i)
{
    imaginar = i;
}
double Complex::getImaginar()
{
    return imaginar;
}
Complex Complex::operator+(Complex c)
```

```
{
    Complex nc;
    nc.setReal(real+c.real);
    nc.setImaginar(imaginar+c.imaginar);
    return nc;
}
```

test_complex.cpp

```
#include <iostream>
using std::cout;
using std::endl;

#include "complex.h"

int main()
{
  Complex n1(2,4);
  Complex n2(1,-4);
  n2 = n1+n2;
  cout << n2.getReal() << ";" << n2.getImaginar() << endl;
  return 0;
}
```

Refolosiţi programul de mai sus şi urmaţi exemplele din curs pentru exerciţiile următoare. Apelaţi in funcţia `main` fiecare dintre cei trei operatori de mai jos pentru a ilustra funcţionarea lor corectă.

2. Implementaţi operatorul binar `*` de înmulţire a două numere complexe ca funcţie nemembră a clasei `Complex`.

3. Implementaţi operatorul unar `!` ca funcţie membră în clasa `Complex`. Acest operator schimbă semnul părţii imaginare.

4. Implementaţi operatorul `<<` pentru clasa `Complex`.