Supraîncărcarea operatorilor

NullPointerNinja și-a propus să combine pasiunea pentru matematică cu cea pentru programare orientată pe obiecte. De data asta, vrea să învețe cum poate face ca polinoamele să se comporte în C++ la fel ca variabilele primitive precum int, float sau double. Astfel, va putea aduna, scădea sau înmulți polinoame folosind operatorii obișnuiți, exact ca în mulțimea numerelor reale.

Un obiect de tip Polinom în C++ va conține numărul de coeficienți și valorile efective ale coeficienților acestuia. Gradul unui polinom va fi numărul de coeficienți minus unu. Așadar vom considera următoarea clasă în limbajul C++:

```
class Polinom
{
    int* coeficienti;
    int nrCoeficienti;

    bool esteInvalid() const;
    int gradMinim(const Polinom& p1, const Polinom& p2) const;

public:

    Polinom(const int& nrCoeficienti = 0, const int* const& coeficienti =
nullptr);
    Polinom(const Polinom& polinom);
    Polinom& operator=(const Polinom& polinom);
    ~Polinom();

    const int getGrad() const;

    // operatorul de indexare (returneaza un coeficient din polinom pe baza
index-ului primit)

    int& operator[](const int& index);
```

```
    const int& operator[](const int& index) const;

    // operatorul sageata

    Polinom* operator->();

    // operatorul - care returneaza opusul unui polinom (-p)

    Polinom operator-() const;

    // operatorii de incrementare / decrementare (formele prefixate si
postfixate)
    // incrementeaza / decrementeaza doar termenul liber

    Polinom& operator++(); // prefixat (++p)
    Polinom& operator--(); // prefixat (--p)

    Polinom operator++(int); // postfixat (p++)
    Polinom operator--(int); // postfixat (p--)

    // operatorii aritmetici supraincarcati ca functii membre

    Polinom operator+(const Polinom& polinom) const;
    Polinom operator-(const Polinom& polinom) const;
    Polinom operator*(const Polinom& polinom) const;

    // operatorii logici de testare a egalitatii

    bool operator ==(const Polinom& polinom) const;
    bool operator !=(const Polinom& polinom) const;

    // operatorii aritmetici supraincarcati ca functii friend

    friend Polinom operator+(const Polinom& polinom, const int& scalar);
    friend Polinom operator+(const int& scalar, const Polinom& polinom);

    friend Polinom operator-(const Polinom& polinom, const int& scalar);
    friend Polinom operator-(const int& scalar, const Polinom& polinom);

    friend Polinom operator*(const Polinom& polinom, const int& scalar);
    friend Polinom operator*(const int& scalar, const Polinom& polinom);

    // operatorul de afisare supraincarcat ca functie friend

    friend std::ostream& operator<<(std::ostream& out, const Polinom& polinom);
};
```

Cerințe

1) Să se implementeze toți operatorii clasei Polinom astfel încât aceștia să funcționeze corect. Fiecare operator valoreaza 0.4p. (8p)

2) În funcția main să se creeze 3 polinoame. Al treilea polinom e pentru a stoca rezultatul operațiilor matematice efectuate cu primele 2 polinoame. (1p)

Se acordă un punct din oficiu. Succes.

Precizări

- Clasa se va implementa în fișiere .h și .cpp. Iar testarea codului se va face în funcția main care va fi într-un fișier separat.

- Neimplementarea în fișiere .h și .cpp atrage după sine pierderea unui punct.

- Erorile de compilare și memory leak-urile atrag fiecare după sine pierderea unui punct.

- Alte depunctări cu privire la modul de implementare sau la corectitudinea declarării metodelor, rămân la latitudinea asistentului care vă corectează.

- Dacă nu s-a specificat vreun mod de implementare a cerințelor rămâne la latitudinea voastră cum vreți să procedați în rezolvare.