

Laborator 3

```
class Fractie
{
private:
    int a; // numarator
    int b; // numitor
public:
    Fractie(int aa = 0, int bb = 0); // constructor cu parametrii impliciti
    Fractie(const Fractie &);        // constructor de copiere
    Fractie &operator=(const Fractie &);
        // se intoarce referinta la obiectul modificat pt a putea face op de genul : int
        // a,b,c,d;    a = (b = (c = (d = 4)));        asociativitate la dreapta
    ~Fractie();
        // constr de copiere, op= si destr se genereaza automat si functioneaza corect
        // implementarea lor va fi facuta doar in scop didactic
    double getValoare();                // cat face a/b
    Fractie getInv();                    // b/a
    void setdata(int, int);              // modifica valorile numaratorului si numitorului
    float getA();                        // returneaza numaratorul
    float getB();                        // returneaza numitorul
    friend Fractie operator+(const Fractie &, const Fractie &); // supradefinire operator
                                                //adunare

    friend Fractie operator-(const Fractie &, const Fractie &);
    friend Fractie operator*(const Fractie &, const Fractie &);
    friend Fractie operator/(const Fractie &, const Fractie &);
    friend Fractie operator-(const Fractie &);                // transforma numerele in inversul
lor.
                                                //ex: 8 -> -8

    Fractie &operator+=(const Fractie &a);
        // supradefinire operator incrementare cu o valoare
        // se intoarce referinta la Fractie pt a putea face operatii ca : int m ; (m+=5)+=3 ;
        // *this = *this + a;
    Fractie &operator-=(const Fractie &);
    Fractie &operator*=(const Fractie &);
    Fractie &operator/=(const Fractie &);
    bool operator==(const Fractie &);                // supradefinire operator de egalitate
    bool operator!=(const Fractie &x);                // supradefinire operator diferit
        // pot sa folosesc in implementare operatorul == implementat anterior
        //{return(!(*this==x));}
    bool operator<(const Fractie &);                // supradefinire operator <
    bool operator>=(const Fractie &);                // supradefinire operator <
```

```
bool operator>(const Fractie &);           // supradefinire operator >
bool operator>=(const Fractie &);        // supradefinire operator <
};
```

Cerinta:

sa implementati metodele si sa le testati in main.