

Programowanie obiektowe C++

Przeciążanie operatorów, konstruktory

Obiekty, konstruktory, operatory

1. Przykład

Dana jest klasa do opisu ułamków zwykłych i podstawowych operacji arytmetycznych na ułamkach:

```
class TUlamek
{
public:
    TUlamek(int licznik = 1, int mianownik = 1);
    TUlamek(TUlamek &ulamek);

    TUlamek operator +();
    TUlamek operator -();
    TUlamek & operator =(const TUlamek &ulamek);
    TUlamek operator +(TUlamek ulamek);
    TUlamek operator -(TUlamek ulamek);
    TUlamek operator *(TUlamek ulamek);
    TUlamek operator /(TUlamek ulamek);

    void uproszcz();
    string napisz();
    double wartosc();

    void ustawLicznik(int licznik);
    void ustawMianownik(int mianownik);

    int wartLicznik() const;
    int wartMianownik() const;

private:
    int licznik;
    int mianownik;

    void poprawZero();
    void poprawZnak();
};
```

w skład której wchodzi:

- prywatne pola: licznik i mianownik – do zapamiętywania wartości licznika i mianownika ułamka,
- konstruktor z parametrami domyślnymi **TUlamek()** pozwalający na zdefiniowanie domyślnej wartości ułamka w postaci $\frac{1}{1}$, ponadto konstruktor pozwala na zdefiniowanie liczby całkowitej w postaci $\frac{\text{licznik}}{1}$, i dodatkowo umożliwia zdefiniowanie ułamka zwykłego w postaci $\frac{\text{licznik}}{\text{mianownik}}$
- dwie metody **ustawLicznik()** i **ustawMianownik()** do zmiany wartości licznika i mianownika ułamka,

- metoda **napisz()** zwracająca wartość ułamka w łańcuchu znaków do wyświetlania,
- metoda **wartosc()** zwracająca numeryczną wartość ułamka,
- metoda **uprosc()** służąca do skracania ułamka (wykorzystuje algorytm szukania największego wspólnego dzielnika dwóch liczb),
- metoda **poprawZero()** służąca do zmiany zerowych ułamków na postać $\frac{0}{1}$,
- metoda **poprawZnak()** przenosząca znak minus z mianownika do licznika,
- operatory jednoargumentowe **+** i **-** do zmiany znaku ułamka,
- dwuargumentowe operatory arytmetyczne: **+**, **-**, ***** i **/**,
- operator wyjścia **<<** pozwalający na wyświetlanie ułamka.

Wygeneruj nowy projekt C++ i skopiuj do pliku źródłowego projektu zawartość pliku tekstowego **ułamki.cpp** ze strony przedmiotu. Uruchom program, prześledź jego działanie.

2. Zadanie

Wykonaj kilka dowolnych operacji arytmetycznych na nowo zdefiniowanych obiektach typu **TUłamek**. Wykonaj dowolne operacje arytmetyczne na tablicach obiektów **TUłamek**.

3. Zadanie

Dopisz dodatkową funkcję operatorową dla operatora wejścia **>>**, która pozwoli na zmianę wartości ułamka wykorzystując wartości ze standardowego wejścia.

Zadania dodatkowe

1. Opracuj typ obiektowy i program do przechowywania i podstawowych operacji arytmetycznych dla liczb zespolonych w postaci: $\Re + \Im \cdot i$.
2. Opracuj typ obiektowy i program do przechowywania i podstawowych operacji na danych reprezentujących czas. Ogranicz się do godzin i minut. Przykładowa deklaracja typu mogłaby wyglądać:

```
class TTime
{
public:
    // stosowne konstruktory

    void addHour(int h);
    void addMinute(int m);
    void timeReset(...); // uzupełnic
    void timeShow(...) const; // uzupełnic

    TTime operator +(const TTime &t) const;
    TTime operator -(const TTime &t) const;

private:
    int hours;
    int minutes;
}
```

3. Opracuj typ obiektowy i program do przechowywania i podstawowych operacji arytmetycznych dla długich liczb całkowitych (więcej niż 20 cyfr w liczbie).