laboratorium: zadanie 8 termin: 8–12 maja 2023 r.

KURS JĘZYKA C++

LICZBY WYMIERNE

Instytut Informatyki Uniwersytetu Wrocławskiego *Paweł Rzechonek*

Prolog.

Liczba wymierna to taka liczba, którą można zapisać w postaci ułamka zwykłego, czyli w postaci $\frac{p}{q}$, gdzie p to dowolna liczba całkowita a q to liczba całkowita różna od 0. Zbiór wszystkich liczb wymiernych oznaczamy symbolem $\mathbb{Q}$ i formalnie można go zdefiniować jako:

$$\mathbb{Q} = \left\{ \frac{p}{q} \ : \ p, q \in \mathbb{Z} \wedge q \neq 0 \right\}$$

Liczby wymierne z operacją dodawania (element neutralny dodawaniato zero) i mnożenia (element neutralny mnożenia to jedynka) stanowią ciało. Szczególnym przypadkiem liczb wymiernych są liczby całkowite.

Zadanie.

Zdefiniuj klasę `wymierna`, reprezentującą liczbę wymierną w postaci pary liczb całkowitych: licznika i mianownika.

```
class wymierna {
    int licz, mian;
    // ...
};
```

Zadbaj o to, aby mianownik zawsze był > 0 oraz aby największy wspólny dzielnik licznika i mianownika zawsze był $= 1$. Udostępnij też gettery, czyli funkcje składowe umożliwiające odczyt licznika i mianownika. Definicję liczby wymiernej umieść w przestrzeni nazw `obliczenia`.

Klasa `wymierna` powinna być wyposażona w konstruktor z licznikiem i mianownikiem, konstruktor konwertujący z wartości typu `int` (możesz zaadoptować do tego celu poprzedni konstruktor definiując drugi argument jako domyślny) oraz konstruktor domyślny, ustawiający wartość liczby wymiernej na zero (czyli ułamek $\frac{0}{1}$).

W klasie `wymierna` zdefiniuj operatory binarne umożliwiające wykonywanie podstawowych obliczeń arytmetycznych (dodawanie `+`, odejmowanie `-`, mnożenie `*` i dzielenie `/`) oraz operatory unarne `-` do zmiany znaku na przeciwny i `!` do wyznaczenia odwrotności (zamiana licznika z mianownikiem oraz pozostawienie znaku liczby w liczniku). Zdefiniuj także operator rzutowania na typ `double` oraz operator jawnego rzutowania na typ `int` (zaokrąglenie do najbliższej liczby całkowitej).

Nie zapomnij przy każdej funkcji składowej, przy konstruktorach i przy operatorach zadeklarować czy zgłaszają one jakieś wyjątki czy nie: w przypadku binarnych operacji arytmetycznych należy zgłosić wyjątek `przekroczenie_zakresu`; gdy wynik nie będzie mógł być wyrażony w postaci ilorazu dwóch liczb typu `int`; w przypadku dzielenia przez 0 podczas operacji dzielenia lub podczas liczenia odwrotności należy zgłosić wyjątek `dzielenie_przez_0`. Zaprojektuj zatem hierarchię klas wyjątków na potrzeby liczb wymiernych zaczynając od klasy bazowej `wyjatek_wymierny` dziedziczącej po `std::logic_error`.

Zaprogramuj także operator strumieniowy do zapisania liczby wymiernej do strumienia wyjściowego `operator<<` w postaci ułamka dziesiętnego okresowego.

```
class wymierna {
    // ...
    friend ostream& operator<< (ostream &wyj, const wymierna &w);
};
```

Na przykład ułamek $\frac{2359348}{99900}$ należy wypisać jako `23.61(709)`.

Na koniec napisz program, który rzetelnie przetestuje wszystkie metody z klasy `wymierna` (wraz ze zgłaszanymi przez nie wyjątkami). Przetestuj także kopiowanie liczb wymiernych (domyślny konstruktor kopiujący i przypisanie kopiujące).

Istotne elementy programu.

- Użycie przestrzeni nazw `obliczenia`.
- Operatory rzutowania i konstruktor konwertujący.
- Definicja operatorów arytmetycznych binarn.ych i unarnych.
- Zapis do strumienia liczby wymiernej w postaci ułamka okresowego.
- Definicja własnej hierarchii klas wyjątków dziedziczących po `std::logic_error`.
- Podział programu na pliki nagłówkowe i pliki źródłowe (wyodrębniony osobny plik z funkcją `main()` z testami).