PJC 11

Zadanie 1
Zdefinuj klasę Resistor, która reprezentuje opornik elektryczny. Klasa powinna zawierać:

-prywatne pole double resistance - opór opronika
-konstruktor domyślny (tworzący opornik o oporze 0)
-konstruktor pobierający jedną wartość double
-metodę double r() const zwracającą wartość oporu
-metodę void r(double) modyfikującą wartość oporu

Ponadto napisz funkcje przeciążające operatory +, * i <<

-operator+ - zwracającą przez wartość obiekt klasy Resistor reprezentujący opór zastępczy dwóch podanych (w postaci obiektów) oporników, zakładając, że są one połączone szeregowo;
-operator* - podobnie, ale dla połączenia równoległego;
-operator<< - pozwalającą na wstawianie obiektów klasy do strumienia wyjściowego.

Następujący fragment

```
Resistor r1, r2{6};
r1.r(3);
std::cout << (r1 + r2) << " " << (r1 * r2) << std::endl;
```

powinien wydrukować

```
9 2
```

Zapewnij aby tworzenie rezystora o ujemnym oporze jest niemożliwe poprzez wykorzystanie wyjątków.

Zadanie 2
Zdefniować klasę Frac opisującą ułamki (liczby wymierne). W szczególności napisać:

-Konstruktor pobierający licznik n i mianownik d i tworzący obiekt reprezentujący ułamek n/d
-Oba argumenty konstruktora powinny mieć wartości domyślne, tak aby
-obiekt Frac(n) odpowiadał liczbie całkowitej n (ułamek o mianowniku 1);
-obiekt Frac() reprezentował zero.

Reprezentacja ułamków powinna być jednoznaczna, czyli niezależnie od wartości argumentów konstruktora powinno być tak, żeby (prywatne) pola num i denom (licznik i mianownik) nie miały wspólnych dzielników (czyli ułamki były zawsze w postaci „skróconej”), denom był zawsze dodatni, a jeśli num jest 0, to denom powinien być 1, aby zapewnić jednoznaczność reprezentacji zera. Zapewnij też poprzez wykorzystanie wyjątków aby tworzenie ułamków dążących do nieskończoności jest niemożliwe. Przyda się pewnie prywatna funkcja znajdująca największy wspólny dzielnik. Zauważmy, że klasa ma tylko pola liczbowe, a więc, jak to zwykle w takich przypadkach bywa, nie musimy nadpisywać dostarczonego przez kompilator konstruktora kopiującego ani przeciążać operatora przypisania.

-Przeciążenia operatorów dodawania, odejmowania, mnożenia i dzielenia.
-Przeciążenie operator<< dla obiektów typu Frac.

Tak więc następująca funkcja main

```
int main() {
Frac a(2), b(4,10), c(24,-15), x(1,-3), y(2,6);
std::cout << -2*((a+b)*5-4)/c << " "
<< (7 + x + y*1114/111) << std::endl;
}
```

powinna wypisać

```
10 3334/333
```