ДЕРЖАВНИЙ УНІВЕРСИТЕТ ІНТЕЛЕКТУАЛЬНИХ ТЕХНОЛОГІЙ І ЗВ'ЯЗКУ

Звіт

з дисципліни Проектний Практикум

Лабораторна робота №4

на тему: «Об'єктно орієнтоване програмування»»

Виконав: студент групи ІПЗ-3.04

Бухта М.М

_____

Перевірив: Багачук Д.Г.

_____

Одеса 2023

Завдання №1

Опис завдання:

a) Напишіть клас Fraction, який має два цілочисельних члени: чисельнік і займенник. Реалізуйте функцію print(), яка виводитиме дріб.

b) Додайте перевантаження оператора множення (*) для виконання операції множення об’єкту класу Fraction на цілочисельне значення і для множення двох об’єктів класу Fraction. Використовуйте спосіб перевантаження оператора через дружню функцію.

Код програми:

main.cpp

```
/************************************
 * Laboratory work #4;              *
 * Student Bukhta Mykyta;           *
 * Grade: 3;                        *
 * Group Software Engineering 3.04; *
 * Task 1;                          *
 ************************************
 */

#include "Fraction.hpp"

#include <utility>
#include <iostream>

using namespace lab_4;

void do_fragment_a(void) {
    std::cout << "Start task 1(a):" << std::endl;
    Fraction f1{1, 4};
    f1.print();

    Fraction f2{1, 2};
    f2.print();
}

void do_fragment_b(void) {
    std::cout << "Start task 1(b):" << std::endl;
    Fraction f1{3, 4};
    f1.print();

    Fraction f2{2, 7};
    f2.print();
```

```
    Fraction f3{f1 * f2};
    f3.print();

    Fraction f4{f1 * 3};
    f4.print();

    Fraction f5{3 * f2};
    f5.print();

    Fraction f6{Fraction{1, 2} * Fraction{2, 3} * Fraction{3, 4}};
    f6.print();
}

int main(int argc, char **argv) {
    do_fragment_a();
    do_fragment_b();

    return 0;
}
```

Fraction.hpp

```
/***********************************
 * Laboratory work #4;             *
 * Student Bukhta Mykyta;          *
 * Grade: 3;                       *
 * Group Software Engineering 3.04; *
 * Task 1;                         *
 ***********************************
 */

#ifndef BUKHTAMYKYTA_LAB_4_TASK_1_FRACTION_HPP
#define BUKHTAMYKYTA_LAB_4_TASK_1_FRACTION_HPP

#include <inttypes.h>

namespace lab_4 {

class Fraction {
public:
    explicit Fraction(int32_t numerator = 0, int32_t pronount = 0);
    Fraction operator* (const Fraction &other) const noexcept;
    Fraction operator* (int32_t numerator) const noexcept;
    friend Fraction operator* (int32_t numerator, const Fraction &fraction)
noexcept;

    void print(void) const;
private:
```

```
    int32_t m_numerator;
    int32_t m_pronount;
};

}   // !lab_4;

#endif  // !BUKHTAMYKYTA_LAB_4_TASK_1_FRACTION_HPP;
```

Fraction.cpp

```
#include "Fraction.hpp"

#include <iostream>

namespace lab_4 {

Fraction::Fraction(int32_t numerator, int32_t pronount)
    : m_numerator{numerator}, m_pronount(pronount)
{

}

Fraction Fraction::operator* (const Fraction &other) const noexcept{
    return Fraction{
        this->m_numerator * other.m_numerator,
        this->m_pronount * other.m_pronount
    };
}

Fraction Fraction::operator* (int32_t numerator) const noexcept {
    return Fraction{
        this->m_numerator * numerator,
        this->m_pronount
    };
}

Fraction operator* (int32_t numerator, const Fraction &fraction) noexcept{
    return fraction * numerator;
}

void Fraction::print(void) const {
    std::cout << m_numerator << '/' << m_pronount << std::endl;
}

}   // lab_4;
```

Результат виконання програми:

```
PS C:\Users\Nikita\Documents\Univercity\University\Grade_3\ProjectPracticum\Lab_4\build> .\task_1\lab_4_task_1.exe
Start task 1(a):
1/4
1/2
Start task 1(b):
3/4
2/7
6/28
9/4
6/7
6/24
```

Завдання №2

Опис завдання:

Використовуючи клас Fraction , представлений у лабораторній роботі, додайте перевантаження операторів << і >> .

Код програми:

main.cpp

```cpp
/************************************
 * Laboratory work #4;              *
 * Student Bukhta Mykyta;           *
 * Grade: 3;                        *
 * Group Software Engineering 3.04; *
 * Task 2;                          *
 ************************************
 */

#include "Fraction.hpp"

#include <iostream>

using namespace lab_4;

int main(int argc, char **argv) {
    Fraction f1;
    std::cout << "Input fraction 1: ";
    std::cin >> f1;

    Fraction f2;
    std::cout << "Input fraction 2: ";
    std::cin >> f2;

    std::cout << f1 << " * " << f2 << " is " << (f1 * f2) << std::endl;
```

```
    return 0;
}
```

Fraction.hpp

```
/***********************************
 * Laboratory work #4;             *
 * Student Bukhta Mykyta;          *
 * Grade: 3;                       *
 * Group Software Engineering 3.04; *
 * Task 2;                         *
 ***********************************
 */

#ifndef BUKHTAMYKYTA_LAB_4_TASK_1_FRACTION_HPP
#define BUKHTAMYKYTA_LAB_4_TASK_1_FRACTION_HPP

#include <istream>
#include <ostream>

namespace lab_4 {

class Fraction {
private:
    int m_numerator;
    int m_denominator;

public:
    Fraction(int numerator = 0, int decominator = 1);

    friend Fraction operator* (const Fraction &f1, const Fraction &f2);
    friend Fraction operator* (const Fraction &f1, int value);
    friend Fraction operator* (int value, const Fraction &f1);
    friend std::istream& operator>> (std::istream & in, Fraction &f1);
    friend std::ostream& operator<< (std::ostream & out, const Fraction &f1);

    static int nod(int a, int b);

    void reduce();
    void print();
};

}   // !lab_4;

#endif  // !BUKHTAMYKYTA_LAB_4_TASK_1_FRACTION_HPP;
```

Fraction.cpp

```
#include "Fraction.hpp"
```

```cpp
#include <iostream>
#include <sstream>
#include <array>

namespace lab_4 {

Fraction::Fraction(int numerator, int decominator):
    m_numerator(numerator), m_denominator(decominator)
{
    // Ми помістили метод reduce() в конструктор, щоб переконатися, що всі дробі, які у нас є, будут зменшені.
    // Всі дробі, які будут перезаписані, повинні бути повторно зменшенні
    reduce();
}

Fraction operator* (const Fraction &f1, const Fraction &f2)
{
    return Fraction(f1.m_numerator * f2.m_numerator, f1.m_denominator * f2.m_denominator);
}

Fraction operator* (const Fraction &f1, int value)
{
    return Fraction(f1.m_numerator * value, f1.m_denominator);
}

Fraction operator* (int value, const Fraction &f1)
{
    return Fraction(f1.m_numerator * value, f1.m_denominator);
}

std::istream& operator>> (std::istream & in, Fraction &f1) {
    std::string segment;
    std::string input;
    in >> input;
    std::stringstream ss(input);
    std::array<int*, 2> metadata = {&f1.m_numerator, &f1.m_denominator};
    int counter{0};
    while (std::getline(ss, segment, '/') && counter != metadata.size()) {
        *metadata[counter++] = std::stoi(segment);
    }

    return in;
}

std::ostream& operator<< (std::ostream & out, const Fraction &f1) {
    out << f1.m_numerator << '/' << f1.m_denominator;
```

```
    return out;
}

int Fraction::nod(int a, int b)
{
    return b == 0 ? a : nod(b, a % b);
}

void Fraction::reduce()
{
    int nod = Fraction::nod(m_numerator, m_denominator);
    m_numerator /= nod;
    m_denominator /= nod;
}

void Fraction::print()
{
    std::cout << m_numerator << "/" << m_denominator << "\n";
}

}   // lab_4;
```

Результат виконання програми:

```
PS C:\Users\Nikita\Documents\Univercity\University\Grade_3\ProjectPracticum\Lab_4\build> .\task_2\lab_4_task_2.exe
Input fraction 1: 3/4
Input fraction 2: 4/9
3/4 * 4/9 is 1/3
```