Московский Авиационный Институт
(Национальный исследовательский Университет)

Факультет: «Информационные технологии и прикладная математика»
Кафедра: 806 «Вычислительная математика и программирование»

Лабораторная работа №1
по курсу «ООП»

Тема:
Простые классы.

Студент:	Вахрамян К.О.
Группа:	М80-206Б-18
Преподаватель:	Журавлев А.А.
Вариант:	3
Оценка:	
Дата:	

Москва
2019

1. Код программы на языке C++:

TRational.h:

```
#ifndef _CLASS_H_
#define _CLASS_H_
#include <iostream>
#include <cstdlib>
#include <math.h>

class TRational {
private:
	int a;
	int b;
public:
	TRational(){}
	TRational(int a, int b);
	int Read(std::istream&);
	 TRational Add(const TRational &d1) const;
	 TRational Div(const TRational &d1) const;
	 TRational Sub(const TRational &d1) const;
	 TRational Mul(const TRational &d1) const;
	int Compare(const TRational &d1) const;
	void Reduce();
	void Print(std::ostream&) const;
};

#endif
```

TRational.cpp:

```
#include "TRational2.h"

TRational operator + (const TRational& d1, const TRational& d2) {
	TRational tmp(d1.a * d2.b + d1.b * d2.a, d1.b * d2.b);
	tmp.Reduce();
	return tmp;
}

TRational operator - (const TRational& d1, const TRational& d2) {
	TRational tmp(d1.a * d2.b - d1.b * d2.a, d1.b * d2.b);
	tmp.Reduce();
	return tmp;
}
```

```
TRational operator / (const TRational& d1, const TRational& d2) {
	TRational tmp(d1.a * d2.b, d1.b * d2.a);
	tmp.Reduce();
	return tmp;
}

TRational operator * (const TRational& d1, const TRational& d2) {
	TRational tmp(d1.a * d2.a, d1.b * d2.b);
	tmp.Reduce();
	return tmp;
}

TRational& TRational::operator*= (unsigned long long num) {
	a = a * num;
	b = b * 1;
	return *this;
}

TRational operator"" _xn(unsigned long long first) {
	TRational P(first, 2);

	return P;
}


std::ostream& operator << (std::ostream& out, const TRational& Rational) {
	return out << Rational.a << "/" << Rational.b;
}
std::istream& operator >> (std::istream &in,  TRational& Rational) {
	char tmp;
	return in >> Rational.a >> tmp >> Rational.b;

}

bool operator > (const TRational& d1,  const TRational& d2) {
	if (d1.a * d2.b > d1.b * d2.a) {

		return true;
	}
	return false;
}
 bool operator == (const TRational& d1, const TRational& d2) {
	if (d1.a * d2.b == d1.b * d2.a) {
		return true;
```

```
        }
        return false;
 }
 bool operator < (const TRational& d1, const TRational& d2) {
        if (d1.a * d2.b < d1.b * d2.a) {
                return true;
        }
        return false;
 }



int TRational::Compare(const TRational &d1) const{
        if (this->b == 0) {
                return 2;
        } else if (d1.b == 0) {
                return 1;
        }
        if (*this > d1) {
                return 1;
        } else if (*this == d1) {
                return 0;
        } else {
                return 2;
        }
}


void TRational::Reduce(){
        int x = abs(this->a);
        int y = abs(this->b);
        if (x == 0 || y == 0) {
                return;
        }
        while (x != 0 && y != 0) {
                if (x > y) {
                        x = x % y;
                } else {
                        y = y % x;
                }
        }
        this->a = this->a / (x + y);
        this->b = this->b / (x + y);
}
```

```
void TRational::Print() const{
	TRational tmp = *this;
	if (tmp.b == 0) {
		std::cout << " -nan\n";
		return;
	} else if (tmp.a == 0) {
		std::cout << " 0\n";
		return;
	} else if (tmp.a < 0 and tmp.b < 0) {
		tmp.a = (-1) * tmp.a;
		tmp.b = (-1) * tmp.b;
		std::cout << tmp << "\n";
		return;
	}
	std::cout << tmp << "\n";
}
```

main.cpp:

```
#include "TRational.h"


int main()
{

	TRational d1, d2, ans;
	if (d1.Read(std::cin) || d2.Read(std::cin) == 1) {
		return 0;
	}


	std::cout << "Add = ";
	ans = d1.Add(d2);
	ans.Print(std::cout);
	std::cout << "\nSub = ";
	ans = d1.Sub(d2);
	ans.Print(std::cout);
	std::cout << "\nMul = ";
	ans = d1.Mul(d2);
	ans.Print(std::cout);
```

```
	std::cout << "\nDiv = ";
	ans = d1.Div(d2);
	ans.Print(std::cout);
	std::cout << "\n";
	if (d1.Compare(d2) == 1) {
		d1.Print(std::cout);
		std::cout << " > ";
		d2.Print(std::cout);
	} else if (d1.Compare(d2) == 0) {
		d1.Print(std::cout);
		std::cout << " = ";
		d2.Print(std::cout);
	} else {
		d1.Print(std::cout);
		std::cout << " < ";
		d2.Print(std::cout);
	}
}
```

CmakeLists.txt:

```
cmake_minimum_required(VERSION 3.1)
project(lab1)

add_executable(lab1 main.cpp Trational.cpp)
```

2. Ссылка на репозиторий на GitHub.

https://github.com/vebcreatex7/oop_exercise_01

3. Набор testcases.

test_01.txt:

```
1/4 5/8
Add = 7/8
Sub = -3/8
Mul = 5/32
Div = 2/5
1/4 < 5/8
```

test_02.txt:

5/12 -1/2
Add = -1/12
Sub = 11/12
Mul = -5/24
Div = 5/-6
5/12 > -1/2

test_03.txt:

5/2 5/2
Add = 5/1
Sub = 0
Mul = 25/4
Div = 1/1
5/2 == 5/2

test_04.txt:

3/4 -3/4
Add = 0
Sub = 3/2
Mul = -9/16
Div = 1/-1
3/4 > -3/4
test_05.txt:

0/4 0/2
Add = 0
Sub = 0
Mul = 0
Div = 0
0/4 == 0/2

test_06.txt:

0/2 2/5
Add = 2/5
Sub = -2/5
Mul = 0
Div = 0
0/2 < 2/5

test_07.txt:

2/0 4/3
Error. Division by zero

test_08.txt
-1/2 -1/2
Add = -1/1
Sub = 0
Mul = 1/4
Div = 1/1
-1/2 = -1/2

5. Объяснение результатов работы программы.

Со стандартного ввода программа считывает две рациональные дроби в виде двух пар чисел, разделенных знаком /. Далее создается два класса d1 и d2, при помощи конструктора приватным переменным класса присваиваются значения. Затем вызываются методы класса в виде: c1.metod(c2), таким образом каждому методу мы передаем два класса (c1 передается при помощи скрытого указателя this*). Сами методы, такие как Add(), Mul(), Div(), Sub(), реализованы согласно правилам арифметики. В методе сокращения дроби Reduce() используется алгоритм Евклида нахождения НОДа через вычитание. Также метод Reduce() осуществляет вывод результата арифметической операции. Метод Compare() производит операцию сравнения. После завершения работы программы на

стандартный вывод подается результат выполнения арифметических операций с рациональными дробями.

6. Вывод.

Выполняя данную лабораторную я получил опыт работы с простыми классами, с системой сборки Cmake, с системой контроля версий GitHub, а также изучил основы работы с классами в C++. Создал класс, соответствующий варианту моего задания, реализовал для него арифметические операции сложения, вычитания, умножения, деления, а также операции сравнения.