Министерство образования и науки Российской Федерации

Федеральное государственное бюджетное образовательное учреждение высшего образования «Московский государственный технический университет имени Н. Э. Баумана (национальный исследовательский университет)

Факультет «Робототехника и комплексная автоматизация»

Кафедра «Системы автоматизированного проектирования»

**Отчет по лабораторной работе**

По курсу «Объектно-ориентированное программирование»

Выполнил: Студент Сергеева Д.К.

Группа РК6-26Б

Проверил: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Дата \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Подпись\_\_\_\_\_\_\_\_\_\_\_\_\_

Москва, 2020 г.

**ПЯВУ Вариант 20R**

**Задание:**

Разработать объектно-ориентированную программу сравнения целого числа и обыкновенной дроби, которые должны передаваться ей двумя аргументами командной строки. Результат сравнения должен отображаться строкой потока стандартного вывода, где заданные целое число и дробь разделяет знак отношения порядка между их величинами. Во всех случаях для записи обыкновенных дробей должен применяться символьный формат, где их числители и знаменатели разделены знаком **'/'**. Программная реализация сравнения должна быть основана на разработке класса обыкновенных дробей с приватными полями данных для целочисленных значений их числителя и знаменателя, а также дружественными методами перегрузки операторов **'>'** и **'<'** в условных выражениях. Конструкторы этого класса должны обеспечивать преобразование обеих сравниваемых величин в указанный числовой формат обыкновенной дроби.Для приведения обыкновенной дроби к несократимому виду следует предусмотреть компонентный метод, реализующий алгоритм Евклида.

**Алгоритм:**

Создаем класс с приватными полями его числителя и знаменателя. После считывания дроби, сокращаем её с помощью алгоритма Евклида (находим наибольший общий делитель числителя и знаменателя, далее сокращаем на этот нод и числитель, и знаменатель). При помощи перегрузки операторов сравнения сравниваем целое число и введённую дробь (для этого сравниваем числитель дроби и целое число, умноженное на знаменатель дроби).

**Входные данные:**

2 числа как аргументы командной строки: целое число и обыкновенная дробь, записанная с разделителем ‘/’.

**Выходные данные:**

Результат сравнения целого числа и обыкновенной дроби.

**Текст программы:**

#include <string.h>

#include <cstdlib>

#include <iostream>

#include <stdio.h>

using namespace std;

class Fraction

{

private:

int \_numerator; //numerator of fraction

int \_denominator; //denominator of fraction

public:

Fraction(int nomerator = 0, int denominator = 1) : \_numerator(nomerator), \_denominator(denominator) {};

Fraction(char\* s);

void reduce(); //reduce fraction

operator char\*(); //type conversion overload

int getDenominator() { return \_denominator; };

friend bool operator<(int n, Fraction& f); //comparison overload '<'

friend bool operator>(int n, Fraction& f); //comparison overload '>'

};

Fraction::Fraction(char\* s)

{

char\* p = strchr(s, '/');

\_denominator = 1;

if (p != NULL)

{

\*p = '\0';

p++;

\_denominator = atoi(p);

}

\_numerator = atoi(s);

}

//Euclide algorithm (search greatest common denominator)

int algEuclide(int num1, int num2)

{

int remainder = 1;

while (num1 != 0)

{

remainder = num2 % num1;

num2 = num1;

num1 = remainder;

}

return num2;

}

//Reduce the fraction

void Fraction::reduce()

{

int gcd = algEuclide(abs(\_numerator), \_denominator);

\_numerator /= gcd;

\_denominator /= gcd;

}

//Comparison overload '<'

bool operator<(int n, Fraction& f)

{

if ((n \* f.\_denominator) < f.\_numerator)

return true;

else

return false;

}

//Comparison overload '>'

bool operator>(int n, Fraction& f)

{

if ((n \* f.\_denominator) > f.\_numerator)

return true;

else

return false;

}

//Type conversion overload

Fraction::operator char\*()

{

static char s[32];

sprintf(s, "%d/%d", \_numerator, \_denominator);

return s;

}

int main(int argc, char\* argv[])

{

if (argc != 3)

{

cout << "Wrong number of arguments." << endl;

return (-1);

}

Fraction frac(argv[2]);

if (frac.getDenominator() == 0)

{

cout << "Incorrect input of fraction." << endl;

return (-1);

}

frac.reduce();

int num = atoi(argv[1]);

if (num > frac)

cout << num << " > " << (char\*)frac << endl;

else if (num < frac)

cout << num << " < " << (char\*)frac << endl;

else

cout << num << " = " << (char\*)frac << endl;

return 0;

}

**Тесты:**

|  |  |  |
| --- | --- | --- |
| **Тест №** | **Входные данные** | **Выходные данные** |
| 1 | -1 -1/2 | -1 < -1/2 |
| 2 | 3 4/12 | 3 > 1/3 |

**Список использованной литературы:**

* Волосатова Т.М., Родионов С.В. Лекции по курсу «Объектно-ориентированное программирование»
* bigor.bmstu.ru