**Московский авиационный институт**

**(Национальный исследовательский университет)**

Факультет: «Информационные технологии и прикладная математика»

Кафедра: 806 «Вычислительная математика и программирование»

Дисциплина: «Объектно-ориентированное программирование»

**Лабораторная работа № 2**

Тема: Перегрузка операторов в С++

Студент: Кулагин Д.Н.

Группа: 80-201

Преподаватель: Чернышов Л.Н.

Дата:

Оценка:

Москва, 2019

1. Постановка задачи: Вариант №3

**Рациональная (несократимая) дробь** представляется парой целых чисел (*a*, *b*), где *a* — числитель, *b* — знаменатель. Создать класс Rational для работы с рациональными дробями. Обязательно должны быть реализованы операции:

- сложения add, (*a*, *b*) + (*c*, *d*) = (*ad* + *bc*, *bd*);

- вычитания sub, (*a*, *b*) – (*c*, d) = (*ad* – *bc*, *bd*);

- умножения mul, (*a*, *b*) ´ (*c*, *d*) = (*ac*, *bd*);

- деления div, (*a*, *b*) / (*c*, *d*) = (*ad*, *bc*);

- операции сравнения.

Должна быть реализована функция сокращения дроби reduce(), которая обязательно вызывается при выполнении арифметических операций.

Операции сложения, вычитания, умножения, деления, сравнения (на равенство, больше и меньше) должны быть выполнены в виде перегрузки операторов.

Необходимо реализовать пользовательский литерал для работы с константами типа **Rational.**

1. Ссылка на github

<https://github.com/AnnyWay775/oop_exercise_02>

1. Описание программы

Программа реализовывает работу с рациональными дробями, используя класс Rational (a - числитель, b - знаменатель). Для начала работы необходимо ввести значения двух дробей (с помощью выбора соответствующих пунктов меню). Затем, также при помощи меню, выбрать нужную нам операцию: сложение, вычитание, умножение, деление и сравнение. Результат будет находиться в третьей рациональной дроби и выводиться в терминале (в случае сравнения (>, < или =) результат опять-таки выводится в виде 1 - “да” и 0 - “нет”). Помимо этих действий в меню предусмотрены пункты для выхода из терминала, печати меню и демонстрации использования пользовательских литералов. Операции +, -, /, \*, >, <, == реализованы с перегрузкой операторов. Пользовательский литерал позволяет десятичное число представить в виде дроби, для дальнейшего сокращения ( reduce() ) и вывода в терминале. Сама функция reduce() реализована с помощью алгоритма Евклида.

1. Набор testcases

Три тект-файла: test\_01.txt, test\_02.txt и test\_03.txt

test\_01.txt - обычный тест, без каких-либо ошибок.

test\_02.txt - тест с отрицательными значениями, а также с применением пользовательских литералов.

test\_03.txt - знаменатель равен нулю, попытка деления на нулевую дробь.

|  |  |  |
| --- | --- | --- |
| test\_01.txt | test\_02.txt | test\_03.txt |
| 2  3  5  3  9  12  4  5  6  7  8  9  10  2  7  19  3  81  29329392  4  5  0 | 2  -7  9  3  -17  -51  4  5  6  7  2  1  3  8  11  0 | 3  10  0  2  4  5  7  0 |

1. Результаты выполнения тестов
   1. test\_01.txt

Программа для работы с рациональной (несократимой) дробью.

Числа a и b вводит пользователь, число c - результат применения оператора над a и b.

Меню команд

0. Выход

1. Печать меню

2. Ввести значение дроби a

3. Ввести значение дроби b

4. Сложение дробей a и b

5. Разность дробей a и b

6. Произведение дробей a и b

7. Частное дробей a и b

8. a == b ???

9. a > b ???

10. a < b ???

11. Пример использования пользовательских литералов

a=0/1 b=0/1 c=0/1

====>2

Числитель = 3

Знаменатель = 5

a=3/5 b=0/1 c=0/1

====>3

Числитель = 9

Знамнатель = 12

a=3/5 b=3/4 c=0/1

====>4

a=3/5 b=3/4 c=27/20

====>5

a=3/5 b=3/4 c=-3/20

====>6

a=3/5 b=3/4 c=9/20

====>7

a=3/5 b=3/4 c=4/5

====>8

a=3/5 b=3/4 c=0/1

====>9

a=3/5 b=3/4 c=0/1

====>10

a=3/5 b=3/4 c=1/1

====>2

Числитель = 7

Знаменатель = 19

a=7/19 b=3/4 c=1/1

====>3

Числитель = 81

Знамнатель = 29329392

a=7/19 b=27/9776464 c=1/1

====>4

a=7/19 b=27/9776464 c=68435761/185752816

====>5

a=7/19 b=27/9776464 c=68434735/185752816

====>0

* 1. test\_02.txt

Программа для работы с рациональной (несократимой) дробью.

Числа a и b вводит пользователь, число c - результат применения оператора над a и b.

Меню команд

0. Выход

1. Печать меню

2. Ввести значение дроби a

3. Ввести значение дроби b

4. Сложение дробей a и b

5. Разность дробей a и b

6. Произведение дробей a и b

7. Частное дробей a и b

8. a == b ???

9. a > b ???

10. a < b ???

11. Пример использования пользовательских литералов

a=0/1 b=0/1 c=0/1

====>2

Числитель = -7

Знаменатель = 9

a=-7/9 b=0/1 c=0/1

====>3

Числитель = -17

Знамнатель = -51

a=-7/9 b=1/3 c=0/1

====>4

a=-7/9 b=1/3 c=-4/9

====>5

a=-7/9 b=1/3 c=-10/9

====>6

a=-7/9 b=1/3 c=-7/27

====>7

a=-7/9 b=1/3 c=-7/3

====>2

Числитель = 1

Знаменатель = 3

a=1/3 b=1/3 c=-7/3

====>8

a=1/3 b=1/3 c=1/1

====>11

12.5 = 25/2

1412.25 = 5649/4

524.125 = 4193/8

a=1/3 b=1/3 c=1/1

====>0

* 1. test\_03.txt

Программа для работы с рациональной (несократимой) дробью.

Числа a и b вводит пользователь, число c - результат применения оператора над a и b.

Меню команд

0. Выход

1. Печать меню

2. Ввести значение дроби a

3. Ввести значение дроби b

4. Сложение дробей a и b

5. Разность дробей a и b

6. Произведение дробей a и b

7. Частное дробей a и b

8. a == b ???

9. a > b ???

10. a < b ???

11. Пример использования пользовательских литералов

a=0/1 b=0/1 c=0/1

====>3

Числитель = 10

Знамнатель = 0

Введённый знаменатель равен нулю.

a=0/1 b=0/1 c=0/1

====>2

Числитель = 4

Знаменатель = 5

a=4/5 b=0/1 c=0/1

====>7

Деление на ноль.

a=4/5 b=0/1 c=0/1

====>0

1. Листинг программы

/\*

Вариант 3

Рациональная (несократимая) дробь представляется парой целых чисел (a, b), где a — числитель, b — знаменатель. Создать класс Rational для работы с рациональными дробями. Обязательно должны быть реализованы операции:

- сложения add, (a, b) + (c, d) = (ad + bc, bd);

- вычитания sub, (a, b) – (c, d) = (ad – bc, bd);

- умножения mul, (a, b) ´ (c, d) = (ac, bd);

- деления div, (a, b) / (c, d) = (ad, bc);

- операции сравнения.

Должна быть реализована функция сокращения дроби reduce(), которая обязательно вызывается при выполнении арифметических операций.

Операции сложения, вычитания, умножения, деления, сравнения (на равенство, больше и меньше) должны быть выполнены в виде перегрузки операторов.

Необходимо реализовать пользовательский литерал для работы с константами типа Rational.

Кулагин Даниил Николаевич

М8О-201Б-18

\*/

#include <iostream>

#include <cstdlib>

class Rational

{

private:

int chisl; //числитель Рациональной Дроби

int znamen; //знаменатель Рациональной Дроби

public:

Rational()

{

chisl = 0;

znamen = 1;

}

Rational(int chislitel, int znamenatel)

{

set(chislitel, znamenatel);

}

Rational operator+(Rational b) //Оператор сложения

{

Rational c(chisl \* b.znamen + znamen \* b.chisl, znamen \* b.znamen);

return c;

}

Rational operator-(Rational b) //Оператор вычитания

{

Rational c(chisl \* b.znamen - znamen \* b.chisl, znamen \* b.znamen);

return c;

}

Rational operator\*(Rational b) //Оператор умножения

{

Rational c(chisl \* b.chisl, znamen \* b.znamen);

return c;

}

Rational operator/(Rational b) //Оператор деления

{

if (b.chisl)

{

Rational c(chisl \* b.znamen, znamen \* b.chisl);

return c;

}

else {

std::cout << "Деление на ноль.\n";

return Rational();

}

}

int operator==(Rational b) //Оператор равенства

{

if (chisl == b.chisl && znamen == b.znamen) return 1;

else return 0;

}

int operator>(Rational b) //Оператор "Больше"

{

if (chisl \* b.znamen - znamen \* b.chisl > 0) return 1;

else return 0;

}

int operator<(Rational b) //Оператор "Меньше"

{

if (chisl \* b.znamen - znamen \* b.chisl < 0) return 1;

else return 0;

}

int get\_chisl()

{

return chisl;

}

int get\_znamen()

{

return znamen;

}

void set(int chislitel, int znamenatel)

{

chisl = chislitel;

if (znamenatel)

znamen = znamenatel;

else

{

std::cout << "Введённый знаменатель равен нулю.\n";

chisl = 0;

znamen = 1;

}

}

int EuclideanAlgorithm(int a, int b) //алгоритм Евклида (поиск НОК)

{

if (b == 0)

return a;

return EuclideanAlgorithm(b, a % b);

}

void reduce() //Сокращение дроби до рациональной

{

if (znamen == 1) return;

int NOK = EuclideanAlgorithm(chisl, znamen);

while (true)

{

if (NOK == 1) break; //Если общих делителей больше нет

chisl /= NOK;

znamen /= NOK;

NOK = EuclideanAlgorithm(chisl, znamen);

}

if (znamen < 0) //Если знаменатель отрицателен, то "отправляем" минус в числитель

{

chisl \*= -1;

znamen \*= -1;

}

}

};

Rational operator "" \_frac (long double a) { //Реализация пользовательского литерала,

int test = (int)a; //переводящего десятичное число

int count = 1; //в дробь

while (test != a) {

count\*=10;

a \*= 10;

test = (int)a;

}

return Rational((int)a, count);

}

void cout\_menu()

{

std::cout << "Меню команд\n";

std::cout << "0. Выход\n";

std::cout << "1. Печать меню\n";

std::cout << "2. Ввести значение дроби a\n";

std::cout << "3. Ввести значение дроби b\n";

std::cout << "4. Сложение дробей a и b\n";

std::cout << "5. Разность дробей a и b\n";

std::cout << "6. Произведение дробей a и b\n";

std::cout << "7. Частное дробей a и b\n";

std::cout << "8. a == b ???\n";

std::cout << "9. a > b ???\n";

std::cout << "10. a < b ???\n";

std::cout << "11. Пример использования пользовательских литералов\n";

}

int main()

{

setlocale(LC\_ALL, "Russian");

std::cout << "Программа для работы с рациональной (несократимой) дробью.\n" <<

"Числа a и b вводит пользователь, число c - результат применения оператора над a и b." << std::endl;

Rational a, b, c, example1(12.5\_frac), example2(1412.25\_frac), example3(524.125\_frac);

int action = 1, num, den;

while (action != 0)

{

switch (action)

{

case 1: cout\_menu(); break;

case 2: std::cout << "Числитель = "; std::cin >> num;

std::cout << "Знаменатель = "; std::cin >> den;

a.set(num, den);

break;

case 3:

std::cout << "Числитель = "; std::cin >> num;

std::cout << "Знамнатель = "; std::cin >> den;

b.set(num, den);

break;

case 4: c = a + b; break;

case 5: c = a - b; break;

case 6: c = a \* b; break;

case 7: c = a / b; break;

case 8: c.set(a == b, 1); break;

case 9: c.set(a > b, 1); break;

case 10: c.set(a < b, 1); break;

case 11:

example1.reduce(); example2.reduce(); example3.reduce();

std::cout << "12.5 = " << example1.get\_chisl() << "/" << example1.get\_znamen() << " \n";

std::cout << "1412.25 = " << example2.get\_chisl() << "/" << example2.get\_znamen() << " \n";

std::cout << "524.125 = " << example3.get\_chisl() << "/" << example3.get\_znamen() << " \n"; break;

default: break;

}

a.reduce(); b.reduce(); c.reduce();

std::cout << "a=" << a.get\_chisl() << "/" << a.get\_znamen() << " ";

std::cout << "b=" << b.get\_chisl() << "/" << b.get\_znamen() << " ";

std::cout << "c=" << c.get\_chisl() << "/" << c.get\_znamen() << " \n====>";

std::cin >> action;

}

return 0;

}

1. Вывод

Я научился перегружать операторы на языке C++, создавать пользовательские литералы, а также создавать парсер для битовых строк.

1. Список литературы

1) Пользовательские литералы в C++11 [электронный ресурс].

URL: <https://habr.com/ru/post/140357/>.

2) Перегрузка операторов в C++ [электронный ресурс].

URL: <https://habr.com/ru/post/132014/>.

3) Сокращение дроби, формулы и примеры решений [электронный ресурс].

URL: <https://www.webmath.ru/poleznoe/formules_12_7.php>