Федеральное государственное бюджетное образовательное учреждение высшего образования

«Сибирский государственный университет

телекоммуникаций и информатики»

Кафедра ПМиК

Современные технологии программирования 1

Лабораторная работа №4.

Абстрактный тип данных простая дробь

Выполнил: студент 4 курса

Ф. ИВТ, группа: ИП-711

Мартасов И. О.

Проверил: доцент кафедры ПМиК

Зайцев Михаил Георгиевич

Новосибирск, 2020

**Содержание**

1. Задание.
2. Текст программы.
3. Тестовые наборы данных для тестирования и результаты тестирования.

**Задание**

1. Реализовать абстрактный тип данных «простая дробь», используя класс

С++ в соответствии с приведенной ниже спецификацией.

2. Протестировать каждую операцию, определенную на типе данных,

используя средства модульного тестирования.

3. Если необходимо, предусмотрите возбуждение исключительных

ситуаций**.**

**Данные**

Простая дробь (тип TFrac) - это пара целых чисел: числитель и знаменатель (a/b). Простые дроби не изменяемые.

# Операции

Операции могут вызываться только объектом простая дробь (тип **TFrac**), указатель на который в них передаётся по умолчанию. При описании операций этот объект называется «сама дробь» this.

|  |  |
| --- | --- |
| ***Конструктор*** |  |
| Вход: | Пара целых чисел (a) и (b). |
| Предусловия: | b не равно 0. В противном случае  возбуждается исключение. |
| Процесс: | Инициализирует поля простой дроби (тип TFrac): числитель значением a, знаменатель - (b). В случае необходимости дробь предварительно сокращается.  Например:  *Конструктор*(6,3) = (2/1)  *Конструктор*(0,3) = (0/3). |
| Выход: | Нет. |
| Постусловия: | Поля объекта проинициализированы  начальными значениями. |
|  | |
| ***Конструктор*** |  |
| Вход: | Строковое представление простой дроби  . Например: ‘7/9’. |
| Предусловия: | b не равно 0. В противном случае  возбуждается исключение. |
| Процесс: | Инициализирует поля простой дроби (тип TFrac) строкой f =’a/b’. Числитель значением a, знаменатель - b. В случае необходимости дробь предварительно сокращается.  Например:  *Конструктор*(‘6/3’) = 2/1  *Конструктор* (‘0/3’) = 0/3 |
| Выход: | Нет. |
| Постусловия: | Поля объекта проинициализированы  начальными значениями. |
|  | |
| **Копировать:** |  |
| Вход: | Нет. |
| Предусловия: | Нет. |
| Процесс: | Создаёт копию самой дроби (тип TFrac) с числителем, и знаменателем такими же, как у самой дроби. |
| Выход: | Простая дробь (тип TFrac).  Например:  c = 2/1, Копировать(c) = 2/1 |
| Постусловия: | Нет. |
|  | |
| ***Сложить*** |  |
| Вход: | Простая дробь d (тип TFrac). |
| Предусловия: | Нет. |
| Процесс: | Создаёт и возвращает простую дробь (тип TFrac), полученную сложением самой дроби q = a1/b1 с d =a2/b2: ((a1/b1)+(a2/b2)=(a1\*b2 + a2\*b1)/( b1\* b2)).  Например: q = 1/2, d = -3/4  q.Сложить(d) = -1/4. |
| Выход: | Простая дробь (тип TFrac). |
| Постусловия: | Нет. |
|  | |
| ***Умножить*** |  |
| Вход: | Простая дробь d (тип TFrac). |
| Предусловия: | Нет. |
| Процесс: | Создаёт простую дробь (тип TFrac), полученную умножением самой дроби q  = a1/b1 на d = a2/b2 ((a1/b1)\*(a2/b2)=(a1\*  a2)/( b1\* b2)). |
| Выход: | Простая дробь (тип TFrac). |
| Постусловия: | Нет. |
|  | |
| ***Вычесть*** |  |
| Вход: | Простая дробь d (тип TFrac). |
| Предусловия: | Нет. |
| Процесс: | Создаёт и возвращает простую дробь (тип TFrac), полученную вычитанием d  = a2/b2 из самой дроби q = a1/b1: ((a1/b1)-(a2/b2)=(a1\* b2-a2\*b1)/(b1\*b2)).  Например: q = (1/2), d = (1/2)  q.Вычесть(d) = (0/1). |
| Выход: | Простая дробь (тип TFrac). |
| Постусловия: | Нет |
|  | |
| ***Делить*** |  |
| Вход: | Простая дробь d (тип TFrac). |
| Предусловия: | Числитель числа d не равно 0. |
| Процесс: | Создаёт и возвращает простую дробь (тип TFrac), полученное делением самой дроби q = a1/b1 на дробь d = a2/b2:  ((a1/b1)/(a2/b2)=(a1\* b2)/( a2\*b1)). |
| Выход: | Простая дробь (тип TFrac). |
| Постусловия: | Нет. |
|  | |
| ***Квадрат*** |  |
| Вход: | Нет. |
| Предусловия: | Нет. |
| Процесс: | Создаёт и возвращает простую дробь (тип TFrac), полученную умножением самой дроби на себя: ((a/b)\*(a/b)=(a\* a)/(  b\* b)). |
| Выход: | Простая дробь (тип TFrac). |
| Постусловия: | Нет. |
|  | |
| ***Обратное*** |  |
| Вход: | Нет. |
| Предусловия: | Нет. |
| Процесс: | Создаёт и возвращает простую дробь  (тип TFrac), полученное делением единицы на саму дробь: 1/((a/b) = b/a. |
| Выход: | Простая дробь (тип TFrac) |
| Постусловия: | Нет. |
|  | |
| ***Минус*** |  |
| Вход: | Нет. |
| Предусловия: | Нет. |
| Процесс: | Создаёт простую дробь, являющуюся разностью простых дробей z и q, где z - простая дробь (0/1), дробь, вызвавшая  метод. |
| Выход: | Простая дробь (тип TFrac). |
| Постусловия: | Нет. |
|  | |
| ***Равно*** |  |
| Вход: | Простая дробь d (тип TFrac). |
| Предусловия: | Нет |
| Процесс: | Сравнивает саму простую дробь q и d. Возвращает значение True, если q и d - тождественные простые дроби, и  значение False - в противном случае. |
| Выход: | Булевское значение. |
| Постусловия: | Нет. |
|  | |
| ***Больше*** |  |
| Вход: | Простая дробь d (тип TFrac). |
| Предусловия: | Нет. |
| Процесс: | Сравнивает саму простую дробь q и d.  Возвращает значение True, если q > d, - значение False - в противном случае. |
| Выход: | Булевское значение. |
| Постусловия: | Нет. |
|  | |
| ***ВзятьЧислительЧисло*** |  |
| Вход: |  |
| Предусловия: | Нет. |
| Процесс: | Возвращает значение числителя дроби в  числовом формате. |
| Выход: | Вещественное значение. |
| Постусловия: | Нет. |
|  | |
| ***ВзятьЗнаменательЧисло*** |  |
| Вход: | Нет. |
| Предусловия: | Нет. |
| Процесс: | Возвращает значение знаменателя дроби  в числовом формате. |
| Выход: | Вещественное значение. |
| Постусловия: | Нет. |
|  | |
| ***ВзятьЧислительСтрока*** |  |
| Вход: | Нет. |
| Предусловия: | Нет. |
| Процесс: | Возвращает значение числителя дроби в  строковом формате. |
| Выход: | Строка. |
| Постусловия: | Нет. |
|  | |
| ***ВзятьЗнаменательСтрока*** |  |
| Вход: | Нет. |
| Предусловия: | Нет. |
| Процесс: | Возвращает значение знаменателя дроби  в строковом формате. |
| Выход: | Строка. |
| Постусловия: | Нет. |
|  | |
| ***ВзятьДробьСтрока*** |  |
| Вход: | Нет. |
| Предусловия: | Нет. |
| Процесс: | Возвращает значение простой дроби, в  строковом формате. |
| Выход: | Строка. |
| Постусловия: | Нет. |
|  |  |

***end TFrac***

**Текст программы**

**TFrac.cpp**

#include "pch.h"

#include "TFrac.h"

int TFrac::gcd(int a, int b)

{

if (a == 0)

{

if (b < 0)

{

return -1 \* b;

}

return b;

}

if (a < 0)

{

a \*= -1;

}

if (b < 0)

{

b \*= -1;

}

while (a != b)

{

if (a > b)

{

a -= b;

}

else

{

b -= a;

}

}

return a;

}

int TFrac::getchisl()

{

return this->chisl;

}

int TFrac::getznam()

{

return this->znam;

}

string TFrac::str\_getchisl()

{

return to\_string(this->chisl);

}

string TFrac::str\_getznam()

{

return to\_string(this->znam);

}

string TFrac::str\_getprime()

{

return to\_string(this->chisl) + "/" + to\_string(this->znam);

}

TFrac::TFrac(int a, int b)

{

try

{

if (b == 0)

{

throw "ERROR 0 ZNAM";

}

}

catch(const char\* exception)

{

std::cerr << "Error: " << exception << endl;

exit(1);

}

this->chisl = (a / gcd(a, b));

this->znam = (b / gcd(a, b));

if (this->znam < 0)

{

this->znam \*= -1;

this->chisl = -this->chisl;

}

}

TFrac::TFrac(string tfrac)

{

int a = 0;

int b = 0;

string delimiter = "/";

string tokenA = tfrac.substr(0, tfrac.find(delimiter));

a = stoi(tokenA);

string tokenB = tfrac.substr(tfrac.find(delimiter) + delimiter.length(), std::string::npos);

b = stoi(tokenB);

try

{

if (b == 0)

{

throw "ERROR 0 ZNAM";

}

}

catch (const char\* exception)

{

std::cerr << "Error: " << exception << endl;

exit(1);

}

this->chisl = (a / gcd(a, b));

this->znam = (b / gcd(a, b));

if (this->znam < 0)

{

this->znam \*= -1;

this->chisl = -this->chisl;

}

}

TFrac TFrac::copy()

{

TFrac q(this->chisl, this->znam);

return q;

}

TFrac TFrac::add(TFrac d)

{

int a = (this->chisl \* d.getznam() + this->znam \* d.getchisl());

int b = (this->znam \* d.getznam());

int c = gcd(this->chisl, this->znam);

a /= c;

b /= c;

TFrac q(a, b);

return q;

}

TFrac TFrac::sub(TFrac d)

{

int a = (this->chisl \* d.getznam() - this->znam \* d.getchisl());

int b = (this->znam \* d.getznam());

int c = gcd(this->chisl, this->znam);

a /= c;

b /= c;

TFrac q(a, b);

return q;

}

TFrac TFrac::mul(TFrac d)

{

int a = (this->chisl \* d.getchisl());

int b = (this->znam \* d.getznam());

int c = gcd(this->chisl, this->znam);

a /= c;

b /= c;

TFrac q(a, b);

return q;

}

TFrac TFrac::div(TFrac d)

{

try

{

if (d.getchisl() == 0)

{

throw "ERROR 0 CHISL IN DIV";

}

}

catch (const char\* exception)

{

std::cerr << "Error: " << exception << endl;

exit(1);

}

int a = (this->chisl \* d.getznam());

int b = (this->znam \* d.getchisl());

int c = gcd(this->chisl, this->znam);

a /= c;

b /= c;

if (b < 0)

{

b \*= -1;

a = -a;

}

TFrac q(a, b);

return q;

}

TFrac TFrac::square()

{

int a = (this->chisl \* this->chisl);

int b = (this->znam \* this->znam);

TFrac q(a, b);

return q;

}

TFrac TFrac::reverse()

{

TFrac d(1, 1);

TFrac q = d.div(\*this);

return q;

}

TFrac TFrac::minus()

{

TFrac d(0, 1);

TFrac q = d.sub(\*this);

return q;

}

bool TFrac::equals(TFrac d)

{

if ((this->chisl \* d.getznam()) == (d.getchisl() \* this->znam))

{

return true;

}

else

{

return false;

}

}

bool TFrac::bigger(TFrac d)

{

if ((this->chisl \* d.getznam()) > (d.getchisl() \* this->znam))

{

return true;

}

else

{

return false;

}

}

**TFrac.h**

#pragma once

#include <string>

#include <iostream>

using namespace std;

class TFrac

{

private:

int chisl;

int znam;

int gcd(int a, int b);

public:

TFrac(int a, int b);

TFrac(string a);

TFrac copy();

TFrac add(TFrac d);

TFrac sub(TFrac d);

TFrac mul(TFrac d);

TFrac div(TFrac d);

TFrac square();

TFrac reverse();

TFrac minus();

bool equals(TFrac d);

bool bigger(TFrac d);

int getchisl();

int getznam();

string str\_getprime();

string str\_getchisl();

string str\_getznam();

};

Тестовые наборы данных для тестирования и результаты тестирования

