Православный Свято-Тихоновский гуманитарный университет

Факультет информатики и прикладной математики

КАФЕДРА ИНФОРМАТИКИ

Лабораторная работа №3

Тема: “[Перегрузка операторов](https://online.pstgu.ru/mod/bigbluebuttonbn/view.php?id=26457). [Конструктор копирования](https://online.pstgu.ru/mod/bigbluebuttonbn/view.php?id=29264)”.

Выполнил: Королев Иван Алексеевич

Проверил: Андрей Сергеевич Евсеев

Группа: 02-1-05

Москва 2021

## 1. Постановка задачи

Необходимо реализовать класс ***Остатки от деления.***Например остаток(100, 7) - остаток от деления на 100, равный 7.  
Можно складывать и вычитать, только если совпадает делитель. Например, можно сложить остаток(100, 88) + остаток(100, 13) = остаток(100, 1) *[88+13=101; 101%100=1]*Но нельзя остаток(2, 1)+остаток(5, 3).

В классе должны быть определены различные конструкторы (по умолчанию, конструктор копирования).

Должны быть перегружены следующие операторы: +, -, ==, !=

В качестве второго операнда выступает объект этого же класса. Однако в некоторых вариантах дополнительно вторым операндом может быть переменная одного из встроенных типов данных - int, double, float, char\*.

## 2. Листинг описания класса

Mode.h

#pragma once

class Mode

{

private:

int resultMod; //результат от деления

int basis; //на что делим

bool checkOnEqualentBase(int basis); //проверка делителей

public:

Mode();

Mode(const Mode& d); //конструктор копирования

Mode(int value, int basis);

Mode operator+(Mode d); // прегрузка оператора +

Mode operator-(Mode d); // прегрузка оператора -

bool operator==(Mode d); // прегрузка оператора ==

bool operator!=(Mode d); // прегрузка оператора !=

int getResultMod() { return resultMod; }

int getBasis() { return basis; }

};

## 3. Листинг реализации класса.

Mode.cpp

#include "Mode.h"

#include <iostream>

bool Mode::checkOnEqualentBase(int basis) //проверка делителей

{

if (this->basis != basis)

return false;

else

return true;

}

Mode::Mode()

{

resultMod = 0;

basis = 0;

}

Mode::Mode(const Mode& d) //конструктор копирования

{

this->resultMod = d.resultMod;

this->basis = d.basis;

}

Mode::Mode(int value, int basis)

{

resultMod = value % basis;

this->basis = basis;

}

Mode Mode::operator+(Mode d) // прегрузка оператора +

{

Mode newObj;

if (checkOnEqualentBase(d.basis))

{

if (resultMod + d.resultMod < 0) {

Mode newObj1((-1 \* (resultMod + d.resultMod)) % basis, basis); // умножаю на -1, т.к. остаток не может быть отрицательным

newObj = newObj1;

}

else {

Mode newObj2((resultMod + d.resultMod) % basis, basis);

newObj = newObj2;

}

return newObj;

}

std::cerr << "Ответ: Разные базисы!!!\n";

}

Mode Mode::operator-(Mode d) // прегрузка оператора -

{

Mode newObj;

if (checkOnEqualentBase(d.basis))

{

if (resultMod - d.resultMod < 0) {

Mode newObj1((-1 \* (resultMod - d.resultMod)) % basis, basis); // умножаю на -1, т.к. остаток не может быть отрицательным

newObj = newObj1;

}

else {

Mode newObj2((resultMod - d.resultMod) % basis, basis);

newObj = newObj2;

}

return newObj;

}

std::cerr << "Ответ: Разные базисы!!!\n";

}

bool Mode::operator==(Mode d) // прегрузка оператора ==

{

if (resultMod == d.resultMod && basis == d.basis)

return true;

return false;

}

bool Mode::operator!=(Mode d) // прегрузка оператора !=

{

if (resultMod != d.resultMod && basis != d.basis)

return true;

return false;

}

Test.cpp

#include "Mode.h"

#include <iostream>

using namespace std;

int main()

{

setlocale(LC\_ALL, "Russian");

Mode obj1(29, 3); // 1 объект

Mode obj2(28, 3); // 2 объект

cout << "Программа, которая умеет складывать и вычитать остатки от деления!" << endl << endl << "Данно: \n";

cout << "obj1: " << obj1.getResultMod() << ", " << obj1.getBasis() << endl;

cout << "obj2: " << obj2.getResultMod() << ", " << obj2.getBasis() << endl;

cout << endl;

//демонстрация работы переопределенного оператора +

cout << "Вопрос: obj3 = obj1 + obj2 ?\n";

Mode obj3 = obj1 + obj2;

if (obj2.getBasis() == obj1.getBasis())

cout << "Ответ: obj3: " << obj3.getResultMod() << ", " << obj3.getBasis() << endl;

cout << endl;

//демонстрация работы переопределенного оператора -

cout << "Вопрос: obj3 = obj1 - obj2 ?\n";

obj3 = obj1 - obj2;

if (obj2.getBasis() == obj1.getBasis())

cout << "Ответ: obj3: " << obj3.getResultMod() << ", " << obj3.getBasis() << endl;

cout << endl;

//демонстрация работы переопределенного оператора ==

cout << "Вопрос: obj1 == obj2 ?\n";

cout << "Ответ: ";

if (obj1 == obj2) {

cout << "obj1 равен obj2 \n";

}

else

cout << "obj1 не равен obj2\n";

cout << endl;

//демонстрация работы переопределенного оператора !=

cout << "Вопрос: obj1 != obj2 ?\n";

cout << "Ответ: ";

if (obj1 != obj2) {

cout << "obj1 не равен obj2 \n";

}

else

cout << "obj1 равен obj2";

cout << endl;

return 0;

## 4. Результаты работы программы проверки класса (какие действия выполнялись, какие результаты были получены).

1) Было создано 3 объекта: “obj1, obj2, obj3”.

2) Протестировано и вывод программы на Рис.1.

Изображение выглядит как текст

Автоматически созданное описание

Рис.1