Министерство науки и высшего образования РФ

Федеральное бюджетное государственное образовательное учреждение высшего профессионального образования

Пермский национальный исследовательский политехнический университет (ПНИПУ)

Факультет: Электротехнический (ЭТФ)

Направление: 09.03.01 «Информатика и вычислительная техника (Автоматизированные системы обработки информации и управления)»

Кафедра: «Информационные технологии и автоматизированных систем» (ИТАС)

Информатика

Лабораторная работа №3

Перегрузка операций

Студент: Балтаев Э. А.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(дата, подпись)

Группа: АСУ-20-1бзу

Работу проверил:

доцент кафедры ИТАС

Полякова О. А.

Оценка: \_\_\_\_\_\_\_\_\_\_\_

Подпись:\_\_\_\_\_\_\_\_\_\_\_

Пермь 2021г.

Постановка задачи

1. Определить пользовательский класс

2. Определить в классе следующие конструкторы: без параметров с параметрами и копирования

3. Определить в классе деструктор

4. Определить в классе компоненты-функции для просмотра и установки полей данных (селекторы и модификаторы)

5. Перегрузить операцию присваивания

6. Перегрузить операцию ввода и вывода объектов с помощью потоков

7. Перегрузить операции указанные в варианте

8. Написать программу, в которой продемонстрировано создание объектов и работу всех перегруженных операций

Задание варианта

Составить класс Money для работы с денежными суммами. Число должно быть представлено двумя полями: типа long для рублей и типа int для копеек. Дробная часть числа при выводе на экран должна быть отделена от целой части запятой. Реализовать:

- деление сумм,

- умножение суммы на дробное число.

Описание класса

#pragma once

#include <iostream>

using namespace std;

class money

{

private:

long rub;

int cop;

public:

money();

money(long r, int c);

money(const money& m);

~money() {}

long get\_rub() { return rub; }

long get\_cop() { return cop; }

void set\_rub(long r);

void set\_cop(int c);

money& operator= (const money& m);

int operator/ (const money& m);

money operator\* (int val);

friend istream& operator>> (istream& in, money& m);

friend ostream& operator<< (ostream& out, const money& m);

};

Определение компонентных функций

#include "money.h"

money::money()

{

rub = 0;

cop = 0;

}

money::money(long r, int c)

{

set\_rub(r);

set\_cop(c);

}

money::money(const money& m)

{

rub = m.rub;

cop = m.cop;

}

void money::set\_rub(long r)

{

if (r < 0)

rub = 0;

else

rub = r;

}

void money::set\_cop(int c)

{

if (c < 0)

cop = 0;

else {

rub += c / 100;

cop = c % 100;

}

}

money& money::operator=(const money& m)

{

if (&m == this)

return \*this;

rub = m.rub;

cop = m.cop;

return \*this;

}

int money::operator/(const money& m)

{

long s1 = rub \* 100 + cop,

s2 = m.rub \* 100 + m.cop;

double res = s1 / s2;

return res;

}

money money::operator\*(int val)

{

int r = rub \* val,

c = cop \* val;

money m(r, c);

return m;

}

istream& operator>>(istream& in, money& m)

{

long r;

int c;

cout << "rub? "; in >> r; m.set\_rub(r);

cout << "cop? "; in >> c; m.set\_cop(c);

return in;

}

ostream& operator<<(ostream& out, const money& m)

{

out << m.rub << "," << m.cop;

return out;

}

Определение главного файла

#include <iostream>

#include "money.h"

int main()

{

system("chcp 1251");

cout << "\nИспользование конструкторов и оператора присвоения\n";

money m1(300, 230);

money m2 = m1;

m2.set\_rub(35);

cout << "\nВвод и вывод\n";

money m3;

cin >> m3;

cout << "m1 - " << m1 << endl;

cout << "m2 - " << m2 << endl;

cout << "m3 - " << m3 << endl;

cout << "\nПерегрузка операций\n";

int val, res1 = m1 / m3;

cout << "m1 / m3 - " << res1 << endl;

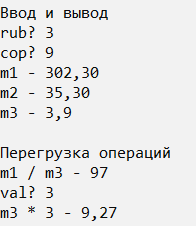
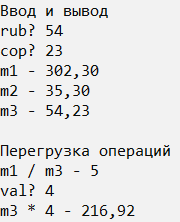
cout << "val? "; cin >> val;

money res2 = m3 \* val;

cout << "m3 \* " << val << " - " << res2 << endl;

}

Результаты программы:

Ответы на контрольные вопросы

1) Чтобы они могли пользоваться private полями и методы вне класса

2) Дружественная функция объявляется внутри класса с ключевым словом friend и в качестве параметра должна передаваться ссылка на объект данного класса. Одна функция может быть дружественным сразу нескольким классам. Дружественная функция может быть обычной функцией и методом другого класса.

3) Как метод класса или дружественную функцию

4) 1

5) 2

6) 1

7) 2

8) Префиксный – возвращает измененную себя, постфиксная возвращает новый объект. Также в префиксном имеется первый параметр int для отличия.

9) С помощью компонентной функции класса, где параметр – ссылка на объект присвоения, возвращает текущий объект.

10) Текущий объект

11) С помощью дружественных функций класса: первый аргумент потоки ввода или вывода, второй – ссылка на объект манипуляции.

12) Постфиксный инкремент

13) Постфиксный инкремент

14) Сравнение объектов a и b (бинарная операция - меньше)

15) Сравнение объектов a и b (бинарная операция - больше)