Министерство науки и высшего образования Российской Федерации

Федеральное государственное автономное образовательное учреждение

высшего образования

**«Пермский национальный исследовательский**

**политехнический университет»**

Электротехнический факультет

Кафедра «Информационные технологии и автоматизированные системы»

**О Т Ч Е Т  
по лабораторной работе на тему Классы №3**

**по дисциплине «Основы алгоритмизации и программирования»**

Выполнил студент гр. РИС-21-1б

Ибрагимов Марк Магамедович

(Фамилия, Имя, Отчество)

(подпись)

Проверил:

доцент

кафедры ИТАС

Полякова О.А.

(оценка) (подпись)

(дата)

Пермь 2022

**Постановка задачи**

Создать 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:

-сложение денежных сумм,

-вычитание денежных сумм.

**Анализ задачи**

1. Необходимо создать класс данных. Также создаются методы класса и перегруженные операторы
2. Пользователь вводит значения рублей и копеек при помощи перегруженных операторов
3. С их же помощью эти значения складываюся, вычитаются и выводятся

**Описание переменных**

Money – класс, конструкторы и деструктор  
rub, kop – объекты класса, принимающие значения рублей и копеек соответственно  
operator – перегружаемый оператор сложения (+), вычитания (-), уравнения (=), ввода (>>), вывода (<<)  
a, b, c, d, t, in, r, k -–переменные для работы с операторами, методами

**Код программы**

#include<iostream>

using namespace std;

class Money

{

long rub;

int kop;

public:

Money() { rub = 0, kop = 0; };

Money(long r, int k) { rub = r, kop = k; };

Money(const Money& t) { rub = t.rub, kop = t.kop; };

~Money() {};

long GetRub() { return rub; };

int GetKop() { return kop; };

void SetRub(long r) { rub = r; };

void SetKop(int k) { kop = k; };

Money& operator =(const Money&);

Money operator +(const Money&);

Money operator -(const Money&);

friend istream& operator>>(istream& in, Money& t);

friend ostream& operator<<(ostream& in, Money& t);

};

Money& Money::operator=(const Money& t)

{

if (&t == this)

return \*this;

rub = t.rub;

kop = t.kop;

return \*this;

}

Money Money::operator+(const Money& t)

{

int temp1 = rub \* 100 + kop;

int temp2 = t.rub \* 100 + t.kop;

Money p;

p.rub = (temp1 + temp2) / 100;

p.kop = (temp1 + temp2) % 100;

return p;

}

Money Money::operator-(const Money& t)

{

int temp1 = rub \* 100 + kop;

int temp2 = t.rub \* 100 + t.kop;

Money n;

n.rub = (temp1 - temp2) / 100;

n.kop = abs(temp1 - temp2) % 100;

return n;

}

istream& operator>>(istream& in, Money& t)

{

cout << "Рубли: ";

in >> t.rub;

cout << "Копейки: ";

in >> t.kop;

return in;

}

ostream& operator<<(ostream& out, Money& t)

{

out << t.rub << ',' << t.kop;

return out;

}

int main()

{

setlocale(LC\_ALL, "rus");

Money a;

Money b;

Money c;

Money d;

cin >> a;

cin >> b;

c = a + b;

d = a - b;

cout << "a=" << a << '\n';

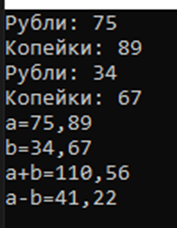
cout << "b=" << b << '\n';

cout << "a+b=" << c << '\n';

cout << "a-b=" << d << '\n';

}

**Примеры выполнения программы**

****

**Анализ результатов**

Программа сработала корректно и вывела необходимые результаты.

**Ответы на вопросы:**

1. **Для чего используются дружественные функции и классы?**

Дружественные функции применяются для доступа к скрытым полям класса и представляют собой альтернативу методам. Метод, как правило, описывает свойство объекта, а в виде дружественных функций оформляются действия, не являющиеся свойствами класса, но концептуально входящие в его интерфейс и нуждающиеся в доступе к его скрытым полям, например, переопределенные операции вывода объектов

1. **Сформулировать правила описания и особенности дружественных функций.**

Правила описания и особенности дружественных функций:

* Дружественная функция объявляется *внутри класса,* к элементам которого ей нужен доступ, с ключевым словом friend. В качестве параметра ей должен передаваться объект или ссылка на объект класса, поскольку указатель this ей не передается.
* Дружественная функция может быть обычной функцией или методом другого ранее определенного класса. На нее не распространяется действие спецификаторов доступа, место размещения ее объявления в классе безразлично.
* Одна функция может быть дружественной сразу нескольким классам.

1. **Каким образом можно перегрузить унарные операции?**

Унарную операцию можно перегрузить:

* + - Как компонентную функцию класса
    - Как внешнюю (глобальную) функцию

1. **Сколько операндов должна иметь унарная функция-операция, определяемая внутри класса?**

Унарная функция-операция, определяемая *внутри класса,* должна быть представлена с помощью нестатического метода без параметров, при этом операндом является вызвавший ее объект.

1. **Сколько операндов должна иметь унарная функция-операция, определяемая вне класса?**

Если функция определяется *вне класса,* она должна иметь один параметр типа класса.

1. **Сколько операндов должна иметь бинарная функция-операция, определяемая внутри класса?**

Бинарная функция-операция, определяемая *внутри класса,* должна быть представлена с помощью нестатического метода с параметрами, при этом вызвавший ее объект считается первым операндом.

1. **Сколько операндов должна иметь бинарная функция-операция, определяемая вне класса?**

Если функция определяется *вне класса,* она должна иметь два параметра типа класса.

1. **Чем отличается перегрузка префиксных и постфиксных унарных операций?**

Разница между ними заключается в возвращаемом ими значении. Перегруженные префиксные операторы возвращают объект после того, как он был инкрементирован или декрементирован. Следовательно, перегрузить их довольно просто. Мы просто увеличиваем или уменьшаем наши переменные-члены, а затем возвращаем \*this. Постфиксные операторы, напротив, должны возвращать состояние объекта до его инкремента или декремента.

1. **Каким образом можно перегрузить операцию присваивания?**

Операция присваивания определена в любом классе по умолчанию как поэлементное копирование. Эта операция вызывается каждый раз, когда одному существующему объекту присваивается значение другого. Если класс содержит поля, память под которые выделяется динамически, необходимо определить собственную операцию присваивания.

1. **Что должна возвращать операция присваивания?**

Чтобы сохранить семантику присваивания, операция-функция должна возвращать ссылку на объект, для которого она вызвана, и принимать в качестве параметра единственный аргумент — ссылку на присваиваемый объект.

1. **Каким образом можно перегрузить операции ввода-вывода?**

Операции ввода-вывода operator>> и operator<< всегда реализуются как внешние дружественные функции, т. к. левым операндом этих операций являются потоки.

1. **В программе описан класс class Student**

**{**

**…**

**Student& operator++();**

**….**

**};**

**и определен объект этого класса Student s;**

**Выполняется операция**

**++s;**

**Каким образом, компилятор будет воспринимать вызов функции-операции?**

Как вызов перегруженного оператора

1. **В программе описан класс class Student**

**{**

**…**

**bool operator<(Student &P);**

**….**

**};**

**и определены объекты этого класса Student a,b;**

**Выполняется операция cout<<a<b;**

**Каким образом, компилятор будет воспринимать вызов функции-операции?**

Компилятор выдаст ошибку, так как операция вывода не переопределена (у оператора << приоритет больше чем у <).

1. **В программе описан класс class Student**

**{**

**…**

**friend bool operator >(const Person&, Person&)**

**….**

**};**

**и определены объекты этого класса Student a,b;**

**Выполняется операция cout<<a>b;**

**Каким образом, компилятор будет воспринимать вызов функции-операции?**

Компилятор выдаст ошибку, так как операция вывода не переопределена (у оператора << приоритет больше чем у >).

1. **В программе описан класс  
   class Student  
   {  
   …  
   friend bool operator >(const Person&, Person&)  
   …  
   };  
   и определены объекты этого класса  
   Student a, b;**

**Выполняется операция  
cout<< a>b;  
Каким образом компилятор будет воспринимать вызов функции-операции?**Никак, класс Person не определен.