МИНИСТЕРСТВО ТРАНСПОРТА РФ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ «РОССИЙСКИЙ УНИВЕРСИТЕТ ТРАНСПОРТА (МИИТ)»

Кафедра АСУ

**Отчет**

По лабораторной работе

по дисциплине   
«Языки программирования высокого уровня»

Тема: «КОНСТРУКТОРЫ И ПЕРЕГРУЗКА ОПЕРАЦИЙ»

Задание № 4

Вариант № 4

Выполнил:   
 ст. гр. УВА-111:

Букин А. А.

Проверил:

Савушкин С.А.

Москва 2020 г.

**Содержание:**

[**Постановка задачи 3**](#_heading=h.30j0zll)

[**Общие требования 3**](#_heading=h.1fob9te)

[**Задание 3**](#_heading=h.3znysh7)

[**Авторское понимание и формализация задачи**](#_heading=h.2et92p0) **3**

[**Текст программы**](#_heading=h.tyjcwt) **4**

[**Результаты тестирования**](#_heading=h.3dy6vkm) **10**

[**Вывод**](#_heading=h.1t3h5sf) **10**

**Постановка задачи**

**Общие требования**

В каждом упражнении требуется реализовать в том или ином виде определение нового класса. Во всех заданиях необходимо реализовать конструктор инициализации (один или несколько) и конструктор без аргументов.

Для демонстрации работы с объектами нового типа во всех заданиях требуется написать главную функцию. В программе обязательно должны быть продемонстрированы различные способы создания объектов и массивов объектов. Программа должна демонстрировать использование всех функций и методов.

**Задание**

Создать класс Money для работы с денежными суммами. Число должно быть представлено двумя полями: типа long для рублей и типа unsigned char - для копеек. Дробная часть (копейки) при выводе на экран должна быть отделена от целой части запятой. Реализовать сложение, вычитание, деление сумм, деление суммы на дробное число, умножение на дробное число и операция сравнения.

**Авторское понимание и формализация задачи**

Мы создаем класс Money, для работы с ним. Вводим определенную сумму, для расчета количества купюр различного достоинства. Так же реализуем сложение, вычитание, операцию сравнения. Деление сумм и деление и умножение суммы на дробное число выполняем как перегруженные функции. И получившийся результат выводим на экран.

**Текст программы**

**Файл – lab4.cpp**

**#include <string>**

**#include <iostream>**

**#include <cmath>**

**using namespace std;**

**class Money {**

**private:**

**long rub; // рубли**

**unsigned char cop; // копейки**

**public:**

**Money(long r, unsigned char c) : rub(r), cop(c) { // конструктор инициализации**

**if (c > 99) {**

**cout << "Ошибочные значения, завершение программы..." << endl;**

**exit(0);**

**}**

**};**

**Money(double n); // конструктор инициализации**

**Money(const Money &m); // конструктор копирования**

**Money(); // конструктор без аргументов**

**string toString();**

**double toDouble();**

**void display();**

**void add(Money a);**

**void substract(Money s);**

**static int compare(Money mon1, Money mon2);**

**static double divide(Money mon1, Money mon2); // перегрузка функции деления**

**void divide(double n);**

**void mult\_num(double n);**

**};**

**Money::Money(double n) {**

**long lrub = (long)n;**

**double cop\_d = n - (double)((long)n);**

**int icop = (unsigned char)abs(round((cop\_d) \* 100.0));**

**if (n < 0.0)**

**{**

**lrub--;**

**icop = 100 - icop;**

**}**

**if (icop > 99) {**

**cout << "Ошибочные значения, завершение программы..." << endl;**

**exit(0);**

**}**

**rub = lrub;**

**cop = icop;**

**}**

**Money::Money(const Money &m) {**

**rub = m.rub;**

**cop = m.cop;**

**}**

**Money::Money() {**

**double sum;**

**cout << "Введите сумму: ";**

**cin >> sum;**

**Money tmp = Money(sum);**

**rub = tmp.rub;**

**cop = tmp.cop;**

**}**

**string Money::toString() {**

**char buf[100];**

**int res = snprintf(buf, sizeof(buf), "%.2f", toDouble());**

**string str = "0.00";**

**if (res >= 0 && res < sizeof(buf))**

**str = buf;**

**size\_t pos = str.find('.');**

**str.replace(pos, 1, ",");**

**return (str);**

**}**

**double Money::toDouble() {**

**int icop = cop;**

**if (rub < 0) {**

**icop -= 100;**

**rub++;**

**}**

**return ((double) rub + (double) icop / 100.0);**

**}**

**void Money::display() {**

**cout << "Сумма: " << toString() << endl;**

**}**

**void Money::add(Money a) {**

**Money tmp = Money(toDouble() + a.toDouble());**

**rub = tmp.rub;**

**cop = tmp.cop;**

**}**

**void Money::substract(Money s) {**

**Money tmp = Money(toDouble() - s.toDouble());**

**rub = tmp.rub;**

**cop = tmp.cop;**

**}**

**int Money::compare(Money mon1, Money mon2) {**

**double m1 = mon1.toDouble();**

**double m2 = mon2.toDouble();**

**if (m1 == m2)**

**return (0);**

**return (m1 > m2 ? 1 : -1);**

**}**

**void Money::mult\_num(double n) {**

**Money tmp = Money(toDouble() \* n);**

**rub = tmp.rub;**

**cop = tmp.cop;**

**}**

**double Money::divide(Money mon1, Money mon2) {**

**double m1 = round(mon1.toDouble() \* 100);**

**double m2 = round(mon2.toDouble() \* 100);**

**if (round(m2) == 0.0)**

**return (0.0);**

**return (m1 / m2);**

**}**

**void Money::divide(double n) {**

**if (n == 0.0) {**

**rub = 0;**

**cop = 0;**

**return ;**

**}**

**Money tmp = Money(toDouble() / n);**

**rub = tmp.rub;**

**cop = tmp.cop;**

**}**

**int main()**

**{**

**cout << "┍━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━┑" << endl;**

**cout << "| Демонстрация работы перегрузок |" << endl;**

**cout << "┕━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━┙" << endl << endl;**

**Money mon1 = Money(2, 2);**

**Money mon2 = Money(3, 55);**

**cout << "mon1: "; mon1.display();**

**cout << "mon2: "; mon2.display();**

**cout << endl;**

**Money sum = mon1;**

**sum.add(mon2);**

**cout << "sum: " + sum.toString() << endl;**

**Money raz = mon1;**

**raz.substract(mon2);**

**cout << "raz: " + raz.toString() << endl;**

**cout << "cmp: " + to\_string(Money::compare(mon1, mon2)) << endl;**

**cout << "div: " + to\_string(Money::divide(mon1, mon2)) << endl; //использование перегруженной функции**

**cout << endl;**

**double num = 1.5;**

**Money input = Money(); // вводится из терминала**

**cout << "number: " + to\_string(num) << endl;**

**cout << "money : " + input.toString() << endl;**

**cout << endl;**

**Money mult = input;**

**mult.mult\_num(num);**

**cout << "multiplication : " + mult.toString() << endl;**

**Money div = input;**

**div.divide(num); //использование перегруженной функции**

**cout << "division : " + div.toString() << endl << endl;**

**cout << "┍━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━┑" << endl;**

**cout << "| Демонстрация работы с классами |" << endl;**

**cout << "┕━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━┙" << endl << endl;**

**Money \*money\_heap = new Money(1, 99);**

**cout << "money\_heap: "; money\_heap->display();**

**int money\_num = 8;**

**Money \*array[money\_num];**

**for (int i = 0; i < money\_num; i++) {**

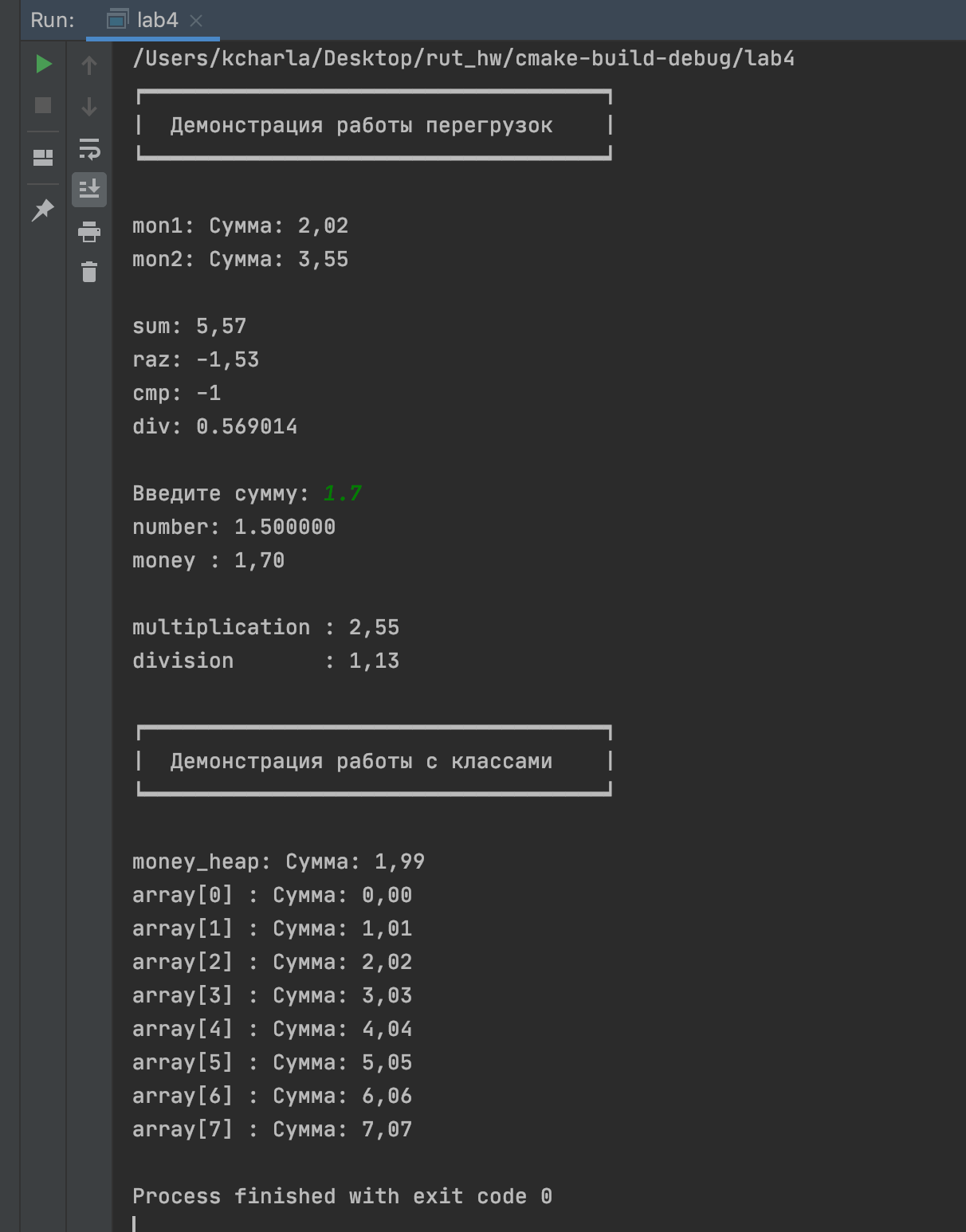
**array[i] = new Money(i, i);**

**cout << "array[" << i << "] : "; array[i]->display();**

**}**

**}**

**Результаты тестирования**

****

**Вывод**

Программа выдает ожидаемые результаты и работает корректно. Поскольку полученные результаты совпали с ожидаемыми, программу можно считать правильной. Я научился научится перегружать операции в классе.