**1.**

//Пищик Е.С. М8О-206Б-19 Лабораторная работа №2 вариант 9

/\*Создать класс BritishMoney для работы с денежными суммами в старой британской система.

Сумма денег должна быть представлено тремя полями: типа unsigned long long для фунтов стерлингов,

типа unsigned long long – для шиллингов, unsigned long long – для пенсов (пенни).

Реализовать сложение сумм, вычитание, деление сумм, деление суммы на дробное число,

умножение на дробное число и операции сравнения. 1 фунт = 20 шиллингов, 1 шиллинг = 12 пенни.

Операции сложения, вычитания, умножения, деления,

сравнения (на равенство, больше и меньше) должны быть выполнены в виде перегрузки операторов.

Необходимо реализовать пользовательский литерал для работы с константами типа BritishMoney.\*/

//В тест файл вводится 9 чисел, где каждые 3 след числа - стерлинги, шиллинги и пенни для экземпляра класса

#include <iostream>

#include <fstream>

#include <string>

typedef unsigned long long ull;

class BritishMoney

{

    //3 приватных поля, ster\_ - стерлинги, shil\_ - шиллинги, penni\_ - пенни

    ull ster\_;

    ull shil\_;

    ull penni\_;

    public:

    // Конструктор класса

    explicit BritishMoney(ull const& ster = 0, ull const& shil = 0, ull const& penni = 0) :

    ster\_(ster), shil\_(shil), penni\_(penni) {}

    //Геттеры

    ull get\_ster() const { return ster\_; }

    ull get\_shil() const { return shil\_; }

    ull get\_penni() const { return penni\_;}

};

//Пользовательский литерал - ster, ввод количества стрелингов, shil - шиллинги, penni - пенни

BritishMoney operator""\_ster(ull obj)

{

    return BritishMoney(obj, 0, 0);

}

BritishMoney operator""\_shil(ull obj)

{

    return BritishMoney(0, obj, 0);

}

BritishMoney operator""\_penni(ull obj)

{

    return BritishMoney(0, 0, obj);

}

//оператор сложения денег

BritishMoney operator+(BritishMoney & lhs, BritishMoney & rhs)

{

    ull dop\_shil = lhs.get\_penni() + rhs.get\_penni() >= 12 ? 1 : 0;

    ull res\_penni = lhs.get\_penni() + rhs.get\_penni() >= 12 ?

    lhs.get\_penni() + rhs.get\_penni() - 12 : lhs.get\_penni() + rhs.get\_penni();

    ull dop\_ster = lhs.get\_shil() + rhs.get\_shil() + dop\_shil  >= 20 ? 1 : 0;

    ull res\_shil = lhs.get\_shil() + rhs.get\_shil() + dop\_shil >= 20 ?

    lhs.get\_shil() + rhs.get\_shil() + dop\_shil - 20: lhs.get\_shil() + rhs.get\_shil() + dop\_shil;

    ull res\_ster = lhs.get\_ster() + rhs.get\_ster() + dop\_ster;

    return BritishMoney(res\_ster, res\_shil, res\_penni);

}

//оператор вычитания

BritishMoney operator-(BritishMoney & lhs, BritishMoney & rhs)

{

    ull res\_lhs = lhs.get\_penni() + 12 \* lhs.get\_shil() + 240 \* lhs.get\_ster();

    ull res\_rhs = rhs.get\_penni() + 12 \* rhs.get\_shil() + 240 \* rhs.get\_ster();

    if(res\_lhs <= res\_rhs) {return BritishMoney(0, 0, 0);}

    ull res = res\_lhs - res\_rhs;

    ull res\_ster = res / 240;

    ull res\_shild = (res - 240 \* res\_ster) / 12;

    ull res\_penni = res - 240 \* res\_ster - 12 \* res\_shild;

    return BritishMoney(res\_ster, res\_shild, res\_penni);

}

//операторы деления

double operator/(BritishMoney & lhs, BritishMoney & rhs)

{

    double res\_lhs = lhs.get\_penni() + 12 \* lhs.get\_shil() + 240 \* lhs.get\_ster();

    double res\_rhs = rhs.get\_penni() + 12 \* rhs.get\_shil() + 240 \* rhs.get\_ster();

    return res\_lhs / res\_rhs;

}

BritishMoney operator/(BritishMoney & lhs, double const& rhs)

{

    ull res = (lhs.get\_penni() + 12 \* lhs.get\_shil() + 240 \* lhs.get\_ster()) / rhs;

    ull res\_ster = res / 240;

    ull res\_shild = (res - 240 \* res\_ster) / 12;

    ull res\_penni = res - 240 \* res\_ster - 12 \* res\_shild;

    return BritishMoney(res\_ster, res\_shild, res\_penni);

}

//операторы сравнения

bool operator<(BritishMoney & lhs, BritishMoney & rhs)

{

    if(lhs.get\_ster() == rhs.get\_ster())

    {

        if(lhs.get\_shil() == rhs.get\_shil()) { return lhs.get\_shil() < rhs.get\_shil(); }

        else { return lhs.get\_shil() < rhs.get\_shil(); }

    }

    else {return lhs.get\_ster() < rhs.get\_ster(); }

}

bool operator>(BritishMoney & lhs, BritishMoney & rhs) { return rhs < lhs; }

bool operator<=(BritishMoney & lhs, BritishMoney & rhs) { return !(rhs < lhs); }

bool operator>=(BritishMoney & lhs, BritishMoney & rhs) { return !(lhs < rhs); }

bool operator==(BritishMoney & lhs, BritishMoney & rhs) { return !(rhs < lhs) && !(lhs < rhs); }

bool operator!=(BritishMoney & lhs, BritishMoney & rhs) { return !(lhs == rhs); }

//Тестирующая функция, в файл с тестом должно быть записано 6 числе типа double

void TestFunction(ull p\_11 = 0, ull p\_12 = 0, ull p\_13 = 0, ull p\_21 = 0,

ull p\_22 = 0, ull p\_23 = 0, ull p\_31 = 0, ull p\_32 = 0, ull p\_33 = 0)

{

    //Создание денег

    BritishMoney s1(p\_11, p\_12, p\_13);

    BritishMoney s2(p\_21, p\_22, p\_23);

    BritishMoney s3(p\_31, p\_32, p\_33);

    //Использование пользовательских литералов для создания объектов типа BritishMoney

    BritishMoney s4 = 135\_ster;

    BritishMoney s5 = 13\_shil;

    BritishMoney s6 = 7\_penni;

    //Создание

    BritishMoney s;

    BritishMoney t;

    BritishMoney r;

    BritishMoney f;

    double div = 0;

    //Сложение, вычитание деление

    s = s1 + s2;

    t = s2 - s1;

    r = s1 - s2;

    div = s1 / s2;

    f = s1 / 2.0;

    std::cout <<"First: ster = " << s1.get\_ster() << " shil = " << s1.get\_shil();

    std::cout << " penni = " << s1.get\_penni() << std::endl;

    std::cout <<"Second: ster = " << s2.get\_ster() << " shil = " << s2.get\_shil();

    std::cout << " penni = " << s2.get\_penni() << std::endl;

    std::cout <<"Third: ster = " << s3.get\_ster() << " shil = " << s3.get\_shil();

    std::cout << " penni = " << s3.get\_penni() << std::endl;

    std::cout << "Fourth created by literal \_ster: ster = " << s4.get\_ster() << " shil = ";

    std::cout << s4.get\_shil() << " penni = " << s4.get\_penni() << std::endl;

    std::cout << "Fifth created by literal \_shil: ster = " << s5.get\_ster() << " shil = ";

    std::cout << s5.get\_shil() << " penni = " << s5.get\_penni() << std::endl;

    std::cout << "Sixth created by literal \_penni: ster = " << s6.get\_ster() << " shil = ";

    std::cout << s6.get\_shil() << " penni = " << s6.get\_penni() << std::endl;

    std::cout << "Sum first and second : ster = " << s.get\_ster() << " shil = " << s.get\_shil();

    std::cout << " penni = " << s.get\_penni() << std::endl;

    std::cout << "Sub second and first : ster = " << t.get\_ster() << " shild = ";

    std::cout << t.get\_shil() << " penni = " << t.get\_penni() << std::endl;

    std::cout << "Sub first and second : ster = " << r.get\_ster() << " shild = ";

    std::cout << r.get\_shil() << " penni = " << r.get\_penni() << std::endl;

    std::cout << "First div second = " << div << std::endl;

    std::cout << "First div double = 2.0 : ster = " << f.get\_ster() << " shild = ";

    std::cout << f.get\_shil() << " penni = " << f.get\_penni() << std::endl;

    //Сравнение

    if (s1 < s2) std::cout << "first < second" << std::endl;

    else std::cout << "first > second" << std::endl;

    if (s1 == s3) std::cout << "first = third" << std::endl;

    else std::cout << "first != third" << std::endl;

    if (s1 == s2) std::cout << "first = second" << std::endl;

    else std::cout << "firste != second" << std::endl;

    std::cout << "Completed\n" << std::endl;

}

int main()

{

    std::string name = "start";

    std::cout << "Take 9 numbers from test\_file - 3 pairs(sterlings, shillings, pennies)" << std::endl;

    std::cout << "Enter q or Q or quit or Quit for close programm" << std::endl;

    //Бесконечный дикл до ввода ключевого слова для выхода

    while(true)

    {

        ull a = 0, d = 0, g = 0;

        ull b = 0, c = 0, e = 0, f = 0, h = 0, k = 0;

        std::cout << "Enter test file name: ";

        std::cin >> name;

        if (name == "Q" || name == "q" || name == "Quit" || name == "quit") break;

        //поток ввода из файла, имя файла хранится в string переменной name

        std::ifstream finp(name.c\_str());

        //взятие из файла 9 чисел и запись в переменные

        finp >> a >> b >> c >> d >> e >> f >> g >> h >> k;

        TestFunction(a, b, c, d, e, f, g, h, k);

        finp.close();

    }

    return 0;

}

**2.** <https://github.com/Pe4enIks/OOP_Pishchik_206>

**3.**

Вводится 9 чисел по 3 тройки (каждая тройка объект BritishMoney)

Тесты:

12 19 7 5 13 11 0 5 8

27 1 11 27 2 8 13 18 4

1 3 0 7 17 4 0 1 0

12 0 0 24 0 0 1 1 1

**4.**

Результат:

Take 9 numbers from test\_file - 3 pairs(sterlings, shillings, pennies)

Enter q or Q or quit or Quit for close programm

Enter test file name: test\_1.txt

First: ster = 12 shil = 19 penni = 7

Second: ster = 5 shil = 13 penni = 11

Third: ster = 0 shil = 5 penni = 8

Fourth created by literal \_ster: ster = 135 shil = 0 penni = 0

Fifth created by literal \_shil: ster = 0 shil = 13 penni = 0

Sixth created by literal \_penni: ster = 0 shil = 0 penni = 7

Sum first and second : ster = 18 shil = 13 penni = 6

Sub second and first : ster = 0 shild = 0 penni = 0

Sub first and second : ster = 7 shild = 5 penni = 8

First div second = 2.27871

First div double = 2.0 : ster = 6 shild = 9 penni = 9

First mul double = 3.0 : ster = 19 shild = 9 penni = 3

first > second

first != third

firste != second

Completed

Enter test file name: test\_2.txt

First: ster = 27 shil = 1 penni = 11

Second: ster = 27 shil = 2 penni = 8

Third: ster = 13 shil = 18 penni = 4

Fourth created by literal \_ster: ster = 135 shil = 0 penni = 0

Fifth created by literal \_shil: ster = 0 shil = 13 penni = 0

Sixth created by literal \_penni: ster = 0 shil = 0 penni = 7

Sum first and second : ster = 54 shil = 4 penni = 7

Sub second and first : ster = 0 shild = 0 penni = 9

Sub first and second : ster = 0 shild = 0 penni = 0

First div second = 0.998618

First div double = 2.0 : ster = 13 shild = 10 penni = 11

First mul double = 3.0 : ster = 40 shild = 12 penni = 9

first < second

first != third

firste != second

Completed

Enter test file name: test\_3.txt

First: ster = 1 shil = 3 penni = 0

Second: ster = 7 shil = 17 penni = 4

Third: ster = 0 shil = 1 penni = 0

Fourth created by literal \_ster: ster = 135 shil = 0 penni = 0

Fifth created by literal \_shil: ster = 0 shil = 13 penni = 0

Sixth created by literal \_penni: ster = 0 shil = 0 penni = 7

Sum first and second : ster = 9 shil = 0 penni = 4

Sub second and first : ster = 6 shild = 14 penni = 4

Sub first and second : ster = 0 shild = 0 penni = 0

First div second = 0.146186

First div double = 2.0 : ster = 0 shild = 11 penni = 6

First mul double = 3.0 : ster = 1 shild = 14 penni = 6

first < second

first != third

firste != second

Completed

Enter test file name: test\_4.txt

First: ster = 12 shil = 0 penni = 0

Second: ster = 24 shil = 0 penni = 0

Third: ster = 1 shil = 1 penni = 1

Fourth created by literal \_ster: ster = 135 shil = 0 penni = 0

Fifth created by literal \_shil: ster = 0 shil = 13 penni = 0

Sixth created by literal \_penni: ster = 0 shil = 0 penni = 7

Sum first and second : ster = 36 shil = 0 penni = 0

Sub second and first : ster = 12 shild = 0 penni = 0

Sub first and second : ster = 0 shild = 0 penni = 0

First div second = 0.5

First div double = 2.0 : ster = 6 shild = 0 penni = 0

First mul double = 3.0 : ster = 18 shild = 0 penni = 0

first < second

first != third

firste != second

Completed

**5.** В программе создается 3 объекта типа BritishMoney и инициализируются значениями из файла. На экран выводятся поля каждого объекта BritishMoney, также суммы, разности 1 и 2 объекта BritishMoney, также результат умножнения, и деления объектов BritishMoney между собой и с числом double, и при помощи операторов сравнений на экран выводится отношения между объектами класса Bottle.