**Московский авиационный институт**

**(Национальный исследовательский университет)**

Институт: «Информационные технологии и прикладная математика»

Кафедра: 806 «Вычислительная математика и программирование»

Дисциплина: «Объектно-ориентированное программирование»

**Лабораторная работа № 1**

Тема: Простые классы на языке С++

Студент: Шараковский Юрий Дмитриевич

Группа: 80-206

Преподаватель: Чернышов Л.Н.

Дата:

Оценка:

Москва, 2020

1. Постановка задачи

Вариант 9: Создать класс BritishMoney для работы с денежными суммами в старой британской системе. Сумма денег должна быть представлена тремя полями: типа unsigned long long для фунтов стерлингов, типа unsigned char - для шиллингов, unsigned char – для пенсов (пенни). Реализовать сложение сумм, вычитание, деление сумм, деление суммы на дробное число, умножение на дробное число и операции сравнения.

1. Описание программы

Программа описывает класс BritishMoney соответствующий условию задачи и определяет методы/функции для работы с ним. В функции main находится интерактивный цикл программы - отвечающий на ввод пользователя операциями с объектами класса BritishMoney.

1. Набор тестов

testcase01:

add 1 2 3 1 2 3

sub 1 2 3 1 2 3

sub 1 2 5 2 2 2

div 10 10 0 0 10 0

muln 0 1 2 3.5

divn 1 0 3 1.67

cmp 1 2 3 1 2 3

cmp 1 2 4 5 0 0

cmp 0 0 2 0 0 0

exit

testcase02:

add 0 0 0 0 0 0

add 999 999 999 1 2 3

add 999 999 999 999 999 99

sub 123 123 123 00 12 34

sub 0 0 0 0 0 0

sub 1 0 0 0 1 9

exit

1. Результаты выполнения тестов

$ ./oop\_exercise\_01 < testcase01

Command example:

> add 1 2 3 1 2 3

result: 2 4 6

Available commands: "add", "sub", "div", "divn", "muln", "cmp"

Enter "quit" or "exit" to exit

> result: (2,4,6)

> result: (0,0,0)

> can't substract greater number from lesser number!

> result: (0,1,9)

> result: (0,4,1)

> result: (0,12,1)

> (1,2,3) is equal to (1,2,3)

> (1,2,4) is less than (5,0,0)

> (0,0,2) is greater than (0,0,0)

>

На каждый запрос программа выводит ответ, проверим некоторые ответы:

1. Сложить A = 1 Фунт 2 Шиллинга 3 Пенни два раза - очевидно что получим 2 Фунта 4 Шиллинга и 6 Пенни.
2. Вычесть из суммы её же - получим 0.
3. Вычесть из суммы большее число - получаем предупреждение.
4. Разделить A = 10 Фунтов 10 Шиллингов на B = 10 Шиллингов - переведем все в наименьшую валюту (пенни), получим A = 10\*20\*12 + 10\*12 = 2520 и B = 120. Теперь разделим A на B и переведем обратно - A/B = 21 = 1 Шиллинг и 9 Пенни.
5. Умножить A = (0,1,2) на дробное число 3.5 - переведем все в пенни и умножим на число A = 14 \* 3.5 = 49 = 4 Шиллинга и 1 Пенни.

$ ./oop\_exercise\_01 < testcase02

Command example:

> add 1 2 3 1 2 3

result: 2 4 6

Available commands: "add", "sub", "div", "divn", "muln", "cmp"

Enter "quit" or "exit" to exit

> result: (0,0,0)

> result: (1054,4,6)

> result: (2102,9,6)

> result: (128,18,5)

> result: (0,0,0)

> result: (0,18,3)

>

1. Листинг программы

*// Шараковский Юрий Дмитриевич. М8О-206Б-19, МАИ. 09.2020.*

*// Лабораторная работа №1. Условие:*

*//*

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

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

*// типа unsigned char – для шиллингов, unsigned char – 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.*

#include <iostream>

#include <ostream>

#include <istream>

#include <string>

#include <cstring>

#define STATUS\_QUIT 1

#define STATUS\_PROCEED 0

class BritishMoney {

private:

unsigned long long pounds;

unsigned char shillings;

unsigned char pennies;

unsigned long long total;

public:

enum result {

less, equal, greater,

};

BritishMoney(unsigned long long pounds = 0, unsigned long long shillings = 0, unsigned long long pennies = 0);

unsigned long long getPounds() const;

unsigned char getShillings() const;

unsigned char getPennies() const;

void setSum(unsigned long long pounds, unsigned long long shillings, unsigned long long pennies);

unsigned long long getTotal() const;

void fromTotal();

static BritishMoney Add(const BritishMoney& A, const BritishMoney& B);

static BritishMoney Sub(const BritishMoney& A, const BritishMoney& B);

static BritishMoney Div(const BritishMoney& A, const BritishMoney& B);

static BritishMoney Divn(const BritishMoney& A, const double B);

static BritishMoney Muln(const BritishMoney& A, const double B);

static result Cmp(const BritishMoney& A, const BritishMoney& B);

};

*// Метод возвращает число пенни в сумме*

unsigned long long BritishMoney::getTotal() const {

unsigned long long total = (unsigned long long)(this->pennies);

total += (unsigned long long)(this->shillings) \* 12;

total += this->pounds \* 12 \* 20;

return total;

}

*// Метод преобразует число пенни в фунты и шиллинги*

void BritishMoney::fromTotal() {

this->pennies = this->total % 12;

this->shillings = this->total / 12 % 20;

this->pounds = this->total / 12 / 20;

}

*// Конструкор*

BritishMoney::BritishMoney(unsigned long long pounds, unsigned long long shillings, unsigned long long pennies) {

this->setSum(pounds, shillings, pennies);

}

BritishMoney BritishMoney::Add(const BritishMoney& A, const BritishMoney& B) {

BritishMoney C = BritishMoney();

C.total = A.getTotal() + B.getTotal();

C.fromTotal();

return C;

}

BritishMoney BritishMoney::Sub(const BritishMoney& A, const BritishMoney& B) {

BritishMoney C = BritishMoney();

if (Cmp(A, B) == less) {

std::cout << "can't substract greater number from lesser number!" << std::endl;

C.pennies = -1;

} else {

C.total = A.getTotal() - B.getTotal();

C.fromTotal();

}

return C;

}

BritishMoney BritishMoney::Div(const BritishMoney& A, const BritishMoney& B) {

BritishMoney C = BritishMoney();

unsigned long long b = B.getTotal();

if (b) {

C.total = A.getTotal() / b;

C.fromTotal();

}

else {

std::cout << "Can't divide by zero!" << std::endl;

}

return C;

}

BritishMoney BritishMoney::Divn(const BritishMoney& A, const double B) {

BritishMoney C = BritishMoney();

if (B) {

C.total = A.getTotal() / B;

C.fromTotal();

} else {

std::cout << "Can't divide by zero!" << std::endl;

}

return C;

}

BritishMoney BritishMoney::Muln(const BritishMoney& A, const double B) {

BritishMoney C = BritishMoney();

if (B) {

C.total = A.getTotal() \* B;

C.fromTotal();

}

return C;

}

*// Метод сравнения (без использования поля total)*

BritishMoney::result BritishMoney::Cmp(const BritishMoney& A, const BritishMoney& B) {

if (A.pounds > B.pounds)

return greater;

else if (A.pounds == B.pounds)

if (A.shillings > B.shillings)

return greater;

else if (A.shillings == B.shillings)

if (A.pennies > B.pennies)

return greater;

else if (A.pennies == B.pennies)

return equal;

else

return less;

else

return less;

else

return less;

}

unsigned long long BritishMoney::getPounds() const {

return this->pounds;

}

unsigned char BritishMoney::getShillings() const {

return this->shillings;

}

unsigned char BritishMoney::getPennies() const {

return this->pennies;

}

void BritishMoney::setSum(unsigned long long pounds, unsigned long long shillings, unsigned long long pennies) {

this->total = pennies;

this->total += shillings \* 12;

this->total += pounds \* 12 \* 20;

this->fromTotal();

}

std::ostream& operator<<(std::ostream& os, const BritishMoney& bm) {

os << "(" << bm.getPounds() << "," << (int)bm.getShillings() << "," << (int)bm.getPennies() << ")";

return os;

}

std::istream& operator>>(std::istream& is, BritishMoney& bm) {

unsigned long long p, a, b;

is >> p >> a >> b;

bm.setSum(p, a, b);

return is;

}

void printInfo() {

std::cout << "Command example: \n\t\t> add\t1 2 3\t1 2 3\n\t\tresult: 2 4 6\n"

<< "Available commands: \"add\", \"sub\", \"div\", \"divn\", \"muln\", \"cmp\""

<< std::endl;

std::cout << "Enter \"quit\" or \"exit\" to exit" << std::endl;

}

int main(int argc, char \*argv[]) {

int status = STATUS\_PROCEED;

BritishMoney sumA = BritishMoney();

BritishMoney sumB = BritishMoney();

BritishMoney result = BritishMoney();

double number;

std::string str;

int errnum = 0;

printInfo();

while (status != STATUS\_QUIT) {

std::cout << "> ";

std::cin >> str;

if (str == "add") {

std::cin >> sumA >> sumB;

result = BritishMoney::Add(sumA, sumB);

std::cout << "result: " << result << std::endl;

}

else if (str == "sub") {

std::cin >> sumA >> sumB;

result = BritishMoney::Sub(sumA, sumB);

if (result.getPennies() != (unsigned char)(-1))

std::cout << "result: "<< result << std::endl;

}

else if (str == "muln") {

std::cin >> sumA >> number;

result = BritishMoney::Muln(sumA, number);

std::cout << "result: "<< result << std::endl;

}

else if (str == "divn") {

std::cin >> sumA >> number;

result = BritishMoney::Divn(sumA, number);

std::cout << "result: "<< result << std::endl;

}

else if (str == "div") {

std::cin >> sumA >> sumB;

result = BritishMoney::Div(sumA, sumB);

std::cout << "result: "<< result << std::endl;

}

else if (str == "cmp") {

std::cin >> sumA >> sumB;

BritishMoney::result result = BritishMoney::Cmp(sumA, sumB);

if (result == BritishMoney::result::less)

std::cout << sumA << " is less than " << sumB;

else if (result == BritishMoney::result::equal)

std::cout << sumA << " is equal to " << sumB;

else

std::cout << sumA << " is greater than " << sumB;

std::cout << std::endl;

}

else if (str == "exit" || str == "quit") {

status = STATUS\_QUIT;

}

else {

std::cout << "input error" << std::endl;

errnum++;

if (errnum >= 10)

status = STATUS\_QUIT;

}

}

return EXIT\_SUCCESS;

}

1. Выводы

В языке C++ можно описать реальные объекты с помощью классов и писать программы так, чтобы код и данные были связаны вместе - это упрощает процесс программирования, но вместе с тем и накладывает новые обязанности на программиста.

1. Список литературы

C++ Primer (5th Edition) Stanley B. Lippman, Josee Lajoie, E. Moo.