МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ
МОСКОВСКИЙ АВИАЦИОННЫЙ ИНСТИТУТ
(НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСТИТЕТ)

ЛАБОРАТОРНАЯ РАБОТА №1 по курсу

объектно-ориентированное программирование I семестр, 2021/22 уч. год

Студент *Клитная Анастасия Викторовна, группа М80-208Б-20*
Преподаватель *Дорохов Евгений Павлович*

Цель:

- Изучение системы сборки на языке C++, изучение систем контроля версии.
- Изучение основ работы с классами в C++;

Порядок выполнения работы

1. Ознакомиться с теоретическим материалом.
2. Получить у преподавателя вариант задания.
3. Реализовать задание своего варианта в соответствии с поставленными требованиями.
4. Подготовить тестовые наборы данных.
5. Создать репозиторий на GitHub.
6. Отправить файлы лабораторной работы в репозиторий.
7. Отчитаться по выполненной работе путём демонстрации работающей программы на тестовых наборах данных (как подготовленных самостоятельно, так и предложенных преподавателем) и ответов на вопросы преподавателя (как из числа контрольных, так и по реализации программы).

Требования к программе

Разработать программу на языке C++ согласно варианту задания. Программа на C++ должна собираться с помощью системы сборки CMake. Программа должна получать данные из стандартного ввода и выводить данные в стандартный вывод.

Необходимо настроить сборку лабораторной работы с помощью CMake. Собранная программа должна называться **oop_exercise_01** (в случае использования Windows **oop_exercise_01.exe)**

Необходимо зарегистрироваться на GitHub (если студент уже имеет регистрацию на GitHubто можно использовтаь ее) и создать репозитарий для задания лабораторной работы.

Описание программы

Исходный код лежит в 3 файлах:

1. main.cpp - исполняемый код.
2. Money.h - специальный файл .h, содержащий прототипы используемых мною функций.
3. Money.cpp - реализация функций для моего задания.
4. CMakeLists.txt - специальный дополнительный файл типа CMakeLists.

Дневник отладки

Во время выполнения лабораторной работы программа не нуждалась в отладке, все ошибки компиляции были исправлены с первой попытки. После их исправления программа работала так, как было задумано изначально.

Недочёты

Недочётов не было обнаружено.

Выводы

Данная лабораторная работа помогла мне использовать полученные на лекциях теоретические знания на практике, и я написала простенький полностью работающий класс.

Исходный код

Money.cpp

```
#include <iostream>
#include "Modulo.h"
#include <cmath>

Money::Money(){
    rub = 1;
    cop = 1;
}

Money::Money(std::istream &is){
    is >> rub;
    is >> cop;
}

Money::Money(unsigned long long first, unsigned long long second){
    rub = first;
    cop = second;
}

Money Money::operator +(Money& a){
    this->rub = this->rub + a.rub;
    this->cop += a.cop;
    return *this;//->rub%this->cop + a.rub%a.cop;
}

Money Money::operator -(Money& a){
    this->rub = this->rub - a.rub;
    this->cop -= a.cop;
    return *this;//->rub%this->cop - a.rub%a.cop;
}
```

```
Money Money::operator *(Money& a){
    this->rub = this->rub * a.rub;
    this->cop *= a.cop;
    return *this;//(this->rub%this->cop) * (a.rub%a.cop);
}

Money Money::operator /(Money& a){
    this->rub = this->rub / a.rub;
    this->cop /= a.cop;
    return *this;//(this->rub%this->cop) / (a.rub%a.cop);
}

Money Money::operator ++(){
    this->cop++;
    this->rub++;
    return *this;
}

Money Money::operator --(){
    this->cop--;
    this->rub--;
    return *this;
}

std::ostream& operator<<(std::ostream& os,const Money& a){
    os << a.rub << "," << a.cop << std::endl;
    return os;
}

bool Money::operator==(const Money& other){
    return this->cop == other.cop && this->rub == other.rub;
}

Money::~Money(){
    std::cout << "Money has deleted" << std::endl;
}

Money.h
#ifndef MONEY_H
#define MONEY_H
#include <iostream>

class Money {
public:
    Money();
    Money(std::istream &is);
    Money(unsigned long long rub, unsigned long long cop);
    Money operator +(Money& a);
    Money operator -(Money& a);
```

```
    Money operator *(Money& a);
    Money operator /(Money& a);
    Money operator ++();
    Money operator --();
    bool operator ==(const Money& other);
    friend std::ostream& operator<<(std::ostream& os,const Money& a);
    ~Money();
private:
    unsigned long long rub;
    unsigned long long cop;
};

#endif // MONEY_H
```

Main.cpp

```
#include <iostream>
#include "Money.h"

int main(){
    Money c(std::cin);
    Money a(10, 6);
    Money b(12, 5);
    std::cout << "Money objects: "<< a << b << c << std::endl;
    std::cout << "Sum: " << a+b << std::endl;
    std::cout << "Division of residues " << a/b << std::endl;
    std::cout << "Multiplication of residuals " << a*b << std::endl;
    std::cout << "Sum " << c+b << std::endl;
    std::cout << "Operator -- : " << --a;
    std::cout << "Operator ++ : " << ++a;
}
```