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

ЛАБОРАТОРНАЯ РАБОТА №2

по курсу “Объектно-ориентированное программирование”

I семестр, 2021/22 учебный год

Студент: *Мерц Савелий Павлович, группа М8О-207Б-20*

Преподаватель: *Дорохов Евгений Павлович, каф. 806*

Задание:

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. Реализовать пользовательский литерал.

Вариант №16:

Создать класс Position для работы с географическими координатами. Координаты задаются двумя числами широта и долгота. Долгота находится в диапазоне от -180 до 180 градусов. Широта находится в диапазоне от -90 до 90 градусов. Реализовать арифметические операции сложения, вычитания, умножения и деления, а также операции сравнения.

Описание программы:

Исходный код разделен на 2 файла:

- CMakeLists.txt - файл с конфигурацией CMake
- main.cpp – основная программа

Дневник отладки:

Проблем не было.

Вывод:

При выполнении работы я на практике познакомился с пользовательскими литералами. Теперь точно можно будет найти пиратский клад, координаты которого записаны с указанием полушариев.

Исходный код:

CMakeLists.txt:

```
add_executable(oop_exercise_02 main.cpp
```

main.cpp:

```
#include <iostream>

class Position
{
private:
    int latitude; //широта(Y)
    int longitude; //долгота(X)
public:
    Position(){}
```

```
Position(int x, int y)
{
    longitude = x;
    latitude = y;
}
friend std::istream& operator>>(std::istream& is, Position& p)
{
    is >> p.longitude >> p.latitude;
    if (p.longitude > 90) p.longitude = p.longitude % 90 - 90;
    if (p.longitude < -90) p.longitude = p.longitude % 90 + 90;
    if (p.latitude > 180) p.latitude = p.latitude % 180 - 180;
    if (p.latitude < -180) p.latitude = p.latitude % 180 + 180;
    return is;
}
friend std::ostream& operator<<(std::ostream& os, const Position& p)
{
    os << "долгота:" << p.longitude << " широта:" << p.latitude;
    return os;
}
Position operator+(const Position& p)
{
    longitude += p.longitude;
    latitude += p.latitude;
    if (longitude > 90) longitude = longitude % 90 - 90;
    if (longitude < -90) longitude = longitude % 90 + 90;
    if (latitude > 180) latitude = latitude % 180 - 180;
    if (latitude < -180) latitude = latitude % 180 + 180;
    return Position(longitude, latitude);
}
Position operator-(const Position& p)
{
    longitude -= p.longitude;
    latitude -= p.latitude;
    if (longitude > 90) longitude = longitude % 90 - 90;
    if (longitude < -90) longitude = longitude % 90 + 90;
    if (latitude > 180) latitude = latitude % 180 - 180;
    if (latitude < -180) latitude = latitude % 180 + 180;
    return Position(longitude, latitude);
}
Position operator*(const Position& p)
{
    longitude *= p.longitude;
    latitude *= p.latitude;
    if (longitude > 90) longitude = longitude % 90 - 90;
    if (longitude < -90) longitude = longitude % 90 + 90;
    if (latitude > 180) latitude = latitude % 180 - 180;
    if (latitude < -180) latitude = latitude % 180 + 180;
    return Position(longitude, latitude);
}
Position operator/(const Position& p)
{
```

```
        longitude /= p.longitude;
        latitude /= p.latitude;
        if (longitude > 90) longitude = longitude % 90 - 90;
        if (longitude < -90) longitude = longitude % 90 + 90;
        if (latitude > 180) latitude = latitude % 180 - 180;
        if (latitude < -180) latitude = latitude % 180 + 180;
        return Position(longitude, latitude);
    }
    bool operator>(const Position& p)
    {
        return (longitude > p.longitude) && (latitude > p.latitude);
    }
    bool operator>=(const Position& p)
    {
        return (longitude >= p.longitude) && (latitude >= p.latitude);
    }
    bool operator<(const Position& p)
    {
        return (longitude < p.longitude) && (latitude < p.latitude);
    }
    bool operator<=(const Position& p)
    {
        return (longitude <= p.longitude) && (latitude <= p.latitude);
    }
    bool operator==(const Position& p)
    {
        return (longitude == p.longitude) && (latitude == p.latitude);
    }
    bool operator!=(const Position& p)
    {
        return (longitude != p.longitude) && (latitude != p.latitude);
    }
    ~Position(){}
};

int operator "" _S(unsigned long long latitude)//южная широта
{
    return -1*latitude;
}
int operator "" _N(unsigned long long latitude)//северная широта
{
    return 1*latitude;
}
int operator "" _E(unsigned long long longitude)//восточная долгота
{
    return 1*longitude;
}
int operator "" _W(unsigned long long longitude)//западная долгота
{
    return -1*longitude;
}
```

```
int main()
{
    Position a(48_N, 2_E),b;
    std::cin >> b;
    std::cout << "позиция а: " << a << std::endl;
    std::cout << "позиция б: " << b << std::endl;
    std::cout << "a + b = " << a + b << std::endl;
    std::cout << "a - b =" << a - b << std::endl;
    std::cout << "a * b = " << a * b << std::endl;
    std::cout << "a / b = " << a / b << std::endl;
    if (a >= b) std::cout << "a >= b" << std::endl;
    if (a <= b) std::cout << "a <= b" << std::endl;
    if (a == b) std::cout << "a == b" << std::endl;
    if (a != b) std::cout << "a != b" << std::endl;
    return 0;
}
```