**Московский авиационный институт**

**(Национальный исследовательский университет)**

Факультет: «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: Федоров Антон Сергеевич

Группа: 80-207

Преподаватель: Чернышов Л.Н.

Дата:

Оценка:

Москва, 2020

1. **Постановка задачи**

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

**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.

1. **Описание программы**

Класс Position имеет 2 приватные переменные типа double: широта(latitude), долгота(longitude), а также приватный метод check\_limits, для взятия по модулю координат объекта. Есть как пустой, так и конструктор с параметрами, один задает позицию [0,0], другой принимает широту и долготу в качестве параметров. Для перегрузки оператора <, <=, >, >= введу понятие нормы. Норма - это корень из суммы квадратов широты и долготы. Перегруженные +, -, \*, / выполняют вышеуказанные операции покоординатно. Оператор == дает истину, когда объекты покоординатно равны (соответственно != даст ложь) и наоборот. Операторы <, > сравнивают норму объектов (если норма отличается на значение меньше эпсилон, то ложь. <=, >= работают аналогично, но истина может быть и при отличии ном меньше чем на эпсилон.

1. **Тесты**

**Тест №1**

a 12

**Тест №2**

12 34

12 34

**Тест №3**

0 0

190 200

**Тест №4**

91 360

18 4

1. **Результаты выполнения тестов**

**Тест №1**

incorrect output

Process finished with exit code 0

**Тест №2**

example1 cords is [12, 34]

example2 cords is [12, 34]

sum is [24, 68]

difference between example1 and example2 is [0, 0]

product of multiplication is[54, 76]

quotient of division between example1 and example2 is [1, 1]

does example1 equal example2? 1

does example1 not equal example2? 0

does example1 >= example2? 1

does example1 <= example2? 1

does example1 > example2? 0

does example1 < example2? 0

Process finished with exit code 0

**Тест №3**

example1 cords is [0, 0]

example2 cords is [10, 20]

sum is [10, 20]

difference between example1 and example2 is [-10, -20]

product of multiplication is[0, 0]

quotient of division between example1 and example2 is [0, 0]

does example1 equal example2? 0

does example1 not equal example2? 1

does example1 >= example2? 0

does example1 <= example2? 1

does example1 > example2? 0

does example1 < example2? 1

Process finished with exit code 0

**Тест №4**

example1 cords is [1, 0]

example2 cords is [18, 4]

sum is [19, 4]

difference between example1 and example2 is [-17, -4]

product of multiplication is[18, 0]

quotient of division between example1 and example2 is [0.0555556, 0]

does example1 equal example2? 0

does example1 not equal example2? 1

does example1 >= example2? 0

does example1 <= example2? 1

does example1 > example2? 0

does example1 < example2? 1

Process finished with exit code 0

1. **Листинг программы**

Класс Position вынесен в отдельный файл. Реализация методов класса вынесена в отдельный cpp файл.

**CMakeLists.txt**

cmake\_minimum\_required(VERSION 3.16)

project(oop\_exercise\_01)

set(CMAKE\_CXX\_STANDARD 20)

add\_executable(oop\_exercise\_01 main.cpp Position.cpp Position.h)

**Position.h**

#ifndef LB1\_POSITION\_H

#define LB1\_POSITION\_H

#include <math.h>

#include <iostream>

#define DBL\_EPSILON 10e-15

class Position{

private:

double latitude;

double longitude;

double norm()const;

void check\_limits();//берет координаты по модулю, чтобы они не выходили за приделы [-90,90] и [-180,180]

public:

Position();

Position(double cur\_latitude, double cur\_longitude);

Position set\_position(double cur\_latitude, double cur\_longitude);

void print\_position();

friend Position operator+(const Position &lhs, const Position &rhs);

friend Position operator-(const Position &lhs, const Position &rhs);

friend Position operator\*(const Position &lhs, const Position &rhs);

friend Position operator/(const Position &lhs, const Position &rhs);

friend bool operator==(const Position &lhs, const Position &rhs);

friend bool operator!=(const Position &lhs, const Position &rhs);

friend bool operator>(const Position &lhs, const Position &rhs);

friend bool operator>=(const Position &lhs, const Position &rhs);

friend bool operator<(const Position &lhs, const Position &rhs);

friend bool operator<=(const Position &lhs, const Position &rhs);

};

#endif

**Position.cpp**

#include "Position.h"

//Конструкторы

void Position::check\_limits(){

latitude=fmod(latitude,90);

longitude=fmod(longitude,180);

};

Position::Position(){latitude=0; longitude=0;}

Position::Position(double cur\_latitude, double cur\_longitude){

latitude=cur\_latitude; longitude=cur\_longitude;

check\_limits();

}

//Методы класса

double Position::norm()const{

return sqrt(latitude \* latitude + longitude \* longitude);

};

Position Position::set\_position(double cur\_latitude, double cur\_longitude){

latitude=cur\_latitude; longitude=cur\_longitude;

check\_limits();

return \*this;

}

void Position::print\_position(){

check\_limits();

std::cout << "[" << latitude << ", " << longitude << "]" << std::endl;

};

//Перегрузка операторов

Position operator+(const Position &lhs, const Position &rhs){

return Position(fmod(lhs.latitude+rhs.latitude,90), fmod(lhs.longitude + rhs.longitude, 180));

}

Position operator-(const Position &lhs, const Position &rhs){

return Position(fmod(lhs.latitude-rhs.latitude,90), fmod(lhs.longitude - rhs.longitude,180));

}

Position operator\*(const Position &lhs, const Position &rhs){

return Position(fmod(lhs.latitude\*rhs.latitude,90), fmod(lhs.longitude \* rhs.longitude,180));

}

Position operator/(const Position &lhs, const Position &rhs){

return Position(fmod(lhs.latitude / rhs.latitude,90), fmod(lhs.longitude / rhs.longitude,180));

}

bool operator==(const Position &lhs, const Position &rhs){

return (lhs.latitude == rhs.latitude && rhs.longitude == lhs.longitude);

}

bool operator!=(const Position &lhs, const Position &rhs){

return (lhs.latitude != rhs.latitude && rhs.longitude != lhs.longitude);

}

bool operator>(const Position &lhs, const Position &rhs){

return (lhs.norm()>rhs.norm())&&!(abs(lhs.norm()-rhs.norm())<DBL\_EPSILON);

}

bool operator>=(const Position &lhs, const Position &rhs){

return (lhs.norm()>rhs.norm())||(abs(lhs.norm()-rhs.norm())<DBL\_EPSILON);

}

bool operator<(const Position &lhs, const Position &rhs){

return (lhs.norm()<rhs.norm())&&!(abs(lhs.norm()-rhs.norm())<DBL\_EPSILON);

}

bool operator<=(const Position &lhs, const Position &rhs){

return (lhs.norm()<rhs.norm())||(abs(lhs.norm()-rhs.norm())<DBL\_EPSILON);

}

**main.cpp**

#include <iostream>

#include "Position.h"

int main(){

Position example1,example2;

double latitude1, longitude1;

std::cin >> latitude1 >> longitude1;

if (std::cin.fail()){

std:: cout << "incorrect output" << std::endl;

return 0;

}

double latitude2, longitude2;

if (std::cin.fail()){

std:: cout << "incorrect output" << std::endl;

return 0;

}

std::cin >> latitude2 >> longitude2;

example1.set\_position(latitude1, longitude1);

example2.set\_position(latitude2, longitude2);

std::cout << "example1 cords is "; example1.print\_position();

std::cout << "example2 cords is "; example2.print\_position();

std::cout << "sum is "; (example1+example2).print\_position();

std::cout << "difference between example1 and example2 is "; (example1-example2).print\_position();

std::cout << "product of multiplication is"; (example1\*example2).print\_position();

std::cout << "quotient of division between example1 and example2 is "; (example1/example2).print\_position();

std::cout << "does example1 equal example2? " << (example1==example2) << std::endl;

std::cout << "does example1 not equal example2? " << (example1!=example2) << std::endl;

std::cout << "does example1 >= example2? " << (example1>=example2) << std::endl;

std::cout << "does example1 <= example2? " << (example1<=example2) << std::endl;

std::cout << "does example1 > example2? " << (example1>example2) << std::endl;

std::cout << "does example1 < example2? " << (example1<example2) << std::endl;

return 0;

}

Выводы

Я познакомился с системой сборки CMake, научился создавать простые классы на языке C++, перегружать операторы через дружественные функции. Познакомился с инструментами ввода и вывода библиотеки iostream.