**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: Абдыкеримов Урмат

Группа: 80-206

Преподаватель: Чернышов Л.Н.

Дата:

Оценка:

Москва, 2020

1. Постановка задачи  
   *Создать класс Angle для работы с углами на плоскости, задаваемыми величиной в градусах и минутах. Обязательно должны быть реализованы: перевод в радианы, приведение к диапазону 0–360, сложение и вычитание углов, деление углов, получение значений тригонометрических функций, сравнение углов.*
2. Описание программы

*В программе реализован класс Angle, с полем degrees и всем методами описанными в постановке задачи. Конструктор принимает два аргумента типа double: градусы и минуты создаваемого угла.*

*Программа принимает на вход четыре переменные типа double: градусы и минуты углов alpha, betta.*

*Далее демонстрируется работа методов класса Angle на углах alpha, betta*

1. Набор тестов

*test\_01:  
0 30 0 30*

*test\_02:  
1 35 2 37*

*test\_03:  
500 59 500 59*

1. Результаты выполнения тестов

*test\_01:*Sin of alpha 0.00872654

alpha / betta = 1

Is betta smaller then alpha 0

Alpha in radians = 0.00872665

Adding betta to alpha

New alpha 1

Subtracting betta from alpha

New alpha 0.5

Alpha in [0;360] degrees 0.5

*test\_02:*Sin of alpha 0.0276309

alpha / betta = 0.605096

Is betta smaller then alpha 0

Alpha in radians = 0.0276344

Adding betta to alpha

New alpha 4.2

Subtracting betta from alpha

New alpha 1.58333

Alpha in [0;360] degrees 1.58333

*test\_03:*Sin of alpha 0.629546

alpha / betta = 1

Is betta smaller then alpha 0

Alpha in radians = 8.74381

Adding betta to alpha

New alpha 1001.97

Subtracting betta from alpha

New alpha 500.983

Alpha in [0;360] degrees 140.983

1. Листинг программы

//Абдыкеримов Урмат Эмирбекович М8О-206Б-19

#include <iostream>

#include <cmath>

using *namespace* std;

const *double* PI = acos(-1), eps = 1e-8;

*class* Angle {

*public:*

*double* degrees;

  Angle();

  Angle(*double* *\_degrees*, *double* *\_minutes*) {

    degrees = *\_degrees* + *\_minutes* / 60.0;

  }

*void* add(Angle *aim*) {

    //увеличиваю degrees на угол aim

    degrees += *aim*.degrees;

  }

*void* subtract(Angle *aim*) {

    //уменьшаю degrees на aim

    degrees -= *aim*.degrees;

  }

*double* inRadians() {

    //значение degrees в радианах

*double* res = degrees / 180.0 \* PI;

    return res;

  }

*double* getSin() {

    return sin(degrees / 180.0 \* PI);

  }

*double* getCos() {

    return cos(degrees / 180.0 \* PI);

  }

*void* to2pi() {

    //привожу degrees к отрезку [0:360] градусов

    if(degrees >= 0) {

      degrees -= (*int*)degrees / 360 \* 360;

    } else {

      degrees += ((*int*)(abs((*int*)degrees) / 360 + 1)) \* 360;

    }

  }

*bool* isBigger(Angle *aim*) {

    //если aim больше degrees возвращает true

    return *aim*.degrees > degrees;

  }

*bool* isSmaller(Angle *aim*) {

    //если aim меньше degrees возвращает true

    return *aim*.degrees < degrees;

  }

*bool* isEqual(Angle *aim*) {

    return abs(*aim*.degrees - degrees) < eps;

  }

*double* div(Angle *aim*) {

    //во сколько раз degrees больше чем aim

    return degrees / *aim*.degrees;

  }

};

*int* main() {

*double* alDegrees, alMinutes, beDegrees, beMinutes;

  cin >> alDegrees >> alMinutes >> beDegrees >> beMinutes;

  Angle alpha(alDegrees, alMinutes), betta(beDegrees, beMinutes);

  cout << "Sin of alpha " << alpha.getSin() << endl;

  cout << "alpha / betta = " << alpha.div(betta) << endl;

  cout << "Is betta smaller then alpha " << alpha.isSmaller(betta) << endl;

  cout << "Alpha in radians = " << alpha.inRadians() << endl;

  cout << "Adding betta to alpha " << endl;

  alpha.add(betta);

  cout << "New alpha "<< alpha.degrees << endl;

  cout << "Subtracting betta from alpha" << endl;

  alpha.subtract(betta);

  cout << "New alpha " << alpha.degrees << endl;

  alpha.to2pi();

  cout << "Alpha in [0;360] degrees " << alpha.degrees << endl;

}