

**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 C++

Студент: Кудинов Сергей

Преподаватель: Журавлев А.А.

Дата:

Оценка:

Москва, 2019

1. Постановка задачи

Создать класс `vector3D`, задаваемый тройкой координат. Обязательно должны быть реализованы: операции сложения и вычитания векторов, векторное произведение векторов, скалярное произведение векторов, умножения на скаляр, сравнение векторов на совпадение, вычисление длины вектора, сравнение длины векторов, вычисление угла между векторами.

2. Репозиторий github

https://github.com/StormStudioAndroid2/oop_exercise_1

3. Описание программы

Реализован класс `Vector3D`, в котором хранятся три переменные, отображающие координаты. Написаны `Get` функции для их получения. Также реализованы функции, указанные в задании, для получения суммы и разности(`plus`, `minus`), и для сравнения различных объектов класса(`equal`). Функции `plus` и `minus` возвращают вектор. Реализованы векторное произведение, скалярное, произведение на число(`vectorPow`, `scalarPow`, `lambdaPow`). В начале выводится сложение первого и второго вектора, затем их вычитание, затем – величина угла между ними, затем длины первого и второго вектора, затем – равны они или нет. Последним выводится значение длины вектора, полученного в результате векторного произведения. Каждая величина выводится с новой строки.

4. Набор testcases

Тестовые файлы: `test_01.test`, `test_02.test`, `test_03.test`

test_01.test:

1 1 1

-1 -1 -1

Проверка правильности для коллинеарных векторов с противоположным направлением

Результат работы программы

0 0 0

2 2 2

180

1.73205

1.73205

Not equal

0

test_02.test :

1 2 3

4 5 6

Проверка корректности работы для случайных векторов

Результат работы программы

5 7 9

-3 -3 -3

12.9332

3.74166

8.77496

Not equal

7.34847

test_03.test:

1 1 1

1 1 1

Проверка корректности работы операций сложения и вычитания для равных векторов

Результат работы программы

2 2 2

0 0 0

0

1.73205

1.73205

Equal

0

5. Результаты выполнения тестов

Все тесты успешно пройдены, программа выдаёт верные результаты, корректно обрабатывает время.

6. Листинг программы

main.cpp

```
#include<iostream>

1.#include "Vector3D.h"
2.#include <iomanip>
3.
4.int main() {
5.    int x1,x2,y1,y2,z1,z2;
6.    std::cin >> x1 >> y1 >> z1;
7.    std::cin >> x2 >> y2 >> z2;
8.    std::cout.precision(6);
9.    Vector3D vector1(x1,y1,z1);
10.   Vector3D vector2(x2,y2,z2);
11.   std::cout << vector1.plus(vector2).getX() << " " << vector1.plus(vector2).ge
tY() << " " << vector1.plus(vector2).getZ() << std::endl;
12.   std::cout << vector1.minus(vector2).getX() << " " << vector1.minus(vector2).
getY() << " " << vector1.minus(vector2).getZ() << std::endl;
13.   std::cout << vector1.getAngle(vector2) << std::endl;
14.   std::cout << vector1.getLength() << std::endl;
15.   std::cout << vector2.getLength() << std::endl;
16.   if (vector1.isEqual(vector2)) {
17.       std::cout << "Equal" << std::endl;
18.   } else {
19.       std::cout << "Not equal" << std::endl;
20.   }
21.   std::cout << vector1.crossProduct(vector2).getLength() << std::endl;
22.   return 0;
23.}
```


Vector3D.h

```
#include <iostream>

1. #include <cmath>
2.
3.
4. class Vector3D
5. {
6. private:
7.     double x;
8.     double y;
9.     double z;
10.
11. public:
12.     Vector3D(double x, double y, double z);
13.     Vector3D();
14.     Vector3D plus(const Vector3D& vector);
15.     Vector3D minus(const Vector3D& vector);
16.     Vector3D crossProduct(const Vector3D& vector);
17.     void lambdaProduct(double lambda);
18.     double scalarProduct(const Vector3D& vector);
19.     bool isEqual(const Vector3D& vector);
20.     double getLength();
21.     double getAngle( Vector3D& vector);
22.     double getX();
23.     double getY();
24.     double getZ();
25.
26. };
```

Vector3D.cpp

```
1. #include "Vector3D.h"
2. #include <iostream>
3. #include <cmath>
4.
5.
6. Vector3D::Vector3D(double x, double y, double z)
7. : x(x), y(y), z(z) {}
8.     Vector3D::Vector3D()
9. : x(0), y(0), z(0) {}
10. Vector3D Vector3D::plus(const Vector3D& vector) {
11.     Vector3D result;
12.     result.x = vector.x+this->x;
13.     result.y = vector.y+this->y;
```

```

14.     result.z = vector.z+this->z;
15.
16.     return result;
17. }
18. Vector3D Vector3D::minus(const Vector3D& vector) {
19.     Vector3D result;
20.     result.x = this->x-vector.x;
21.     result.y = this->y-vector.y;
22.     result.z = this->z-vector.z;
23.     return result;
24. }
25. Vector3D Vector3D::crossProduct(const Vector3D& vector) {
26.     Vector3D result;
27.     result.x = this->y*vector.z-this->z*vector.y;
28.     result.y = this->z*vector.x-this->x*vector.z;
29.     result.z = this->x*vector.y-this->y*vector.x;
30.
31.     return result;
32. }
33. void Vector3D::lambdaProduct(double lambda) {
34.     this->x*=lambda;
35.     this->y*=lambda;
36.     this->z*=lambda;
37.
38. }
39. double Vector3D::scalarProduct(const Vector3D& vector) {
40.     return this->x*vector.x+this->y*vector.y+this->z*vector.z;
41. }
42. bool Vector3D::isEqual(const Vector3D& vector) {
43.     return (this->x==vector.x && this->y==vector.y && this->z==vector.z);
44. }
45. double Vector3D::getLength() {
46.     return sqrt(this->x*this->x+this->y*this->y+this->z*this->z);
47. }
48.
49. double Vector3D::getAngle( Vector3D& vector) {
50.     if ((vector.getLength()==0) || (this->getLength()==0) {
51.         return 0;
52.     }
53.     const double halfC = 180/M_PI;
54.     double cos1 = (this->scalarProduct(vector)/(this->getLength()*vector.getLength())) ;
55.     if (cos1 < -1) {
56.         return 180;
57.     }
58.     if (cos1>1) {
59.         return 0;
60.     }

```

```
61.     return halfC*acos(cos1);
62.
63. }
64.
65. double Vector3D::getX() {
66.     return x;
67. }
68. double Vector3D::getY() {
69.     return y;
70. }
71. double Vector3D::getZ() {
72.     return z;
73. }
```

7. Вывод

Реализована программа, включающая в себя простой класс с методами и переменными. Также получены навыки работы с git и stake.

Список литературы

1. Шилдт, Герберт. С++: базовый курс, 3-е изд. : Пер. с англ. - М. : ООО "И.Д. Вильямс", 2018. - 624 с. : ил. - Парал. тит. англ.