Московский Авиационный Институт
(Национальный исследовательский Университет)

Факультет: «Информационные технологии и прикладная математика»
Кафедра: 806 «Вычислительная математика и программирование»

Лабораторная работа
по курсу «ООП»

Тема:
Простые классы.

Студент:	Голубев В.С.
Группа:	М80-206Б-18
Преподаватель:	Журавлев А.А.
Вариант:	Quaternion
Оценка:	
Дата:	

Москва
2019

quaternion.hpp:

```
#ifndef LAB1_QUATERNION_HPP
                            #define LAB1_QUATERNION_HPP
                            struct quaternion {
                                quaternion ();
                                quaternion (double real, double x, double y,
                            double z);
                                void print();
                                quaternion sum(const quaternion& a);
                                quaternion diff(const quaternion& a);
                                quaternion mult(const quaternion& b);
                                quaternion div(const quaternion &b);
                            private:
                                double array[4];
                            };
                            #endif //LAB1_QUATERNION_HPP
```

quaternion.cpp:

```
                        #include <iostream>
                         #include <cmath>


                         #include "quaternion.hpp"
                         quaternion::quaternion() {
                             for (int i = 0; i < 4; ++i) {
                                 array[i] = 0;
                             }
                         }
                         quaternion::quaternion (double real, double x,
                         double y, double z) {
                             array[0] = real;
                             array[1] = x;
                             array[2] = y;
                             array[3] = z;
                         }
                         quaternion quaternion::sum(const quaternion &a) {
                             quaternion res;
                             for (int i = 0; i < 4; ++i) {
                                 res.array[i] = array[i] + a.array[i];
                             }
                             return res;
                         }
                         quaternion quaternion::diff(const quaternion &a) {
                             quaternion res;
```

```
    for (int i = 0; i < 4; ++i) {
        res.array[i] = array[i] - a.array[i];
    }
    return res;
}
quaternion quaternion::mult(const quaternion &b) {
    quaternion res;
    res.array[0] = array[0] * b.array[0] - array[1]
* b.array[1] - array[2] * b.array[2] - array[3] *
b.array[3];
    res.array[1] = array[0] * b.array[1] + array[1]
* b.array[0] + array[2] * b.array[3] - array[3] *
b.array[2];
    res.array[2] = array[0] * b.array[2] + array[2]
* b.array[0] + array[3] * b.array[1] - array[1] *
b.array[3];
    res.array[3] = array[0] * b.array[3] + array[3]
* b.array[0] +  array[1] * b.array[2] - array[2] *
b.array[1];
    return res;
}
quaternion quaternion::div(const quaternion &b) {
    quaternion conjugate;
    double VectNorm;
    quaternion reverse;
    quaternion res;
    for (int i = 0; i < 4; ++i) {
        if (i == 0) {
            conjugate.array[i] = b.array[i];
        } else {
            conjugate.array[i] = -b.array[i];
        }
    }
    VectNorm = sqrt(
            b.array[0] * b.array[0] + b.array[1] *
b.array[1] + b.array[2] * b.array[2] + b.array[3] *
b.array[3]);
    for (int i = 0; i < 4; ++i) {
        reverse.array[i] = conjugate.array[i] /
(VectNorm * VectNorm);
    }

  res = mult(reverse);
 return res;
}
```

```
void quaternion::print() {
    std::cout << array[0] << " " "+" << " " << "("
<< array[1] << ")" << "i" <<" " "+" <<" " << "(" <<
array[2]<< ")" "j" << " " "+" << " " << "(" <<
array[3] << ")" << "k" << std::endl;
}
```

main.cpp:

```
#include <iostream>

#include <cmath>
#include "quaternion.hpp"
int main() {
    double q1, x1, y1, z1;
    double q2, x2, y2, z2;
    std::cout << "Enter quaternions" << std::endl;
    std::cin >> q1 >> x1 >> y1 >> z1 >> q2 >> x2 >> y2 >> z2;
    quaternion V1{q1, x1, y1, z1};
    quaternion V2{q2, x2, y2, z2};
    std::cout << " H1 :" << std::endl;
    V1.print();
    std::cout << "H2:" << std::endl;
    V2.print();
    std::cout << "Sum is :" << std::endl;
    V1.sum(V2).print();
    std::cout << "Difference is :" << std::endl;
    V1.diff(V2).print();
    std::cout << "Product is :" << std::endl;
    V1.mult(V2).print();
    std::cout << "Quotient is :" << std::endl;
    V1.div(V2).print();
    return 0;
}
```

CmakeLists.txt:

```
cmake_minimum_required(VERSION 3.14)
 set(CMAKE_CXX_STANDARD 14)
 add_executable(lab1 main.cpp
 quaternion.cpp quaternion.hpp)
```

2. Ссылка на репозиторий на GitHub

https://github.com/VSGolubev-bald /oop_exercise_01

3. Набор testcases.

test_00:

```
1 2 3 4
4 3 2 1
```

test_01:

```
7.8 9.3 6 5.298
1.1 2.2 3.3 4.40
```

test_02:

```
0 2.67 3.01 4.703
1.23 2.00 4.5 7
```

test_03:

```
1124.5 3.5 6.7 4.7
103 21 65.678 3.456
```

test_04:

```
0 0 0 0
4.5 6.789 11.89 3.00
```

4. Результаты выполнения тестов.

test_00:

```
H1 :
1 + (2)i + (3)j + (4)k
H2:
5 + (4)i + (3)j + (2)k
Sum is :
6 + (6)i + (6)j + (6)k
Difference is :
-4 + (-2)i + (0)j + (2)k
Product is :
-20 + (13)i + (30)j + (16)k
Quotient is :
0.0925926 + (-0.148148)i + (-0.166667)j + (-0.148148)k

Process finished with exit code 0
```

test_01:

```
H1 :
7.8 + (9.3)i + (6)j + (5.298)k
H2:
1.1 + (2.2)i + (3.3)j + (4.4)k
Sum is :
8.9 + (11.5)i + (9.3)j + (9.698)k
Difference is :
6.7 + (7.1)i + (2.7)j + (0.898)k
Product is :
-54.9912 + (45.192)i + (3.0756)j + (57.6378)k
Quotient is :
0.236364 + (-0.563636)i + (-0.545455)j + (-0.642182)k

Process finished with exit code 0
```

test_02:

```
H1 :
0 + (2.67)i + (3.01)j + (4.703)k
H2:
1.23 + (2)i + (4.5)j + (7)k
Sum is :
1.23 + (4.67)i + (7.51)j + (11.703)k
Difference is :
-1.23 + (0.67)i + (-1.49)j + (-2.297)k
Product is :
-51.806 + (15.1511)i + (-5.5817)j + (11.7797)k
Quotient is :
0 + (-0.0714258)i + (-0.181173)j + (-0.440339)k

Process finished with exit code 0
```

test_03:

```
H1 :
1124.5 + (3.5)i + (6.7)j + (4.7)k
H2:
103 + (21)i + (65.678)j + (3.456)k
Sum is :
1227.5 + (24.5)i + (72.378)j + (8.156)k
Difference is :
1021.5 + (-17.5)i + (-58.978)j + (1.244)k
Product is :
115294 + (23927.8)i + (74631.6)j + (4459.55)k
Quotient is :
7.53297 + (-0.00478032)i + (-0.0286196)j + (-0.00105643)k

Process finished with exit code 0
```

test_04

```
H1 :
0 + (0)i + (0)j + (0)k
H2:
4.5 + (6.789)i + (11.89)j + (3)k
Sum is :
4.5 + (6.789)i + (11.89)j + (3)k
Difference is :
-4.5 + (-6.789)i + (-11.89)j + (-3)k
```

```
Product is :
0 + (-11.89)i + (0)j + (0)k
Quotient is :
0 + (-0)i + (-0)j + (-0)k

Process finished with exit code 0
```

5. Объяснение результатов работы программы.

1) При запуске программы с аргументом test_??.txt объекты V1 и V2 получают компоненты q, отвечающие за действительную часть кватерниона, и компоненты x, y, z , отвечающие за мнимую векторную часть, из файлов test_??.txt.
2) Вывод данных объектов в стандартный поток вывода.
3) Объекты V1 и V2 складываются с помощью метода sum() класса quaternion , и результат выводится в стандартный поток вывода с помощью метода print класса quaternion.
4) Из объекта V1 вычитается объект V2 с помощью функции-члена difference() класса quaternion, и результат выводится в стандартный поток вывода с помощью метода print.
5) Объект V1 умножается на объект V2 с помощью метода mult класса quaternion, и выводится в стандартный поток вывода с помощью метода print.
6) Объект V1 делится на объект V2 с помощью метода div класса quaternion, и выводится в стандартный поток вывода с помощью метода print.

6. Вывод.

Выполняя данную лабораторную, я получил опыт работы с простыми классами, с системой сборки Cmake, с системой контроля версий GitHub, а также изучил основы работы с классами в C++. Попытался вникнуть в суть понятия «кватернион». В очередной раз убедился, насколько интересна, сложна и познавательна математика.