

## Домашняя работа №2

### Знакомство с C++

#### Инструментарий и требования к работе

Работа выполняется на C++. На сервере сборка под C++20.

#### [ППА] Модификация

Тесты группы "extra\_functions" теперь являются обязательными к прохождению.

#### Задание

Напишите реализацию шаблонного класса кватерниона `Quat` с параметром `T` – тип данных для хранения частей числа (гарантируется, что в тестах `T` – вещественный тип).

Необходимо реализовать:

- операторы (список ниже).
- методы (список ниже).
- конструктор из 4 значений типа `T`: `a`, `b`, `c`, `d` (нотация `a + bi + cj + dk`)
- конструктор, создающий кватернион поворота из 3 аргументов: `T` (угол поворота), `bool` (true: угол передан в радианах, иначе: в градусах) и вектора (3 координаты, задающие ось вращения).

Нужно реализовать возможность создания объекта класса `Quat` без аргументов (все компоненты заполняются 0).

В репозитории будет `tests.cpp` для тестирования (google test) и шаблон для класса кватерниона `quat.hpp`. Файл `tests.cpp` менять нельзя. В файле `quat.hpp` нельзя менять выданное, но можно (и нужно) добавлять свои методы/операторы `Quat` и необходимые заголовочные файлы.

Информацию про кватернионы и формальное описание реализации его операций можно посмотреть [здесь](#) и [здесь](#). Обратите внимание, что кватернион может быть представлен как комбинация скаляра и вектора из трёх компонентов. Соответственно, операции по кватерниону и скаляру/вектору следует рассматривать с этой точки зрения.

Для отправки на проверку необходимо, чтобы проходили все тесты, кроме тестов `rotation_matrix`, `matrix`, `angle`, `apply`, за которые можно получить дополнительные баллы.

Список операторов:

Оператор	Аргументы	Пояснение
<code>+</code> , <code>+=</code> , <code>-</code> , <code>-=</code>	<code>Quat</code>	Сложение/вычитание двух кватернионов (поэлементно)
<code>*</code>	<code>Quat</code>	Умножение кватернионов
<code>~</code>	<code>-</code>	Сопряжение
<code>==</code> , <code>!=</code>	<code>Quat</code>	Сравнение на (не)равенство
приведения типа к <code>T</code>	<code>-</code>	Вычисление модуля кватерниона Примечание: оператор должен быть <code>explicit</code>

В дополнение к вышеописанному должно работать умножение на скаляр и вектор (типа `T`). Это может быть реализовано как через перегрузку операторов, так и через дополнительные конструкторы.

Список методов:

Метод	Аргументы	Пояснение
<code>data</code>	-	Получение указателя на 4 компонента кватерниона в порядке <code>b, c, d, a</code> (в другой нотации: <code>x, y, z, w</code> ).
<code>rotation_matrix</code>	-	Получение матрицы поворота.
<code>matrix</code>	-	Получение вещественного матричного представления.
<code>angle</code>	<code>bool</code>	Нахождение угла кватернионами поворота. Принимает флажок, вернуть значение в градусах или нет, аналогично конструктору из угла.
<code>apply</code>	<code>vector3_t</code>	Применение кватерниона к вектору. Поворот переданного в аргументах вектора на текущий кватернион.

Матрицы хранятся в порядке row-major.