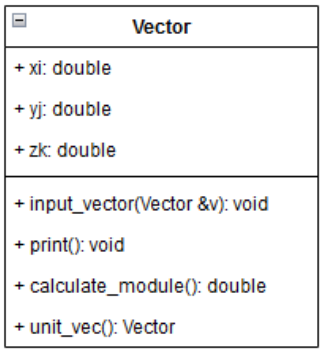
Список 01 – Лабораторная практика

Классы и объекты

Предмед: Алгоритмизация и программирование

Преподаватель: Хольгер Эспинола Ривера

1. Вектор как класс. Рассмотрим UML-диаграмму класса Vector, которая представляет трехмерного вектора с компонентами **i**, **j**, **k**.



Реализуйте следующие операции на C++:

[1]. Создать класс, содержащий трехмерный вектор координат (x, y, z)

[2]. Реализовать конструкторы и деконструкторы

[3]. Реализовать ввод координат каждого вектора как метод класса

[4]. Реализовать в качестве публичных методов операции печати, вычисления модуля и единичного вектора.

[5]. Реализуйте следующие операции между двумя векторами:

- скалярное произведение

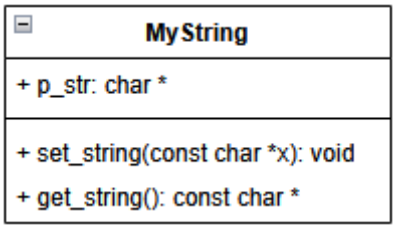
- векторное произведение

- угол между 2 векторами

- Евклидово расстояние

- расстояние от Манхэттена

2. Классы и объекты для строк. Постройте класс MyString согласно следующей диаграмме UML, определив соответствующие атрибуты и реализуя методы:

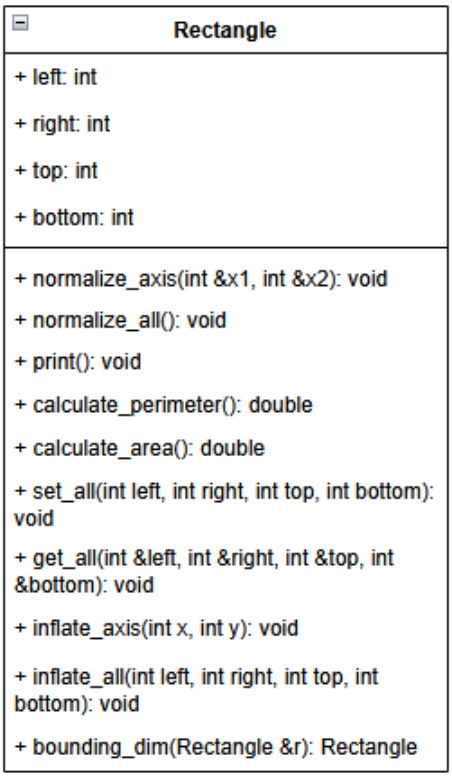


[1]. Используйте конструктор(ы) и деструктор, чтобы гарантировать правильная инициализация и деактивация объекта.

[2]. Создайте метод **get_string**, который предоставит доступ к сохраненной строке. Используя cout и метод **get_string**, выведите строку из объекта str.

[3]. Реализуйте метод **set_string**, чтобы иметь возможность вводить новую строку символов и заменять текущую, сохраненную в созданном объекте класса MyString.

3. Прямоугольник 2D как класс. Постройте класс Rectangle согласно следующей диаграмме UML, определив соответствующие атрибуты и реализуя методы:



[1]. Объявление класса. Создание экземпляра класса. Создание конструкторов для класса Rectangle. Рассмотрим 2 типа конструкторов: конструктор, который может устанавливать значения для каждого параметра класса, и один конструктор, который может устанавливать значения по умолчанию, если какой-либо параметр не был указан.

[2]. Перегрузка конструкторов и методов класса. Используйте конструктор по умолчанию, конструктор с параметрами и конструктор с копией объекта.

[3]. Реализуйте функцию для нормализации границ. Это означает, что функция должна проверять, что по некоторой оси первое значение в некоторой координате должно быть меньше второго значения. По оси X: левое должно быть меньше правого, а по оси Y: нижнее должно быть меньше верхнего.

[4]. Постройте метод **print**, чтобы можно было отобразить каждое значение для параметров, связанных с объектом класса Rectangle.

[5]. Реализовать функцию для вычисления периметра прямоугольника.

[6]. Реализуйте функцию для вычисления площади прямоугольника.

[7]. Определите метод **inflate**, который принимает аргументы и расширяет стороны прямоугольника в соответствии с указанными приращениями. Вызовите и выполните функцию inflate для сценариев с разными экземплярами параметров.

[8]. Реализуйте методы **set** и **get** для класса Rectangle. Метод set устанавливает все 4 границы прямоугольника и нормализует их. Метод get извлекает текущую границу прямоугольника и сохраняет их в предоставленных ссылочных параметрах. Используйте метод set для передачи значений аргументов классу и используйте метод get для получения значений закрытых переменных класса.

[9]. Спецификаторы доступа. Инкапсуляция. Создайте глобальный метод для получения нового прямоугольника, который берет размеры из сравнения границ между 2 прямоугольниками. Размеры третьего прямоугольника должны выбирать самые низкие значения из левых и нижних значений и самые высокие значения из правых и верхних значений. Реализуйте функцию, чтобы можно было передавать эти экземпляры класса как функцию параметра и передавать объекты по ссылке.