**Часть Данича:**

В модели ООП для данного проекта было создано 1 интерфейс и 3 основных класса:

**class Test:**

Поля:

public AlgorithmItnerface Instance { set; get; }

– данное поле нужно, чтобы класс тестирования принимал в себя любой экземпляр класса, созданного на основе интерфейса AlgorithmInterface

int rangeOfRandomNumbers{ set; get; }

- данное поле задает диапазон случайных чисел от 0 до rangeOfRandomNumbers

int vectorLength { set; get; }

- задает длину вектора, подаваемого на вход алгоритмам (на графике будет прогоняться время от 0 до vectorLength)

int numberOfStarts { set; get; }

- данное поле задает количество запусков одного и того же алгоритма, с одинаковым набором параметров

float[] vector { set; get; }

- данное поле нужно для хранения вектора нужного на вход алгоритмам

Методы:

public float[] StartAlgorithm(

- данный метод запускает прогон экземпляра класса наследуемого от AlgorithmInterface, заданное пользователем количество раз, с длиной вектора от 0 до введенной пользователем величины, после чего возвращает массив результатов вычисленного времени работы алгоритма, расположенных в порядке от 0 до длины вектора

(например, нулевой элемент этого массива – это время, полученное при подаче алгоритму вектора длины 0)

**Class Timer:**

Поля:

float[] vector { set; get; }

- данное поле нужно для хранения вектора нужного на вход алгоритмам

public AlgorithmItnerface Instance { set; get; }

- данное поле нужно, чтобы класс тестирования принимал в себя любой экземпляр класса, созданного на основе интерфейса AlgorithmInterface

Методы:

public float CalculateTime()

- на основе алгоритма наследуемого от интерфейса AlgorithmItnerface, а также длины вектора, вычисляет время работы алгоритма и возвращает его

**Class Vector**

Поля:

int vectorLength { set; get; }

- задает длину вектора, подаваемого на вход алгоритмам (на графике будет прогоняться время от 0 до vectorLength)

int rangeOfRandomNumbers { set; get; }

- данное поле задает диапазон случайных чисел от 0 до rangeOfRandomNumbers

Методы:

public float[] GenerateRandomVector()

- заполняет вектор заданной длины случайными числами (от 0 до rangeOfRandomNumbers)

public float[] GenerateShuffleVector()

- заполняет вектор числами от 0 до vectorLength и перемешивает их в случайном порядке

**Class Algorithm1: AlgorithmItnerface**

- Класс алгоритма номер один, наследуется от интерфейса AlgorithmItnerface, принимает на вход вектор и возвращает f(v) = 1

**Interface AlgorithmItnerface**

- имеет один единственный метод int ExecuteAlgorithm(float[] vector), на основе данного интерфейса будут создаваться классы алгоритмы