Тест требования

# Позитивное тестирование

**Тест CreateCarControlTest**

Описание: в этом тесте проверяется корректность создания объекта класса CarControl с помощью массива объектов класса Car.

Входные данные:

* Car[] cars, заполнен данными;
* CarControl carControl.

Ожидаемые выходные данные:

* carControl не равен null;
* carControl.Length = 4.

Сценарий теста:

1. создание объекта класса CarControl и его заполнение с помощью готовых данные;
2. проверка корректности создания объекта.

**Тест SortCarControlTest**

Описание: в этом тесте проверяется корректность работы метода сортировки в классе CarControl.

Входные данные:

* Car[] unsortedArray, заполнен данными;
* Car[] sortedArray, заполнен данными, отсортирован по возрастанию;
* CarControl carControl.

Ожидаемые выходные данные:

* carControl отсортирован и равен sortedArray.

Сценарий теста:

1. создание объекта класса CarControl и его заполнение с помощью готовых данные;
2. сортировка объекта;
3. проверка объекта на корректность сортировки.

**Тест SaveCarControlTest**

Описание: в этом тесте проверяется корректность работы метода сохранения в файл в классе CarControl.

Входные данные:

* Car[] cars, заполнен данными.
* CarControl carControl.

Ожидаемые выходные данные:

* В директории тестового проекта существует файл cars.txt.

Сценарий теста:

1. создание объекта класса CarControl и его заполнение с помощью готовых данные;
2. сортировка объекта;
3. сохранение объекта в файл;
4. проверка файла на существование.

# Негативное тестирование

**Тестовый пример 4**

Описание: в этом тесте проверяется поведение программы на ввод некорректных данных.

Входные данные:

* int Legth

Ожидаемые выходные данные:

* Программа завершает свою работу с сообщением о вводе некорректных данных.

Сценарий теста:

1. запуск программы;
2. ввод в числовое поле не числа.