

Тест-требования:

1. Проверить реакцию на ввод неположительного значения количества элементов в массиве.
2. Проверить реакцию на ввод некорректных данных при заполнении массива.
3. Проверить корректность сортировки массива.
4. Проверить корректность записи результата сортировки в файл.

Тестовый пример 1.

Тест-требование 1.

Описание теста: проверить реакцию программы на ввод пользователем неположительного значения количества элементов в массиве.

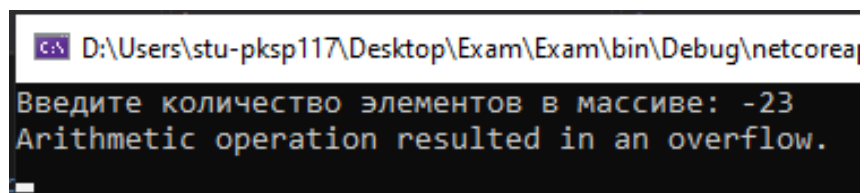
Сценарий теста:

1. Запустить программу.
2. Ввести количество элементов в массиве -2.

Входные данные: количество элементов в массиве (-23).

Ожидаемые выходные данные: сообщение об ошибке и прерывание выполнения программы.

Фактические выходные данные: сообщение об ошибке и прерывание выполнения программы.



```
C:\> D:\Users\stu-pksp117\Desktop\Exam\Exam\bin\Debug\netcorea|
Введите количество элементов в массиве: -23
Arithmetic operation resulted in an overflow.
```

Тестовый пример 2.

Тест-требование 2.

Описание теста: проверить реакцию программы на ввод пользователем некорректных данных при заполнении массива.

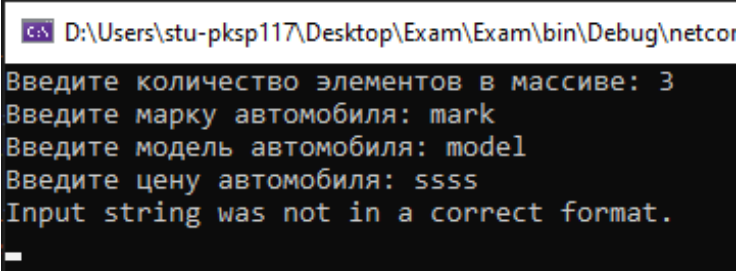
Сценарий теста:

1. Запустить программу.
2. Ввести количество элементов в массиве (3).
3. Ввести марку автомобиля (mark).
4. Ввести модель автомобиля (model).
5. Ввести цену автомобиля (ssss).

Входные данные: цена автомобиля: ssss.

Ожидаемые выходные данные: сообщение об ошибке и прерывание выполнения программы.

Фактические выходные данные: сообщение об ошибке и прерывание выполнения программы.



```
D:\Users\stu-pksp117\Desktop\Exam\Exam\bin\Debug\netco...
Введите количество элементов в массиве: 3
Введите марку автомобиля: mark
Введите модель автомобиля: model
Введите цену автомобиля: ssss
Input string was not in a correct format.
_
```

Тестовый пример 3.

Тест-требование 3.

Описание теста: проверить корректность сортировки введённого массива по сочетанию полей «модель» + «цена».

Сценарий теста:

1. Запустить программу.
2. Ввести количество элементов в массиве (5).
3. Ввести элементы массива (см. входные данные).
4. Открыть txt-файл с результатом.

Входные данные:

1. Марка: Renault

Модель: Logan

Цена: 300000

2. Марка: Chevrolet

Модель: Aveo

Цена: 150000

3. Марка: Lada

Модель: Priora

Цена: 150000

4. Марка: Toyota

Модель: Avensis

Цена: 600000

5. Марка: Chevrolet

Модель: Aveo

Цена: 180000

Ожидаемые выходные данные:

1. Марка: Toyota

Модель: Avensis

Цена: 600000

2. Марка: Chevrolet

Модель: Aveo

Цена: 180000

3. Марка: Chevrolet

Модель: Aveo

Цена: 150000

4. Марка: Renault

Модель: Logan

Цена: 300000

5. Марка: Lada

Модель: Priora

Цена: 150000

Фактические выходные данные:

1. Марка: Toyota

Модель: Avensis

Цена:600000

2. Марка: Chevrolet

Модель: Aveo

Цена:180000

3. Марка: Chevrolet

Модель: Aveo

Цена:150000

4. Марка: Renault

Модель: Logan

Цена:300000

5. Марка: Lada

Модель: Priora

Цена:150000

Тестовый пример 4.

Тест-требование 4.

Описание теста: проверить корректность записи отсортированного массива в файл формата txt.

Сценарий теста:

1. Запустить программу.

2. Ввести количество элементов в массиве (5).

3. Ввести элементы массива (см. входные данные).

4. Открыть txt-файл с результатом.

Входные данные:

1. Марка: Renault

Модель: Logan

Цена: 300000

2. Марка: Chevrolet

Модель: Aveo

Цена: 150000

3. Марка: Lada

Модель: Priora

Цена: 150000

4. Марка: Toyota

Модель: Avensis

Цена: 600000

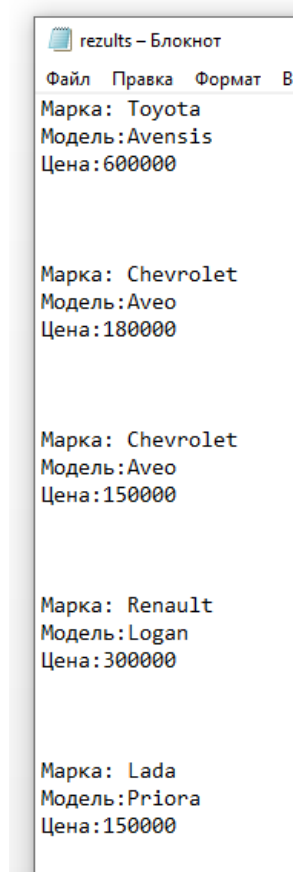
5. Марка: Chevrolet

Модель: Aveo

Цена: 180000

Ожидаемые выходные данные: txt-файл, хранящий отсортированный массив.

Фактические выходные данные: txt-файл, хранящий отсортированный массив.



```
results - Блокнот
Файл  Правка  Формат  В
Марка: Toyota
Модель: Avensis
Цена: 600000

Марка: Chevrolet
Модель: Aveo
Цена: 180000

Марка: Chevrolet
Модель: Aveo
Цена: 150000

Марка: Renault
Модель: Logan
Цена: 300000

Марка: Lada
Модель: Priora
Цена: 150000
```