|  |  |
| --- | --- |
| Gerb-BMSTU_01 | **Министерство науки и высшего образования Российской Федерации**  **Федеральное государственное бюджетное образовательное учреждение**  **высшего образования**  **«Московский государственный технический университет**  **имени Н.Э. Баумана**  **(национальный исследовательский университет)»**  **(МГТУ им. Н.Э. Баумана)** |

ФАКУЛЬТЕТ **Информатика и системы управления**

КАФЕДРА **Компьютерные системы и сети (ИУ6)**

НАПРАВЛЕНИЕ ПОДГОТОВКИ **09.03.04 Программная инженерия**

**Отчет**

|  |  |
| --- | --- |
| **по лабораторной работе №** | 1 |

**Название:**

Введение в курс

**Дисциплина:** Архитектура ЭВМ

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Студент | ИУ7-52Б |  |  | Брянская Е.В. |
|  | (Группа) |  | (Подпись, дата) | (И.О. Фамилия) |
|  |  |  |  |  |
| Преподаватель |  |  |  | Попов А.Ю. |
|  |  |  | (Подпись, дата) | (И.О. Фамилия) |

Москва, 2020

**Задание 1**

Создать хранилище в оперативной памяти для хранения информации о детях.

Необходимо хранить информацию о ребенке: фамилия и возраст.

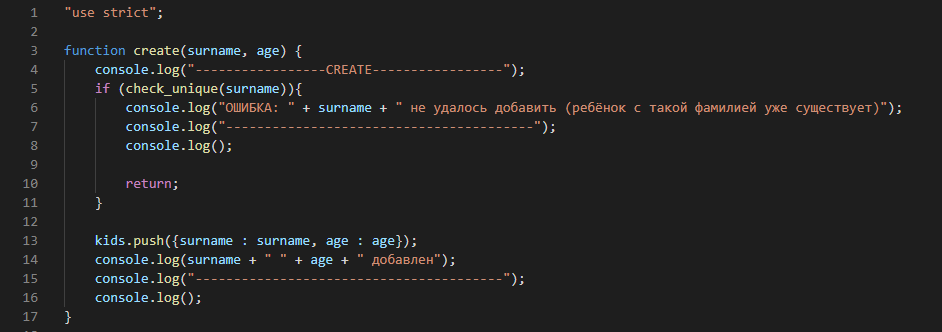
Необходимо обеспечить уникальность фамилий детей.

Реализовать функции:

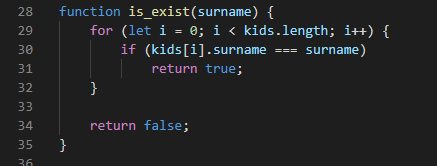
* CREATE READ UPDATE DELETE для детей в хранилище
* Получение среднего возраста детей
* Получение информации о самом старшем ребенке
* Получение информации о детях, возраст которых входит в заданный отрезок
* Получение информации о детях, фамилия которых начинается с заданной буквы
* Получение информации о детях, фамилия которых длиннее заданного количества символов
* Получение информации о детях, фамилия которых начинается с гласной буквы

CREATE READ UPDATE DELETE для детей в хранилище:

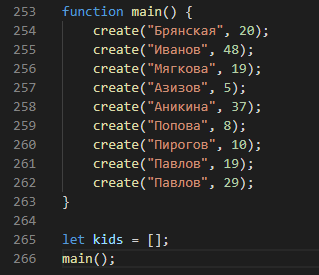
1. CREATE



Реализована вспомогательная функция, которая определяет, есть ли ребёнок с текущей фамилий в хранилище.



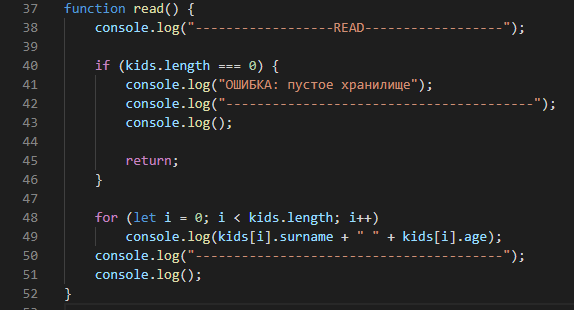
**Тесты:**

****

**Результат:**

****

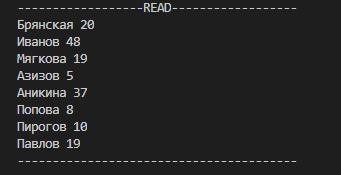
1. READ



**Тесты:**



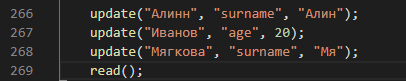
**Результат:**

****

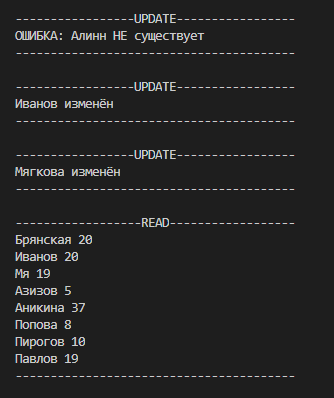
1. UPDATE



**Тесты:**



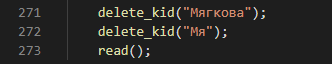
**Результат:**



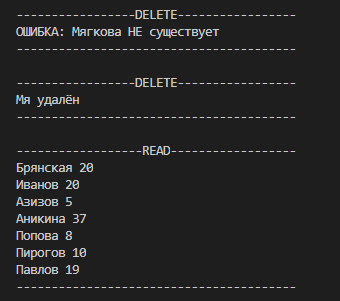
1. DELETE



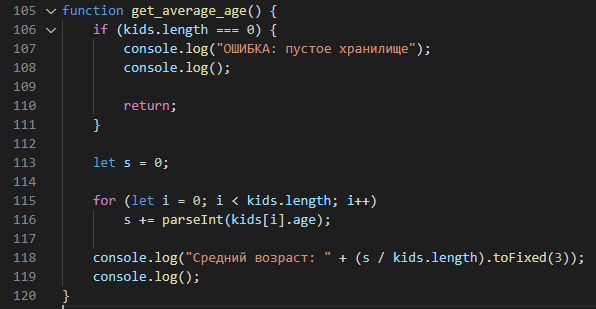
**Тесты:**



**Результат:**

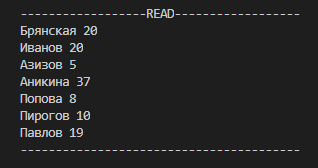


Получение среднего возраста детей



**Тесты:**

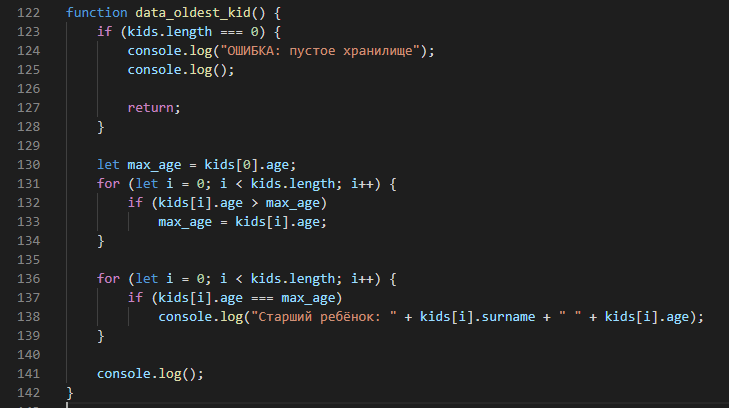




**Результат:**

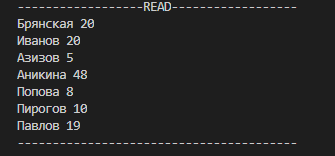
****

Получение информации о самом старшем ребенке



**Тест 1:**

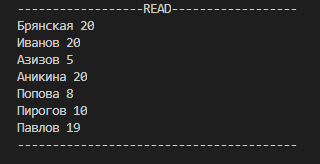




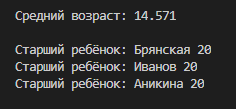
**Результат:**



**Тест 2**



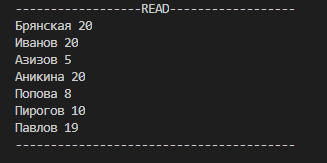
**Результат:**

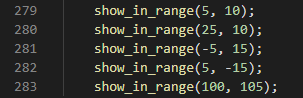


Получение информации о детях, возраст которых входит в заданный отрезок

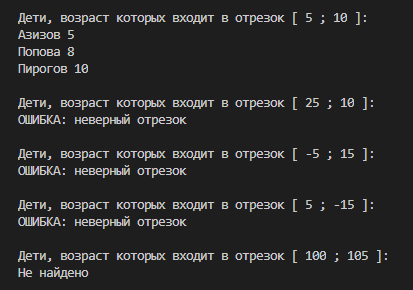


**Тесты:**

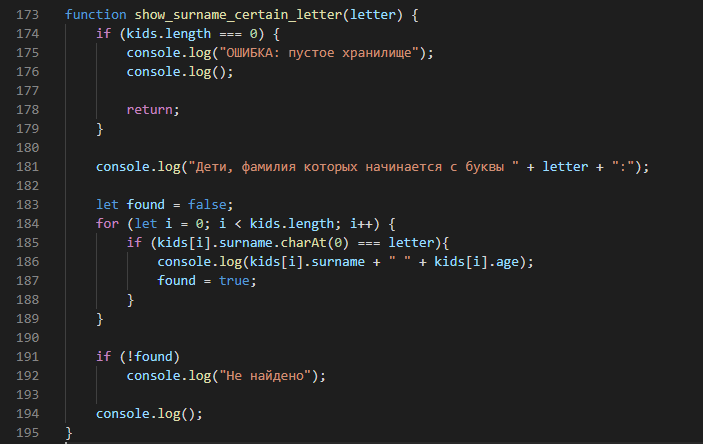
****



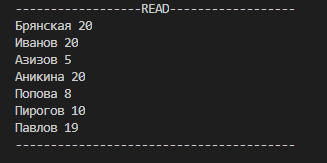
**Результат:**

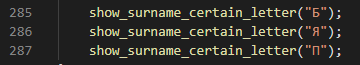


Получение информации о детях, фамилия которых начинается с заданной буквы

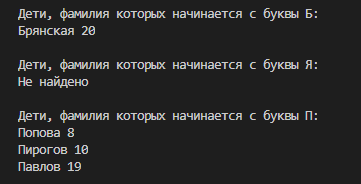


**Тесты:**

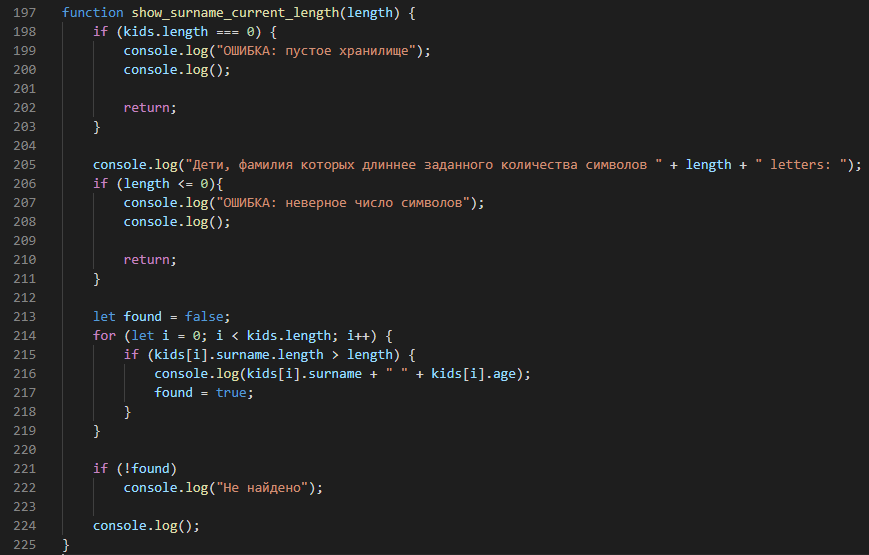
****



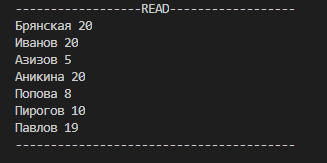
**Результат:**

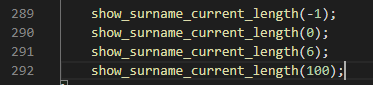


Получение информации о детях, фамилия которых длиннее заданного количества символов

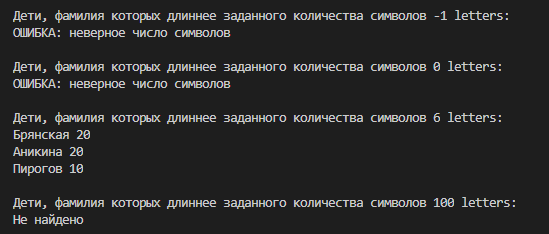


**Тесты:**

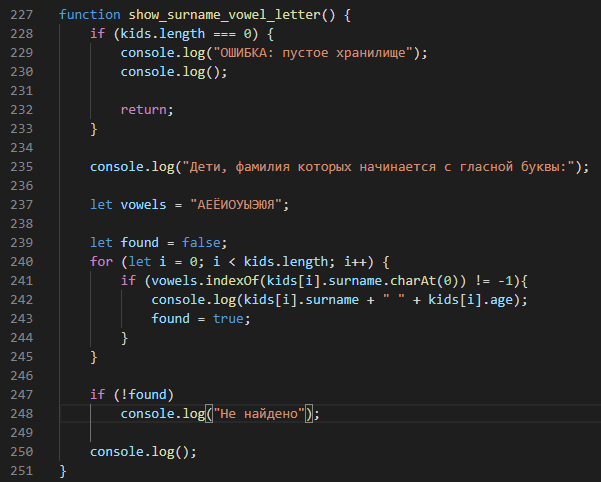
****



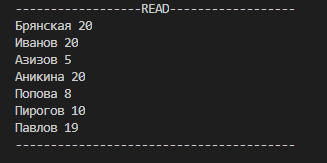
**Результат:**



Получение информации о детях, фамилия которых начинается с гласной буквы

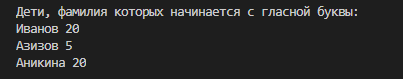


**Тесты:**

****



**Результат:**



**Задание 2**

Создать хранилище в оперативной памяти для хранения информации о студентах.

Необходимо хранить информацию о студенте: название группы, номер студенческого билета, оценки по программированию.

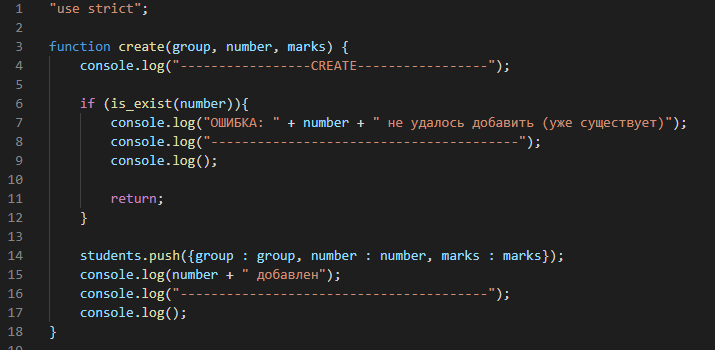
Необходимо обеспечить уникальность номеров студенческих билетов.

Реализовать функции:

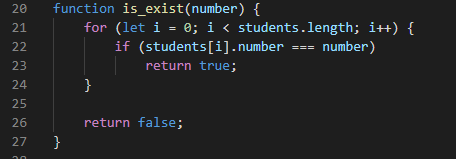
* CREATE READ UPDATE DELETE для студентов в хранилище
* Получение средней оценки заданного студента
* Получение информации о студентах в заданной группе
* Получение студента, у которого наибольшее количество оценок в заданной группе
* Получение студента, у которого нет оценок

CREATE READ UPDATE DELETE для студентов в хранилище

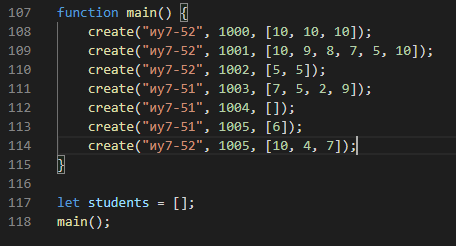
1. CREATE



Реализована вспомогательная функция, которая определяет, есть ли ребёнок с текущей фамилий в хранилище.



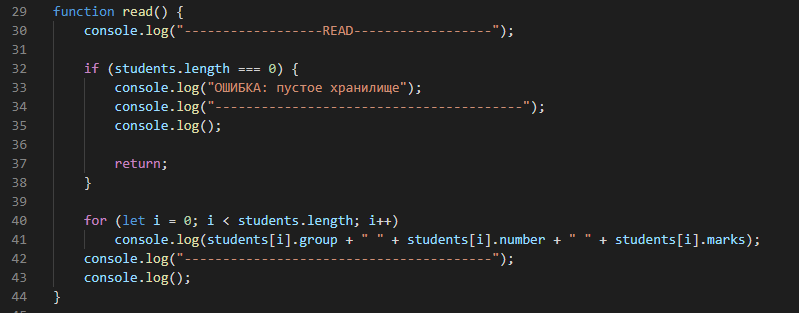
**Тесты:**



**Результат:**



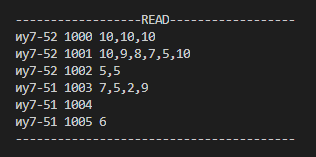
1. READ



**Тесты:**



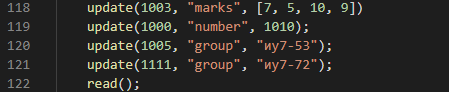
**Результат:**



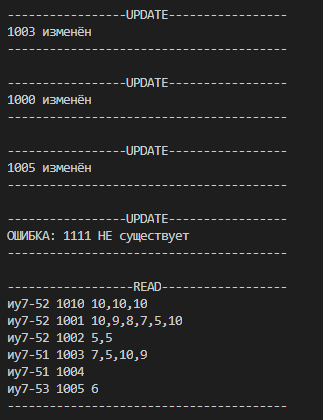
1. UPDATE



**Тесты:**



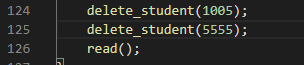
**Результат:**



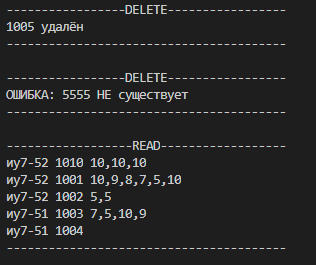
1. DELETE



**Тесты:**



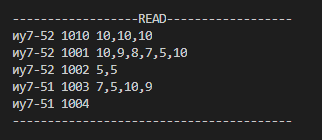
**Результат:**



Получение средней оценки заданного студента

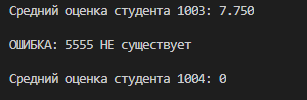


**Тесты:**

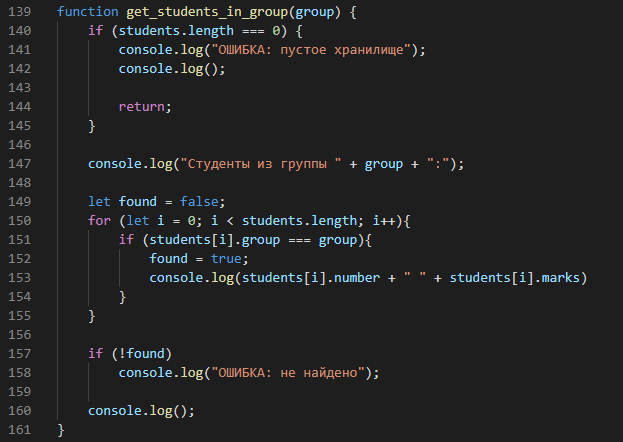




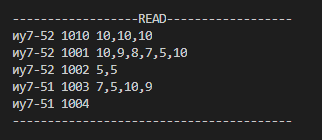
**Результат:**



Получение информации о студентах в заданной группе

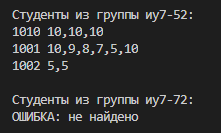


**Тесты:**

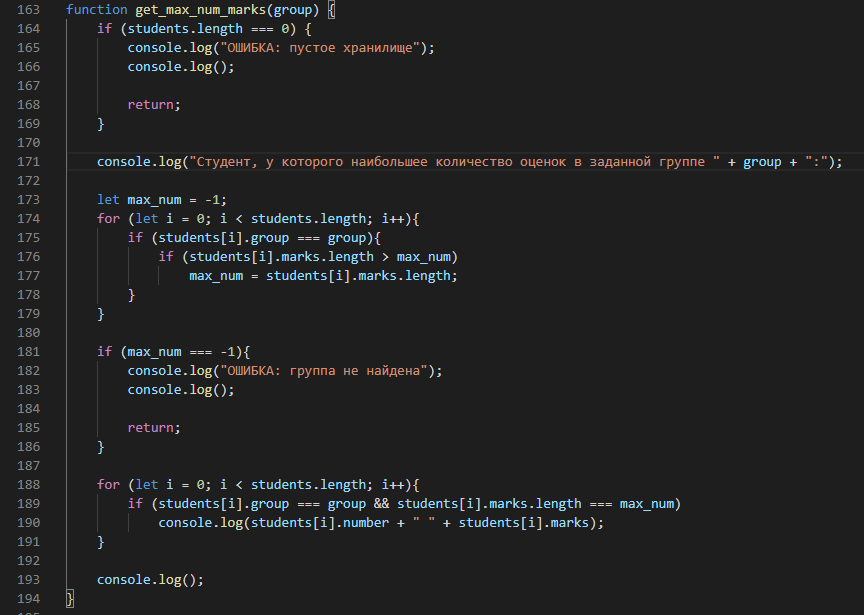




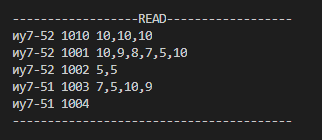
**Результат:**



Получение студента, у которого наибольшее количество оценок в заданной группе

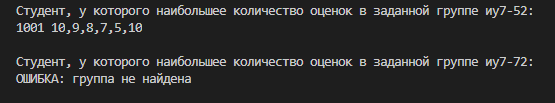


**Тесты:**

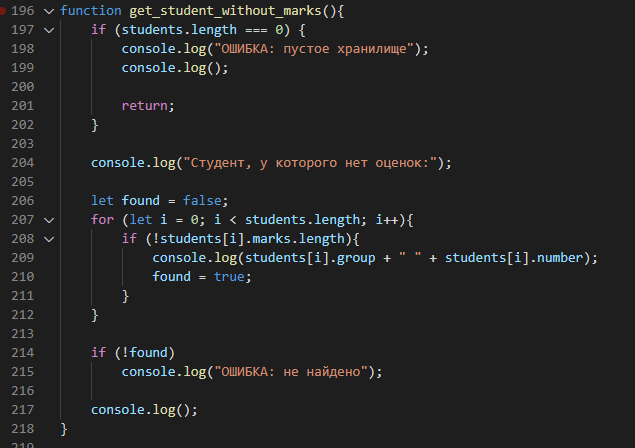




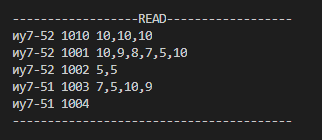
**Результат:**



Получение студента, у которого нет оценок



**Тесты:**





**Результат:**



**Задание 3**

Создать хранилище в оперативной памяти для хранения точек.

Необходимо хранить информацию о точке: имя точки, позиция X и позиция Y.

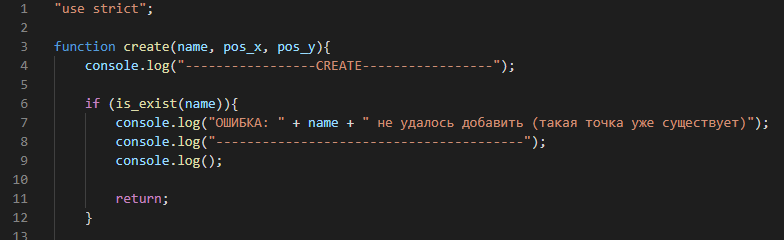
Необходимо обеспечить уникальность имен точек.

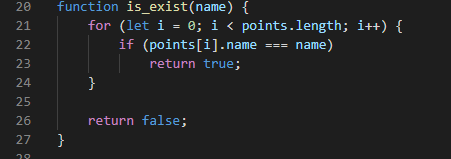
Реализовать функции:

* CREATE READ UPDATE DELETE для точек в хранилище
* Получение двух точек, между которыми наибольшее расстояние
* Получение точек, находящихся от заданной точки на расстоянии, не превышающем заданную константу
* Получение точек, находящихся выше / ниже / правее / левее заданной оси координат
* Получение точек, входящих внутрь заданной прямоугольной зоны

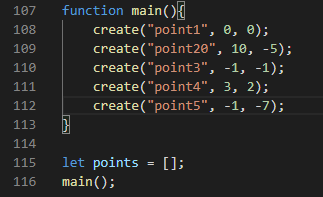
CREATE READ UPDATE DELETE для точек в хранилище

1. CREATE

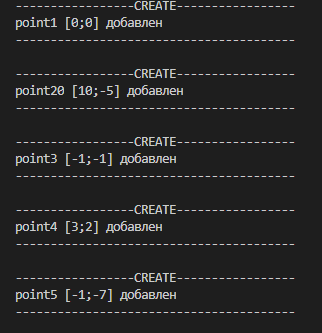




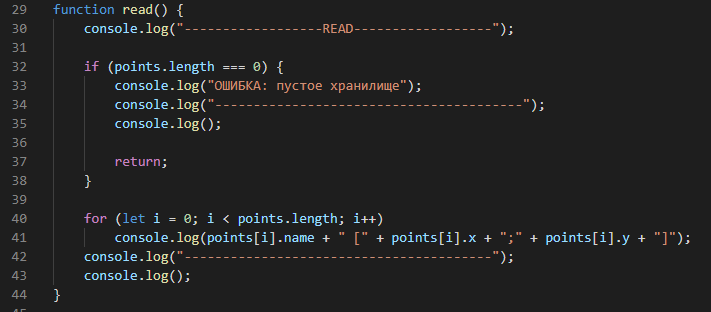
**Тесты:**

****

**Результат:**

****

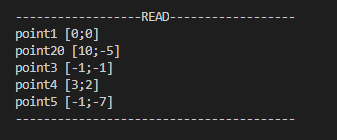
1. READ



**Тесты:**

****

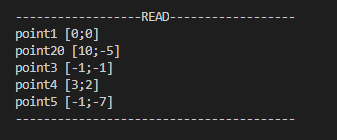
**Результат:**

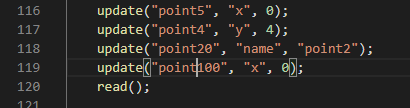


1. UPDATE

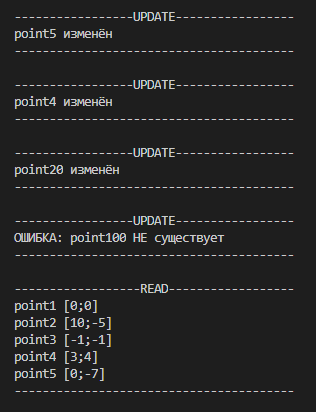


**Тесты:**



****

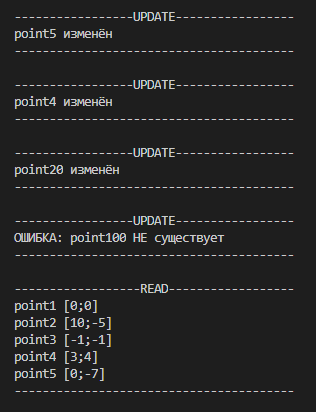
**Результат:**

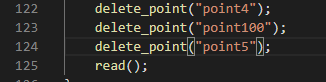


1. DELETE

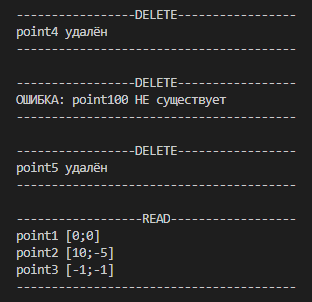


**Тесты:**



****

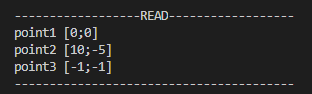
**Результат:**

****

Получение двух точек, между которыми наибольшее расстояние



**Тесты:**

****

****

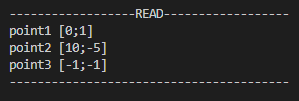
**Результат:**

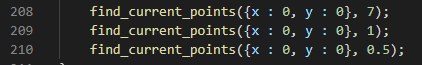
****

Получение точек, находящихся от заданной точки на расстоянии, не превышающем заданную константу

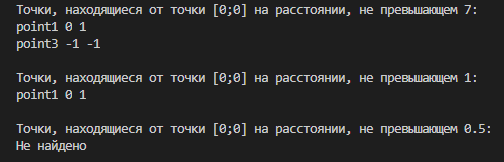


**Тесты:**

****

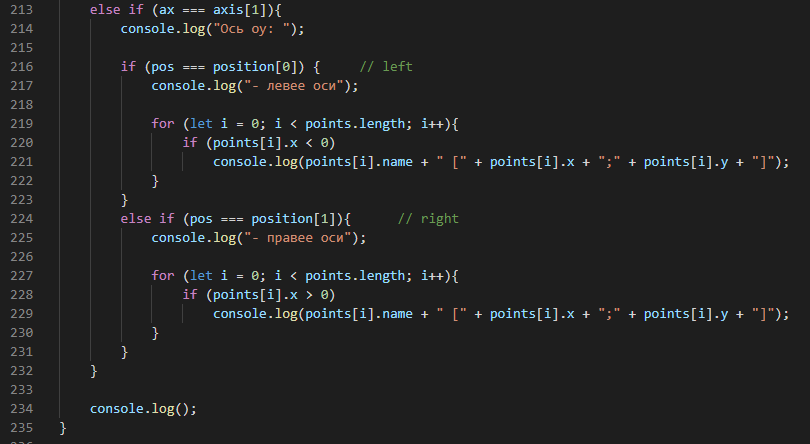
****

**Результат:**

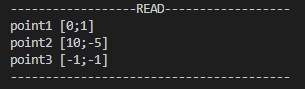
****

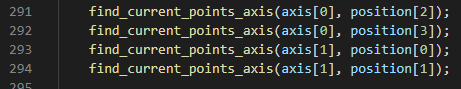
Получение точек, находящихся выше / ниже / правее / левее заданной оси координат

****

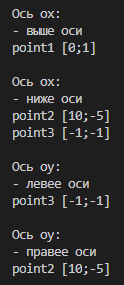
****

**Тесты:**

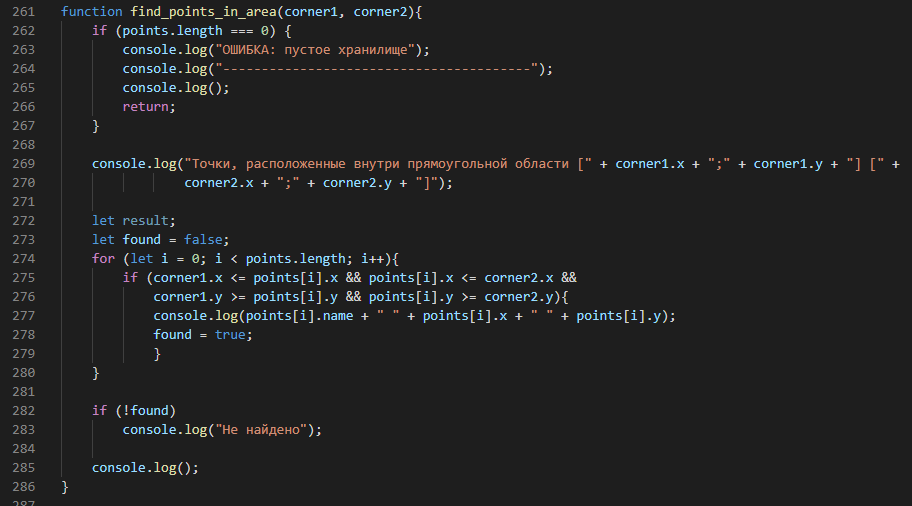
****

****

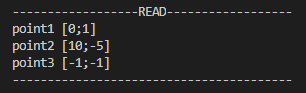
**Результат:**

****

Получение точек, входящих внутрь заданной прямоугольной зоны

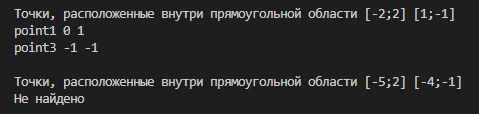


**Тесты:**

****

****

**Результат:**



**Вторая часть лабораторной работы**

### **Задание 1**

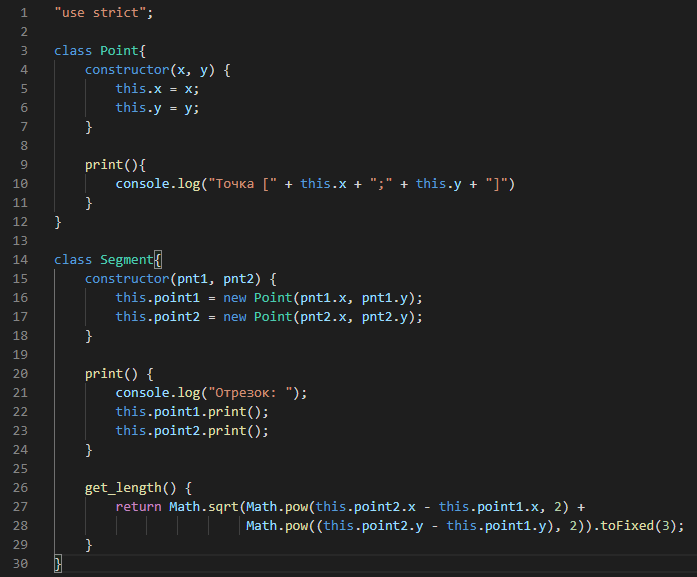
Создать класс Точка.

Добавить классу Точка метод инициализации полей и метод вывода полей на экран

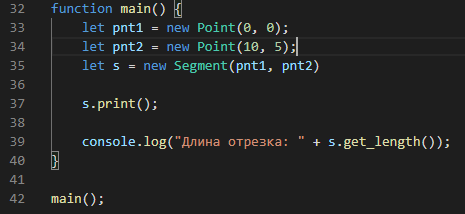
Создать класс Отрезок.

У класса Отрезок должны быть поля, являющиеся экземплярами класса Точка.

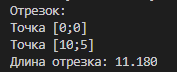
Добавить классу Отрезок метод инициализации полей, метод вывода информации о полях на экран, а также метод получения длины отрезка.



**Тесты:**



**Результат:**



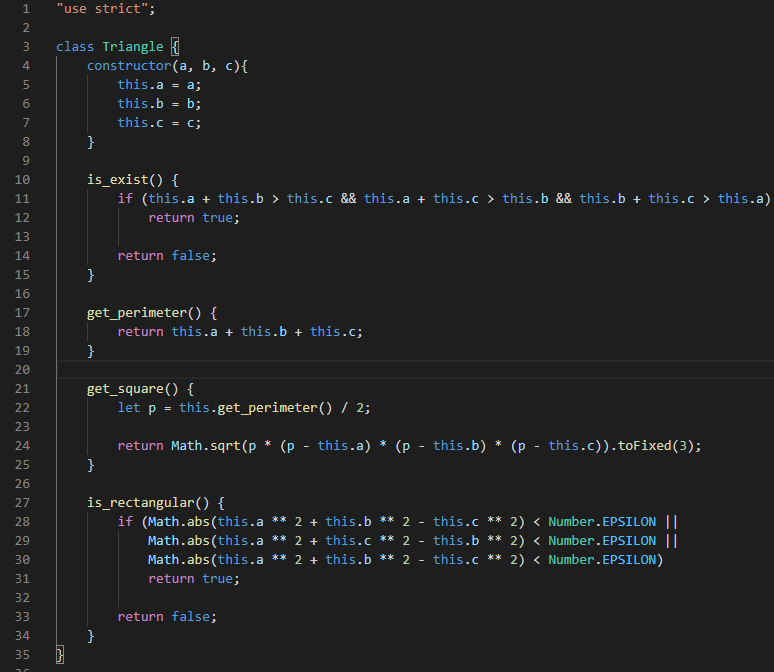
### **Задание 2**

Создать класс Треугольник.

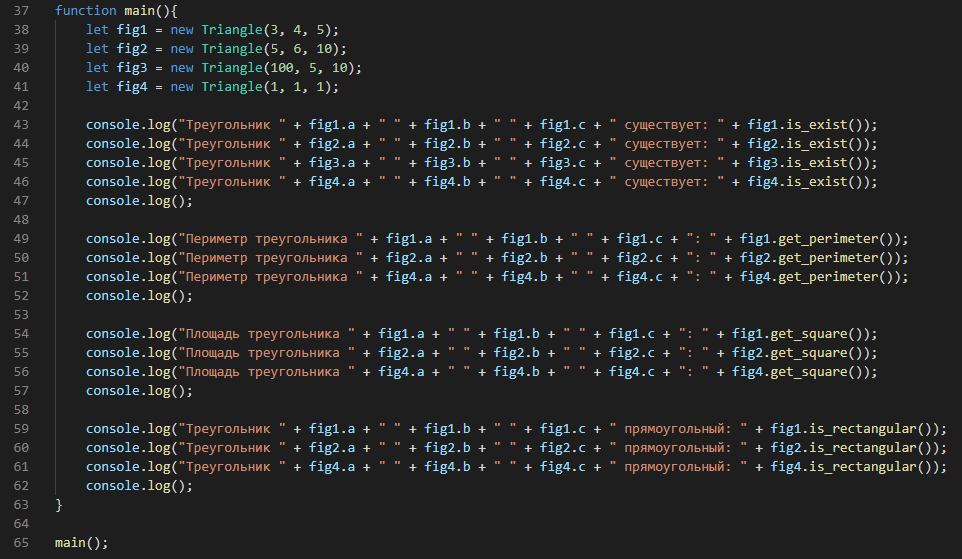
Класс Треугольник должен иметь поля, хранящие длины сторон треугольника.

Реализовать следующие методы:

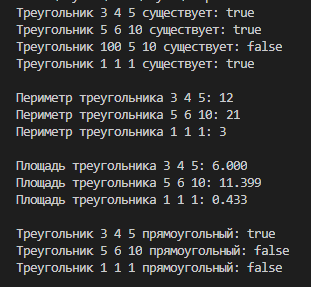
* Метод инициализации полей
* Метод проверки возможности существования треугольника с такими сторонами
* Метод получения периметра треугольника
* Метод получения площади треугольника
* Метод для проверки факта: является ли треугольник прямоугольным



**Тесты:**

****

**Результаты:**

****

### **Задание 3**

Реализовать программу, в которой происходят следующие действия:

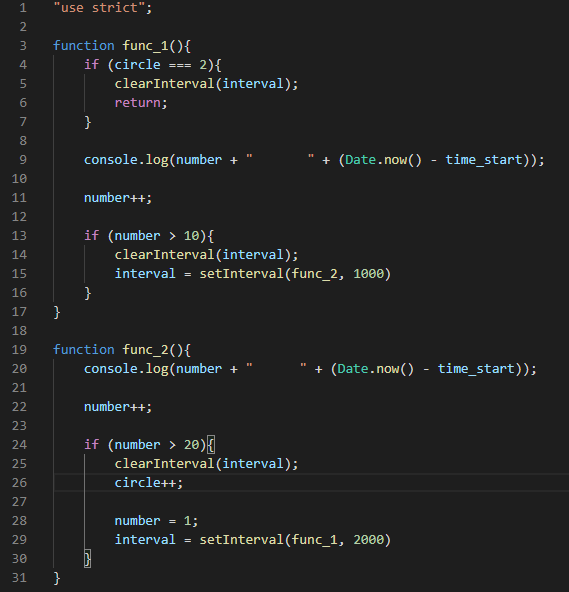
Происходит вывод целых чисел от 1 до 10 с задержками в 2 секунды.

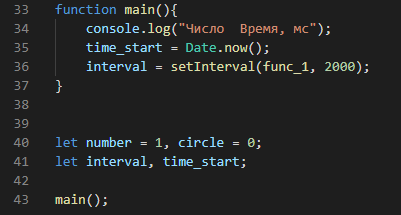
После этого происходит вывод от 11 до 20 с задержками в 1 секунду.

Потом опять происходит вывод чисел от 1 до 10 с задержками в 2 секунды.

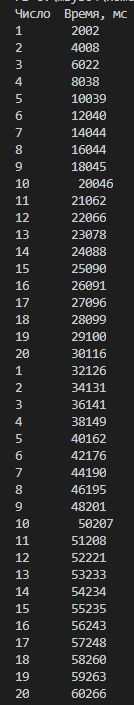
После этого происходит вывод от 11 до 20 с задержками в 1 секунду.

Это должно происходить циклически.





**Результат:**

****

Примечание**:** второй столбик – время, которое прошло с момента запуска программы