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**

О Т Ч Е Т

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

Цель: изучить основы синтаксиса языка JS, научиться использовать массивы, объекты, классы для решения задач. Исследовать особенности функции setInterval и использовать её для решения конкретной задачи.

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

```
"use strict";

function create(surname, age) {
    console.log("-----------------CREATE-----------------");
    if (check_unique(surname)){
        console.log("ОШИБКА: " + surname + " не удалось добавить (ребёнок с такой фамилией уже существует)");
        console.log("----------------------------------------");
        console.log();

        return;
    }

    kids.push({surname : surname, age : age});
    console.log(surname + " " + age + " добавлен");
    console.log("----------------------------------------");
    console.log();
}
```

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

```
function is_exist(surname) {
    for (let i = 0; i < kids.length; i++) {
        if (kids[i].surname === surname)
            return true;
    }

    return false;
}
```

Тесты:

```
function main() {
    create("Брянская", 20);
    create("Иванов", 48);
    create("Мягкова", 19);
    create("Азизов", 5);
    create("Аникина", 37);
    create("Попова", 8);
    create("Пирогов", 10);
    create("Павлов", 19);
    create("Павлов", 29);
}

let kids = [];
main();
```

Результат:

```
----------CREATE----------
Брянская 20 добавлен
----------------------------------------

----------CREATE----------
Иванов 48 добавлен
----------------------------------------

----------CREATE----------
Мягкова 19 добавлен
----------------------------------------

----------CREATE----------
Азизов 5 добавлен
----------------------------------------

----------CREATE----------
Аникина 37 добавлен
----------------------------------------

----------CREATE----------
Попова 8 добавлен
----------------------------------------

----------CREATE----------
Пирогов 10 добавлен
----------------------------------------

----------CREATE----------
Павлов 19 добавлен
----------------------------------------

----------CREATE----------
ОШИБКА: Павлов не удалось добавить (ребёнок с такой фамилией уже существует)
----------------------------------------
```

2) READ

```
37  function read() {
38      console.log("----------READ----------");
39
40      if (kids.length === 0) {
41          console.log("ОШИБКА: пустое хранилище");
42          console.log("----------------------------------------");
43          console.log();
44
45          return;
46      }
47
48      for (let i = 0; i < kids.length; i++)
49          console.log(kids[i].surname + " " + kids[i].age);
50      console.log("----------------------------------------");
51      console.log();
52  }
```

Тесты:

```
264     read();
```

Результат:

```
------------------READ------------------
Брянская 20
Иванов 48
Мягкова 19
Азизов 5
Аникина 37
Попова 8
Пирогов 10
Павлов 19
----------------------------------------
```

3) UPDATE

```
54  function update(surname, key, value) {
55      console.log("----------------UPDATE----------------");
56
57      if (kids.length === 0) {
58          console.log("ОШИБКА: пустое хранилище");
59          console.log("----------------------------------------");
60          console.log();
61
62          return;
63      }
64
65      if (!is_exist(surname)) {
66          console.log("ОШИБКА: " + surname + " НЕ существует");
67          console.log("----------------------------------------");
68          console.log();
69
70          return;
71      }
72
73      for (let i = 0; i < kids.length; i++) {
74          if (kids[i].surname === surname) {
75              kids[i][key] = value;
76              break;
77          }
78      }
79
80      console.log(surname + " изменён");
81      console.log("----------------------------------------");
82      console.log();
83  }
84
```

Тесты:

```
266     update("Алинн", "surname", "Алин");
267     update("Иванов", "age", 20);
268     update("Мягкова", "surname", "Мя");
269     read();
```

Результат:

```
-----------------UPDATE-----------------
ОШИБКА: Алинн НЕ существует
----------------------------------------

-----------------UPDATE-----------------
Иванов изменён
----------------------------------------

-----------------UPDATE-----------------
Мягкова изменён
----------------------------------------

------------------READ------------------
Брянская 20
Иванов 20
Мя 19
Азизов 5
Аникина 37
Попова 8
Пирогов 10
Павлов 19
----------------------------------------
```

4) DELETE

```
85  function delete_kid(surname) {
86      console.log("-----------------DELETE-----------------");
87
88      if (kids.length === 0) {
89          console.log("ОШИБКА: пустое хранилище");
90          console.log("----------------------------------------");
91          console.log();
92
93          return;
94      }
95
96      if (!is_exist(surname)) {
97          console.log("ОШИБКА: " + surname + " НЕ существует");
98          console.log("----------------------------------------");
99          console.log();
100
101         return;
102     }
103
104     for (let i = 0; i < kids.length; i++) {
105         if (kids[i].surname === surname)
106             kids.splice(i, 1);
107     }
108
109     console.log(surname + " удалён");
110     console.log("----------------------------------------");
111     console.log();
112 }
```

Тесты:

```
271     delete_kid("Мягкова");
272     delete_kid("Мя");
273     read();
```

Результат:

```
----------------DELETE----------------
ОШИБКА: Мягкова НЕ существует
--------------------------------------

----------------DELETE----------------
Мя удалён
--------------------------------------

----------------READ----------------
Брянская 20
Иванов 20
Азизов 5
Аникина 37
Попова 8
Пирогов 10
Павлов 19
--------------------------------------
```

Получение среднего возраста детей

```
105 function get_average_age() {
106     if (kids.length === 0) {
107         console.log("ОШИБКА: пустое хранилище");
108         console.log();
109 
110         return;
111     }
112 
113     let s = 0;
114 
115     for (let i = 0; i < kids.length; i++)
116         s += parseInt(kids[i].age);
117 
118     console.log("Средний возраст: " + (s / kids.length).toFixed(3));
119     console.log();
120 }
```

Тесты:

```
275     get_average_age();
```

```
----------------READ----------------
Брянская 20
Иванов 20
Азизов 5
Аникина 37
Попова 8
Пирогов 10
Павлов 19
--------------------------------------
```

Результат:

```
Средний возраст: 17.000
```

Получение информации о самом старшем ребенке

```
122  function data_oldest_kid() {
123      if (kids.length === 0) {
124          console.log("ОШИБКА: пустое хранилище");
125          console.log();
126
127          return;
128      }
129
130      let max_age = kids[0].age;
131      for (let i = 0; i < kids.length; i++) {
132          if (kids[i].age > max_age)
133              max_age = kids[i].age;
134      }
135
136      for (let i = 0; i < kids.length; i++) {
137          if (kids[i].age === max_age)
138              console.log("Старший ребёнок: " + kids[i].surname + " " + kids[i].age);
139      }
140
141      console.log();
142  }
```

Тест 1:

```
277      data_oldest_kid();
```

```
------------------READ------------------
Брянская 20
Иванов 20
Азизов 5
Аникина 48
Попова 8
Пирогов 10
Павлов 19
----------------------------------------
```

Результат:

```
Старший ребёнок: Аникина 48
```

Тест 2

```
------------------READ------------------
Брянская 20
Иванов 20
Азизов 5
Аникина 20
Попова 8
Пирогов 10
Павлов 19
----------------------------------------
```

Результат:

```
Старший ребёнок: Брянская 20
Старший ребёнок: Иванов 20
Старший ребёнок: Аникина 20
```

Получение информации о детях, возраст которых входит в заданный отрезок

```
function show_in_range(start, end) {
    if (kids.length === 0) {
        console.log("ОШИБКА: пустое хранилище");
        console.log();

        return;
    }

    console.log("Дети, возраст которых входит в отрезок [ " + start + " ; " + end + " ]:");
    if (start > end || start < 0){
        console.log("ОШИБКА: неверный отрезок");
        console.log();

        return;
    }

    let found = false;
    for (let i = 0; i < kids.length; i++)
        if (start <= kids[i].age && kids[i].age <= end) {
            console.log(kids[i].surname + " " + kids[i].age);
            found = true;
        }

    if (!found)
        console.log("Не найдено");

    console.log();
}
```

Тесты:

```
-----------------READ-----------------
Брянская 20
Иванов 20
Азизов 5
Аникина 20
Попова 8
Пирогов 10
Павлов 19
--------------------------------------
```

```
    show_in_range(5, 10);
    show_in_range(25, 10);
    show_in_range(-5, 15);
    show_in_range(5, -15);
    show_in_range(100, 105);
```

Результат:

```
Дети, возраст которых входит в отрезок [ 5 ; 10 ]:
Азизов 5
Попова 8
Пирогов 10

Дети, возраст которых входит в отрезок [ 25 ; 10 ]:
ОШИБКА: неверный отрезок

Дети, возраст которых входит в отрезок [ -5 ; 15 ]:
ОШИБКА: неверный отрезок

Дети, возраст которых входит в отрезок [ 5 ; -15 ]:
ОШИБКА: неверный отрезок

Дети, возраст которых входит в отрезок [ 100 ; 105 ]:
Не найдено
```

Получение информации о детях, фамилия которых начинается с заданной буквы

```
173  function show_surname_certain_letter(letter) {
174      if (kids.length === 0) {
175          console.log("ОШИБКА: пустое хранилище");
176          console.log();
177
178          return;
179      }
180
181      console.log("Дети, фамилия которых начинается с буквы " + letter + ":");
182
183      let found = false;
184      for (let i = 0; i < kids.length; i++) {
185          if (kids[i].surname.charAt(0) === letter){
186              console.log(kids[i].surname + " " + kids[i].age);
187              found = true;
188          }
189      }
190
191      if (!found)
192          console.log("Не найдено");
193
194      console.log();
195  }
```

Тесты:

```
------------------READ------------------
Брянская 20
Иванов 20
Азизов 5
Аникина 20
Попова 8
Пирогов 10
Павлов 19
----------------------------------------
```

```
285      show_surname_certain_letter("Б");
286      show_surname_certain_letter("Я");
287      show_surname_certain_letter("П");
```

Результат:

```
Дети, фамилия которых начинается с буквы Б:
Брянская 20

Дети, фамилия которых начинается с буквы Я:
Не найдено

Дети, фамилия которых начинается с буквы П:
Попова 8
Пирогов 10
Павлов 19
```

Получение информации о детях, фамилия которых длиннее заданного количества символов

```
function show_surname_current_length(length) {
    if (kids.length === 0) {
        console.log("ОШИБКА: пустое хранилище");
        console.log();

        return;
    }

    console.log("Дети, фамилия которых длиннее заданного количества символов " + length + " letters: ");
    if (length <= 0){
        console.log("ОШИБКА: неверное число символов");
        console.log();

        return;
    }

    let found = false;
    for (let i = 0; i < kids.length; i++) {
        if (kids[i].surname.length > length) {
            console.log(kids[i].surname + " " + kids[i].age);
            found = true;
        }
    }

    if (!found)
        console.log("Не найдено");

    console.log();
}
```

Тесты:

```
-----------------READ-----------------
Брянская 20
Иванов 20
Азизов 5
Аникина 20
Попова 8
Пирогов 10
Павлов 19
--------------------------------------
```

```
    show_surname_current_length(-1);
    show_surname_current_length(0);
    show_surname_current_length(6);
    show_surname_current_length(100);
```

Результат:

```
Дети, фамилия которых длиннее заданного количества символов -1 letters:
ОШИБКА: неверное число символов

Дети, фамилия которых длиннее заданного количества символов 0 letters:
ОШИБКА: неверное число символов

Дети, фамилия которых длиннее заданного количества символов 6 letters:
Брянская 20
Аникина 20
Пирогов 10

Дети, фамилия которых длиннее заданного количества символов 100 letters:
Не найдено
```

Получение информации о детях, фамилия которых начинается с гласной буквы

```
227  function show_surname_vowel_letter() {
228      if (kids.length === 0) {
229          console.log("ОШИБКА: пустое хранилище");
230          console.log();
231
232          return;
233      }
234
235      console.log("Дети, фамилия которых начинается с гласной буквы:");
236
237      let vowels = "АЕЁИОУЫЭЮЯ";
238
239      let found = false;
240      for (let i = 0; i < kids.length; i++) {
241          if (vowels.indexOf(kids[i].surname.charAt(0)) != -1){
242              console.log(kids[i].surname + " " + kids[i].age);
243              found = true;
244          }
245      }
246
247      if (!found)
248          console.log("Не найдено");
249
250      console.log();
251  }
```

Тесты:

```
-----------------READ-----------------
Брянская 20
Иванов 20
Азизов 5
Аникина 20
Попова 8
Пирогов 10
Павлов 19
--------------------------------------
```

```
294      show_surname_vowel_letter();
```

Результат:

```
Дети, фамилия которых начинается с гласной буквы:
Иванов 20
Азизов 5
Аникина 20
```

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

```
"use strict";

function create(group, number, marks) {
    console.log("-----------------CREATE-----------------");

    if (is_exist(number)){
        console.log("ОШИБКА: " + number + " не удалось добавить (уже существует)");
        console.log("----------------------------------------");
        console.log();

        return;
    }

    students.push({group : group, number : number, marks : marks});
    console.log(number + " добавлен");
    console.log("----------------------------------------");
    console.log();
}
```

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

```
20  function is_exist(number) {
21      for (let i = 0; i < students.length; i++) {
22          if (students[i].number === number)
23              return true;
24      }
25
26      return false;
27  }
```

Тесты:

```
107 function main() {
108     create("иу7-52", 1000, [10, 10, 10]);
109     create("иу7-52", 1001, [10, 9, 8, 7, 5, 10]);
110     create("иу7-52", 1002, [5, 5]);
111     create("иу7-51", 1003, [7, 5, 2, 9]);
112     create("иу7-51", 1004, []);
113     create("иу7-51", 1005, [6]);
114     create("иу7-52", 1005, [10, 4, 7]);
115 }
116
117 let students = [];
118 main();
```

Результат:

```
-----------------CREATE-----------------
1000 добавлен
----------------------------------------

-----------------CREATE-----------------
1001 добавлен
----------------------------------------

-----------------CREATE-----------------
1002 добавлен
----------------------------------------

-----------------CREATE-----------------
1003 добавлен
----------------------------------------

-----------------CREATE-----------------
1004 добавлен
----------------------------------------

-----------------CREATE-----------------
1005 добавлен
----------------------------------------

-----------------CREATE-----------------
ОШИБКА: 1005 не удалось добавить (уже существует)
----------------------------------------
```

2) READ

```
29  function read() {
30      console.log("------------------READ------------------");
31
32      if (students.length === 0) {
33          console.log("ОШИБКА: пустое хранилище");
34          console.log("----------------------------------------");
35          console.log();
36
37          return;
38      }
39
40      for (let i = 0; i < students.length; i++)
41          console.log(students[i].group + " " + students[i].number + " " + students[i].marks);
42      console.log("----------------------------------------");
43      console.log();
44  }
```

Тесты:

```
116     read();
```

Результат:

```
------------------READ------------------
иу7-52 1000 10,10,10
иу7-52 1001 10,9,8,7,5,10
иу7-52 1002 5,5
иу7-51 1003 7,5,2,9
иу7-51 1004
иу7-51 1005 6
----------------------------------------
```

3) UPDATE

```
function update(number, key, value) {
    console.log("-----------------UPDATE-----------------");

    if (students.length === 0) {
        console.log("ОШИБКА: пустое хранилище");
        console.log("----------------------------------------");
        console.log();

        return;
    }

    if (!is_exist(number)) {
        console.log("ОШИБКА: " + number + " НЕ существует");
        console.log("----------------------------------------");
        console.log();

        return;
    }

    for (let i = 0; i < students.length; i++) {
        if (students[i].number === number) {
            students[i][key] = value;
            break;
        }
    }

    console.log(number + " изменён");
    console.log("----------------------------------------");
    console.log();
}

```

Тесты:

```
    update(1003, "marks", [7, 5, 10, 9])
    update(1000, "number", 1010);
    update(1005, "group", "иу7-53");
    update(1111, "group", "иу7-72");
    read();
```

Результат:

```
-----------------UPDATE-----------------
1003 изменён
----------------------------------------

-----------------UPDATE-----------------
1000 изменён
----------------------------------------

-----------------UPDATE-----------------
1005 изменён
----------------------------------------

-----------------UPDATE-----------------
ОШИБКА: 1111 НЕ существует
----------------------------------------

------------------READ------------------
иу7-52 1010 10,10,10
иу7-52 1001 10,9,8,7,5,10
иу7-52 1002 5,5
иу7-51 1003 7,5,10,9
иу7-51 1004
иу7-53 1005 6
----------------------------------------
```

4) DELETE

```
77  function delete_student(number) {
78      console.log("-----------------DELETE-----------------");
79
80      if (students.length === 0) {
81          console.log("ОШИБКА: пустое хранилище");
82          console.log("----------------------------------------");
83          console.log();
84
85          return;
86      }
87
88      if (!is_exist(number)) {
89          console.log("ОШИБКА: " + number + " НЕ существует");
90          console.log("----------------------------------------");
91          console.log();
92
93          return;
94      }
95
96      for (let i = 0; i < students.length; i++) {
97          if (students[i].number === number)
98              students.splice(i, 1);
99      }
100
101     console.log(number + " удалён");
102     console.log("----------------------------------------");
103     console.log();
104 }
```

Тесты:

```
    delete_student(1005);
    delete_student(5555);
    read();
```

Результат:

```
----------------DELETE----------------
1005 удалён
--------------------------------------

----------------DELETE----------------
ОШИБКА: 5555 НЕ существует
--------------------------------------

-----------------READ-----------------
иу7-52 1010 10,10,10
иу7-52 1001 10,9,8,7,5,10
иу7-52 1002 5,5
иу7-51 1003 7,5,10,9
иу7-51 1004
--------------------------------------
```

Получение средней оценки заданного студента

```
function get_average_mark(student) {
    if (students.length === 0) {
        console.log("ОШИБКА: пустое хранилище");
        console.log();

        return;
    }

    if (!is_exist(student)) {
        console.log("ОШИБКА: " + student + " НЕ существует");
        console.log("--------------------------------------");
        console.log();

        return;
    }

    let s = 0, count = 0;
    for (let i = 0; i < students.length; i++) {
        if (students[i].number === student) {
            for (let j = 0; j < students[i].marks.length; j++) {
                s += students[i].marks[j];
                count++;
            }
            break;
        }
    }

    if (s)
        console.log("Средний оценка студента " + student + ": " + (s / count).toFixed(3));
    else
        console.log("Средний оценка студента " + student + ": " + s);
    console.log();
}
```

Тесты:

```
------------------READ------------------
иу7-52 1010 10,10,10
иу7-52 1001 10,9,8,7,5,10
иу7-52 1002 5,5
иу7-51 1003 7,5,10,9
иу7-51 1004
----------------------------------------
```

```
161     get_average_mark(1003);
162     get_average_mark(5555);
163     get_average_mark(1004);
```

Результат:

```
Средний оценка студента 1003: 7.750

ОШИБКА: 5555 НЕ существует

Средний оценка студента 1004: 0
```

Получение информации о студентах в заданной группе

```
139 function get_students_in_group(group) {
140     if (students.length === 0) {
141         console.log("ОШИБКА: пустое хранилище");
142         console.log();
143 
144         return;
145     }
146 
147     console.log("Студенты из группы " + group + ":");
148 
149     let found = false;
150     for (let i = 0; i < students.length; i++){
151         if (students[i].group === group){
152             found = true;
153             console.log(students[i].number + " " + students[i].marks)
154         }
155     }
156 
157     if (!found)
158         console.log("ОШИБКА: не найдено");
159 
160     console.log();
161 }
```

Тесты:

```
------------------READ-----------------
иу7-52 1010 10,10,10
иу7-52 1001 10,9,8,7,5,10
иу7-52 1002 5,5
иу7-51 1003 7,5,10,9
иу7-51 1004
---------------------------------------
```

```
193     get_students_in_group("иу7-52");
194     get_students_in_group("иу7-72");
```

Результат:

```
Студенты из группы иу7-52:
1010 10,10,10
1001 10,9,8,7,5,10
1002 5,5

Студенты из группы иу7-72:
ОШИБКА: не найдено
```

Получение студента, у которого наибольшее количество оценок в заданной группе

```
163 function get_max_num_marks(group) {
164     if (students.length === 0) {
165         console.log("ОШИБКА: пустое хранилище");
166         console.log();
167 
168         return;
169     }
170 
171     console.log("Студент, у которого наибольшее количество оценок в заданной группе " + group + ":");
172 
173     let max_num = -1;
174     for (let i = 0; i < students.length; i++){
175         if (students[i].group === group){
176             if (students[i].marks.length > max_num)
177                 max_num = students[i].marks.length;
178         }
179     }
180 
181     if (max_num === -1){
182         console.log("ОШИБКА: группа не найдена");
183         console.log();
184 
185         return;
186     }
187 
188     for (let i = 0; i < students.length; i++){
189         if (students[i].group === group && students[i].marks.length === max_num)
190             console.log(students[i].number + " " + students[i].marks);
191     }
192 
193     console.log();
194 }
```

Тесты:

```
------------------READ------------------
иу7-52 1010 10,10,10
иу7-52 1001 10,9,8,7,5,10
иу7-52 1002 5,5
иу7-51 1003 7,5,10,9
иу7-51 1004
----------------------------------------
```

```
225     get_max_num_marks("иу7-52");
226     get_max_num_marks("иу7-72");
```

Результат:

```
Студент, у которого наибольшее количество оценок в заданной группе иу7-52:
1001 10,9,8,7,5,10

Студент, у которого наибольшее количество оценок в заданной группе иу7-72:
ОШИБКА: группа не найдена
```

Получение студента, у которого нет оценок

```
196 function get_student_without_marks(){
197     if (students.length === 0) {
198         console.log("ОШИБКА: пустое хранилище");
199         console.log();
200 
201         return;
202     }
203 
204     console.log("Студент, у которого нет оценок:");
205 
206     let found = false;
207     for (let i = 0; i < students.length; i++){
208         if (!students[i].marks.length){
209             console.log(students[i].group + " " + students[i].number);
210             found = true;
211         }
212     }
213 
214     if (!found)
215         console.log("ОШИБКА: не найдено");
216 
217     console.log();
218 }
```

Тесты:

```
-----------------READ-----------------
иу7-52 1010 10,10,10
иу7-52 1001 10,9,8,7,5,10
иу7-52 1002 5,5
иу7-51 1003 7,5,10,9
иу7-51 1004
--------------------------------------
```

```
252     get_student_without_marks();
```

Результат:

```
Студент, у которого нет оценок:
иу7-51 1004
```

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

```
"use strict";

function create(name, pos_x, pos_y){
    console.log("-----------------CREATE-----------------");

    if (is_exist(name)){
        console.log("ОШИБКА: " + name + " не удалось добавить (такая точка уже существует)");
        console.log("----------------------------------------");
        console.log();

        return;
    }
```

```
function is_exist(name) {
    for (let i = 0; i < points.length; i++) {
        if (points[i].name === name)
            return true;
    }

    return false;
}
```

Тесты:

```
function main(){
    create("point1", 0, 0);
    create("point20", 10, -5);
    create("point3", -1, -1);
    create("point4", 3, 2);
    create("point5", -1, -7);
}

let points = [];
main();
```

Результат:

```
-----------------CREATE-----------------
point1 [0;0] добавлен
----------------------------------------

-----------------CREATE-----------------
point20 [10;-5] добавлен
----------------------------------------

-----------------CREATE-----------------
point3 [-1;-1] добавлен
----------------------------------------

-----------------CREATE-----------------
point4 [3;2] добавлен
----------------------------------------

-----------------CREATE-----------------
point5 [-1;-7] добавлен
----------------------------------------
```

2) READ

```
function read() {
    console.log("------------------READ------------------");

    if (points.length === 0) {
        console.log("ОШИБКА: пустое хранилище");
        console.log("----------------------------------------");
        console.log();

        return;
    }

    for (let i = 0; i < points.length; i++)
        console.log(points[i].name + " [" + points[i].x + ";" + points[i].y + "]");
    console.log("----------------------------------------");
    console.log();
}
```

Тесты:

```
    read();
```

Результат:

```
------------------READ------------------
point1 [0;0]
point20 [10;-5]
point3 [-1;-1]
point4 [3;2]
point5 [-1;-7]
----------------------------------------
```

3) UPDATE

```
46  function update(name, key, value) {
47      console.log("------------------UPDATE------------------");
48
49      if (points.length === 0) {
50          console.log("ОШИБКА: пустое хранилище");
51          console.log("----------------------------------------");
52          console.log();
53
54          return;
55      }
56
57      if (!is_exist(name)) {
58          console.log("ОШИБКА: " + name + " НЕ существует");
59          console.log("----------------------------------------");
60          console.log();
61
62          return;
63      }
64
65      for (let i = 0; i < points.length; i++) {
66          if (points[i].name === name) {
67              points[i][key] = value;
68              break;
69          }
70      }
71
72      console.log(name + " изменён");
73      console.log("----------------------------------------");
74      console.log();
75  }
```

Тесты:

```
------------------READ------------------
point1 [0;0]
point20 [10;-5]
point3 [-1;-1]
point4 [3;2]
point5 [-1;-7]
----------------------------------------
```

```
116     update("point5", "x", 0);
117     update("point4", "y", 4);
118     update("point20", "name", "point2");
119     update("point100", "x", 0);
120     read();
```

Результат:

```
-----------------UPDATE-----------------
point5 изменён
----------------------------------------

-----------------UPDATE-----------------
point4 изменён
----------------------------------------

-----------------UPDATE-----------------
point20 изменён
----------------------------------------

-----------------UPDATE-----------------
ОШИБКА: point100 НЕ существует
----------------------------------------

------------------READ------------------
point1 [0;0]
point2 [10;-5]
point3 [-1;-1]
point4 [3;4]
point5 [0;-7]
----------------------------------------
```

4) DELETE

```
77  function delete_point(name) {
78      console.log("-----------------DELETE-----------------");
79
80      if (points.length === 0) {
81          console.log("ОШИБКА: пустое хранилище");
82          console.log("----------------------------------------");
83          console.log();
84
85          return;
86      }
87
88      if (!is_exist(name)) {
89          console.log("ОШИБКА: " + name + " НЕ существует");
90          console.log("----------------------------------------");
91          console.log();
92
93          return;
94      }
95
96      for (let i = 0; i < points.length; i++) {
97          if (points[i].name === name)
98              points.splice(i, 1);
99      }
100
101     console.log(name + " удалён");
102     console.log("----------------------------------------");
103     console.log();
104 }
```

Тесты:

```
-----------------READ-----------------
point1 [0;0]
point2 [10;-5]
point3 [-1;-1]
point4 [3;4]
point5 [0;-7]
--------------------------------------
```

```
122     delete_point("point4");
123     delete_point("point100");
124     delete_point("point5");
125     read();
```

Результат:

```
----------------DELETE----------------
point4 удалён
--------------------------------------

----------------DELETE----------------
ОШИБКА: point100 НЕ существует
--------------------------------------

----------------DELETE----------------
point5 удалён
--------------------------------------

-----------------READ-----------------
point1 [0;0]
point2 [10;-5]
point3 [-1;-1]
--------------------------------------
```

Получение двух точек, между которыми наибольшее расстояние

```
110  function find_points_max_distance()
111  {
112      if (points.length === 0) {
113          console.log("ОШИБКА: пустое хранилище");
114          console.log("----------------------------------------");
115          console.log();
116          return;
117      }
118
119      if (points.length === 1){
120          console.log("ОШИБКА: в хранилище только одна точка");
121          console.log("----------------------------------------");
122          console.log();
123          return;
124      }
125
126      console.log("Точки с наибольшем расстоянием между ними: ");
127
128      let max_distance = -1;
129      let temp_distance;
130      let pnt1 = [], pnt2 = [];
131      for (let i = 0; i < points.length; i++){
132          for (let j = i + 1; j < points.length; j++){
133              temp_distance = find_distance(points[i], points[j]);
134              if (temp_distance > max_distance){
135                  max_distance = temp_distance;
136                  pnt1 = [];
137                  pnt2 = [];
138                  pnt1.push(points[i]);
139                  pnt2.push(points[j]);
140              }
141              else if (Math.abs(temp_distance - max_distance) < 1e-5){
142                  pnt1.push(points[i]);
143                  pnt2.push(points[j]);
144              }
145          }
146      }
147
148      for (let i = 0; i < pnt1.length; i++)
149          console.log(pnt1[i].name + " [" + pnt1[i].x + ";" + pnt1[i].y + "] и "
150                      + pnt2[i].name + " [" + pnt2[i].x + ";" + pnt2[i].y + "]");
151      console.log();
152  }
```

Тесты:

```
------------------READ------------------
point1 [0;0]
point2 [10;-5]
point3 [-1;-1]
----------------------------------------
```

```
174      find_points_max_distance();
```

Результат:

```
Точки с наибольшем расстоянием между ними:
point2 [10;-5] и point3 [-1;-1]
```

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

```
function find_current_points(pnt, cur_distance){
    if (points.length === 0) {
        console.log("ОШИБКА: пустое хранилище");
        console.log("---------------------------------------");
        console.log();
        return;
    }

    console.log("Точки, находящиеся от точки [" + pnt.x + ";" + pnt.y + "] на расстоянии, не превышающем " + cur_distance + ":");

    if (cur_distance < 0){
        console.log("ОШИБКА: неверно заданное расстояние");
        console.log();
        return;
    }

    let found = false;
    for (let i = 0; i < points.length; i++){
        if (find_distance(pnt, points[i]) <= cur_distance)
        {
            found = true;
            console.log(points[i].name + " " + points[i].x + " " + points[i].y);
        }
    }

    if (!found)
        console.log("Не найдено");

    console.log();
}
```

Тесты:

```
------------------READ------------------
point1 [0;1]
point2 [10;-5]
point3 [-1;-1]
----------------------------------------
```

```
    find_current_points({x : 0, y : 0}, 7);
    find_current_points({x : 0, y : 0}, 1);
    find_current_points({x : 0, y : 0}, 0.5);
```

Результат:

```
Точки, находящиеся от точки [0;0] на расстоянии, не превышающем 7:
point1 0 1
point3 -1 -1

Точки, находящиеся от точки [0;0] на расстоянии, не превышающем 1:
point1 0 1

Точки, находящиеся от точки [0;0] на расстоянии, не превышающем 0.5:
Не найдено
```

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

```
function find_current_points_axis(ax, pos){
    if (points.length === 0) {
        console.log("ОШИБКА: пустое хранилище");
        console.log("---------------------------------------");
        console.log();
        return;
    }

    if (ax === axis[0]){
        console.log("Ось ox: ");

        if (pos === position[2]) {     // up
            console.log("- выше оси");

            for (let i = 0; i < points.length; i++){
                if (points[i].y > 0)
                    console.log(points[i].name + " [" + points[i].x + ";" + points[i].y + "]");
            }
        }
        else if (pos === position[3]){      // down
            console.log("- ниже оси");

            for (let i = 0; i < points.length; i++){
                if (points[i].y < 0)
                        console.log(points[i].name + " [" + points[i].x + ";" + points[i].y + "]");
            }
        }
    }
```

```
    else if (ax === axis[1]){
        console.log("Ось oy: ");

        if (pos === position[0]) {     // left
            console.log("- левее оси");

            for (let i = 0; i < points.length; i++){
                if (points[i].x < 0)
                    console.log(points[i].name + " [" + points[i].x + ";" + points[i].y + "]");
            }
        }
        else if (pos === position[1]){      // right
            console.log("- правее оси");

            for (let i = 0; i < points.length; i++){
                if (points[i].x > 0)
                    console.log(points[i].name + " [" + points[i].x + ";" + points[i].y + "]");
            }
        }
    }

    console.log();
}
```

Тесты:

```
-----------------READ-----------------
point1 [0;1]
point2 [10;-5]
point3 [-1;-1]
---------------------------------------
```

```
    find_current_points_axis(axis[0], position[2]);
    find_current_points_axis(axis[0], position[3]);
    find_current_points_axis(axis[1], position[0]);
    find_current_points_axis(axis[1], position[1]);
```

Результат:

```
Ось ox:
- выше оси
point1 [0;1]

Ось ox:
- ниже оси
point2 [10;-5]
point3 [-1;-1]

Ось oy:
- левее оси
point3 [-1;-1]

Ось oy:
- правее оси
point2 [10;-5]
```

Получение точек, входящих внутрь заданной прямоугольной зоны

```
261  function find_points_in_area(corner1, corner2){
262      if (points.length === 0) {
263          console.log("ОШИБКА: пустое хранилище");
264          console.log("----------------------------------------");
265          console.log();
266          return;
267      }
268
269      console.log("Точки, расположенные внутри прямоугольной области [" + corner1.x + ";" + corner1.y + "] [" +
270                  corner2.x + ";" + corner2.y + "]");
271
272      let result;
273      let found = false;
274      for (let i = 0; i < points.length; i++){
275          if (corner1.x <= points[i].x && points[i].x <= corner2.x &&
276              corner1.y >= points[i].y && points[i].y >= corner2.y){
277              console.log(points[i].name + " " + points[i].x + " " + points[i].y);
278              found = true;
279              }
280      }
281
282      if (!found)
283          console.log("Не найдено");
284
285      console.log();
286  }
```

Тесты:

```
-----------------READ-----------------
point1 [0;1]
point2 [10;-5]
point3 [-1;-1]
----------------------------------------
```

```
318      find_points_in_area({x : -2, y : 2}, {x : 1, y : -1});
319      find_points_in_area({x : -5, y : 2}, {x : -4, y : -1});
```

Результат:

```
Точки, расположенные внутри прямоугольной области [-2;2] [1;-1]
point1 0 1
point3 -1 -1

Точки, расположенные внутри прямоугольной области [-5;2] [-4;-1]
Не найдено
```

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.

```
"use strict";

class Point{
    constructor(x, y) {
        this.x = x;
        this.y = y;
    }

    print(){
        console.log("Точка [" + this.x + ";" + this.y + "]")
    }
}

class Segment{
    constructor(pnt1, pnt2) {
        this.point1 = new Point(pnt1.x, pnt1.y);
        this.point2 = new Point(pnt2.x, pnt2.y);
    }

    print() {
        console.log("Отрезок: ");
        this.point1.print();
        this.point2.print();
    }

    get_length() {
        return Math.sqrt(Math.pow(this.point2.x - this.point1.x, 2) +
                         Math.pow((this.point2.y - this.point1.y), 2)).toFixed(3);
    }
}
```

Тесты:

```
function main() {
    let pnt1 = new Point(0, 0);
    let pnt2 = new Point(10, 5);
    let s = new Segment(pnt1, pnt2)

    s.print();

    console.log("Длина отрезка: " + s.get_length());
}

main();
```

Результат:

```
Отрезок:
Точка [0;0]
Точка [10;5]
Длина отрезка: 11.180
```

Задание 2

Создать класс *Треугольник*.

Класс *Треугольник* должен иметь поля, хранящие длины сторон треугольника.

Реализовать следующие методы:

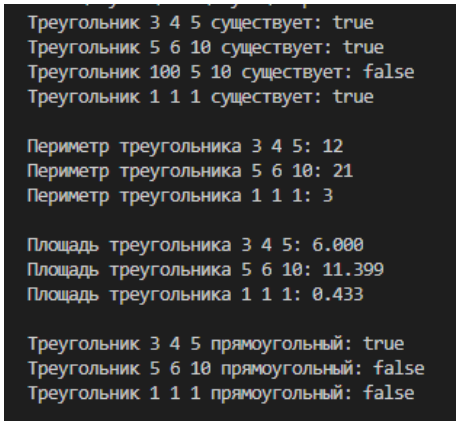
- Метод инициализации полей
- Метод проверки возможности существования треугольника с такими сторонами
- Метод получения периметра треугольника
- Метод получения площади треугольника
- Метод для проверки факта: является ли треугольник прямоугольным

```
"use strict";

class Triangle {
    constructor(a, b, c){
        this.a = a;
        this.b = b;
        this.c = c;
    }

    is_exist() {
        if (this.a + this.b > this.c && this.a + this.c > this.b && this.b + this.c > this.a)
            return true;

        return false;
    }

    get_perimeter() {
        return this.a + this.b + this.c;
    }

    get_square() {
        let p = this.get_perimeter() / 2;

        return Math.sqrt(p * (p - this.a) * (p - this.b) * (p - this.c)).toFixed(3);
    }

    is_rectangular() {
        if (Math.abs(this.a ** 2 + this.b ** 2 - this.c ** 2) < Number.EPSILON ||
            Math.abs(this.a ** 2 + this.c ** 2 - this.b ** 2) < Number.EPSILON ||
            Math.abs(this.a ** 2 + this.b ** 2 - this.c ** 2) < Number.EPSILON)
            return true;

        return false;
    }
}
```

Тесты:

```
function main(){
    let fig1 = new Triangle(3, 4, 5);
    let fig2 = new Triangle(5, 6, 10);
    let fig3 = new Triangle(100, 5, 10);
    let fig4 = new Triangle(1, 1, 1);

    console.log("Треугольник " + fig1.a + " " + fig1.b + " " + fig1.c + " существует: " + fig1.is_exist());
    console.log("Треугольник " + fig2.a + " " + fig2.b + " " + fig2.c + " существует: " + fig2.is_exist());
    console.log("Треугольник " + fig3.a + " " + fig3.b + " " + fig3.c + " существует: " + fig3.is_exist());
    console.log("Треугольник " + fig4.a + " " + fig4.b + " " + fig4.c + " существует: " + fig4.is_exist());
    console.log();

    console.log("Периметр треугольника " + fig1.a + " " + fig1.b + " " + fig1.c + ": " + fig1.get_perimeter());
    console.log("Периметр треугольника " + fig2.a + " " + fig2.b + " " + fig2.c + ": " + fig2.get_perimeter());
    console.log("Периметр треугольника " + fig4.a + " " + fig4.b + " " + fig4.c + ": " + fig4.get_perimeter());
    console.log();

    console.log("Площадь треугольника " + fig1.a + " " + fig1.b + " " + fig1.c + ": " + fig1.get_square());
    console.log("Площадь треугольника " + fig2.a + " " + fig2.b + " " + fig2.c + ": " + fig2.get_square());
    console.log("Площадь треугольника " + fig4.a + " " + fig4.b + " " + fig4.c + ": " + fig4.get_square());
    console.log();

    console.log("Треугольник " + fig1.a + " " + fig1.b + " " + fig1.c + " прямоугольный: " + fig1.is_rectangular());
    console.log("Треугольник " + fig2.a + " " + fig2.b + " " + fig2.c + " прямоугольный: " + fig2.is_rectangular());
    console.log("Треугольник " + fig4.a + " " + fig4.b + " " + fig4.c + " прямоугольный: " + fig4.is_rectangular());
    console.log();
}

main();
```

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.

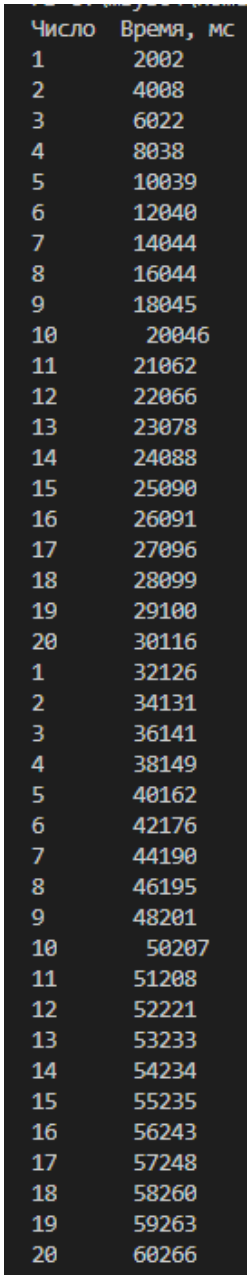
После этого происходит вывод от 11 до 20 с задержками в 1 секунду.

Это должно происходить циклически.

```
"use strict";

function func_1(){
    if (circle === 2){
        clearInterval(interval);
        return;
    }

    console.log(number + "       " + (Date.now() - time_start));

    number++;

    if (number > 10){
        clearInterval(interval);
        interval = setInterval(func_2, 1000)
    }
}

function func_2(){
    console.log(number + "      " + (Date.now() - time_start));

    number++;

    if (number > 20){
        clearInterval(interval);
        circle++;

        number = 1;
        interval = setInterval(func_1, 2000)
    }
}
```

```
function main(){
    console.log("Число  Время, мс");
    time_start = Date.now();
    interval = setInterval(func_1, 2000);
}


let number = 1, circle = 0;
let interval, time_start;

main();
```

Результат:



Число	Время, мс
1	2002
2	4008
3	6022
4	8038
5	10039
6	12040
7	14044
8	16044
9	18045
10	20046
11	21062
12	22066
13	23078
14	24088
15	25090
16	26091
17	27096
18	28099
19	29100
20	30116
1	32126
2	34131
3	36141
4	38149
5	40162
6	42176
7	44190
8	46195
9	48201
10	50207
11	51208
12	52221
13	53233
14	54234
15	55235
16	56243
17	57248
18	58260
19	59263
20	60266

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

Вывод: в ходе лабораторной работы были изучены основы синтаксиса языка JS, рассмотрены массивы, объекты, классы для решения задач. Рассмотрена функция setInterval и применена для решения поставленной задачи.