



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 «Программное обеспечение ЭВМ и информационные технологии»

ОТЧЕТ ПО ЛАБОРАТОРНОЙ РАБОТЕ №2
По курсу: "Архитектура ЭВМ"

Студент _____ Сукочева Алис
Группа _____ ИУ7-53Б
Название предприятия _____ МГТУ им. Н. Э. Баумана, каф. ИУ7
Тема _____ Создание классов. Таймер.

Студент:	_____	Сукочева А.
	подпись, дата	Фамилия, И.О.
Преподаватель:	_____	Попов А. Ю.
	подпись, дата	Фамилия, И. О.

TASK_1. Задание 1

Техническое задание:

- 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

Листинг 1 — Код программы. Задание 1.

```
1 "use strict";
2
3 class Kids {
4     constructor() {
5         this.arr = [];
6     }
7
8     add(surname, age) {
9         // function func(elem) {
10            // return elem.surname === surname;
11            // }
12         if (!this.arr.find(elem => elem.surname === surname)) {
13             let new_kid = { surname, age }; // { surname: surname, age :
14                age }
15             this.arr.push(new_kid);
16         }
17
18         log() {
19             for (let i = 0; i < this.arr.length; i++)
20                 console.log(this.arr[i].surname, " ", this.arr[i].age);
21             console.log("\n")
```

```

22     }
23
24     read(surname) {
25         return this.arr.find(elem => elem.surname === surname);
26         // for (let i = 0; i < this.arr.length; i++)
27         //   if (this.arr[i].surname === surname)
28         //     return this.arr[i];
29         // return ""
30     }
31
32     update(surname, new_age, new_surname = false) {
33         for (let i = 0; i < this.arr.length; i++) {
34             if (this.arr[i].surname === surname) {
35                 this.arr[i].age = new_age;
36                 if (new_surname && !this.arr.find(elem => elem.surname ===
37                     new_surname))
38                     this.arr[i].surname = new_surname;
39                 return;
40             }
41         }
42
43         delete(surname) {
44             this.arr = this.arr.filter(elem =>
45                 elem.surname !== surname);
46         }
47
48         get_avg() {
49             let sum = 0;
50             let len = this.arr.length;
51             for (let i = 0; i < len; i++)
52                 sum += this.arr[i].age;
53             return len ? sum / len : 0;
54         }
55
56         get_eldest() {
57             let len = this.arr.length;
58             if (!len)
59                 return;
60             let max = this.arr[0];
61             for (let i = 1; i < len; i++) {
62                 if (this.arr[i].age > max.age)
63                     max = this.arr[i];
64             }
65             return max;
66         }
67

```

```

68     get_age_range(begin, end) {
69         return this.arr.filter(elem => elem.age >= begin
70             && elem.age <= end);
71     }
72
73     get_surname_letter(letter) {
74         return this.arr.filter(elem => elem.surname.charAt(0) === letter);
75     }
76
77     get_surname_len(len) {
78         return this.arr.filter(elem => elem.surname.length > len)
79     }
80
81     get_surname_vowels() {
82         const VOWELS = ['A', 'E', 'I', 'O', 'U'];
83         return this.arr.filter(elem =>
84             VOWELS.indexOf(elem.surname.charAt(0).toUpperCase()) !== -1)
85     };

```

Листинг 2 — Код тестов. Задание 1.

```

1  function test_kids() {
2      let child = new Kids();
3      child.add("Sukocheva", 2);
4      child.add("Sukocheva", 3);
5      child.add("Namestnik", 1);
6      child.add("Vinogradov", 4);
7      child.add("Volkov", 3);
8      child.add("Orbitov", 7);
9
10
11     child.log();
12
13     console.log(child.read("Sukocheva"));
14     child.update("Sukocheva", 4)
15     console.log(child.read("Sukocheva"));
16
17     child.log();
18     child.delete("Vinogradov");
19     child.log();
20
21     child.read("Sukocheva")
22
23     console.log(child.get_avg());
24     console.log(child.get_eldest());
25

```

```
26     console.log(child.get_age_range(1, 3))
27     console.log(child.get_surname_letter('S'));
28
29     console.log(child.get_surname_len(6));
30
31     console.log(child.get_surname_vowels());
32 }
```

TASK_1. Задание 2

Техническое задание:

- 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

Листинг 3 — Код программы. Задание 1.

```
1  "use strict";
2
3  class Students {
4      constructor() {
5          this.arr = [];
6      }
7
8      add(id, group, arr_score) {
9          if (!this.arr.find(elem => elem.id === id)) {
10             let new_student = { group, id, arr_score };
11             this.arr.push(new_student);
12         }
13     }
14
15     log() {
16         console.log(this.arr);
17     }
18
19     read(id) {
20         return this.arr.find(elem => elem.id === id);
21     }
22
23     update(id, group, arr_score) {
24         let student = this.read(id);
25         if (student) {
```

```

26         student.group = group;
27         student.arr_score = arr_score;
28     }
29 }
30
31 delete(id) {
32     this.arr = this.arr.filter(elem =>
33         elem.id !== id);
34 }
35
36 get_avg(id) {
37     let student = this.read(id);
38     if (!student)
39         return;
40
41     let len = student.arr_score.length;
42     let sum = 0;
43     for (let i = 0; i < len; i++)
44         sum += student.arr_score[i];
45
46     return len ? sum / len : 0;
47 }
48
49 get_info_group(group) {
50     return this.arr.filter(elem => elem.group === group);
51 }
52
53 get_student_max_count_score(group) {
54     let students = this.get_info_group(group);
55     if (!students)
56         return;
57     let max = students[0];
58     for (let i = 1; i < students.length; i++) {
59         if (students[i].arr_score.length > max.arr_score.length)
60             max = students[i];
61     }
62     return max
63 }
64
65 get_student_no_score() {
66     return this.arr.filter(elem => elem.arr_score.length === 0);
67 }
68 };

```

Листинг 4 — Код тестов. Задание 2.

```

1 function test_students() {

```

```

2      let students = new Students();
3
4      students.add(123456, "UI7-43b", [5, 5, 5]);
5      students.add(123456, "UI7-43b", [5, 5, 5]);
6
7      students.add(123457, "UI7-43b", []);
8      students.add(123451, "UI7-43b", [1, 5, 5, 2]);
9      students.add(123000, "UI7-44b", [4, 3, 4]);
10     students.add(123444, "UI7-45b", [5, 4, 5]);
11     students.add(123442, "UI7-45b", [2, 3, 3]);
12     students.add(123441, "UI7-42b", [2, 3, 5]);
13     students.add(123443, "UI7-41b", [5, 2, 3]);
14
15     students.log();
16
17     console.log(students.read(123456));
18
19     students.update(123441, "UI7-42b", [5, 5, 5]);
20     students.log();
21
22     students.delete(123442);
23     students.log();
24
25     console.log(students.get_avg(123441));
26     console.log(students.get_info_group("UI7-43b"));
27
28     console.log(students.get_student_max_count_score("UI7-43b"));
29     console.log(students.get_student_no_score());
30 }

```


TASK_1. Задание 3

Техническое задание:

- 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

Листинг 5 — Код программы. Задание 3

```
1  "use strict";
2
3  class Points {
4      constructor() {
5          this.arr = [];
6      }
7
8      add(name_point, x, y) {
9          if (!this.arr.find(elem => elem.name_point === name_point)) {
10             let new_point = { name_point, x, y };
11             this.arr.push(new_point);
12         }
13     }
14
15     log() {
16         console.log(this.arr);
17     }
18
19     read(name_point) {
20         return this.arr.find(elem => elem.name_point === name_point);
21     }
22
23     update(name_point, x, y) {
24         let point = this.read(name_point);
25         if (point) {
```

```

26         point.x = x;
27         point.y = y;
28     }
29 }
30
31 delete(name_point) {
32     this.arr = this.arr.filter(elem =>
33         elem.name_point !== name_point);
34 }
35
36 get_distance(p1, p2) {
37     let dx = p1.x - p2.x;
38     let dy = p1.y - p2.y;
39     return Math.sqrt(dx * dx + dy * dy);
40 }
41
42 min_distance() {
43     if (this.arr.length < 2)
44         return;
45
46     let len = this.arr.length;
47     let p1 = this.arr[0];
48     let p2 = this.arr[1];
49     let min_dist = this.get_distance(p1, p2), current_dist;
50     console.log(min_dist);
51
52     for (let i = 0; i < len - 1; i++)
53         for (let j = i + 1; j < len; j++) {
54             current_dist = this.get_distance(this.arr[i],
55                 this.arr[j]);
56             if (current_dist < min_dist)
57                 min_dist = current_dist;
58         }
59     return min_dist;
60 }
61
62 // Получение точек, находящихся от заданной точки на расстоянии,
63 // не превышающем заданную константу
64 get_points_less(point, max_len) {
65     return this.arr.filter(elem =>
66         this.get_distance(point, elem) <= max_len);
67 }
68
69 // axis: 0-x, 1-y;
70 // direction: 0-'+', 1-'-';
71 get_points_axis(axis, direction) {

```

```

72         let func;
73
74         if (!axis && !direction)
75             func = p => p.x > 0;
76         else if (!axis && direction)
77             func = p => p.x < 0;
78         else if (!direction)
79             func = p => p.y > 0;
80         else
81             func = p => p.y < 0;
82
83         return this.arr.filter(func);
84     }
85     get_points_inside_rectangle(min_x, max_x, min_y, max_y) {
86         return this.arr.filter(p =>
87             p.x > min_x && p.x < max_x &&
88             p.y > min_y && p.y < max_y);
89     }
90 }

```

Листинг 6 — Код тестов. Задание 3.

```

1  function test_points() {
2      let points = new Points();
3      points.add("p", 0, 0);
4      points.add("p0", 3, 4);
5      points.add("p1", 1, 1);
6      points.add("p2", 10, 10);
7      points.add("p3", 1, 10);
8      points.add("p4", 10, 1);
9      points.add("p5", 12, 0);
10     points.add("p6", -12, 1);
11     points.add("p7", 12, -1);
12
13     points.log();
14
15     console.log(points.read("p1"));
16     console.log(points.read("p3"));
17     console.log(points.read("p13"));
18     console.log(points.read("p5"));
19
20     points.update("p5", 100, 12);
21     points.log();
22     points.delete("p5");
23     points.log();
24
25     console.log(points.min_distance());

```

```
26
27     let p = points.read("p"); // 0, 0
28     console.log(points.get_points_less(p, 5));
29
30     console.log("Points:\n")
31     points.log();
32     console.log("axis X-:\n", points.get_points_axis(0, 1));
33     console.log("axis Y-:\n", points.get_points_axis(1, 1));
34
35     console.log("rectangle: -10, 10, -10, 10\n",
36         points.get_points_inside_rectangle(-10, 10, -10, 10));
37 }
```

TASK_2. Задание 1

Техническое задание:

- Создать класс Точка.
- Добавить классу точка 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 так же метод получения длины отрезка.

Листинг 7 — Код программы. Задание 1.

```
1  class Point {
2      constructor(x, y) {
3          this.set_data(x, y);
4      }
5
6      set_data(x, y) {
7          this.x = x;
8          this.y = y;
9      }
10
11     log() {
12         console.log("X: ", this.x, "Y: ", this.y);
13     }
14 }
15
16 class Line {
17     constructor(x1, y1, x2, y2) {
18         this.set_data(x1, y1, x2, y2);
19     }
20
21     set_data(x1, y1, x2, y2) {
22         this.start_point = new Point(x1, y1);
23         this.end_point = new Point(x2, y2);
24     }
25
26     get_distance() {
27         const dx = this.start_point.x - this.end_point.x;
28         const dy = this.start_point.y - this.end_point.y;
29         return Math.sqrt(dx * dx + dy * dy);
30     }
31 }
```

```
32     log () {  
33         console.log ("Начальная точка:");  
34         this.start_point.log ();  
35         console.log ("Конечная точка:");  
36         this.end_point.log ();  
37     }  
38 }
```

TASK_2. Задание 2

Техническое задание:

- Создать класс Треугольник.
- Класс Треугольник должен иметь поля, хранящие длины сторон треугольника.
- Необходимо обеспечить уникальность номеров студенческих билетов.

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

Листинг 8 — Код программы. Задание 1.

```
1  class Triangle {
2      constructor(a, b, c) {
3          this.set_data(a, b, c);
4      }
5
6      set_data(a, b, c) {
7          this.a = a;
8          this.b = b;
9          this.c = c;
10     }
11
12     check_existence() {
13         if ((this.a < this.b + this.c) &&
14             (this.b < this.a + this.c) &&
15             (this.c < this.b + this.a))
16             return true;
17         return false;
18     }
19
20     get_perimeter() {
21         if (!this.check_existence())
22             return;
23         return this.a + this.b + this.c;
24     }
25
26     area() {
```

```

27         if (!this.check_existence())
28             return;
29         let semi_perimeter = this.get_perimeter() / 2;
30         return Math.sqrt(semi_perimeter * (semi_perimeter - this.a) *
31             (semi_perimeter - this.b) * (semi_perimeter - this.c));
32     }
33
34     check_rectangular() {
35         let EPS = 1e-5;
36         if (!this.check_existence())
37             return;
38         let a = this.a, b = this.b, c = this.c;
39
40         if ((a * a + b * b - c * c < EPS) ||
41             (a * a + c * c - b * b < EPS) ||
42             (c * c + b * b - a * a < EPS))
43             return true;
44         return false;
45     }
46 }

```


TASK_2. Задание 3

Техническое задание:

- 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.

Листинг 9 — Код программы. Задание 3

```
1  "use strict"
2
3  const TIME_FIRST = 1000
4  const TIME_SECOND = 500
5
6  function timer() {
7      let a = 1;
8
9      let funcFirst = () => {
10         console.log(a++);
11         if (a > 10) {
12             clearInterval(interval1);
13             interval2 = setInterval(funcSecond, TIME_SECOND)
14         }
15     }
16
17     let funcSecond = () => {
18         console.log(a++);
19         if (a > 20) {
20             clearInterval(interval2);
21             a = 1;
22             interval1 = setInterval(funcFirst, TIME_FIRST)
23         }
24     }
25
26     let interval1 = setInterval(funcFirst, TIME_FIRST);
27     let interval2;
28 }
```