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 **09.03.01 Информатика и вычислительная техника;**

О Т Ч Е Т

по лабораторной работе № 1

Дисциплина: Архитектура ЭВМ

Студент ИУ7-52б _____ Кузин А.А.
(Группа) (Подпись, дата) (И.О. Фамилия)

Преподаватель _____ Попов А.Ю.
(Подпись, дата) (И.О. Фамилия)

Москва, 2020

Цель

Целью данной лабораторной работы является ознакомление с основами Javascript.

Часть 1

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

Листинг программы:

```
"use strict";

function create(arr, surname, age){
    let child = {};
    for (let i = 0; i < arr.length; i++){
        if (arr[i]["surname"] === surname){
            console.log("There is already such surname");
            return;
        }
    }
    child["surname"] = surname;
    child["age"] = age;
```

```
    arr.push(child);
}

function read(arr, surname){
    if (surname){
        let found = false;
        for (let i = 0; i < arr.length; i++) {
            if (surname === arr[i]["surname"]){
                found = true;
                console.log("Surname: " + arr[i]["surname"] + "; Age: " + arr[i]["age"] + ";");
                break;
            }
        }
        if (!found)
            console.log("There is no child with such surname");
    }
    else{
        for (let i = 0; i < arr.length; i++) {
            console.log("Surname: " + arr[i]["surname"] + ", Age: " + arr[i]["age"] + ";");
        }
    }
}

function update(arr, key, surname, age){
    let changed = false;
    for (let i = 0; i < arr.length; i++) {
        if (arr[i]["surname"] === key) {
            changed = true;
            if (surname)
                arr[i]["surname"] = surname;
            if (age)
                arr[i]["age"] = age;
        }
    }
    if (!changed)
        console.log("There is no child with such surname");
}

function deleteRecord(arr, key){
    let deleted = false;
    for (let i = 0; i < arr.length; i++) {
        if (arr[i]["surname"] === key) {
            deleted = true;
            arr.splice(i, 1);
        }
    }
    if (!deleted)
        console.log("There is no child with such surname");
}

function getAverageAge(arr){
    let sum = 0;
```

```
    for (let i = 0; i < arr.length; i++){
        sum += arr[i]["age"];
    }
    let av = sum / arr.length;
    return av;
}

function findOldest(arr){
    let oldest = {};
    oldest["surname"] = "";
    oldest["age"] = 0;
    for (let i = 0; i < arr.length; i++) {
        if (arr[i]["age"] > oldest["age"]){
            oldest = arr[i];
        }
    }
    return oldest;
}

function findInRange(arr, min, max){
    let res = [];
    let found = 0;
    for (let i = 0; i < arr.length; i++) {
        if (arr[i]["age"] >= min && arr[i]["age"] <= max){
            res.push(arr[i]);
            found += 1;
        }
    }
    if (found === 0)
        console.log("There are no children in that range");
    return res;
}

function findFirstLetter(arr, letter){
    let res = [];
    let found = 0;
    for (let i = 0; i < arr.length; i++) {
        if (arr[i]["surname"].charAt(0) === letter){
            res.push(arr[i]);
            found = found + 1;
        }
    }

    if (found === 0)
        console.log("There are no children whose surname starts with that letter");
    return res;
}

function findLonger(arr, len){
    let res = [];
    let found = 0;
    for (let i = 0; i < arr.length; i++) {
```

```
        if (arr[i]["surname"].length > len){
            res.push(arr[i]);
            found = found + 1;
        }
    }

    if (found === 0)
        console.log("There are no children whose surname is longer");
    return res;
}

function findVowel(arr){
    let res = [];
    vowels = ["A", "E", "I", "O", "U"];
    let found = 0;
    for (let i = 0; i < arr.length; i++) {
        for (letter in vowels){
            if (arr[i]["surname"].charAt(0) === vowels[letter]){
                res.push(arr[i]);
                found = found + 1;
                break;
            }
        }
    }

    if (found === 0)
        console.log("There are no children whose surname starts with vowel");
    return res;
}
```

Тесты

1. Создание детей в хранилище:

```
create(children, "Kuzin", 10);
create(children, "Ivanov", 12);
create(children, "Zhukov", 6);
create(children, "Alin", 15);
create(children, "Zhukov", 8);
```

Хранилище после выполнения:

```
Surname: Kuzin, Age: 10;
Surname: Ivanov, Age: 12;
Surname: Zhukov, Age: 6;
Surname: Alin, Age: 15;
There is already such surname
```

2. Считывание хранилища:

```
read(children);
read(children, "Ivanov");
read(children, "ASdaw");
```

Результат:

```
Surname: Kuzin, Age: 10;
```

Surname: Ivanov, Age: 12;
Surname: Zhukov, Age: 6;
Surname: Alin, Age: 15;

Surname: Ivanov; Age: 12;
There is no child with such surname

3. Обновление записей:

```
update(children, "Asda");
update(children, "Kuzin", undefined, 11);
update(children, "Zhukov", "Zhukovkov", 11);
update(children, "Alin", "Alina", undefined)
```

Результат:

There is no child with such surname
Surname: Kuzin, Age: 11;
Surname: Ivanov, Age: 12;
Surname: Zhukovkov, Age: 11;
Surname: Alina, Age: 15;

4. Удаление записей:

```
deleteRecord(children, "Kuzin");
deleteRecord(children, "Zhukov");
```

Результат:

There is no child with such surname
Surname: Ivanov, Age: 12;
Surname: Zhukovkov, Age: 11;
Surname: Alina, Age: 15;

5. Получение среднего возраста:

```
getAverageAge(children);
create(children, "Smirnov", 9);
getAverageAge(children);
```

Результат:

12.666666666666666
11.75

6. Получение информации о самом старшем ребенке

```
findOldest(children);
```

Результат:

{ surname: 'Alina', age: 15 }

7. Получение информации о детях, возраст которых входит в заданный отрезок

```
findInRange(children, 10, 16);
findInRange(children, 13, 16);
findInRange(children, 16, 17);
```

Результат:

Surname: Ivanov, Age: 12;
Surname: Zhukovkov, Age: 11;

Surname: Alina, Age: 15;

Surname: Alina, Age: 15;

There are no children in that range

8. Получение информации о детях, фамилия которых начинается с заданной буквы

findFirstLetter(children, "Z");
create(children, "Zhuk", 11);
findFirstLetter(children, "Z");
findFirstLetter(children, "B");

Результат:

Surname: Zhukovkov, Age: 11;

Surname: Zhukovkov, Age: 11;
Surname: Zhuk, Age: 11;
There are no children whose surname starts with that letter

9. Получение информации о детях, фамилия которых длиннее заданного количества символов

findLonger(children, 5);
findLonger(children, 10);

Результат:

Surname: Ivanov, Age: 12;
Surname: Zhukovkov, Age: 11;
Surname: Smirnov, Age: 9;
There are no children whose surname is longer

10. Получение информации о детях, фамилия которых начинается с гласной буквы

findVowel(children);

Результат:

Surname: Ivanov, Age: 12;
Surname: Alina, Age: 15;

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

Листинг программы:

```
function create(arr, group, id, marks) {
    let stud = {};
    for (let i = 0; i < arr.length; i++) {
        if (arr[i]["id"] === id) {
            console.log("There is already such id");
            return;
        }
    }
    stud["group"] = group;
    stud["id"] = id;
    stud["marks"] = marks;
    arr.push(stud);
}

function read(arr, id) {
    if (id) {
        let found = false;
        for (let i = 0; i < arr.length; i++) {
            if (id === arr[i]["id"]) {
                found = true;
                let tmp = ""
                for (let j in arr[i]["marks"])
                    tmp += arr[i]["marks"][j] + ", ";
                console.log("group: " + arr[i]["group"] + "; Id: " + arr[i]["id"] + ";" + " Marks: " +
tmp);
                break;
            }
        }
        if (!found)
            console.log("There is no student with such id");
    }
    else {
        for (let i = 0; i < arr.length; i++) {
            let tmp = ""
            for (let j in arr[i]["marks"])
                tmp += arr[i]["marks"][j] + ", ";
            console.log("group: " + arr[i]["group"] + "; Id: " + arr[i]["id"] + ";" + " Marks: " + tmp);
```

```
        }
    }
}

function update(arr, key, group, id, marks) {
    let changed = false;
    for (let i = 0; i < arr.length; i++) {
        if (arr[i]["id"] === key) {
            changed = true;
            if (group)
                arr[i]["group"] = group;
            if (id)
                arr[i]["id"] = id;
            if (marks)
                arr[i]["marks"] = marks;
        }
    }
    if (!changed)
        console.log("There is no student with such id");
}

function deleteRecord(arr, key) {
    let deleted = false;
    for (let i = 0; i < arr.length; i++) {
        if (arr[i]["id"] === key) {
            deleted = true;
            arr.splice(i, 1);
        }
    }
    if (!deleted)
        console.log("There is no student with such id");
}

function averageMark(arr, id){
    let av = 0;
    let l = 0;
    let found = false;
    for (let i = 0; i < arr.length; i++){
        if (arr[i]["id"] === id){
            found = true
            l = arr[i]["marks"].length;
            for (let j in arr[i]["marks"])
                av += arr[i]["marks"][j];
            break;
        }
    }
    av = av / l;
    return av;
}

function readGroup(arr, group){
    let found = false;
```

```
    let res = [];

    for (let i = 0; i < arr.length; i++) {
        if (arr[i]["group"] === group){
            found = true;
            res.push(arr[i]);
        }
    }

    if (!found)
        console.log("There is no such group");
    return  res;
}

function findMoreMarks(arr, group) {
    let found = false;
    let res = {};
    let l = 0;

    for (let i = 0; i < arr.length; i++) {
        if (arr[i]["group"] === group && arr[i]["marks"].length > l){
            found = true;
            res = arr[i];
            l = arr[i]["marks"].length;
        }
    }

    if (!found)
        console.log("There is no such group");
    return res;
}

function findNoMarks(arr) {
    let found = false;
    let res = {};

    for (let i = 0; i < arr.length; i++) {
        if (arr[i]["marks"].length === 0) {
            found = true;
            res = arr[i];
        }
    }

    if (!found)
        console.log("There is no such group");
    return res;
}
```

Тесты

1. Создание записи

```
create(students, 12, 132, [4, 4, 5]);
create(students, 12, 113, [4, 3, 3, 5]);
```

```
create(students, 11, 126, [5, 5, 3, 5, 4]);
create(students, 13, 125, [3, 3, 3]);
read(students);
create(students, 13, 132, [3, 3, 3]);
```

Результат:

```
group: 12; Id: 132; Marks: 4, 4, 5,
group: 12; Id: 113; Marks: 4, 3, 3, 5,
group: 11; Id: 126; Marks: 5, 5, 3, 5, 4,
group: 13; Id: 125; Marks: 3, 3, 3,
There is already such id
```

2. Чтение хранилища

```
read(students);
console.log();
read(students, 132);
read(students, 1);
```

Результат:

```
group: 12; Id: 132; Marks: 4, 4, 5,
group: 12; Id: 113; Marks: 4, 3, 3, 5,
group: 11; Id: 126; Marks: 5, 5, 3, 5, 4,
group: 13; Id: 125; Marks: 3, 3, 3,

group: 12; Id: 132; Marks: 4, 4, 5,
There is no student with such id
```

3. Обновление записи

```
update(students, 132, 13, 133, [5, 4, 5]);
update(students, 133, 12, undefined, [4, 4, 5]);
read(students);
console.log();
update(students, 1, 12, undefined, [4, 4, 5]);
```

Результат:

```
group: 12; Id: 133; Marks: 4, 4, 5,
group: 12; Id: 113; Marks: 4, 3, 3, 5,
group: 11; Id: 126; Marks: 5, 5, 3, 5, 4,
group: 13; Id: 125; Marks: 3, 3, 3,

There is no student with such id
```

4. Удаление записи

```
deleteRecord(students, 125);
deleteRecord(students, 100);
```

Результат:

```
group: 12; Id: 133; Marks: 4, 4, 5,
group: 12; Id: 113; Marks: 4, 3, 3, 5,
group: 11; Id: 126; Marks: 5, 5, 3, 5, 4,
There is no student with such id
```

5. Получение средней оценки заданного студента

```
console.log(averageMark(students, 133));
console.log(averageMark(students, 126));
```

```
console.log(averageMark(students, 12));
```
Результат:
```
4.333333333333333
4.4
There is no such student
NaN
```
6. Получение информации о студентах в заданной группе
```
read(readGroup(students, 12));
read(readGroup(students, 100));
```
Результат:
```
group: 12; Id: 133; Marks: 4, 4, 5,
group: 12; Id: 113; Marks: 4, 3, 3, 5,
There is no such group
```

7. Получение студента, у которого наибольшее количество оценок в заданной группе
```
console.log(findMoreMarks(students, 12));
```
Результат:
```
{ group: 12, id: 113, marks: [ 4, 3, 3, 5 ] }
```

7. Получение студента, у которого нет оценок
```
create(students, 15, 150, []);
console.log(findNoMarks(students));
```
Результат:
```
{ group: 15, id: 150, marks: [] }
```

Задание 3

Создать хранилище в оперативной памяти для хранения точек.

Неоходимо 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 программы:

```
function create(arr, name, x, y) {
    let point = {};
    for (let i = 0; i < arr.length; i++) {
        if (arr[i]["name"] === name) {
            console.log("There is already such name");
            return;
        }
    }
    point["name"] = name;
    point["x"] = x;
    point["y"] = y;
    arr.push(point);
}

function read(arr, name) {
    if (name) {
        let found = false;
        for (let i = 0; i < arr.length; i++) {
            if (name === arr[i]["name"]) {
                found = true;
                console.log("Name: " + arr[i]["name"] + "; X: " + arr[i]["x"] + ";" + " Y: " +
arr[i]["y"]);
                break;
            }
        }
        if (!found)
            console.log("There is no point with such name");
    }
    else {
        for (let i = 0; i < arr.length; i++) {
            console.log("Name: " + arr[i]["name"] + "; X: " + arr[i]["x"] + ";" + " Y: " + arr[i]["y"]);
        }
    }
}

function update(arr, key, name, x, y) {
    let changed = false;
    for (let i = 0; i < arr.length; i++) {
        if (arr[i]["name"] === key) {
            changed = true;
            if (name)
                arr[i]["name"] = name;
            if (x)
                arr[i]["x"] = x;
            if (y)
                arr[i]["y"] = y;
        }
    }
    if (!changed)
```

```
        console.log("There is no point with such name");
}

function deleteRecord(arr, key) {
    let deleted = false;
    for (let i = 0; i < arr.length; i++) {
        if (arr[i]["name"] === key) {
            deleted = true;
            arr.splice(i, 1);
        }
    }
    if (!deleted)
        console.log("There is no student with such name");
}

function findDist(x1, y1, x2, y2){
    let d = Math.sqrt(Math.pow(x2 - x1, 2) + Math.pow(y2 - y1, 2));
    return d;
}

function maxDist(arr){
    let dist = 0;
    let p1, p2 = {};
    for (let i = 0; i < arr.length - 1; i++){
        for (let j = i + 1; j < arr.length; j++){
            let tmp = findDist(arr[i]["x"], arr[i]["y"], arr[j]["x"], arr[j]["y"]);
            if (tmp > dist){
                dist  = tmp;
                p1 = arr[i];
                p2 = arr[j];
            }
        }
    }
    return [p1, p2];
}

function closer(arr, name, dist){
    let res = [];
    let ind = -1;
    for (let i = 0; i < arr.length; i++){
        if (arr[i]["name"] == name){
            ind = i;
            break;
        }
    }
    if (ind === -1){
        console.log("There is no point with such name");
        return res;
    }

    for (let i = 0; i < arr.length; i++){
        if (i != ind)
```

```
        {
            let tmp = findDist(arr[ind]["x"], arr[ind]["y"], arr[i]["x"], arr[i]["y"]);
            if (tmp < dist)
                res.push(arr[i]);
        }
    }
    return res;
}

function side(arr, axes, side){
    let res = [];
    for (let i = 0; i < arr.length; i++){
        if (axes === 1){
            if (side === 1 && arr[i]["y"] > 0)
                res.push(arr[i]);
            else if (side === -1 && arr[i]["y"] < 0)
                res.push(arr[i]);
        }
        else if (axes === 2){
            if (side === 1 && arr[i]["x"] > 0)
                res.push(arr[i]);
            else if (side === -1 && arr[i]["x"] < 0)
                res.push(arr[i]);
        }
    }
    return res;
}

function inside(arr, xl, xr, yt, yb){
    let res = [];
    for (let i = 0; i < arr.length; i++){
        if ((xl < arr[i]["x"] && arr[i]["x"] < xr) && (yb < arr[i]["y"] && arr[i]["y"] < yt))
            res.push(arr[i]);
    }
    return res;
}
```

Тесты

1. Создание записи

```
create(points, "first", 1, 1);
create(points, "second", 1, -1);
create(points, "third", -2, -2);
create(points, "fourth", -2, 1);
read(points);
create(points, "first", 1, 2);
```

Результат:

```
Name: first; X: 1; Y: 1
Name: second; X: 1; Y: -1
Name: third; X: -2; Y: -2
Name: fourth; X: -2; Y: 1
There is already such name
```

2. Чтение хранилища

```
read(points);
console.log();
read(points, "first");
read(points, "no");
```

Результат:

```
Name: first; X: 1; Y: 1
Name: second; X: 1; Y: -1
Name: third; X: -2; Y: -2
Name: fourth; X: -2; Y: 1

Name: first; X: 1; Y: 1
There is no point with such name
```

3. Обновление записи

```
update(points, "second", "second2", 1, 2);
read(points);
console.log();
update(points, "second2", "second");
read(points);
update(points, "second2");
```

Результат:

```
Name: first; X: 1; Y: 1
Name: second2; X: 1; Y: 2
Name: third; X: -2; Y: -2
Name: fourth; X: -2; Y: 1

Name: first; X: 1; Y: 1
Name: second; X: 1; Y: 2
Name: third; X: -2; Y: -2
Name: fourth; X: -2; Y: 1
There is no point with such name
```

4. Удаление записи

```
deleteRecord(points, "first");
read(points);
deleteRecord(points, "first");
```

Результат:

```
Name: second; X: 1; Y: 2
Name: third; X: -2; Y: -2
Name: fourth; X: -2; Y: 1
There is no point with such name
```

5. Получение двух точек, между которыми наибольшее расстояние

```
console.log(maxDist(points));
```

Результат:

```
[ { name: 'second', x: 1, y: 2 }, { name: 'third', x: -2, y: -2 } ]
```

6. Получение точек, находящихся от заданной точки на расстоянии, не превышающем заданную константу
```
create(points, "new", 10, 2);
read(closer(points, "second", 4.5));
```
Результат:
```
Name: fourth; X: -2; Y: 1
```

7. Получение точек, находящихся выше / ниже / правее / левее заданной оси координат
```
read(side(points, 1, 1));
console.log();
read(side(points, 2, -1));
```
Результат:
```
Name: second; X: 1; Y: 2
Name: fourth; X: -2; Y: 1
Name: new; X: 10; Y: 2

Name: third; X: -2; Y: -2
Name: fourth; X: -2; Y: 1
```

8. Получение точек, входящих внутрь заданной прямоугольной зоны
```
read(inside(points, -3, 3, 3, -3));
```

Результат:
```
Name: second; X: 1; Y: 2
Name: third; X: -2; Y: -2
Name: fourth; X: -2; Y: 1
```

Часть 2

Задание 1

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 программы

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

   output(){
      let msg = "X = " + this.x + "; Y = " + this.y;
      return msg;
   }
}

class Section{
   constructor(start, end){
      this.start = start;
      this.end = end;
   }

   output(){
      let msg1 = "Start: " + this.start.output() + "; end: " + this.end.output();
      console.log(msg1);
   }

   length(){
      let d = Math.sqrt(Math.pow(this.start.x - this.end.x, 2) + Math.pow(this.start.y - this.end.y,
2));
      return d;
   }
}
```

Тест:

```
let sec = new Section(new Point(0, 0), new Point(5, 0));
sec.output();
console.log("Length = " + sec.length());
```

Результат

```
Start: X = 0; Y = 0; end: X = 5; Y = 0
Length = 5
```

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

Листинг программы

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

   isPossible() {
      if (this.a === this.b && this.b === this.c)
         return true;

      if (this.a < this.b + this.c && this.b < this.a + this.c && this.c < this.b + this.a)
         return true;
      else
         return false;
   }

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

   square() {
      let p = this.perimeter() / 2;

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

   isRight() {
      if (this.a * this.a + this.b * this.b == this.c * this.c || this.a * this.a
         + this.c * this.c == this.b * this.b || this.b * this.b + this.c * this.c == this.a * this.a)
         return true;
      else
         return false;
   }
}
```

Тест:

```
let tri = new Triangle(1, 1, 1);

if (tri.isPossible())
   console.log("Is possible");
else
   console.log("Is not possible");
console.log("Perimeter = " + tri.perimeter());

console.log(tri.square());

if (tri.isRight())
   console.log("Right");
else
   console.log("Isn't right");

let tri2 = new Triangle(1, 1, Math.sqrt(2));
console.log(tri2.square());

if (tri2.isRight())
   console.log("Right");
else
   console.log("Isn't right");
```

Результат

```
Is possible
Perimeter = 3
0.4330127018922193
Isn't right
0.49999999999999983
Isn't right
```

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.

Листинг программы

```
"use strict";
```

```
let seconds = 0;
let rep = 0;

let int1 = setInterval(() => {
   seconds++;
   console.log(seconds);
   if (seconds === 10)
      clearInterval(int1);
}, 1000);

setTimeout(() => {
   let int2 = setInterval(() => {
      seconds++;
      console.log(seconds);
      if (seconds === 20) {
         clearInterval(int2);
      }
   }, 500);
}, 10000);


function stop(){
   if (rep === 2)
      clearInterval(interval);
}

let interval = setInterval(() => {
   seconds = 0;
   let int1 = setInterval(() => {
      seconds++;
      console.log(seconds);
      if (seconds === 10)
         clearInterval(int1);
   }, 1000);

   setTimeout(() => {
      let int2 = setInterval(() => {
         seconds++;
         console.log(seconds);
         if (seconds === 20) {
            clearInterval(int2);
            rep++;
            stop();
         }
      }, 500);
   }, 10000);
}, 15100);
```

Вывод

В результате выполнения лабораторной работы я ознакомился с основами Javascript.