Министерство образования и науки Российской Федерации

Федеральное государственное бюджетное образовательное учреждение

высшего профессионального образования

«Владимирский государственный университет

имени Александра Григорьевича и Николая Григорьевича Столетовых»

(ВлГУ)

Кафедра информационных систем и программной инженерии

Лабораторная работа №1

по дисциплине "Графический и веб-дизайн"

ТЕМА РАБОТЫ:

Язык JavaScript

Выполнил:

студент гр. ПРИ-120

Парахин К.В.

Принял:

Шамышев А.

Владимир 2023 г.

Цель работы:

Цель работы: получить навыки работы с языком JavaScript; освоить основные типы данных и операторы JavaScript; изучить механизм прототипирования.

Задания:

1. Изучить методический материал;
2. Создать собственный JS-скрипт с использованием переменных и условным оператором («if» и «?»). Запустить скрипт в браузере. JS-скрипт должен быть создан отдельным файлом и подключен к HTML-странице;
3. Выведите простые числа с помощью цикла for на интервале от 2 до n, где n любое число больше 10
4. Создать собственный JS-скрипт с примерами использования forEach(), map(), some(), reduce(). Запустить скрипт в браузере. JSскрипт должен быть создан отдельным файлом и подключен к HTMLстранице;
5. Создайте JS-скрипт с примером использованием объектов и их прототипов;
6. Реализуйте несколько (2 или больше) классов и продемонстрируйте на них 3 основные парадигмы ООП (наследование, инкапсуляция и полиморфизм);
7. Добавьте метод и поле в уже созданный класс из задания 2 с помощью prototype;
8. Определить класс «Трапеция», имеющий координаты 4-х точек, конструктор и метод для расчета площади трапеции. Создать массив экземпляров этого класса и определить, в скольких трапециях площадь больше заданного числа x;
9. Выполнить задание 8 с использованием объектов;

Выполнение работы:

2. Создать собственный JS-скрипт с использованием переменных и условным оператором («if» и «?»). Запустить скрипт в браузере. JS-скрипт должен быть создан отдельным файлом и подключен к HTML-странице;

Скрипт:

alert('Попробуйте сыграть в случайное число, для этого укажите диапазон');

let range = prompt("Введите диапазон");

let certainNumber = prompt("Введите число из диапазона");

let randomNumber = Math.floor(Math.random() \* range);

if (randomNumber == certainNumber){

    alert("Вам повезло в случайном числе");

}

else {

    Math.abs(certainNumber - randomNumber) == 1

        ? alert("Вы были очень близки")

        : alert("Вы даже близко не угадали число");

}

<html>

    <script src="number2.js">

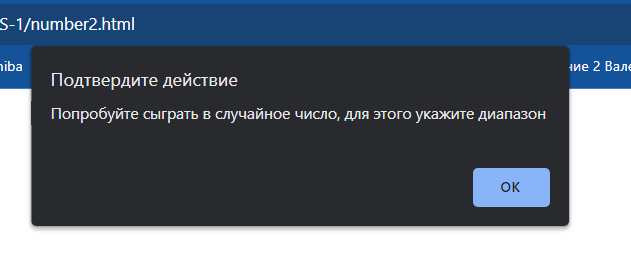
    </script>

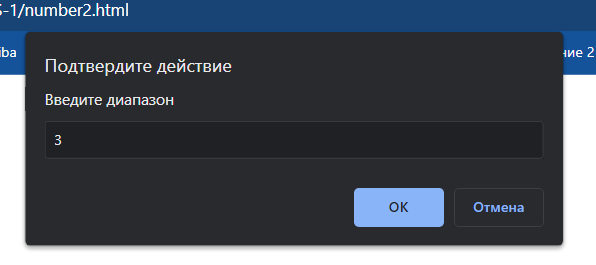
    <body>

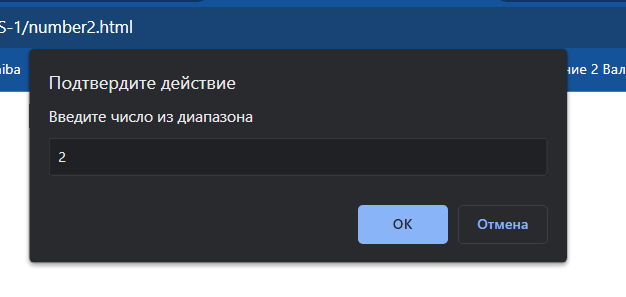
        <h3>Номер 2</h3>

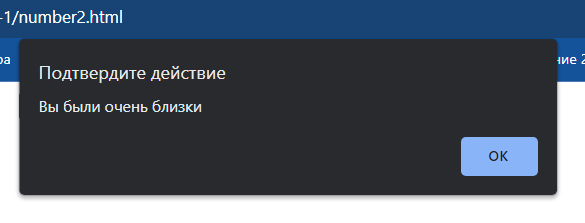
    </body>

</html>









1. Выведите простые числа с помощью цикла for на интервале от 2 до n, где n любое число больше 10

let n = 11;

console.log(`Простые числа от 2 до ${n}`);

for(let i = 2; i <= n; i++)

{

    let temp = 2;

    while (i % temp != 0){

        temp += 1

    }

    if (i == temp){

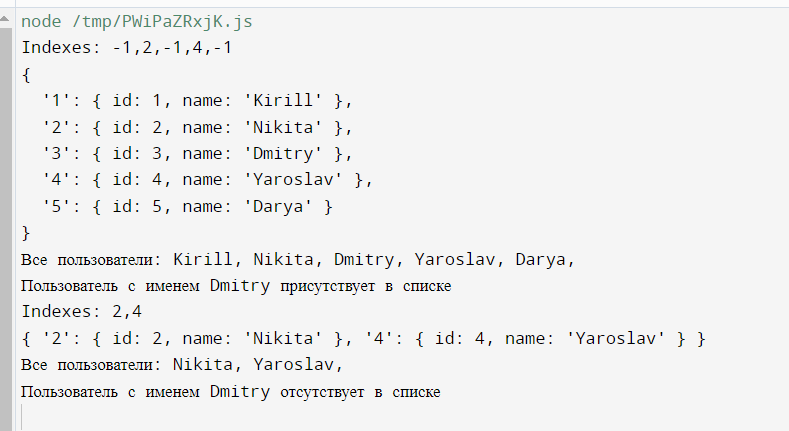
        console.log(i);

    }

}



1. Создать собственный JS-скрипт с примерами использования forEach(), map(), some(), reduce(). Запустить скрипт в браузере. JSскрипт должен быть создан отдельным файлом и подключен к HTMLстранице;
2. const users = [
3. { id: 1, name: "Kirill" },
4. { id: 2, name: "Nikita" },
5. { id: 3, name: "Dmitry" },
6. { id: 4, name: "Yaroslav"},
7. { id: 5, name: "Darya" }
8. ]
9. const usersOdd = users.filter((x) => {
10. if (x.id % 2 == 0){
11. return x;
12. }
13. });
14. usersOutput(users);
15. usersOutput(usersOdd);
16. function usersOutput(users){
17. let string = '';
18. users.forEach(x => string += x.name + ', ');
19. const indexes = users.map((x) => x.id % 2 == 0 ? x.id : -1);
20. alert(`Indexes: ${indexes}`);
21. const indexedUsers = users.reduce((x, y) => {
22. const { id } = y;
23. return { ...x, [id]: y}
24. }, {});
25. console.log(indexedUsers);
26. alert(`Все пользователи: ${string}`);
27. const certainName = "Dmitry";
28. if (users.some((x) => x.name == certainName)){
29. alert(`Пользователь с именем ${certainName} присутствует в списке`);
30. }
31. else {
32. alert(`Пользователь с именем ${certainName} отсутствует в списке`);
33. }
34. }



5. Создайте JS-скрипт с примером использованием объектов и их прототипов;

const berry = {

    isSmall : true,

    isEatable: true

}

const strawberry = {

    color: "red",

}

strawberry.\_\_proto\_\_ = berry;

const melon = {

    isSmall: false,

    color: "green",

}

melon.\_\_proto\_\_ = berry;

const honeysuckle = {

    isEatable: false,

    color: "black"

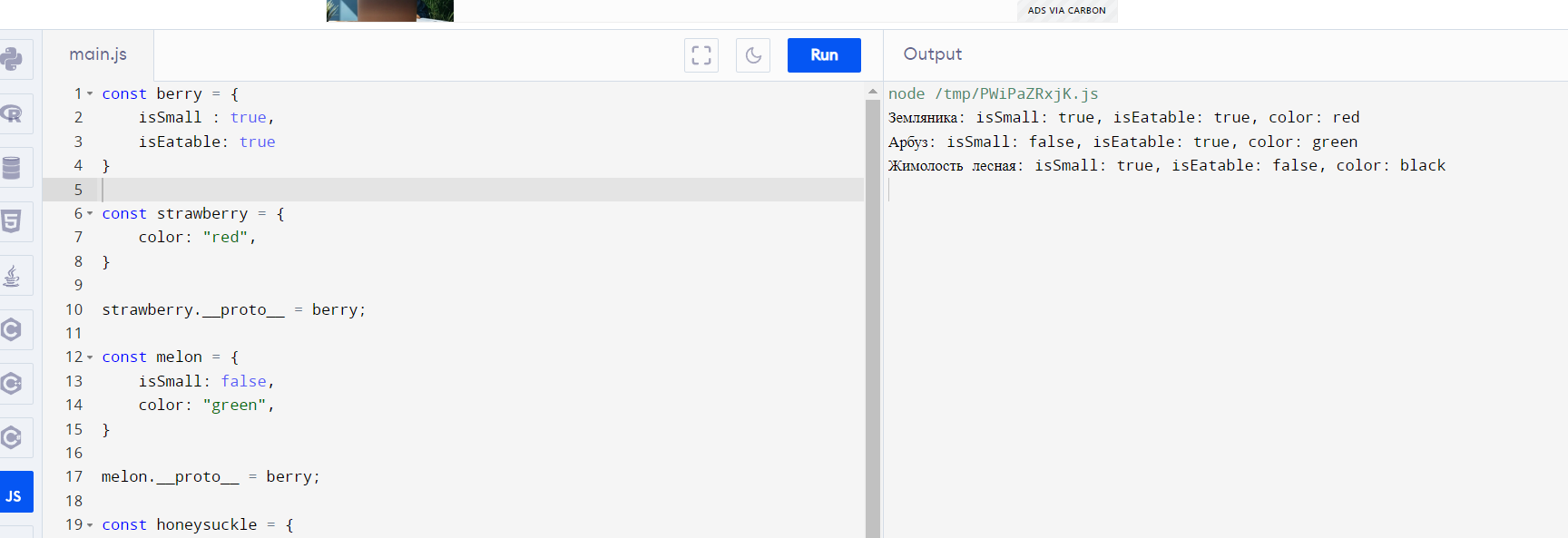
}

honeysuckle.\_\_proto\_\_ = berry;

console.log(`Земляника: isSmall: ${strawberry.isSmall}, isEatable: ${strawberry.isEatable}, color: ${strawberry.color}`);

console.log(`Арбуз: isSmall: ${melon.isSmall}, isEatable: ${melon.isEatable}, color: ${melon.color}`);

console.log(`Жимолость лесная: isSmall: ${honeysuckle.isSmall}, isEatable: ${honeysuckle.isEatable}, color: ${honeysuckle.color}`);



6. Реализуйте несколько (2 или больше) классов и продемонстрируйте на них 3 основные парадигмы ООП (наследование, инкапсуляция и полиморфизм);

class Vehicle{

    #mass;

    #speed;

    #wheelsCount;

    constructor(mass, speed, wheelsCount){

        this.#mass = mass;

        this.#speed = speed;

        this.#wheelsCount = wheelsCount;

    }

    descriptVehicle(){

    }

    get getMass(){

        return this.#mass;

    }

    get getSpeed(){

        return this.#mass;

    }

    get getWheelsCount(){

        return this.#mass;

    }

}

Vehicle.prototype.sayCommon = function() {

    console.log(`Im vehicle with

        mass = ${this.getMass} kg,

        speed = ${this.getSpeed} m/s,

        wheels count = ${this.getWheelsCount}`);

}

const orientation = {

    left : 'left',

    right : "right"

}

class Car extends Vehicle{

    #steeringOrientation;

    constructor(mass, speed, wheelsCount, steeringOrientation) {

        super(mass, speed, wheelsCount);

        this.#steeringOrientation = steeringOrientation;

    }

    descriptVehicle(){

        const description = `Its car with ${this.#steeringOrientation} rule orientation`;

        console.log(description);

    }

}

class Bus extends Vehicle{

    #floorsNumber;

    constructor(mass, speed, wheelsCount, floorsNumber) {

        super(mass, speed, wheelsCount);

        this.#floorsNumber = floorsNumber;

    }

    descriptVehicle(){

        const description = `Its bus with ${this.#floorsNumber} floors`;

    }

}

let car = new Car(2000, 200, 4, orientation.left);

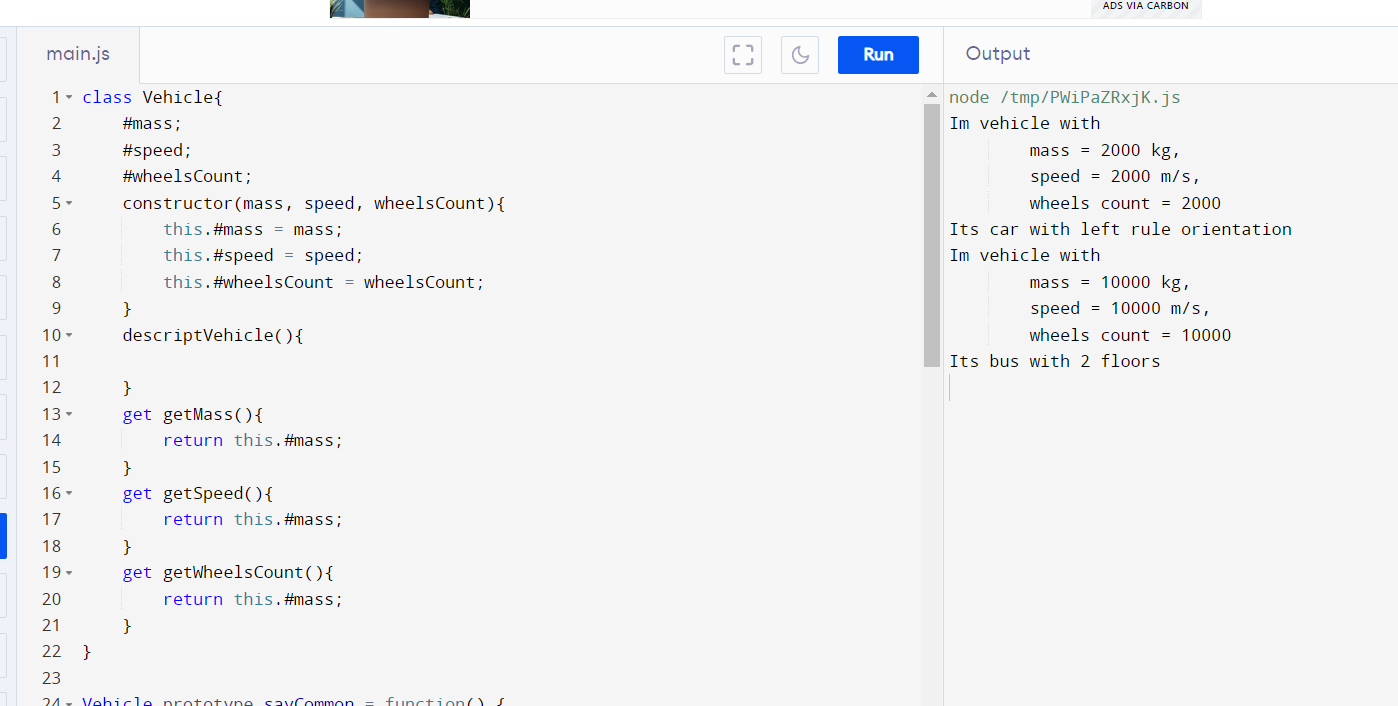
let bus = new Bus(10000, 100, 4, 2);

car.sayCommon();

car.descriptVehicle();

bus.sayCommon();

bus.descriptVehicle();



7. Добавьте метод и поле в уже созданный класс из задания 2 с помощью prototype;

class PlayRandomGame{

    #range

    #certainNumber

    playRandom()

    {

        alert('Попробуйте сыграть в случайное число, для этого укажите диапазон');

        this.#range = prompt("Введите диапазон");

        this.#certainNumber = prompt("Введите число из диапазона");

        let randomNumber = Math.floor(Math.random() \* range);

        if (randomNumber == certainNumber){

            alert("Вам повезло в случайном числе");

        }

        else {

            Math.abs(certainNumber - randomNumber) == 1

                ? alert("Вы были очень близки")

                : alert("Вы даже близко не угадали число");

        }

    }

}

PlayRandomGame.prototype.author = "Kirill";

PlayRandomGame.prototype.playRandomWithoutInput = function(){

    alert("Сейчас вы сыграете в случайную игру без ввода");

    const range = 5;

    const certainNumber = 2;

    const randomNumber = Math.floor(Math.random() \* range);

    if (randomNumber == certainNumber){

        alert("Вам повезло в случайном числе");

    }

    else {

        Math.abs(certainNumber - randomNumber) == 1

            ? alert("Вы были очень близки")

            : alert("Вы даже близко не угадали число");

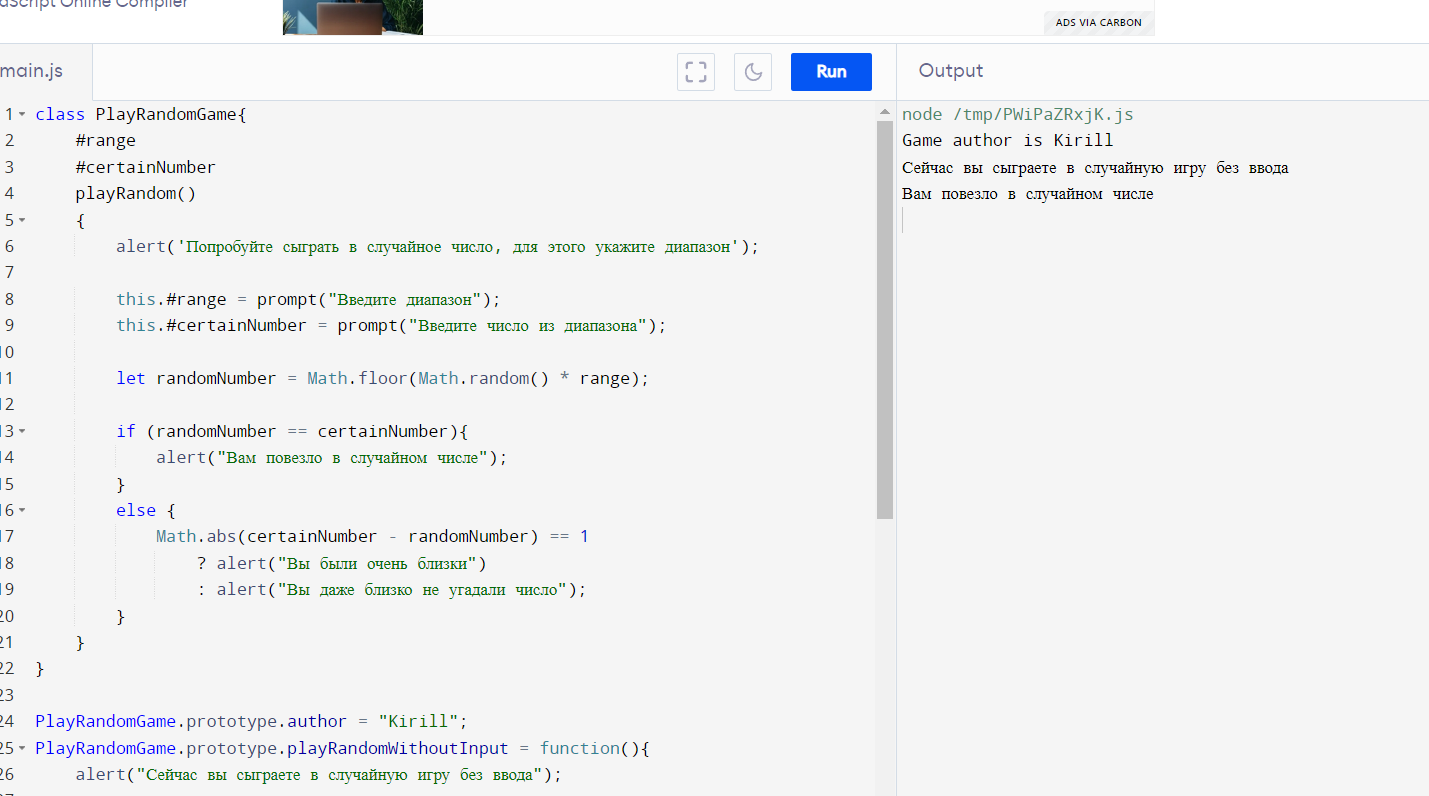
    }

}

var game = new PlayRandomGame();

console.log(`Game author is ${game.author}`);

game.playRandomWithoutInput();



8. Определить класс «Трапеция», имеющий координаты 4-х точек, конструктор и метод для расчета площади трапеции. Создать массив экземпляров этого класса и определить, в скольких трапециях площадь больше заданного числа x;

class Trapesia {

    #x1;

    #y1;

    #x2;

    #y2;

    #x3;

    #y3;

    #x4;

    #y4;

    constructor(x1, y1, x2, y2, x3, y3, x4, y4){

        this.#x1 = x1;

        this.#y1 = y1;

        this.#x2 = x2;

        this.#y2 = y2;

        this.#x3 = x3;

        this.#y3 = y3;

        this.#x4 = x4;

        this.#y4 = y4;

    }

    calculateSquare(){

        const a = Math.sqrt((this.#x2 - this.#x1)\*(this.#x2 - this.#x1)

            + (this.#y2 - this.#y1)\* (this.#y2 - this.#y1));

        const b = Math.sqrt((this.#x3 - this.#x2)\*(this.#x3 - this.#x2)

            + (this.#y3 - this.#y2)\* (this.#y3 - this.#y2));

        const c = Math.sqrt((this.#x4 - this.#x3)\*(this.#x4 - this.#x3)

            + (this.#y4 - this.#y3)\* (this.#y4 - this.#y3));

        const d = Math.sqrt((this.#x1 - this.#x4)\*(this.#x1 - this.#x4)

            + (this.#y1 - this.#y4)\* (this.#y1 - this.#y4));

        const p = (a + b + c + d) / 2;

        return Math.sqrt((p - a)\*(p - b)\*(p - c)\*(p - d));

    }

}

const range = 25;

const figures = [

    new Trapesia(1, 2, 3, 4, 3, 7, 1, 8),

    new Trapesia(1, 2, 32, 4, 3, 27, 1, 8),

    new Trapesia(11, 2, 3, 4, 3, 73, 1, 8),

    new Trapesia(1, 23, 3, 34, 3, 7, 0, 8),

    new Trapesia(1, 2, 5, 4, 3, 7, 1, 8)

];

const number = figures.reduce((x, y) => {

    var square = y.calculateSquare();

    console.log(square);

    if (square > range){

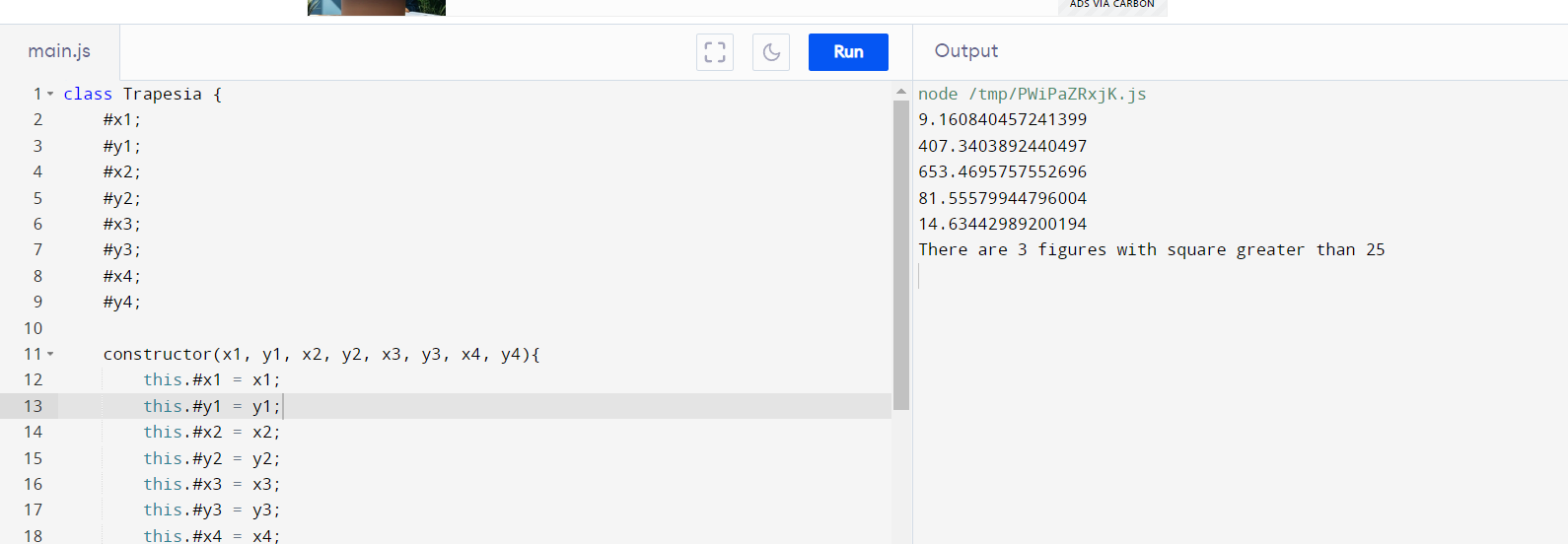
        return x + 1;

    }

    return x

}, 0);

console.log(`There are ${number} figures with square greater than ${range}`);



Выполнить задание 8 с использованием объектов (без использования возможности объявления отдельного класса);

const figure1 = {

    x1: 1,

    y1: 2,

    x2: 3,

    y2: 4,

    x3: 3,

    y3: 7,

    x4: 1,

    y4: 8,

    calculateSquare: function(){

        const a = Math.sqrt((this.x2 - this.x1)\*(this.x2 - this.x1)

            + (this.y2 - this.y1)\* (this.y2 - this.y1));

        const b = Math.sqrt((this.x3 - this.x2)\*(this.x3 - this.x2)

            + (this.y3 - this.y2)\* (this.y3 - this.y2));

        const c = Math.sqrt((this.x4 - this.x3)\*(this.x4 - this.x3)

            + (this.y4 - this.y3)\* (this.y4 - this.y3));

        const d = Math.sqrt((this.x1 - this.x4)\*(this.x1 - this.x4)

            + (this.y1 - this.y4)\* (this.y1 - this.y4));

        const p = (a + b + c + d) / 2;

        return Math.sqrt((p - a)\*(p - b)\*(p - c)\*(p - d));

    }

};

const range = 25;

const figure2 = Object.assign({}, figure1);

figure2.x2 = 32;

figure2.y3 = 27;

const figure3 = Object.assign({}, figure1);

figure3.x1 = 11;

figure1.y3 = 27;

const figure4 = Object.assign({}, figure1);

figure4.y1 = 23;

figure4.y2 = 34;

figure4.x4 = 0;

const figure5 = Object.assign({}, figure1);

figure5.x2 = 5;

const figures = [

    figure1,

    figure2,

    figure3,

    figure4,

    figure5

];

const number = figures.reduce((x, y) => {

    var square = y.calculateSquare();

    console.log(square);

    if (square > range){

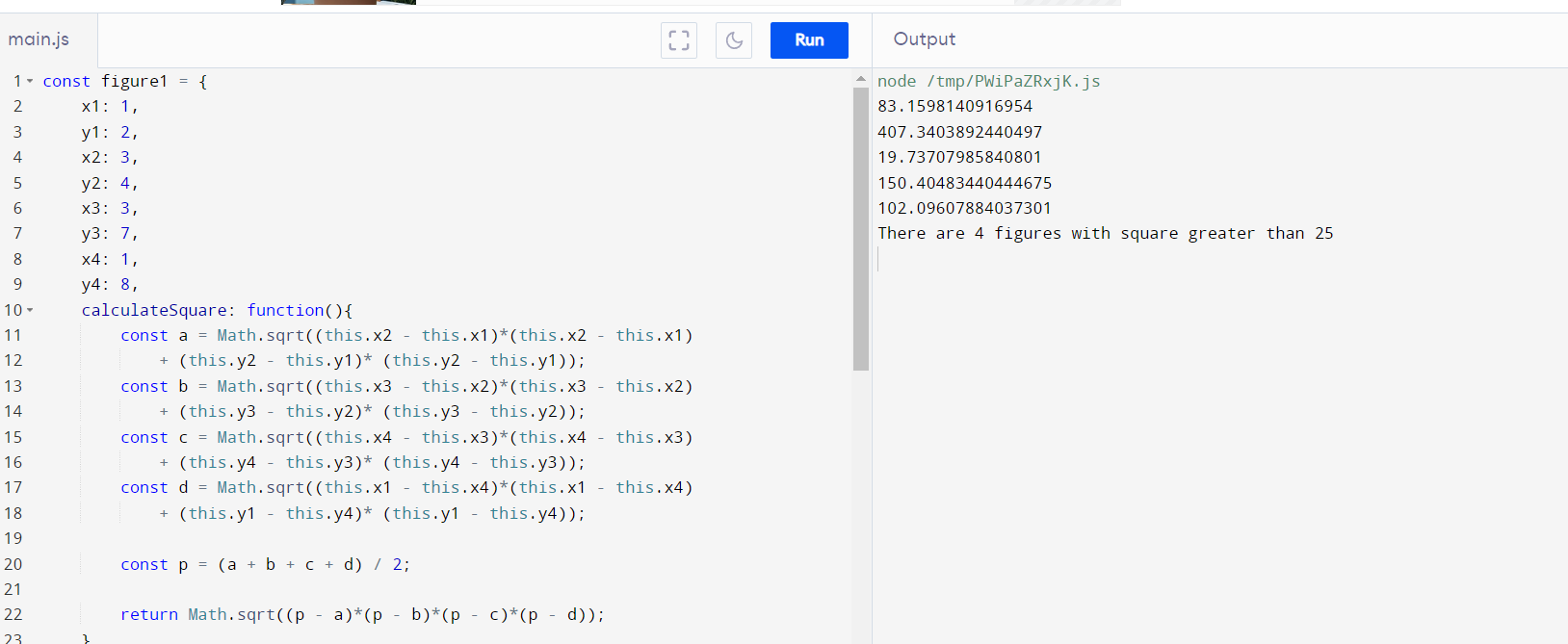
        return x + 1;

    }

    return x

}, 0);

console.log(`There are ${number} figures with square greater than ${range}`);



Вывод

В результате выполнения работы, я получил основные навыки работы с языком JavaScript, освоил основные типы данных и операторы JavaScript, а также изучил механизмы прототипирования и принципы ООП при работе с классами.