**МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ**

###### ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

###### КЕМЕРОВСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ

**ИНСТИТУТ ЦИФРЫ**

**ОТЧЁТ**

**О ВЫПОЛНЕНИИ ЛАБОРАТОРНОЙ РАБОТЫ**

«Объекты и классы»

Студентки 2 курса, ФИТ-211 группы

**Колесник Полины Олеговны**

Направление 02.03.02 – «Фундаментальная информатика и информационные технологии»

Руководитель:

Доцент Зимин А. И.

Работа защищена

« »

“ ” 2023 г.

Кемерово 2023 г.

**ОТЧЁТ О ПРОДЕЛАННОЙ РАБОТЕ**

**Объекты**

**1 задание**

function Object() {

    let user = {};

    user.name = "John";

    user.surname = "Smith"

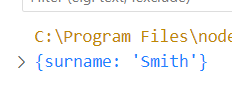
    user.name = "Pete"

    delete user.name

    console.log(user)

}

Object()



**2 задание**

function Object() {

    let MyBrowser = {

        name: "Microsoft Internet Explorer",

        version: "9.0"

    }

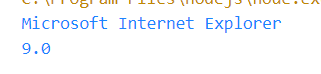
    for(let i in MyBrowser) {

        console.log(MyBrowser[i])

    }

}

Object()



**3 задание**

let obj = {

   // name: "mbkv"

}

function isEmpty(obj) {

    for (let i in obj) {

        if (obj.hasOwnProperty(i)) {

            return false

        }

    }

    return true

}

console.log(isEmpty(obj))

****

**4 задание**

const user = {

    name: "John"

    };

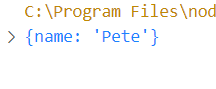
    // это будет работать?

    user.name = "Pete"; //работает

    // а это?

    //user = 123; нет

console.log(user)



**5 задание**

let obj = {

    name: "fmkd",

    age: 23,

    "number a": 45

}

function multiplyNumeric(obj) {

    for (let i in obj) {

        if(typeof obj[i] === 'number') {

            obj[i] \*= 2

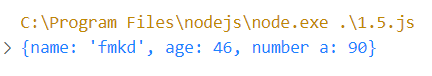
        }

    }

    console.log(obj)

}

multiplyNumeric(obj)



**6 задание**

let obj = {

    read(a, b) {

        this.a = a

        this.b = b

    },

    sum() {

        return this.a + this.b

    },

    mul() {

        return this.a \* this.b

    }

}

obj.read(15, 5)

console.log(obj.sum())

console.log(obj.mul())

****

**7 задание**

let ladder = {

    step: 0,

    up() {

        this.step++;

        return this;

    },

    down() {

        this.step--;

        return this;

    },

    showStep: function() { // показывает текущую ступеньку

        console.log(this.step);

        return this;

    }

};

ladder.up().up().down().showStep().down().showStep();



**8 задание**

'use strict';

function Browser(name, version ) {

    this.name = name;

    this.version = version;

    this.aboutBrowser = function() {

        console.log(this.name);

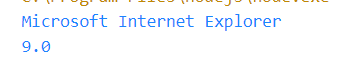
        console.log(this.version);

    }

}

let myBrowser = new Browser("Microsoft Internet Explorer", "9.0");

myBrowser.aboutBrowser();



**9 задание**

function Employee(name, division, phone, salary) {

    this.name = name

    this.division = division

    this.phone = phone

    this.salary = salary

    this.aboutEmp = function() {

        console.log(this.name + ", " + this.division + ", " + this.phone + ", " + this.salary)

    }

}

let emp = new Employee("Петров", "Лаборант", "8-800-555-35-35", "8950")

emp.aboutEmp()



**10 задание**

function Calculator() {

    this.read = function(a,b) {

        this.a = a

        this.b = b

    }

    this.sum = function() {

        return this.a + this.b

    }

    this.mul = function() {

        return this.a \* this.b

    }

}

let res = new Calculator()

res.read(11, 5)

console.log(res.sum())

console.log(res.mul())



**11 задание**

function Accumulator(stringValue) {

    this.value = stringValue;

    this.read = function(a) {

        this.a = a

        this.value +=  +this.a;

    }

}

let res = new Accumulator(1);

res.read(7)

res.read(9)

console.log(res.value)



**Работа с прототипами**

**1 задание**

let animal = {

    jumps: null

};

let rabbit = {

    \_\_proto\_\_: animal,

    jumps: true

};

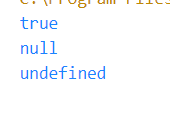
console.log(rabbit.jumps); // true, так как в объекте значение свойства было как true

delete rabbit.jumps;

console.log(rabbit.jumps); // null, так как после удаления свойства jumps, вывелось значение свойство объекта animal

delete animal.jumps;

console.log(rabbit.jumps); //undefined, так как ни в первом ни во втором объекте больше нет такого свойства



**2 задание**

let animal = {

    eat() {

        this.full = true;

    }

};

let rabbit = {

    \_\_proto\_\_: animal,

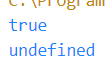
};

 //rabbit

rabbit.eat();

console.log(rabbit.full);

console.log(animal.full);



**3 задание**

let hamster = {

    stomach: [],

    eat(food) {

        this.stomach.push(food);

    }

};

let speedy = {

    \_\_proto\_\_: hamster,

    stomach: [] //свой животик

};

let lazy = {

    \_\_proto\_\_: hamster,

    stomach: [] //и у этого свой животик

};

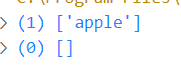
// Этот хомяк нашёл еду

speedy.eat("apple");

console.log(speedy.stomach); // apple

// У этого хомяка тоже есть еда. Почему? Исправьте

console.log(lazy.stomach); // apple



**4 задание**

// Добавление свойства по умолчанию к встроенному объекту

String.prototype.color = "black";

// Добавление (изменение) метода к встроенному объекту

String.prototype.size = "15";

String.prototype.write = stringWrite;

function stringWrite() {

    console.log("Цвет текста: " + this.color);

    console.log("Размер шрифта: " + this.size);

    console.log("Текст: " + this.toString());

}

// используем измененный класс

let s = new String("Это строка");

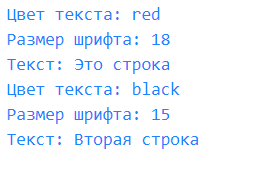
s.color = "red";

s.size = "18";

s.write();

let s2 = new String("Вторая строка");

s2.write();



**5 задание**

function Rabbit() {}

Rabbit.prototype = {

    eats: true

};

let rabbit = new Rabbit();

//Rabbit.prototype = {};  //true

//Rabbit.prototype.eats = false; //false ссылаемся на созданный объект, а не на прототип

//delete rabbit.eats; //true, т.к. пытаемся удалить свойство из объекта rabbit, а там и так его нет//

 delete Rabbit.prototype.eats; //undefined удалили свойство у прототипа, поэтому при вызове его уже не существует

console.log(rabbit.eats); // true

**Классы(Class)**

**1 задание**

class Clock {

    constructor(hour, minute, second) {

        this.hour = hour;

        this.minute = minute;

        this.second = second;

    }

    update() {

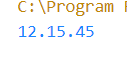
        console.log(this.hour +"."+ this.minute +"."+ this.second)

    }

}

let cl = new Clock(12, 15, 45);

cl.update();



**2 задание**

class Animal {

    constructor(name) {

        this.name = name;

    }

}

class Rabbit extends Animal {

    constructor(name) {

        //this.name = name;

        super(name);

        this.created = Date.now();

    }

}

let rabbit = new Rabbit("Белый кролик"); // Error: this is not defined

console.log(rabbit.name);

****

**3 задание**

class Clock {

    constructor(template) {

        this.template = template;

    }

    render() {

        let date = new Date();

        let hours = date.getHours();

        if (hours < 10) hours = '0' + hours;

        let mins = date.getMinutes();

        if (mins < 10) mins = '0' + mins;

        let secs = date.getSeconds();

        if (secs < 10) secs = '0' + secs;

        let output = this.template

            .replace('h', hours)

            .replace('m', mins)

            .replace('s', secs);

            console.log(output);

    }

    stop() {

        clearInterval(this.timer);

       }

    start() {

        this.render();

        this.timer = setInterval(() => this.render(), 1000);

    }

}

class ExtendedClock extends Clock {

    constructor(fdate) {

        super(fdate);

      let { precision = 1000 } = fdate;

      this.precision = precision;

    }

    start() {

      this.render();

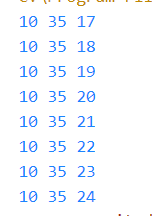
      this.timer = setInterval(() => this.render(), this.precision);

    }

  };

  let clock = new Clock("h m s");

  clock.start();

****

**4 задание**

class Stock {

    constructor() {

        this.listW = [];

        this.listV = [];

    }

    add(w, v) {

        this.listW.push(w);

        this.listV.push(v);

    }

    getByW(min\_w) {

        let setW = new Set();

        for (let index = 0; index < this.listW.length; index++) {

            if (this.listW[index] < min\_w) {

                setW.add(this.listW[index]);

            }

        }

        let min = Math.min( ...setW );

        return this.listW.indexOf(min);

    }

    getByV(min\_v) {

        let setV = new Set();

        for (let index = 0; index < this.listV.length; index++) {

            if (this.listV[index] < min\_v) {

                setV.add(this.listV[index]);

            }

        }

        let min = Math.min( ...setV );

        return this.listV.indexOf(min);

    }

}

let stock1 = new Stock();

stock1.add(5, 7);

stock1.add(4, 8);

stock1.add(1, 8);

console.log(stock1.getByV(2));

console.log(stock1.getByW(6));

