**Основы JS**

**Задание 1.** Нужно написать функцию, которая возвращает тип входного аргумента

На входе: один аргумент, любого значения

На выходе: строка, с типом входного аргумента

**Код**:

function getArgumentType(arg) {

  return typeof arg;

}

// входные данные

console.log(getArgumentType(3)); // "number"

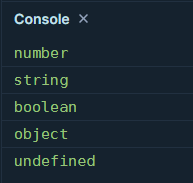
console.log(getArgumentType("Google")); // "string"

console.log(getArgumentType(true)); // "boolean"

console.log(getArgumentType(null)); // "object"

console.log(getArgumentType(undefined)); // "undefined"

**Вывод:**



**Задание 2.** Что выведется в консоль и почему?

const a = () => console.log("a");

const b = () => console.log("b");

const c = a || b ? a() : b();

console.log(c);

**Вывод:**

****

**Пояснение:**

1. Создаются две стрелочные функции a и b, которые выводят в консоль строки "a" и "b" соответственно.

2. Далее происходит вычисление выражения a || b. Оператор «||» выполняет логическое ИЛИ. В данном случае, так как функции a и b являются истинными значениями, результатом выражения будет «a», т.к. «a» находится слева от оператора.

3. Затем вызывается функция a() и выводится в консоль «a».

4. Но, т. к. функция a() не возвращает явно никакого значения, результатом вызова будет undefined.

5. Результат, который присваивается переменной c, будет «undefined», что и выводится в консоль.

**Функции**

**Задание 1.** Нужно реализовать функцию, возвращающую сумму всех аргументов, если тип аргумента - число. Вызовов ВСЕГДА 5

**Код:**

function p(num) {

  if (typeof num !== 'number') {

    return p;

  }

  let sum = num;

  function nextp(nextNum) {

    if (typeof nextNum === 'number') {

      sum += nextNum;

    }

    return nextp;

  }

  nextp.toString = function () {

    return sum.toString();

  };

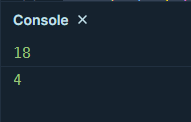
  return nextp;

}

console.log(p(3)(2)(1)(5)(7).toString()); // входные данные

console.log(p(3)(null)(1)('asad')().toString()); // входные данные

**Вывод:**

****

**Объекты и Массивы**

**Задание 1.** Реализовать функцию convertArrToObj, которая на вход принимаетмассив значений, а на выход отдает объект, где ключ — это индекс элемента вмассиве, а значение - сам элемент.

**Код:**

function convertArrToObj(arr) {

  const obj = {};

  for (let i = 0; i < arr.length; i++) {

    obj[i] = arr[i];

  }

  return obj;

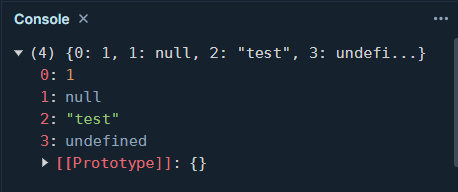
}

const arr = [1, null, 'test', undefined];

const obj = convertArrToObj(arr);

console.log(obj); // { 0: 1, 1: null, 2: 'test', 3: undefined }

**Вывод:**

****

**Задание 2.** Требуется реализовать функцию countFromArr, которая позволяет

подсчитать значения массива. На вход поступает массив, на выход получаем

объект, где ключ — это элемент массива, а значение - его кол-во в массиве.

**Код:**

function countFromArr(arr) {

  const x = {};

  for (let i = 0; i < arr.length; i++) {

    const elmnt = arr[i];

    if (x[elmnt]) {

      x[elmnt]++;

    } else {

      x[elmnt] = 1;

    }

  }

  return x;

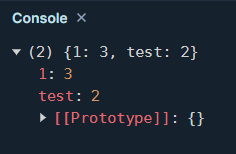
}

const arr = [1, 1, 1, 'test', 'test'];

const res = countFromArr(arr);

console.log(res); // { 1: 3, test: 2 }

**Вывод:**

****

**Задание 3**. Требуется реализовать функцию groupByField, которая позволяет

сгруппировать массив объектов по определенному полю этого объекта.

На вход поступает массив объектов и строка с названием нужного поля, на выход

объект - где ключом является значение поля объекта, а значением - массив

объектов, где значение в поле равно этому ключу.

**Код:**

function groupByField(arr, field) {

const result = {};

for (const obj of arr) {

const value = obj[field];

if (value) {

if (result[value]) {

result[value].push(obj);

} else {

result[value] = [obj];

}

}

}

return result;

}

const arr = [{test: 1}, {test: 2}, {test: 3}, {test: 1}, {test: 1}];

const res = groupByField(arr, 'test');

console.log(res); // { 1: [{test: 1}, {test: 1}, {test: 1}], 2: [{test: 2}], 3: [{test: 3}] }

**Вывод:**

