**Сумма свойств**

Есть объект salaries с зарплатами. Напишите код, который выведет сумму всех зарплат.

Если объект пустой, то результат должен быть 0.

Например:

var salaries = {

"Вася": 100,

"Петя": 300,

"Даша": 250

};

//... ваш код выведет 650

**Свойство с наибольшим значением**

Есть объект salaries с зарплатами. Напишите код, который выведет имя сотрудника, у которого самая большая зарплата.

Если объект пустой, то пусть он выводит «нет сотрудников».

Например:

"use strict";

var salaries = {

"Вася": 100,

"Петя": 300,

"Даша": 250

};

// ... ваш код выведет "Петя"

**Умножьте численные свойства на 2**

Создайте функцию multiplyNumeric, которая получает объект и умножает все численные свойства на 2. Например:

// до вызова

var menu = {

width: 200,

height: 300,

title: "My menu"

};

multiplyNumeric(menu);

// после вызова

menu = {

width: 400,

height: 600,

title: "My menu"

};

P.S. Для проверки на число используйте функцию:

function isNumeric(n) {

return !isNaN(parseFloat(n)) && isFinite(n)

}

**Создайте калькулятор для введённых значений**

Напишите код, который:

Запрашивает по очереди значения при помощи prompt и сохраняет их в массиве.

Заканчивает ввод, как только посетитель введёт пустую строку, не число или нажмёт «Отмена».

При этом ноль 0 не должен заканчивать ввод, это разрешённое число.

Выводит сумму всех значений массива

**Поиск в массиве**

Создайте функцию find(arr, value), которая ищет в массиве arr значение value и возвращает его номер, если найдено, или -1, если не найдено.

Например:

arr = ["test", 2, 1.5, false];

find(arr, "test"); // 0

find(arr, 2); // 1

find(arr, 1.5); // 2

find(arr, 0); // -1

Открыть песочницу с тестами для задачи.

**Фильтр диапазона**

Создайте функцию filterRange(arr, a, b), которая принимает массив чисел arr и возвращает новый массив, который содержит только числа из arr из диапазона от a до b. То есть, проверка имеет вид a ≤ arr[i] ≤ b. Функция не должна менять arr.

Пример работы:

var arr = [5, 4, 3, 8, 0];

var filtered = filterRange(arr, 3, 5);

// теперь filtered = [5, 4, 3]

// arr не изменился

Открыть песочницу с тестами для задачи.

**Решето Эратосфена**

Целое число, большее 1, называется простым, если оно не делится нацело ни на какое другое, кроме себя и 1.

Древний алгоритм «Решето Эратосфена» для поиска всех простых чисел до n выглядит так:

Создать список последовательных чисел от 2 до n: 2, 3, 4, ..., n.

Пусть p=2, это первое простое число.

Зачеркнуть все последующие числа в списке с разницей в p, т.е. 2\*p, 3\*p, 4\*p и т.д. В случае p=2 это будут 4,6,8....

Поменять значение p на первое не зачеркнутое число после p.

Повторить шаги 3-4 пока p2 < n.

Все оставшиеся не зачеркнутыми числа – простые.

Посмотрите также анимацию алгоритма.

Реализуйте «Решето Эратосфена» в JavaScript, используя массив.

Найдите все простые числа до 100 и выведите их сумму.

**Подмассив наибольшей суммы**

На входе массив чисел, например: arr = [1, -2, 3, 4, -9, 6].

Задача – найти непрерывный подмассив arr, сумма элементов которого максимальна.

Ваша функция должна возвращать только эту сумму.

Например:

getMaxSubSum([-1, 2, 3, -9]) = 5 (сумма выделенных)

getMaxSubSum([2, -1, 2, 3, -9]) = 6

getMaxSubSum([-1, 2, 3, -9, 11]) = 11

getMaxSubSum([-2, -1, 1, 2]) = 3

getMaxSubSum([100, -9, 2, -3, 5]) = 100

getMaxSubSum([1, 2, 3]) = 6 (неотрицательные - берем всех)

Если все элементы отрицательные, то не берём ни одного элемента и считаем сумму равной нулю:

getMaxSubSum([-1, -2, -3]) = 0

**Добавить класс в строку**

В объекте есть свойство className, которое содержит список «классов» – слов, разделенных пробелом:

var obj = {

className: 'open menu'

}

Создайте функцию addClass(obj, cls), которая добавляет в список класс cls, но только если его там еще нет:

addClass(obj, 'new'); // obj.className='open menu new'

addClass(obj, 'open'); // без изменений (класс уже существует)

addClass(obj, 'me'); // obj.className='open menu new me'

alert( obj.className ); // "open menu new me"

P.S. Ваша функция не должна добавлять лишних пробелов.

**Перевести текст вида border-left-width в borderLeftWidth**

Напишите функцию camelize(str), которая преобразует строки вида «my-short-string» в «myShortString».

То есть, дефисы удаляются, а все слова после них получают заглавную букву.

Например:

camelize("background-color") == 'backgroundColor';

camelize("list-style-image") == 'listStyleImage';

camelize("-webkit-transition") == 'WebkitTransition';

Такая функция полезна при работе с CSS.

P.S. Вам пригодятся методы строк charAt, split и toUpperCase.

**Функция removeClass**

У объекта есть свойство className, которое хранит список «классов» – слов, разделенных пробелами:

var obj = {

className: 'open menu'

};

Напишите функцию removeClass(obj, cls), которая удаляет класс cls, если он есть:

removeClass(obj, 'open'); // obj.className='menu'

removeClass(obj, 'blabla'); // без изменений (нет такого класса)

Функция должна корректно обрабатывать дублирование класса в строке:

obj = {

className: 'my menu menu'

};

removeClass(obj, 'menu');

alert( obj.className ); // 'my'

Лишних пробелов после функции образовываться не должно.

**Фильтрация массива "на месте"**

Создайте функцию filterRangeInPlace(arr, a, b), которая получает массив с числами arr и удаляет из него все числа вне диапазона a..b. То есть, проверка имеет вид a ≤ arr[i] ≤ b. Функция должна менять сам массив и ничего не возвращать.

Например:

arr = [5, 3, 8, 1];

filterRangeInPlace(arr, 1, 4); // удалены числа вне диапазона 1..4

alert( arr ); // массив изменился: остались [3, 1]

**Сортировать в обратном порядке**

Как отсортировать массив чисел в обратном порядке?

var arr = [5, 2, 1, -10, 8];

// отсортируйте

alert( arr ); // 8, 5, 2, 1, -10

**Скопировать и отсортировать массив**

Есть массив строк arr. Создайте массив arrSorted – из тех же элементов, но отсортированный.

Исходный массив не должен меняться.

var arr = ["HTML", "JavaScript", "CSS"];

// ... ваш код ...

alert( arrSorted ); // CSS, HTML, JavaScript

alert( arr ); // HTML, JavaScript, CSS (без изменений)

Постарайтесь сделать код как можно короче.

**Случайный порядок в массиве**

Используйте функцию sort для того, чтобы «перетрясти» элементы массива в случайном порядке.

var arr = [1, 2, 3, 4, 5];

arr.sort(ваша функция);

alert( arr ); // элементы в случайном порядке, например [3,5,1,2,4]

**Сортировка объектов**

Напишите код, который отсортирует массив объектов people по полю age.

Например:

var vasya = { name: "Вася", age: 23 };

var masha = { name: "Маша", age: 18 };

var vovochka = { name: "Вовочка", age: 6 };

var people = [ vasya , masha , vovochka ];

... ваш код ...

// теперь people: [vovochka, masha, vasya]

alert(people[0].age) // 6

Выведите список имён в массиве после сортировки.

**Вывести односвязный список**

Односвязный список – это структура данных, которая состоит из элементов, каждый из которых хранит ссылку на следующий. Последний элемент может не иметь ссылки, либо она равна null.

Например, объект ниже задаёт односвязный список, в next хранится ссылка на следующий элемент:

var list = {

value: 1,

next: {

value: 2,

next: {

value: 3,

next: {

value: 4,

next: null

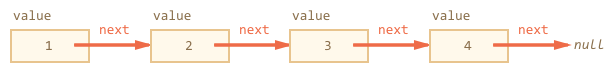
}

}

}

};

Графическое представление этого списка:



Альтернативный способ создания:

var list = { value: 1 };

list.next = { value: 2 };

list.next.next = { value: 3 };

list.next.next.next = { value: 4 };

Такая структура данных интересна тем, что можно очень быстро разбить список на части, объединить списки, удалить или добавить элемент в любое место, включая начало. При использовании массива такие действия требуют обширных перенумерований.

Задачи:

Напишите функцию printList(list), которая выводит элементы списка по очереди, при помощи цикла.

Напишите функцию printList(list) при помощи рекурсии.

Напишите функцию printReverseList(list), которая выводит элементы списка в обратном порядке, при помощи рекурсии. Для списка выше она должна выводить 4,3,2,1

Сделайте вариант printReverseList(list), использующий не рекурсию, а цикл.

**Как лучше – с рекурсией или без?**

**Отфильтровать анаграммы**

Анаграммы – слова, состоящие из одинакового количества одинаковых букв, но в разном порядке. Например:

воз - зов

киборг - гробик

корсет - костер - сектор

Напишите функцию aclean(arr), которая возвращает массив слов, очищенный от анаграмм.

Например:

var arr = ["воз", "киборг", "корсет", "ЗОВ", "гробик", "костер", "сектор"];

alert( aclean(arr) ); // "воз,киборг,корсет" или "ЗОВ,гробик,сектор"

Из каждой группы анаграмм должно остаться только одно слово, не важно какое.

**Оставить уникальные элементы массива**

Пусть arr – массив строк.

Напишите функцию unique(arr), которая возвращает массив, содержащий только уникальные элементы arr.

Например:

function unique(arr) {

/\* ваш код \*/

}

var strings = ["кришна", "кришна", "харе", "харе",

"харе", "харе", "кришна", "кришна", "8-()"

];

alert( unique(strings) ); // кришна, харе, 8-()