Домашнее задание (массивы - функции)

Задание 1

принимает значение, которым заполнять массив, а вторым - сколько элементов должно быть в массиве. Пример: arrayFill('x', 5) сделает массив ['x', 'x', 'x', 'x', 'x'].

Задание 2

Дан двухмерный массив с числами, например [[1, 2, 3], [4, 5], [6]]. Найдите сумму элементов этого массива. Массив, конечно же, может быть произвольным.

Задание 3

Дан трехмерный массив с числами, например [[[1, 2], [3, 4]], [[5, 6], [7, 8]]]. Найдите сумму элементов этого массива. Массив, конечно же, может быть произвольным.

Задание 4

Сделайте функцию isNumberInRange, которая параметром принимает число и проверяет, что оно больше нуля и меньше 10. Если это так - пусть функция возвращает true, если не так - false.

Задание 5

Сделайте функцию isEven() (even - это четный), которая параметром принимает целое число и проверяет: четное оно или нет. Если четное - пусть функция возвращает true, если нечетное - false.

Задание 6

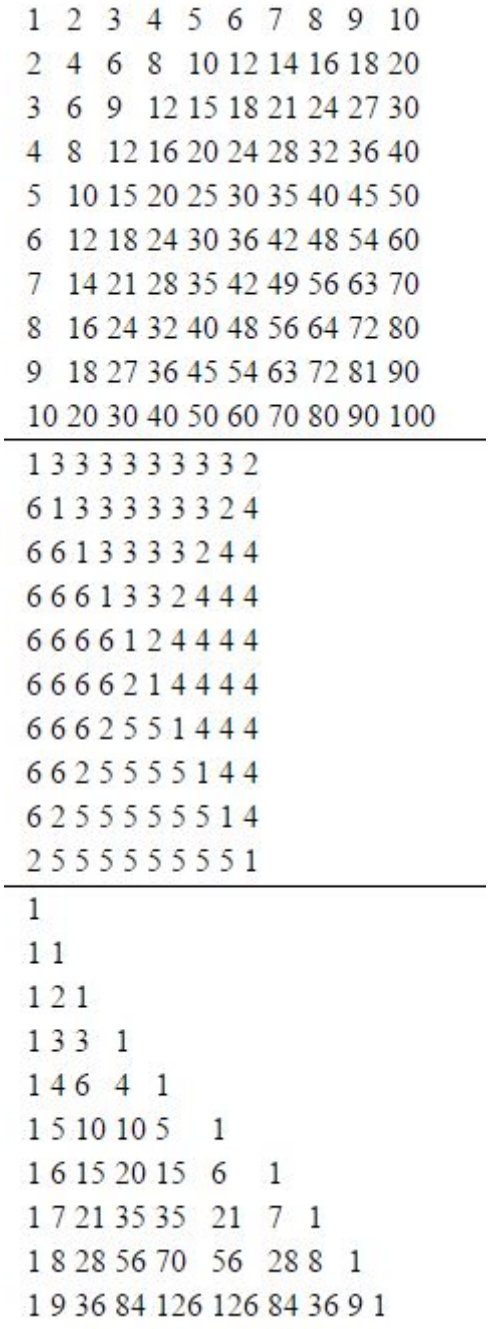
Дан массив с целыми числами. Создайте из него новый массив, где останутся лежать только четные из этих чисел. Для этого используйте вспомогательную функцию isEven из предыдущей задачи.

Задание 7

Сделайте функцию getDivisors, которая параметром принимает число и возвращает массив его делителей (чисел, на которое делится данное число).

Задание 8

Сформировать 3 матрицы изображенные на картинке



Задание 9

Создайте функцию, которая получает два параметра – число и степень числа. Используя Math.Pow внутри функции, возведите число в степень и выведите с помощью alert.

Задание 10

Функция принимает параметр - возраст пользователя. Если число больше 16 – то выводим «добро пожаловать», если меньше – “вы еще молоды”.

Задание 11

Модифицируйте предыдущий пример – учтите вариант, если пользователь не передал параметр в функцию. В таком случае выведите сообщение – “Введите возраст”. Реализуйте два вида проверки наличия аргумента – проверка на undefined и оператор ||.

Задание 12

Создайте функцию, которая считает длину массива и возвращает ее в виде числа. Массив в функцию передается как аргумент. Если аргумент не задан – выводится сообщение об ошибке.

Задание 13

Пользователь вводит числа. Если число больше 10, то функция возвращает квадрат числа, если меньше 7 – пишет, что число меньше 7. Если 8, 9 – то возвращает соответственно 7 или 8. Реализуйте решение с несколькими return.

Задание 14

Дана строка. Сделайте заглавным первый символ каждого слова этой строки. Для этого сделайте вспомогательную функцию `ucfirst`, которая будет получать строку, делать первый символ этой строки заглавным и возвращать обратно строку с заглавной первой буквой.

Задание 15

Дана строка вида `'var_text_hello'`. Сделайте из него текст `'varTextHello'`.

Задание 15

Сделайте функцию `inArray`, которая определяет, есть в массиве элемент с заданным текстом или нет. Функция первым параметром должна принимать текст элемента, а вторым - массив, в котором делается поиск. Функция должна возвращать `true` или `false`. Показать решение.

Задание 16

Дана строка, например, `'123456'`. Сделайте из нее `'214365'`.

Задание 17

Напиши функцию создания генератора `sequence(start, step)`. Она при вызове возвращает другую функцию-генератор, которая при каждом вызове дает число на 1 больше, и так до бесконечности. Начальное число, с которого начинать отсчет, и шаг, задается при создании генератора. Шаг можно не указывать, тогда он будет равен одному. Начальное значение по умолчанию равно 0. Генераторов можно создать сколько угодно.

```
var generator = sequence(10, 3);
var generator2 = sequence(7, 1);

console.log(generator()); // 10
console.log(generator()); // 13

console.log(generator2()); // 7

console.log(generator()); // 16

console.log(generator2()); // 8
```

Задание 18

Также, нужна функция `take(gen, x)` которая вызвает функцию `gen` заданное число (`x`) раз и возвращает массив с результатами вызовов.

```
var gen2 = sequence(0, 2);
console.log(take(gen2, 5)); // [0, 2, 4, 6, 8 ]
```

Задание 19

Напиши функцию `map(fn, array)`, которая принимает на вход функцию и массив, и обрабатывает каждый элемент массива этой функцией, возвращая новый массив. Пример:

```
function square(x) { return x * x; } // возведение в квадрат
console.log(map(square, [1, 2, 3, 4])); // [1, 4, 9, 16]
console.log(map(square, [])); // []
```

Обрати внимание: функция не должна изменять переданный ей массив:

```
var arr = [1, 2, 3];
console.log(map(square, arr)); // [1, 4, 9]
console.log(arr); // [1, 2, 3]
```

Задание 20

Напиши функцию `fmap(a, gen)`, которая принимает на вход 2 функции, `a` и `gen`, где `gen` — функция-генератор вроде той, что была в 17 задании. `fmap` возвращает новую функцию-генератор, которая при каждом вызове берет следующее значение из `gen` и пропускает его через функцию `a`. Пример:

```
var gen = sequence(1, 1);
function square(x) { return x * x; }
var squareGen = fmap(square, gen);

console.log(squareGen()); // 1
console.log(squareGen()); // 4
console.log(squareGen()); // 9
console.log(squareGen()); // 16
```

А, еще, сделайте тогда, чтобы в качестве `gen` можно было указать функцию с аргументами, и при вызове

```
function add(a, b) {
    return a + b;
}

// Мы получаем новую функцию, которая вызвает add, и результат пропускает через функцию square
var squareAdd = fmap(square, add);
console.log(squareAdd(2, 3)); // 25 = (2 + 3) ^ 2
console.log(squareAdd(5, 7)); // 144 = (5 + 7) ^ 2
```

Эти аргументы бы передавались функции `gen`. Аргументов может быть любое количество.

Подсказка: если непонятно, как сделать функцию, принимающую произвольное число аргументов, то стоит погуглить про псевдопеременную `arguments` (псевдопеременная значит, что она выглядит как переменная, но формально ей не является). Чтобы понять, как вызвать функцию с заранее неизвестным числом аргументов, можно погуглить `Function.prototype.call` и `Function.prototype.apply`. В JS функции - это объекты, и у них есть полезные методы и свойства.

Задание 21

Написать функцию `filter`, которая принимает функцию-предикат и массив. Возвращает она массив значений, для которых предикат вернет true.

```
var input = [1, 2, 3, 4, 5, 6];
function isEven(x) { return x % 2 == 0; } // проверяет на четность
console.log(filter(input, isEven)); // [2, 4, 6]
```

Функция не должна изменять исходный массив