1. Написать Array#duplicate(), такую что:

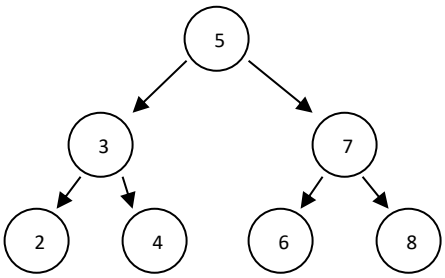
```
var list= [1, 2, 3, …, N];
list.duplicate();
console.log(list); // 1, 2, 3, …, N, 1, 2, 3, …, N
```

2. Имеется следующая структура, представляющая собой элемент дерева:

```
var node = {
   value: 9, // Числовое значение элемента
   left: {…}, // Указатель на левого потомка
   right: {…} // Указатель на правого потомка
}
```

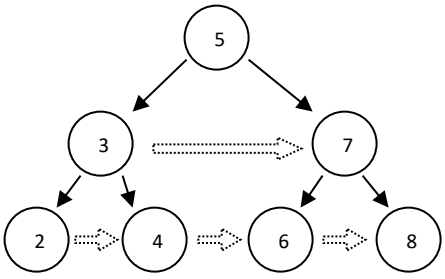
В свойствах ***left*** и ***right*** находятся такие же структуры, представляющие потомков элемента. Если потомка нету в свойстве находится значение ***null***.

Написать функцию ***tree_to_string(parent)***, проходящую по всему дереву и возвращающую строку со значениями всех элементов дерева через запятую в любом порядке. Например, для структуры:



функция должна вернуть, например, строку "2, 4,3,6,8,7,5".

3. Усложнение для задачи 2. Написать реализацию функции, проходящую дерево "в ширину" (сначала первый уровень слева направо, потом и второй и т.д.).
Результатом для следующего дерева должна быть строка "5, 3, 7, 2, 4, 6, 8".



4. Написать функцию **run_parallel()**, принимающую на вход массив функций-задач, которые выполняются асинхронно. Функция должна запустить параллельное выполнение всех задач и оповестить, когда все задачи будут завершены. Предложите свой интерфейс для функции run_parallel() и для функции-задачи.
5. Написать функцию **spy()** принимающую на вход оригинальную функцию или метод. create_spy_fn() должна возвращать функцию-декоратор над оригинальной функцией и должна дать возможность подсчитать количество вызовов этого декоратора без использования глобальной переменной. Функция-декоратор должна вызывать оригинальную функцию с сохранением её контекста и любого количества параметров.
6. Написать функцию для вращения матрицы NxN на 90 градусов по часовой стрелке. Вращение должно происходить в исходной матрице, без создания вспомогательных матриц и массивов.

7. Написать функцию, которая выводит в консоль все возможные варианты расстановки + и - между цифрами от 1 до 9, так чтобы в результате вычислений получалось 100 (например, 123+4-5+67-89=100).
8. Усложнение для задачи 7. Написать два варианта решения: наиболее компактный и максимально быстрый.
9. Написать генератор случайных лабиринтов. В лабиринте должен быть один и только один путь прохода из любой точки в любую. Функция **create_labyrinth(n, m)** принимает в себя ширину и высоту лабиринта и возвращает матрицу NxM объектов следующего вида:

```
{
     top: false,
     left: true,
     bottom: true,
     right: false
}
```

где top, left, right и bottom это соответствующие стенки ячейки лабиринта, и значения true/false обозначают наличие или отсутствие этой стенки.