First problem

```
/**

 * @param {string} val
 * @return {Object}
 */
var expect = function(val) {
    return {
        toBe: function(expected) {
            if (val === expected) return true;
            throw new Error("Not Equal");
        },
        notToBe: function(expected) {
            if (val !== expected) return true;
            throw new Error("Equal");
        }
    };
};
/**
 * expect(5).toBe(5); // true
 * expect(5).notToBe(5); // throws "Equal"
 */
```

Second problem

```
/**

 * @param {integer} init
 * @return { increment: Function, decrement: Function, reset: Function }
 */
var createCounter = function(init) {
    let current = init;

    return {
        increment: function() {
            return ++current;
        },
        decrement: function() {
            return --current;
        },
        reset: function() {
            current = init;
            return current;
        }
    };
};
```

Third problem

```
/**
 * @param {number[]} nums
 * @return {number}
 */
var removeDuplicates = function(nums) {
    let uniqueSet = new Set(nums);

    nums.splice(0, nums.length, ...uniqueSet);

    return uniqueSet.size;
};
```

Fourth problem

```
/**
 * @param {number[]} prices
 * @return {number}
 */
var maxProfit = function(prices) {
    let min = prices[0];
    let maxProfit = 0;
    for (let i = 1; i < prices.length; i++) {
        if (prices[i] < min) {
            min = prices[i];
        }
        let profit = prices[i] - min;
        if (profit > maxProfit) {
            maxProfit = profit;
        }
    }
    return maxProfit;
};
```

Problem 5

```
var majorityElement = function(nums) {
    let candidate = null;
    let count = 0;
    for (let num of nums) {
        if (count === 0) {
            candidate = num;
        }

        count += (num === candidate) ? 1 : -1;
    }
    return candidate;
};
```