Friyay4

Warm Up Problem

Work individually, and we'll come back at 9:20.

Write a function called **removeDuplicates** that given an array of items, returns the array without any duplicates. So, removeDuplicates([1, 2, 3, 4, 4, 4, 5, 6, 6, 7]) should return [1, 2, 3, 4, 5, 6, 7].

```
function removeDuplicates(arr) {
    // setup
    const newArr = [];

    // do work (remove duplicates)
    arr.forEach(function (item) {
        if(newArr.indexOf(item) === -1) {
            newArr.push(item);
        }
    })

    // return the completed work
    return newArr;
}

// OR

function removeDuplicates2(arr) {
    const arrObj = {}
    arr.forEach(function (item) {
        arrObj[item] = true;
    })
    return Object.keys(arrObj);
}
```

Group whiteboarding

Write a function called **sort** that given an array of numbers, returns a sorted array. So sort([10, 5, 9, 4, 17]) should return [4, 5, 9, 10, 17].

Sorting!

BubbleSort

```
function bubbleSort(arr) {
    // work through array and if number greater, swap
    for (let i=0; i<arr.length; i++) {
        // have to go through array multiple times
        for (let j=1;j<arr.length; j++) {
            let current = arr[j];
            let previous = arr[j-1];
            // if current number is greater, do nothing
            if (current > previous) {
                continue;
            } else {
                // if prev greater, swap current with previous
                let temp = current;
                current = previous;
                previous = temp;
                // fancy version es6+
                // [current, previous] = [previous, current]
            }
        }
    }
}
```

InsertionSort

```
function insertionSort (inputArr) {
    let length = inputArr.length;
    // work through array
    for (let i = 1; i < length; i++) {
        // key is current item in array we are trying to insert into sorted portion
        let key = inputArr[i];
        // j is last index in sorted portion
        let j = i - 1;
        // finds the first item in sorted portion of array that's less than key
        while (j >= 0 && inputArr[j] > key) {
            // shifts array one to the right until you find correct place
            inputArr[j + 1] = inputArr[j];
            j -= 1;
        }
        // inserts key into sorted portion in correct position
        inputArr[j + 1] = key;
    }
    return inputArr;
};
```

SelectionSort

```
function selectionSort(arr) {
    const len = arr.length;
    // loop through array
    for (let i=0; i<len; i++) {
        // set min as index at first position each loop
        let min = i;
        // check rest of array (items greater than i less than length of list)
        for (let j=i+1; j<len; j++) {
            // if any number in array is less than current min, set min position to be
            the position for the smallest item in rest of array
            if (arr[j] < arr[min]) {
                min = j
            }
        }
        // if min is different than i (something else is smaller than the current number)
        if (i !== min) {
            // swap!
            let temp = arr[i];
            arr[i] = arr[min];
            arr[min] = temp;
            // fancy es6
            // [arr[i], arr[min]] = [arr[min], arr[i]]
        }
    }
    return arr;
}
```

What's the O(?) for these sorting methods?

O(n^2)

Examples:

https://visualgo.net/bn/sorting

https://www.toptal.com/developers/sorting-algorithms

Links:

- Bubble Sort: https://dev.to/ryan_dunton/bubble-sorting-for-beginners-in-js-2opp
- Insertion Sort: https://dev.to/ryan_dunton/insertion-sorting-for-beginners-in-js------fkg
- Selection Sort: https://medium.com/javascript-algorithms/javascript-algorithms-selection-sort-54da919d0513