**Assignment24**

**1.**

function romanToInt(s) {

const symbolValues = new Map([

['I', 1],

['V', 5],

['X', 10],

['L', 50],

['C', 100],

['D', 500],

['M', 1000]

]);

let total = 0;

for (let i = 0; i < s.length; i++) {

const currentSymbolValue = symbolValues.get(s[i]);

const nextSymbolValue = symbolValues.get(s[i + 1]);

if (nextSymbolValue && currentSymbolValue < nextSymbolValue) {

total -= currentSymbolValue;

} else {

total += currentSymbolValue;

}

}

return total;

}

// Example:

console.log(romanToInt("III"));

Output: 3

console.log(romanToInt("LVIII"));

Output: 58

**2.**

function lengthOfLongestSubstring(s) {

const charSet = new Set();

let maxLength = 0;

let left = 0;

let right = 0;

while (right < s.length) {

const currentChar = s.charAt(right);

if (!charSet.has(currentChar)) {

charSet.add(currentChar);

maxLength = Math.max(maxLength, right - left + 1);

right++;

} else {

charSet.delete(s.charAt(left));

left++;

}

}

return maxLength;

}

// Example:

console.log(lengthOfLongestSubstring("abcabcbb"));

Output: 3

console.log(lengthOfLongestSubstring("bbbbb"));

Output: 1

console.log(lengthOfLongestSubstring("pwwkew"));

Output: 3

**3.**

function majorityElement(nums) {

let count = 0;

let candidate = null;

for (let num of nums) {

if (count === 0) {

candidate = num;

count = 1;

} else if (num === candidate) {

count++;

} else {

count--;

}

}

return candidate;

}

// Example:

console.log(majorityElement([3, 2, 3]));

Output: 3

console.log(majorityElement([2, 2, 1, 1, 1, 2, 2]));

Output: 2

**4.**

function groupAnagrams(strs) {

const map = new Map()

for (let str of strs) {

const sortedStr = str.split('').sort().join('');

if (map.has(sortedStr)) {

map.get(sortedStr).push(str);

} else {

map.set(sortedStr, [str]);

}

}

return Array.from(map.values());

}

// Example:

console.log(groupAnagrams(["eat", "tea", "tan", "ate", "nat", "bat"]));

Output: [["eat", "tea", "ate"], ["tan", "nat"], ["bat"]]

console.log(groupAnagrams([""]));

Output: [[""]]

console.log(groupAnagrams(["a"]));

Output: [["a"]]

**5.**

function nthUglyNumber(n) {

const ugly = new Array(n);

ugly[0] = 1;

let p2 = 0;

let p3 = 0;

let p5 = 0;

for (let i = 1; i < n; i++) {

const next2 = ugly[p2] \* 2;

const next3 = ugly[p3] \* 3;

const next5 = ugly[p5] \* 5;

const min = Math.min(next2, next3, next5);

ugly[i] = min;

if (min === next2) {

p2++;

}

if (min === next3) {

p3++;

}

if (min === next5) {

p5++;

}

}

return ugly[n - 1];

}

// Example:

console.log(nthUglyNumber(10));

Output: 12

console.log(nthUglyNumber(1));

Output: 1

**6.**

function topKFrequent(words, k) {

const freqMap = new Map();

for (let word of words) {

freqMap.set(word, (freqMap.get(word) || 0) + 1);

}

const sortedWords = [...freqMap.keys()].sort((a, b) => {

const freqA = freqMap.get(a);

const freqB = freqMap.get(b);

if (freqA === freqB) {

return a.localeCompare(b);

}

return freqB - freqA;

});

return sortedWords.slice(0, k);

}

// Example:

console.log(topKFrequent(["i", "love", "leetcode", "i", "love", "coding"], 2));

Output: ["i", "love"]

console.log(topKFrequent(["the", "day", "is", "sunny", "the", "the", "the", "sunny", "is", "is"], 4));

Output: ["the", "is", "sunny", "day"]

**7.**

function maxSlidingWindow(nums, k) {

const n = nums.length;

const result = [];

const deque = [];

for (let i = 0; i < n; i++) {

while (deque.length && deque[0] <= i - k) {

deque.shift();

}

while (deque.length && nums[deque[deque.length - 1]] <= nums[i]) {

deque.pop();

}

deque.push(i);

if (i >= k - 1) {

result.push(nums[deque[0]]);

}

}

return result;

}

// Example:

console.log(maxSlidingWindow([1, 3, -1, -3, 5, 3, 6, 7], 3));

Output: [3, 3, 5, 5, 6, 7]

console.log(maxSlidingWindow([1], 1));

Output: [1]

**8.**

function findClosestElements(arr, k, x) {

let left = 0;

let right = arr.length - 1;

while (right - left + 1 > k) {

if (Math.abs(arr[left] - x) > Math.abs(arr[right] - x)) {

left++;

} else {

right--;

}

}

return arr.slice(left, right + 1);

}

// Example:

console.log(findClosestElements([1, 2, 3, 4, 5], 4, 3));

Output: [1, 2, 3, 4]

console.log(findClosestElements([1, 2, 3, 4, 5], 4, -1));

Output: [1, 2, 3, 4]