**Assignment10**

**1.**

function isPowerOfThree(n) {

if (n <= 0) {

return false;

}

while (n % 3 === 0) {

n /= 3;

}

return n === 1;

}

**Test cases**

console.log(isPowerOfThree(27));

**Output:** true

console.log(isPowerOfThree(0));

**Output:** false

console.log(isPowerOfThree(-1));

**Output:** false

**2.**

function lastRemaining(n) {

let leftToRight = true;

let remaining = n;

let step = 1;

let head = 1;

while (remaining > 1) {

if (leftToRight || remaining % 2 === 1) {

head += step;

}

step \*= 2;

remaining = Math.floor(remaining / 2);

leftToRight = !leftToRight;

}

return head;

}

**Test cases**

console.log(lastRemaining(9));

**Output:** 6

console.log(lastRemaining(1));

**Output:** 1

**3.**

function printSubsets(set, current = "", index = 0) {

if (index === set.length) {

console.log(current);

return;

}

printSubsets(set, current + set[index], index + 1);

printSubsets(set, current, index + 1);

}

**Test case**

printSubsets("abc");

output: { “”, “a”, “b”, “c”, “ab”, “ac”, “bc”, “abc”}

**4.**

function calculateLength(str) {

// Base case: If the string is empty, return 0

if (str === "") {

return 0;

}

return 1 + calculateLength(str.slice(1));

}

**Test cases**

console.log(calculateLength("abcd"));

**Output:** 4

console.log(calculateLength("GEEKSFORGEEKS"));

**Output:** 13

**5.**

function countContiguousSubstrings(S) {

let count = 0;

const n = S.length;

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

count++;

let j = i - 1;

let k = i + 1;

while (j >= 0 && k < n && S[j] === S[k]) {

count++;

j--;

k++;

}

}

return count;

}

**Test cases**

console.log(countContiguousSubstrings("abcab"));

**Output:** 7

console.log(countContiguousSubstrings("aba"));

**Output:** 4

**6.**

function towerOfHanoi(n, fromRod, toRod, auxRod) {

if (n === 1) {

console.log(`Move disk 1 from rod ${fromRod} to rod ${toRod}`);

return 1;

}

let count = 0;

count += towerOfHanoi(n - 1, fromRod, auxRod, toRod);

console.log(`Move disk ${n} from rod ${fromRod} to rod ${toRod}`);

count++;

count += towerOfHanoi(n - 1, auxRod, toRod, fromRod);

return count;

}

**Test cases**

console.log(towerOfHanoi(2, 1, 3, 2));

**Output:** 3

console.log(towerOfHanoi(3, 1, 3, 2));

**Output:** 7

**7.**

function printPermutations(str) {

const result = [];

const visited = new Array(str.length).fill(false);

const permutation = [];

generatePermutations(str, visited, permutation, result);

return result;

}

function generatePermutations(str, visited, permutation, result) {

if (permutation.length === str.length) {

result.push(permutation.join(''));

return;

}

}

}

**8.**

function countConsonants(str) {

const vowels = ['a', 'e', 'i', 'o', 'u'];

let count = 0;

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

const char = str[i].toLowerCase();

if (char >= 'a' && char <= 'z' && !vowels.includes(char)) {

count++;

}

}

return count;

}

**Test cases**

console.log(countConsonants("abc de"));

**Output:** 3

console.log(countConsonants("geeksforgeeks portal"));

**Output:** 12