**Assignment7**

**1.**

function isIsomorphic(s, t) {

if (s.length !== t.length) {

return false;

}

const sToT = {};

const tToS = {};

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

const sChar = s[i];

const tChar = t[i];

if ((sChar in sToT && sToT[sChar] !== tChar) || (tChar in tToS && tToS[tChar] !== sChar)) {

return false;

}

sToT[sChar] = tChar;

tToS[tChar] = sChar;

}

return true;

}

const s = "egg";

const t = "add";

console.log(isIsomorphic(s, t));

**Output:** true

**2.**

function isStrobogrammatic(num) {

const strobogrammaticPairs = {

'0': '0',

'1': '1',

'6': '9',

'8': '8',

'9': '6',

};

let left = 0;

let right = num.length - 1;

while (left <= right) {

const leftChar = num[left];

const rightChar = num[right];

if (!strobogrammaticPairs[leftChar] || strobogrammaticPairs[leftChar] !== rightChar) {

return false;

}

left++;

right--;

}

if (num.length % 2 === 1 && !strobogrammaticPairs[num[Math.floor(num.length / 2)]]) {

return false;

}

return true;

}

const num = "69";

console.log(isStrobogrammatic(num));

**Output**: true

**3.**

function addStrings(num1, num2) {

let i = num1.length - 1;

let j = num2.length - 1;

let carry = 0;

let result = "";

while (i >= 0 || j >= 0 || carry > 0) {

const digit1 = i >= 0 ? parseInt(num1[i]) : 0;

const digit2 = j >= 0 ? parseInt(num2[j]) : 0;

const sum = digit1 + digit2 + carry;

result = (sum % 10) + result;

carry = Math.floor(sum / 10);

i--;

j--;

}

return result;

}

const num1 = "11";

const num2 = "123";

console.log(addStrings(num1, num2));

**Output:** "134"

**4.**

function reverseWords(s) {

const words = s.split(' ');

const reversedWords = words.map(word => word.split('').reverse().join(''));

return reversedWords.join(' ');

}

const s = "Let's take LeetCode contest";

console.log(reverseWords(s));

**Output:** "s'teL ekat edoCteeL tsetnoc"

**5.**

function reverseStr(s, k) {

const chars = s.split('');

for (let start = 0; start < chars.length; start += 2 \* k) {

let left = start;

let right = Math.min(start + k - 1, chars.length - 1);

while (left < right) {

[chars[left], chars[right]] = [chars[right], chars[left]];

left++;

right--;

}

}

return chars.join('');

}

const s = "abcdefg";

const k = 2;

console.log(reverseStr(s, k));

**Output:** "bacdfeg"

**6.**

function rotateString(s, goal) {

if (s.length !== goal.length) {

return false;

}

const concatenated = s + s;

return concatenated.includes(goal);

}

const s = "abcde";

const goal = "cdeab";

console.log(rotateString(s, goal));

**Output:** true

**7.**

function processString(str) {

const stack = [];

for (let char of str) {

if (char !== '#') {

stack.push(char);

} else if (stack.length > 0) {

stack.pop();

}

}

return stack;

}

function backspaceCompare(s, t) {

const stackS = processString(s);

const stackT = processString(t);

while (stackS.length > 0 && stackT.length > 0) {

if (stackS.pop() !== stackT.pop()) {

return false;

}

}

return stackS.length === 0 && stackT.length === 0;

}

const s = "ab#c";

const t = "ad#c";

console.log(backspaceCompare(s, t));

**Output:** true

**8.**

function checkStraightLine(coordinates) {

const [x1, y1] = coordinates[0];

const [x2, y2] = coordinates[1];

const slope1 = (y2 - y1) / (x2 - x1);

for (let i = 2; i < coordinates.length; i++) {

const [x, y] = coordinates[i];

const slope2 = (y - y1) / (x - x1);

if (slope1 !== slope2) {

return false;

}

}

return true;

}

const coordinates = [[1,2],[2,3],[3,4],[4,5],[5,6],[6,7]];

console.log(checkStraightLine(coordinates));

**Output:** true