**1**. function fizzBuzz() {

for (let i = 1; i <= 100; i++) {

if (i % 3 === 0 && i % 5 === 0) {

console.log("FizzBuzz");

} else if (i % 3 === 0) {

console.log("Fizz");

} else if (i % 5 === 0) {

console.log("Buzz");

} else {

console.log(i);

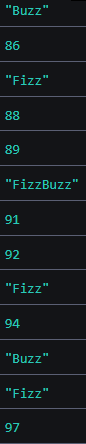
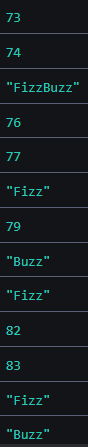
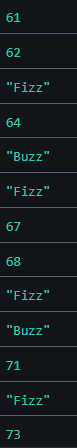
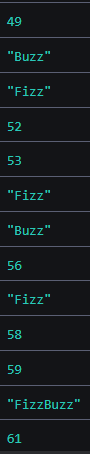
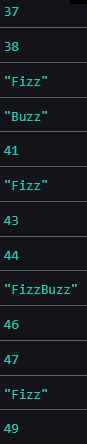
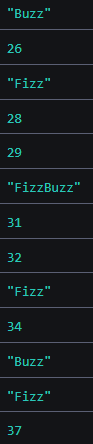
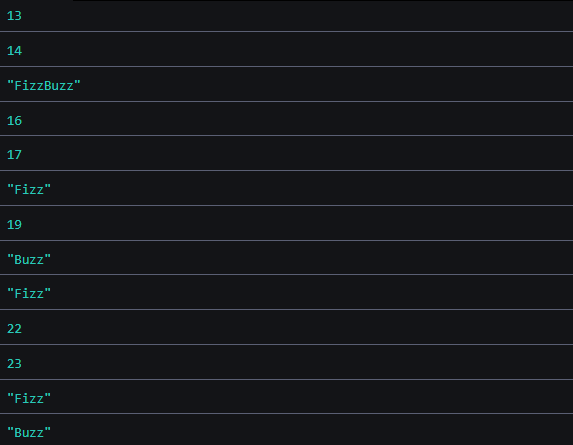
}

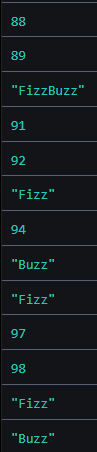
}

}

fizzBuzz()

**Output:**





**2**. function arithmeticExpression() {

const exp = prompt(

"Please input arithmetic expression in the format number1+number2 OR number1-number2"

);

console.log(exp);

const regex = /\+|\-/;

const isTrue = regex.test(exp);

if (isTrue) {

const operation = exp.match(regex).toString();

const arr = exp.split(operation);

const num1 = Number(arr[0].trim());

const num2 = Number(arr[1].trim());

if (isNaN(num1) || isNaN(num2)) {

return `invalid input`;

} else {

if (operation === "+") {

return num1 + num2;

}

if (operation === "-") {

return num1 - num2;

}

}

} else {

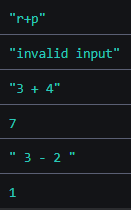
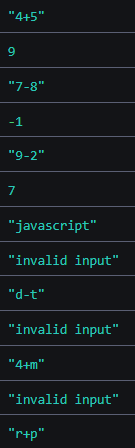
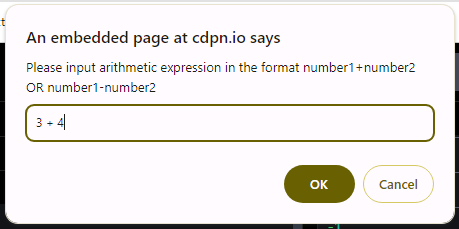
return `invalid input`;

}

}

console.log(arithmeticExpression());

**Output:**



**3**.

function flatArray(nestedArray) {

const output = new Array();

function flat(array){

for (const item of array) {

if (Array.isArray(item)) {

flat(item);

} else {

output.push(item);

}

}

}

if (Array.isArray(nestedArray)) {

flat(nestedArray);

return output;

} else {

return "Invalid input";

}

}

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

console.log(flatArray([[1, 2, 3], [4, 5], [6]]));

console.log(flatArray([]));

console.log(flatArray([1, 2, 3]));

console.log(flatArray([[], [[], [[], []]]]));

console.log(flatArray([1, [2], 'a', [true, [null, undefined]]]));

console.log(flatArray([{ o: 1 }]));

console.log(flatArray(123));

console.log(flatArray('string'));

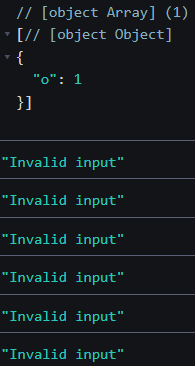
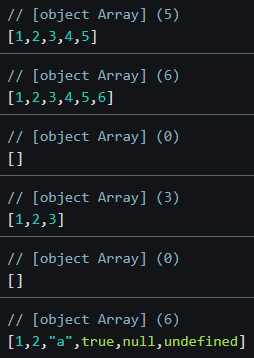
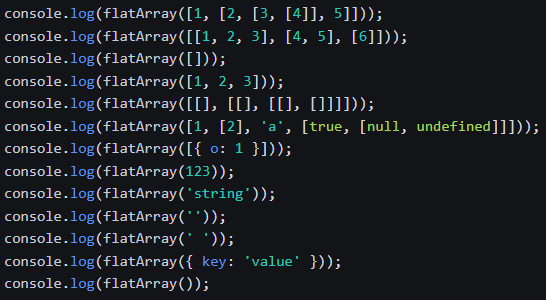
console.log(flatArray(''));

console.log(flatArray(' '));

console.log(flatArray({ key: 'value' }));

console.log(flatArray());

**Output:**



**4**. function anagrams(str1, str2) {

console.log(str1, str2);

if (typeof str1 !== "string" || typeof str2 !== "string") {

return "Invalid input(input must be string)";

}

if (str1.length !== str2.length) {

return "Given strings are not anagrams(length of two strings is not equal)";

}

const newstr1 = str1.toLowerCase();

const newstr2 = str2.toLowerCase();

let isTrue = false;

const arr1 = Array.from(newstr1);

const arr2 = Array.from(newstr2);

if (arr1.sort().join() === arr2.sort().join()) {

return `${str1} and ${str2} are anagrams of each other`;

} else {

return `${str1} and ${str2} are not anagrams of each other`;

}

}

console.log(anagrams("angel", "glean"));

console.log(anagrams(123, 1010));

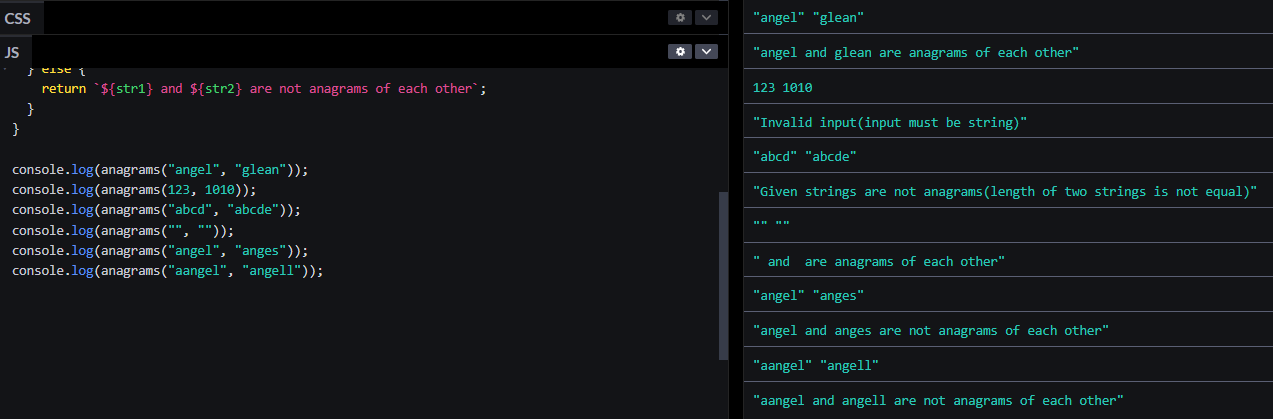
console.log(anagrams("abcd", "abcde"));

console.log(anagrams("", ""));

console.log(anagrams("angel", "anges"));

console.log(anagrams("aangel", "angell"));

**Output:**



**5**. function distinctArray(arr) {

if (!Array.isArray(arr)) {

return "Invalid input";

}

const newArray = [];

for (const i of arr) {

if (!newArray.includes(i)) {

newArray.push(i);

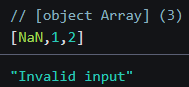
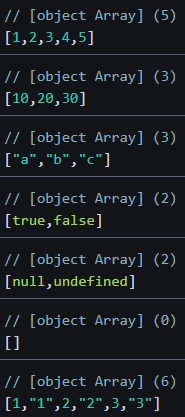
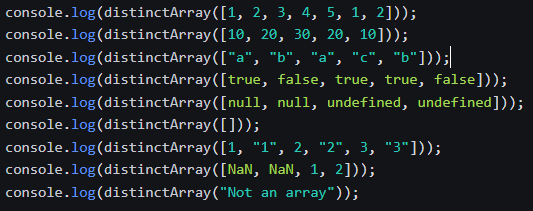
}

}

return newArray;

}

**Output:**



**6**. function capitalize(str) {

if (typeof str !== "string") {

return "Invalid input(input must be string)";

}

const arr = str.split(" ");

let capitalize\_arr = [];

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

let word = arr[i];

if (word.length > 0) {

word = word[0].toUpperCase() + word.slice(1);

}

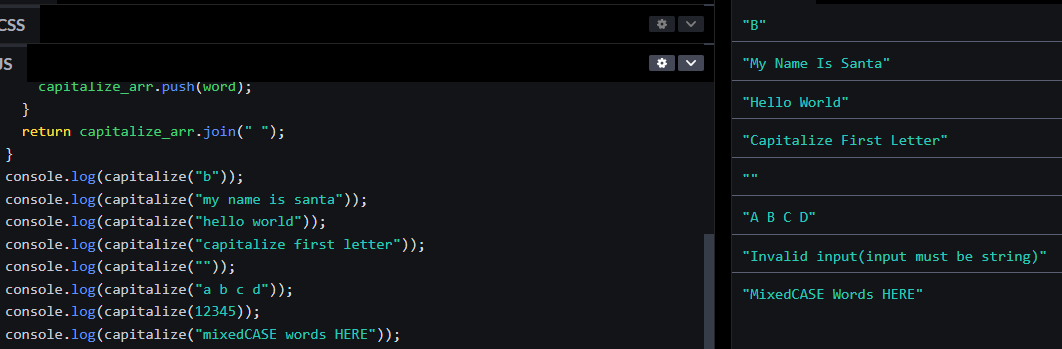
capitalize\_arr.push(word);

}

return capitalize\_arr.join(" ");

}

**Output:**



**7**. function fibonacci(n) {

if (n <= 0) {

return [];

} else if (n === 1) {

return [0];

} else if (n === 2) {

return [0, 1];

}

const sequence = [0, 1];

for (let i = 2; i < n; i++) {

const nextNumber = sequence[i - 1] + sequence[i - 2];

sequence.push(nextNumber);

}

return sequence;

}

const n = 15;

console.log(fibonacci(n));

**Output:**



**8**. class HashMap {

constructor() {

this.size = 20;

this.buckets = new Array(this.size).fill(null).map(() => []);

}

\_hash(key) {

let hash = 0;

for (let char of key) {

hash += char.charCodeAt(0);

}

return hash % this.size;

}

put(key, value) {

const hash = this.\_hash(key);

const bucket = this.buckets[hash];

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

const [k, v] = bucket[i];

if (k === key) {

bucket[i] = [key, value];

return;

}

}

bucket.push([key, value]);

}

get(key) {

const hash = this.\_hash(key);

const bucket = this.buckets[hash];

for (let [k, v] of bucket) {

if (k === key) {

return v;

}

}

return undefined;

}

remove(key) {

const hash = this.\_hash(key);

const bucket = this.buckets[hash];

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

const [k, v] = bucket[i];

if (k === key) {

bucket.splice(i, 1);

return true;

}

}

return false;

}

}

const map = new HashMap();

map.put('key1', 'value1');

console.log(map.get('key1'));

map.remove('key1');

console.log(map.get('key1'));

**Output:**



**9**. function filterArray(arr){

if (!Array.isArray(arr)) {

return "Invalid input";

}

const filteredArray = arr.filter(num => num%2===0);

return filteredArray;

}

**Output:**



**10**. function toTitleCase(str) {

if (typeof str !== "string") {

return "Invalid input(input must be string)";

}

return str.replace(/\w\S\*/g, function (txt) {

return txt.charAt(0).toUpperCase() + txt.substr(1).toLowerCase();

});

}

**Output:**

