

# ADVANCED JAVASCRIPT

## Problem 1 :

Create a Map in JavaScript and perform the following operations :

- Add key-value pairs to the Map.
- Check if a specific key exists.
- Retrieve the value associated with a given key.
- Iterate through all key-value pairs.

## CODE:

```
const myMap = new Map();

myMap.set('name', 'Alice');
myMap.set(1, 'One');
myMap.set('isStudent', true);

console.log("1. Map after adding elements:", myMap);

const keyToCheck = 'name';
const exists = myMap.has(keyToCheck);
console.log(`2. Does key '${keyToCheck}' exist? ${exists}`); // true

const nonExistentKey = 'age';
console.log(`2. Does key '${nonExistentKey}' exist?
${myMap.has(nonExistentKey)}`); // false

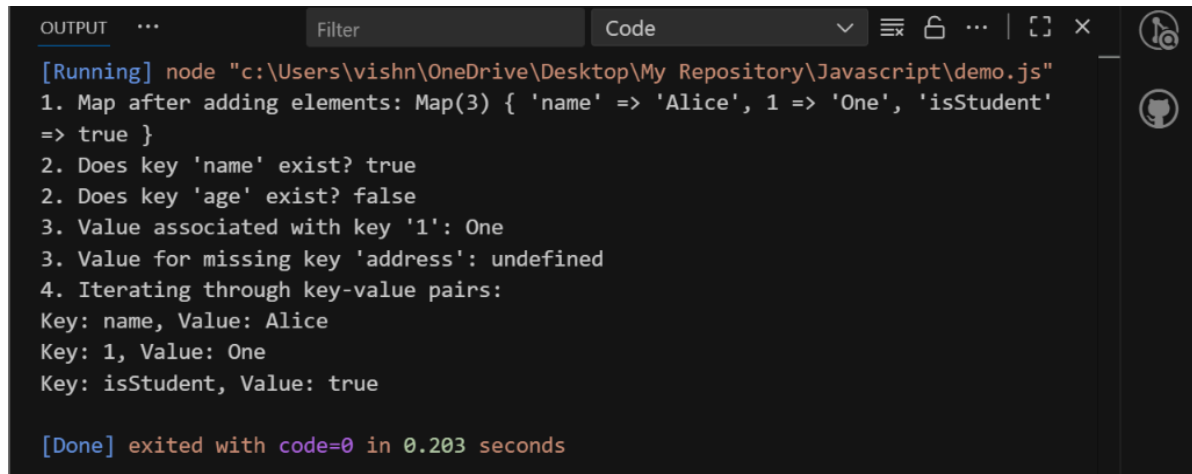
const keyToRetrieve = 1;
const value = myMap.get(keyToRetrieve);
console.log(`3. Value associated with key '${keyToRetrieve}':
${value}`); // One

const missingValue = myMap.get('address');
console.log(`3. Value for missing key 'address': ${missingValue}`); //
undefined

console.log("4. Iterating through key-value pairs:");
```

```
myMap.forEach((value, key) => {  
    console.log(`Key: ${key}, Value: ${value}`);  
});
```

## OUTPUT:



```
OUTPUT ... Filter Code  
[Running] node "c:\Users\vishn\OneDrive\Desktop\My Repository\Javascript\demo.js"  
1. Map after adding elements: Map(3) { 'name' => 'Alice', 1 => 'One', 'isStudent'  
=> true }  
2. Does key 'name' exist? true  
2. Does key 'age' exist? false  
3. Value associated with key '1': One  
3. Value for missing key 'address': undefined  
4. Iterating through key-value pairs:  
Key: name, Value: Alice  
Key: 1, Value: One  
Key: isStudent, Value: true  
  
[Done] exited with code=0 in 0.203 seconds
```

## Problem 2 :

Create a Map to store contact information (name, age, email, location) and implement a function to retrieve contact details by name.

## CODE:

```
// Create a Map to store contact information  
const contactMap = new Map();  
  
// Function to add a contact  
function addContact(name, age, email, location) {  
    contactMap.set(name, {  
        age: age,  
        email: email,  
        location: location  
    });  
}  
  
// Add sample contacts  
addContact('John Doe', 30, 'john@example.com', 'New York');  
addContact('Jane Smith', 24, 'jane@example.com', 'Los Angeles');  
addContact('Peter Jones', 45, 'peter@example.com', 'Chicago');  
  
console.log("Problem 2 - Contact Map:", contactMap);
```

```
// Implement a function to retrieve contact details by name
function getContactDetailsByName(name) {
    if (contactMap.has(name)) {
        return contactMap.get(name);
    } else {
        return `Contact '${name}' not found.`;
    }
}

// Test the retrieval function
console.log("\nRetrieving John Doe's details:");
console.log(getContactDetailsByName('John Doe'));

console.log("\nRetrieving a non-existent contact:");
console.log(getContactDetailsByName('Mike Williams'));
```

## OUTPUT:

```
OUTPUT  ...  Filter  Code  v  ≡  🔒  ...  |  [ ]  x
[Running] node "c:\Users\vishn\OneDrive\Desktop\My Repository\Javascript\demo.js"
Problem 2 - Contact Map: Map(3) {
  'John Doe' => { age: 30, email: 'john@example.com', location: 'New York' },
  'Jane Smith' => { age: 24, email: 'jane@example.com', location: 'Los Angeles' },
  'Peter Jones' => { age: 45, email: 'peter@example.com', location: 'Chicago' }
}

Retrieving John Doe's details:
{ age: 30, email: 'john@example.com', location: 'New York' }

Retrieving a non-existent contact:
Contact 'Mike Williams' not found.

[Done] exited with code=0 in 0.227 seconds
```

## Problem 3 :

You are given an array of numbers named myArray. Create a function that takes any number of arguments and adds them to the existing array. Use the spread and rest operator.

## CODE:

```
const myArray = [10, 20, 30];
function addElementsToArray(arr, ...newElements) {
    return [...arr, ...newElements];
}
```

```
const newArray1 = addElementsToArray(myArray, 40, 50);
console.log("1. New array with 40 and 50:", newArray1);

const newArray2 = addElementsToArray(myArray, 100, 200, 300, 400);
console.log("2. New array with multiple elements:", newArray2);

const newArray3 = addElementsToArray(myArray);
console.log("3. New array with no new elements:", newArray3);
```

## OUTPUT:

```
OUTPUT ... Filter Code
[Running] node "c:\Users\vishn\OneDrive\Desktop\My Repository\Javascript\demo.js"
1. New array with 40 and 50: [ 10, 20, 30, 40, 50 ]
2. New array with multiple elements: [
  10, 20, 30, 100,
  200, 300, 400
]
3. New array with no new elements: [ 10, 20, 30 ]

[Done] exited with code=0 in 0.169 seconds
```

## Problem 4 :

Create an object car with properties brand, model, and a method displayDetails that prints "Brand: [brand], Model: [model]". Test the method using this keyword.

## CODE:

```
const car = {
  brand: 'Toyota',
  model: 'Camry',

  displayDetails: function() {
    console.log(`Brand: [${this.brand}], Model: [${this.model}]`);
  }
};

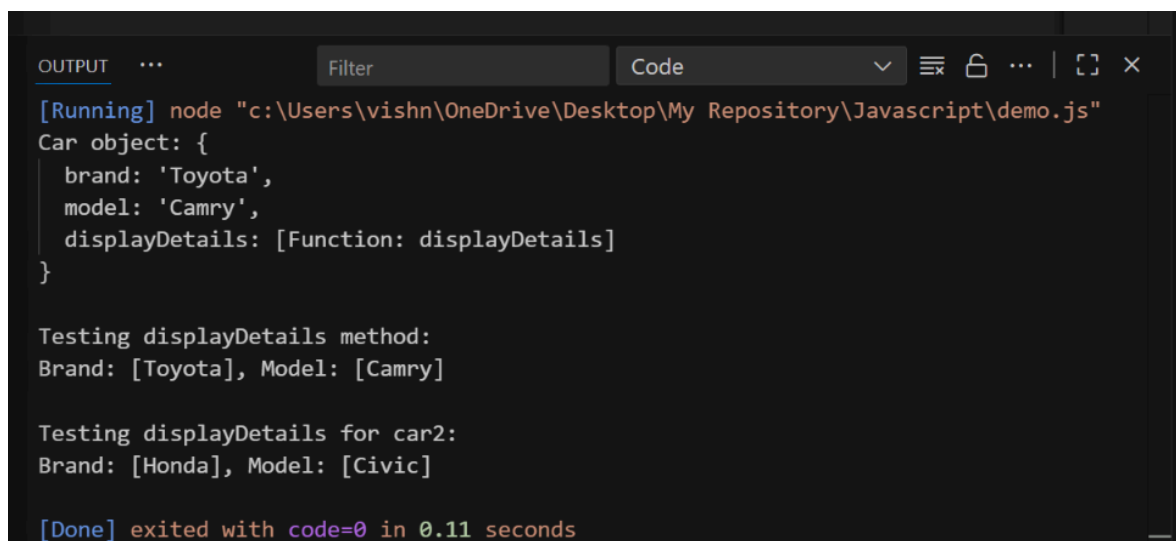
console.log("Car object:", car);

console.log("\nTesting displayDetails method:");
car.displayDetails();
```

```
const car2 = {
  brand: 'Honda',
  model: 'Civic',
  displayDetails: car.displayDetails
};

console.log("\nTesting displayDetails for car2:");
car2.displayDetails();
```

## OUTPUT:



```
OUTPUT ... Filter Code
[Running] node "c:\Users\vishn\OneDrive\Desktop\My Repository\Javascript\demo.js"
Car object: {
  brand: 'Toyota',
  model: 'Camry',
  displayDetails: [Function: displayDetails]
}

Testing displayDetails method:
Brand: [Toyota], Model: [Camry]

Testing displayDetails for car2:
Brand: [Honda], Model: [Civic]

[Done] exited with code=0 in 0.11 seconds
```

## Problem 5 :

Create two objects person1 and person2 with properties name and age. Create a function "introduce" that prints "Hello, I'm [name], and I'm [age] years old." Use the call method to make person2 introduce itself using the introduce function.

## CODE:

```
const person1 = {
  name: 'Rohit',
  age: 28,
};

const person2 = {
  name: 'Priya',
  age: 32,
```

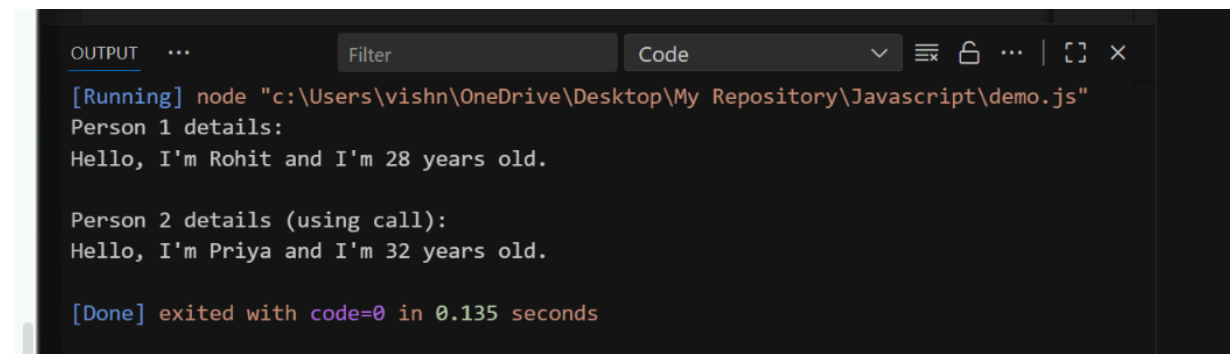
```
};

function introduce() {
    console.log(`Hello, I'm ${this.name} and I'm ${this.age} years old.`);
}

console.log("Person 1 details:");
introduce.call(person1);

console.log("\nPerson 2 details (using call):");
introduce.call(person2);
```

## OUTPUT:



```
OUTPUT  ...  Filter  Code  v  ≡  🔒  ...  |  [ ]  x

[Running] node "c:\Users\vishn\OneDrive\Desktop\My Repository\Javascript\demo.js"
Person 1 details:
Hello, I'm Rohit and I'm 28 years old.

Person 2 details (using call):
Hello, I'm Priya and I'm 32 years old.

[Done] exited with code=0 in 0.135 seconds
```