Bahria University,

Karachi Campus



LAB EXPERIMENT NO.

\_\_\_08\_\_\_

LIST OF TASKS

|  |  |
| --- | --- |
| TASK NO | OBJECTIVE |
| **01** | Write a javascript program that implements the array.**map()** that aim to produces an array containing square roots of the numbers in the original array. |
| **02** | Create a class named 'Member' having the members: Name, Age, Salary. It also has a method named 'printSalary' which prints the salary of the members. Create child class 'Employee' that inherits the 'Member' class. The 'Employee' classes have data member 'department'. Now, assign name, age and salary to an employee by making an object of child class and print the same. |
| **03** | Write a JavaScript program to implement the concept of nullish coalesing operator by using the below object properties. |
| 04 | Write a javascript program to create the Promise that resolve in 10 seconds and check the status by returning the “Promise is resolved successfully” string if the number is even otherwise reject the promise by returning the string “Promise is rejected”. Convert this task unto async await as well and compare the results. |

Submitted On

14-12-2023

(Date: DD/MM/YY)

**Task 01:** Write a javascript program that implements the array.**map()** that aim to produces an array containing square roots of the numbers in the original array.

**Solution:**

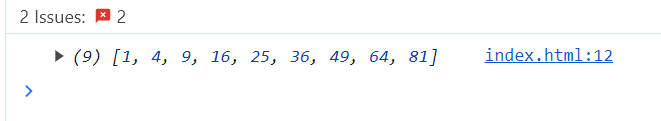
let array1=[1,2,3,4,5,6,7,8,9];

let array2= array1.map((value,index,arr)=>{

return value\*value})

console.log(array2);

**Output:**



**Task 02:** Create a class named 'Member' having the members: Name, Age, Salary. It also has a method named 'printSalary' which prints the salary of the members. Create child class 'Employee' that inherits the 'Member' class. The 'Employee' classes have data member 'department'. Now, assign name, age and salary to an employee by making an object of child class and print the same.

**Solution:**

class Member{

constructor(name,age,salary){

this.name=name;

this.age=age;

this.salary=salary;}

printSalary(){

console.log("The Salary of member is"+ this.salary);}}

class Employee extends Member{

constructor(name,age,salary,department){

super(name,age,salary);

this.department=department;}}

const employee1 = new Employee("John Doe", 30, 50000, "IT");

console.log(`${employee1.name} is ${employee1.age} years old.`);

console.log(`${employee1.name} works in the ${employee1.department} department.`);

employee1.printSalary();

**Output:**

A close-up of a computer screen

Description automatically generated

**Task 03:** Write a JavaScript program to implement the concept of nullish coalesing operator by using the below object properties.

**Solution:**

const response = {

data: {

name: "Ronaldo",

occupation: null,

lies: 0,

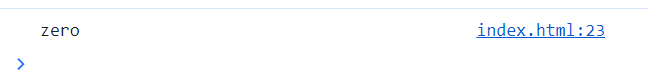
},

};

let reply=response?.data?.lies || 'zero'

console.log(reply);

**Output:**



**Task 04:** Write a javascript program to create the Promise that resolve in 10 seconds and check the status by returning the “Promise is resolved successfully” string if the number is even otherwise reject the promise by returning the string “Promise is rejected”. Convert this task unto async await as well and compare the results.

**Solution:**

const promiseExample = new Promise((resolve, reject) => {

setTimeout(() => {

const randomNumber = Math.floor(Math.random() \* 100);

if (randomNumber % 2 === 0) {

resolve(`Promise is resolved successfully with even number: ${randomNumber}`);

} else {

reject('Promise is rejected with an odd number');}

}, 10000); // Resolves after 10 seconds});

promiseExample

.then(result => console.log(result))

.catch(error => console.log(error));

**Async/Await:**

const asyncExample = async () => {

try {

const result = await new Promise((resolve, reject) => {

setTimeout(() => {

const randomNumber = Math.floor(Math.random() \* 100);

if (randomNumber % 2 === 0) {

resolve(`Promise is resolved successfully with even number: ${randomNumber}`);

} else {

reject('Promise is rejected with an odd number');}}, 10000);});

console.log(result);

} catch (error) {

console.log(error);}};

asyncExample();

**Output:**

A close up of a text

Description automatically generated

A close up of a text

Description automatically generated