PART-B

1. **Develop a javascript to sort and accessing the array elements.**

   Input: n=5

   Unsorted array elements: 10,1,-5,4,15

   Sorted Array: -5,1, 4, 10, 15

```
<html>
<body>
<h2>JavaScript Arrays</h2>
<p id="demo"></p>
<script>
const points = [10, 1, -5, 4, 15];
alert("The actual elemets are "+ points)
var points1= points.sort(function(a, b){return a-b});
alert("The sorted elements are "+ points1);
</script>
</body>
</html>
```

2. **Create a class by the name rectangle with 2 attributes length and breadth. Include a parameterized constructor to assign values to data members and a function to calculate area of the rectangle. Demonstrate creation of object of class rectangle and display its area.**

   Input: length=5, breadth=6

   Output: Area=30

```
class Rectangle {
constructor(length, breadth) {
this.length = length;
this.breadth = breadth;
```

```
}
 calculateArea() {
 return this.length * this.breadth;
 }
}
//Creating an object of the Rectangle class
let myRectangle = new Rectangle(5, 10);
// Displaying the area of the rectangle
console.log("Length:", myRectangle.length);
console.log("Breadth:", myRectangle.breadth);
console.log("Area:", myRectangle.calculateArea());
```

3. **Develop a javascript to demonstrate the working of callback and async functions.**

Callback:

```
hello(goodbye);
function hello(callback){
    console.log("hello");
    callback();

}
function goodbye(){
    console.log("bye");
}
```

Async:

```
 console.log("start");
 setTimeout(() => {
     console.log("hey");
 }, 2000);
 console.log("end");
```

4. **Develop an arrow function in javascript that checks whether a year is leap year, alert the user with true if the year is leap year and false if year is non leap year. Validate centuries also.**

Input: 2000, Output: Leap year

Input: 2100, Output: Non Leap year

Input: 2004, Output: Leap year

Input: 2006, Output: Non Leap year

```
<!DOCTYPE html>
<html lang="en">
<head>
 <meta charset="UTF-8">
 <meta name="viewport" content="width=device-width, initial-scale=1.0">
 <title>Leap Year Checker</title>
 <script>
 const isLeapYear = year => {
 if (year % 4 === 0) {
 if (year % 100 === 0) {
 return year % 400 === 0;
 }
 return true;
 }
 return false;
 };
 function checkLeapYear() {
 const userYear = prompt("Enter a year:");
 const year = parseInt(userYear);
 if (!isNaN(year)) {
 const result = isLeapYear(year);
 alert(`Is ${year} a leap year? ${result}`);
 } else {
```

```
alert("Invalid input. Please enter a valid year.");
}
}
</script>
</head>
<body>
<h1>Leap Year Checker</h1>
<button onclick="checkLeapYear()">Check Leap Year</button>
</body>
</html>
```

5. **Develop a javascript that accepts length and breadth of rectangle as parameter of an arrow functions. Call the function using spread and rest operator and alert the user with a perimeter of the rectangle.**

```
// Arrow function to calculate perimeter of rectangle

const Perimeter = (length, breadth) => 2 * (length + breadth);



// Using spread operator to pass parameters

const rectangleSpread = [10, 5];

const perimeterSpread = Perimeter(...rectangleDimensionsSpread);

console.log(`Perimeter_spread: ${perimeterSpread}`);



// Using rest operator to pass parameters

const rectangleRest = [8, 6];
```

```
const perimeterRest = Perimeter(...rectangleDimensionsRest);

console.log(`perimeter_rest: ${perimeterRest}`);
```

6. **Develop a javascript to demonstrate the usage of optional and default parameters in a function.**

Optional parameters:

```
function disp(a,b,c)

{

   var c = c || 10;

   console.log(a + b + c);

}

disp(10,10);
```

Default parameters:

```
function hello(a,b=1)

{

   console.log(a + b);

}

Hello(10);
```

7. **Create a class by the name box with parameters length, breadth, and height. Create a class boxweight that extends box and include a new parameter weight. Create another class by the name boxcost that extends boxweight and has a**

parameter by the name shipmentcost. Include constructors in all the classes. Create an object of boxcost and display values of all parameters that represent multilevel inheritance.

```
class Box {
   constructor(length, breadth, height) {
     this.length = length;
     this.breadth = breadth;
     this.height = height;
   }
 }
 class BoxWeight extends Box {
   constructor(length, breadth, height, weight) {
     super(length, breadth, height);
     this.weight = weight;
   }
 }
 class BoxCost extends BoxWeight {
   constructor(length, breadth, height, weight, shipmentCost) {
     super(length, breadth, height, weight);
     this.shipmentCost = shipmentCost;
   }
 }
const myBoxCost = new BoxCost(10, 5, 3, 2, 20);
 console.log("Length:", myBoxCost.length);
 console.log("Breadth:", myBoxCost.breadth);
 console.log("Height:", myBoxCost.height);
 console.log("Weight:", myBoxCost.weight);
 console.log("Shipment Cost:", myBoxCost.shipmentCost);
```