Question 1: Answer The Following Questions (10 points)

1. What is an Event Loop?

- Answer: The event loop is a mechanism in JavaScript that handles asynchronous operations. It continuously checks the message queue and executes any pending tasks, ensuring that non-blocking operations like I/O, timers, and network requests do not hinder the execution of other code.

2. How do you add an element at the beginning of an array? How do you add one at the end? Hint: 2 ways.

- Answer:

- To add an element at the beginning of an array:

```
let arr = [2, 3, 4];
arr.unshift(1); // Using unshift method
arr = [1, ...arr]; // Using spread operator
```

- To add an element at the end of an array:

```
let arr = [1, 2, 3];
arr.push(4); // Using push method
arr = [...arr, 4]; // Using spread operator
```

Question 2: What will the following code output? (5 points)

a.

```
var b = 1;
function outer() {
  var b = 2;
```

```
  function inner() {
    b++;
    var b = 3;
    console.log(b);
  }
  inner();
}
outer();
```

- Answer: The output will be `3`. Inside the `inner` function, `b` is incremented before it is defined with `var b = 3`, so the increment affects the local `b`, not the `b` in the `outer` function.

b.

```
for (let i = 0; i < 5; i++) {
  setTimeout(function() {
    console.log(i);
  }, i * 1000);
}
```

- Answer: The output will be:

```
0
1
2
3
4
```

Each `console.log(i)` will be executed after `i * 1000` milliseconds, and because `let` is block-scoped, `i` retains its value in each iteration.

c.

```
let arr = ['foo', 'bar'];

arr.length = 0;

arr.push('baz');

console.log(arr);
```

- Answer: The output will be `['baz']`. Setting `arr.length = 0` clears the array, and then `'baz'` is added to the empty array.

d.

```
function func() {
   for (let key in arguments) {
      console.log(arguments[key]);
   }
}

func(1, "Hello", true);
```

- Answer: The output will be:

```
1
Hello
true
```

The `for...in` loop iterates over the `arguments` object, logging each argument passed to `func`.

e.

```
let car = { carName:"BMW" , carPrice:"1000000" };

console.log(car instanceof Object);

console.log(Object.entries(car));
```

- Answer: The output will be:

```
 true

 [["carName", "BMW"], ["carPrice", "1000000"]]
```

`car instanceof Object` evaluates to `true` because `car` is an object. `Object.entries(car)` returns an array of key-value pairs from the `car` object.

Question 3: (85 points)

1. Create Function `sumObjectValues()` that will sum all values of the fields that contain numbers. Ensure that iteration is done only over own property of the object. (15 points)

- Answer:

```
function sumObjectValues(obj) {
    let sum = 0;
    for (let key in obj) {
        if (obj.hasOwnProperty(key) && typeof obj[key] === 'number') {
            sum += obj[key];
        }
    }
```

```
    return sum;
}
```

2. Show the execution of 3 asynchronous block of code, one after the other in sequence. (10 points)

- Answer:

```
function async1() {
    return new Promise((resolve) => {
        setTimeout(() => {
            console.log('Async 1');
            resolve();
        }, 1000);
    });
}

function async2() {
    return new Promise((resolve) => {
        setTimeout(() => {
            console.log('Async 2');
            resolve();
        }, 1000);
    });
}

function async3() {
    return new Promise((resolve) => {
        setTimeout(() => {
```

```
            console.log('Async 3');
            resolve();
        }, 1000);
    });
}

async1().then(async2).then(async3);
```

3. Get the maximum value from a numbers array along with its index. (5 points)

- Answer:

```
function getMaxValue(arr) {
    let max = Math.max(...arr);
    let index = arr.indexOf(max);
    return { max, index };
}
```

4. Write a function which accepts two valid dates and returns the difference between them as number of days. (10 points)

- Answer:

```
function dateDifference(date1, date2) {
    let diffInTime = new Date(date2).getTime() - new Date(date1).getTime();
    let diffInDays = diffInTime / (1000 * 3600 * 24);
    return diffInDays;
}
```

5. Design a Calculator interface for 2 number inputs which can perform sum, difference, product, and dividend whenever invoked on the same interface. (15 points)

- Answer:

```
<html>
<body>
    <input type="number" id="num1" placeholder="Number 1">
    <input type="number" id="num2" placeholder="Number 2">
    <button onclick="calculate()">Calculate</button>
    <div id="result"></div>

    <script>
        function calculate() {
            let num1 = parseFloat(document.getElementById('num1').value);
            let num2 = parseFloat(document.getElementById('num2').value);
            let result = `
                Sum: ${num1 + num2}<br>
                Difference: ${num1 - num2}<br>
                Product: ${num1 * num2}<br>
                Dividend: ${num1 / num2}
            `;
            document.getElementById('result').innerHTML = result;
        }
    </script>
</body>
</html>
```

6. Write a function which can return multiple values from a function. (10 points)

- Answer:

```
function multipleValues() {
    return { value1: 'First Value', value2: 'Second Value' };
}

let { value1, value2 } = multipleValues();
```

7. Write a function to reverse an array. For Example: `reverse([1, 2, 3, 4])` → `[4, 3, 2, 1]` (10 points)

- Answer:

```
function reverseArray(arr) {
    return arr.reverse();
}
```

8. Write a function that converts an object into an array, where each element represents a key-value pair in the form of an array. For Example: `({ a: 1, b: 2 })` → `[["a", 1], ["b", 2]]` (10 points)

- Answer:

```
function objectToArray(obj) {
    return Object.entries(obj);
}
```

Bonus

1. Write a function which can convert the time input given in 12 hours format to 24 hours format (10 points)

   - Answer:

```
function convertTo24Hours(time12h) {
    const [time, modifier] = time12h.split(' ');
    let [hours, minutes] = time.split(':');
    if (hours === '12') {
        hours = '00';
    }
    if (modifier === 'PM') {
        hours = parseInt(hours, 10) + 12;
    }
    return `${hours}:${minutes}`;
}
```

2. Make this syntax possible: `var a = add(2)(3); //5` (5 points)

   - Answer:

```
function add(x) {
    return function(y) {
        return x + y;
    }
}
let a = add(2)(3); // 5
```

3. Check if the user with the name "John" exists in the array of objects (5 points)

- Answer:

```
let users = [
    { name: 'John', age: 25 },
    { name: 'Jane', age: 30 },
    { name: 'Joe', age: 35 }
];
let userExists = users.some(user => user.name === 'John');
```