Name : Omar Saber Abdo

Id : 20191149

**Question 1:**

**What is an Event Loop?**

Event loop constantly monitors a queue of events and tasks. When an event or task is completed or becomes ready to execute, the event loop picks it up and processes it. This allows programs to handle multiple tasks concurrently, making them responsive and efficient.

**How do you add an element at the beginning of an array?**

**How do you add one at the end?**

To add an element at the beginning of an array, you can use the **unshift()** method or the spread operator (**...**) along with the **concat()** method. To add an element at the end of an array, you can use the **push()** method or the spread operator with array literals.

Here are the two ways for each operation:

**Adding an element at the beginning:**

1. Using the **unshift()** method:

**Adding an element at the end:**

2.Using the push() method:

**Question 2:**

1. 3
2. 0 1 2 3 4
3. Baz
4. 1 , Hello , true
5. True , [carName , bmw] ,[ carPrice ,100000]

**Question 3:**

1

function sumObjectValues(obj) {

let sum = 0;

for (const key in obj) {

if (obj.hasOwnProperty(key) && typeof obj[key] === 'number') {

sum += obj[key];

}

}

return sum;

}

const sampleObject = {

field1: 10,

field2: 'text',

field3: 20,

field4: null,

};

const result = sumObjectValues(sampleObject);

console.log(result); // Output: 30 (10 + 20)

/////////////////////////////////////////////////////////////////////////////////////

2

function asyncTask(time, id, callback) {

setTimeout(() => {

console.log(`Async task ${id} completed after ${time} ms`);

callback();

}, time);

}

function runSequentially() {

console.log('Starting sequence...');

asyncTask(2000, 1, () => {

asyncTask(1500, 2, () => {

asyncTask(1000, 3, () => {

console.log('Sequence completed.');

});

});

});

}

runSequentially();

/////////////////////////////////////////////////////////////////////////////////////

3

function getMaxValueWithIndex(numbers) {

1. if (numbers.length === 0) {
2. return { value: undefined, index: -1 };
3. }
4. let max = numbers[0];
5. let maxIndex = 0;
6. for (let i = 1; i < numbers.length; i++) {
7. if (numbers[i] > max) {
8. max = numbers[i];
9. maxIndex = i;
10. }
11. }
12. return { value: max, index: maxIndex };
13. }
14. const numbersArray = [34, 12, 53, 27, 89, 5];
15. const result = getMaxValueWithIndex(numbersArray);
16. console.log(`Maximum value: ${result.value}`);

console.log(`Index of maximum value: ${result.index}`);

/////////////////////////////////////////////////////////////////////////////////////

4

function dateDifferenceInDays(date1, date2) {

// Calculate the time difference in milliseconds

const timeDifference = Math.abs(date2 - date1);

// Calculate the number of milliseconds in a day

const millisecondsPerDay = 24 \* 60 \* 60 \* 1000;

// Calculate the difference in days

const differenceInDays = Math.floor(timeDifference / millisecondsPerDay);

return differenceInDays;

}

// Example usage

const startDate = new Date('2023-08-20');

const endDate = new Date('2023-08-25');

const daysDifference = dateDifferenceInDays(startDate, endDate);

console.log(`The difference between the dates is ${daysDifference} days.`);

/////////////////////////////////////////////////////////////////////////////////////

5

const readline = require('readline');

const rl = readline.createInterface({

input: process.stdin,

output: process.stdout

});

function add(a, b) {

return a + b;

}

function subtract(a, b) {

return a - b;

}

function multiply(a, b) {

return a \* b;

}

function divide(a, b) {

if (b === 0) {

return 'Cannot divide by zero';

}

return a / b;

}

function calculator() {

rl.question('Enter the first number: ', (num1) => {

rl.question('Enter the second number: ', (num2) => {

const a = parseFloat(num1);

const b = parseFloat(num2);

console.log(`Sum: ${add(a, b)}`);

console.log(`Difference: ${subtract(a, b)}`);

console.log(`Product: ${multiply(a, b)}`);

console.log(`Division: ${divide(a, b)}`);

rl.close();

});

});

}

calculator();

/////////////////////////////////////////////////////////////////////////////////////

6

function multipleValuesWithObject() {

return {

a: 15,

b: 25,

c: 35

};

}

const { a, b, c } = multipleValuesWithObject();

console.log(a, b, c); // Output: 15 25 35z

/////////////////////////////////////////////////////////////////////////////////////

7

function reverseArrayInPlace(arr) {

arr.reverse();

}

const originalArray = [1, 2, 3, 4];

reverseArrayInPlace(originalArray);

console.log(originalArray); // Output: [4, 3, 2, 1]

/////////////////////////////////////////////////////////////////////////////////////

8

function objectToArray(obj) {

const result = [];

for (const key in obj) {

if (obj.hasOwnProperty(key)) {

result.push([key, obj[key]]);

}

}

return result;

}

const inputObject = { a: 1, b: 2 };

const outputArray = objectToArray(inputObject);

console.log(outputArray); // Output: [["a", 1], ["b", 2]]

/////////////////////////////////////////////////////////////////////////////////////

Bonus++

/////////////////////////////////////////////////////////////////////////////////////

1

function convertTo24HourFormat(time12Hour) {

const [time, period] = time12Hour.split(' ');

const [hours, minutes] = time.split(':');

let hours24 = parseInt(hours);

if (period.toLowerCase() === 'pm' && hours24 !== 12) {

hours24 += 12;

} else if (period.toLowerCase() === 'am' && hours24 === 12) {

hours24 = 0;

}

return `${hours24.toString().padStart(2, '0')}:${minutes}`;

}

const time12Hour = "02:30 PM";

const time24Hour = convertTo24HourFormat(time12Hour);

console.log(time24Hour); // Output: "14:30"

/////////////////////////////////////////////////////////////////////////////////////

2

function add(x) {

return function(y) {

return x + y;

};

}

var a = add(2)(3); // This will result in a = 5

console.log(a);

/////////////////////////////////////////////////////////////////////////////////////

3

const users = [

{ id: 1, name: 'Alice' },

{ id: 2, name: 'Bob' },

{ id: 3, name: 'John' },

{ id: 4, name: 'Eve' }

];

const userNameToFind = 'John';

const userExists = users.some(user => user.name === userNameToFind);

if (userExists) {

console.log(`User with name "${userNameToFind}" exists.`);

} else {

console.log(`User with name "${userNameToFind}" does not exist.`);}