**Q1) Answer The Following Questions:**

**1) What is an Event Loop?**

**A:** The event loop is the secret by which JavaScript gives us an illusion of being multithreaded even though it is single-threaded, JS executes all operations on a single thread, but using a few smart data structures, it gives us the illusion of multi-threading. The event loop works by making a request to some internal or external "event provider", then calls the relevant event handler.

**2) How do you add an element at the beginning of an array? How do you add one at the end?**

**A:** To add an element at the beginning of an array, we use the **unshift()** method, which is a counterpart of the **push()** method.

Hence, to add an element at the end of the array, we use the **push()** method

**Q2) What is the output?**

**1)** The output will be 3

**2)** The output is 5 5 5 5 5

**3)** Not defined

**4)** 1 hello true

**5)** True

carName Bmw

carPrice 1000000

**Q3)**

**Number 1)**

function obj\_values(object) {

var results = [];

for (var property in object)

results.push(object[property]);

return results;

}

function list\_sum( list ){

return list.reduce(function(previousValue, currentValue, index, array){

return previousValue + currentValue;

});

}

function sumObjectValues(obj){

return list\_sum(obj\_values(obj));

}

**Number 2)**

**Since we have asynchronous functions that we need to execute syncronously, we must use callbacks:**

some\_3secs\_function(some\_value, function() {

some\_5secs\_function(other\_value, function() {

some\_8secs\_function(third\_value, function() {

Assuming every function is created like this:

function some\_3secs\_function(value, callback){

callback();

The specific function in this example doesn’t particularly do anything.

**Number 3)**

function indexOfMax(arr) {

if (arr.length === 0) {

return -1;

}

var max = arr[0];

var maxIndex = 0;

for (var i = 1; i < arr.length; i++) {

if (arr[i] > max) {

maxIndex = i;

max = arr[i];

}

}

return maxIndex;

}

**We can also achieve this using Math.max() to get the max value, and IndexOf() to get the index.**

**Number 4)**

function parseDate(str) {

var mdy = str.split('/');

return new Date(mdy[2], mdy[0]-1, mdy[1]);

}

function datediff(first, second) {

// Take the difference between the dates and divide by milliseconds per day.

// Round to nearest whole number to deal with DST.

return Math.round((second-first)/(1000\*60\*60\*24));

}

alert(datediff(parseDate(first.value), parseDate(second.value)));

**Number 5)**

**Interface (HTML Code):**

<html>

<head>

<meta charset=utf-8 />

<title>Simple Javascript Calculator</title>

<style type="text/css">

body {margin: 30px;}

</style>

</head>

<body>

<form>

1st Number : <input type="text" id="firstNumber" /><br>

2nd Number: <input type="text" id="secondNumber" /><br>

<input type="button" onClick="multiplyBy()" Value="Multiply"/>

<input type="button" onClick="divideBy()" Value="Divide"/>

<input type="button" onClick="AddNumber()" Value="Add"/>

<input type="button" onClick="SubtractNumber()" Value="Subtract"/>

</form>

<p>The Result is : <br>

<span id = "result"></span>

</p>

</body>

</html>

**Javascript Code:**

function AddNumber()

{

num1 = document.getElementById("firstNumber").value;

num2 = document.getElementById("secondNumber").value;

document.getElementById("result").innerHTML = num1 + num2;

}

function SubtractNumber()

{

num1 = document.getElementById("firstNumber").value;

num2 = document.getElementById("secondNumber").value;

document.getElementById("result").innerHTML = num2 – num1;

}

function multiplyBy()

{

num1 = document.getElementById("firstNumber").value;

num2 = document.getElementById("secondNumber").value;

document.getElementById("result").innerHTML = num1 \* num2;

}

function divideBy()

{

num1 = document.getElementById("firstNumber").value;

num2 = document.getElementById("secondNumber").value;

document.getElementById("result").innerHTML = num1 / num2;

}

**Number 6)**

It’s not possible to return more than one value in a function, what we could do is return an array containing all the values we need to return.

For example:

**function getValues() {**

**return [getFirstValue(), getSecondValue()];**

**}**

Then we can access them in this method:

**var values = getValues();**

**var first = values[0];**

**var second = values[1];**

**Number 7)**

function reverseArr(input) {

var ret = new Array;

for(var i = input.length-1; i >= 0; i--) {

ret.push(input[i]);

}

return ret;

}

var a = [3,5,7,8]

var b = reverseArr(a);

**Number 8**

We can use Object.keys() and map() to do this:

var obj = {"1":5,"2":7,"3":0,"4":0,"5":0,"6":0,"7":0,"8":0,"9":0,"10":0,"11":0,"12":0}

var result = Object.keys(obj).map((key) => [Number(key), obj[key]]);

console.log(result);