Assignment 4

Q1) Hoisting: In JS, the variables and function declarations are moved to the top of their containing scope during the compilation phase, before the code is executed, means we can use them before they are actually declared in the code.

Only declarations are hoisted and not the initialization.

Q2) Temporal Dead Zone: during this time we can’t access the variables because it will result in reference error. It happens when we try to access variables declared using let and const before their actual declarations in the code which means hoisting can’t be performed with let and const it can only be used with var keyword.

Q3) var have global scope and var can be redeclared within the same scope without any errors.

While the let have block scope. Let doesn’t allow redeclaration with the same name which results in syntax error. Let helps avoid common issues related to hoisting and variable re-declaration.

Q4) ES6: it provides arrow functions for writing function expressions which makes it easier and less words to write.

Let and const were provided by ES6 to declare variables with block scoping, allowing better control over variable scope and reducing issues related to hoisting.

Template literals allow embedding with the strings using backticks. They support multiline, expression evaluation.

Standardized module system were introduced, using ‘import’ and ‘export’ keywords, which provides a more organized and reusable way to share JS code across files.

Destructuring : it allows us to extract values from array or objects into distinct values.

Spread operator: we use this to allow expanding an array into individual elements. It can be used to combine arrays, pass multiple arguments to functions and create copies of arrays and objects.

Rest: it allow capturing multiple function arguments into an array, simplifying the handling of functions.

Q5) ‘let’ and ‘const’: let can be reassigned. Let can be declared without initializing and it has block scope meaning they are accessible only within the block in which they are defined.

Const: it can’t be reassigned, once a value is assigned it can’t be changed. Const must be initialized at the time of declaration, they can’t be declared without an initial value.

Q6) Template literals: they allow the embedding of expressions within strings using backticks. It supports multi-line strings. We can embed expressions using `${expression}` . The expressions are evaluated and their values are inserted into the resulting string.

Q7) map(): it is used when we want to transform each element of an array and create a new array with the transformed values of the same length.

Ex:

let salaries = [2000, 4000, 6000, 7000, 8000];

let newSalaries = salaries.map((salary) => {

  let incrementAmount = salary / 2;

  return salary + incrementAmount;

});

console.log(newSalaries);

console.log(salaries);

forEach(): is used when you want to perform an operation on each element of an array without creating a new array. Ex:

let dishes = ["Biryani", "chaap", "mutton", "kebab"];

for (let i = 0; i < dishes.length; i++) {

  console.log(dishes[i]);

}

console.log("\*\*\*\*\*\*\*\*");

//forEach

//function as a parameter which contains element: also known as functional programming

dishes.forEach(function (element) {

  console.log(element);

});

Q8) Destructuring Arrays: enclose the desired variables inside square brackets and assign them to the corresponding elements of the array in the same order.

EX: const nums = [1,2,3,4,5]

Const [first, second] = nums

Console.log(first);

Console.log(second);

Destructuring Objects: enclose the desired variables inside curly braces and assign them to the corresponding properties of the object using the same names

Const car = {

Color: ‘blue’,

Model: 2022,

Company: ‘honda’

}

Const {color, company} = person;

Console.log(color)

Console.log(company)

Q10) Spread operator: it provides a concise and powerful syntax for manipulating arrays, objects and function in JS. It makes the code more expressive and readable.

Ex:

Const arr1 = [1,2,3]

Const arr2 = [5,6,7]

Const concat = […arr1, …arr2];

Console.log(concat);