**Assignment 4 Solutions**

**Q.1 Explain Hoisting in JavaScript.  
Ans.** Hoisting in JavaScript refers to the process in which interpreter moves the declaration of functions, variables or classes to the top of their scope. Because of hoisting we can use variables or functions even before they are declared.  
Variables declared with var are hoisted with initial value of undefined whereas variables declared with let and const are said to be in **Temporal Dead Zone** till they are declared in this zone if we try to access them it will throw reference error.

**Q.2 Explain Temporal Dead Zone?  
Ans.** A temporal dead zone (TDZ) is the area of a block where a variable is inaccessible until the moment the interpreter initializes it with a value.It is the zone in which the variables declared with let and const are said to be in until they are initialized.

**Q.3 Difference between var & let?  
Ans.** The variables that are defined with **var** statement have function scope, whereas the variables that are defined with **let** statement have block scope.  
We can declare a variable with **var** again even if it has been defined previously in the same scope, but we cannot declare a variable with **let** more than once if we defined that previously in the same scope.  
Variables declared with var are hoisted with undefined value whereas variable declared with let are hoisted in temporal dead zone.

**Q.4 What are the major features introduced in ECMAScript 6?  
Ans.   
1. let and const keywords :** The keyword "let" enables the users to define variables and on the other hand, "const" enables the users to define constants.  
**2. Arrow Functions:** ES6 provides a feature known as Arrow Functions. It provides a more concise syntax for writing function expressions by removing the "function" and "return" keywords.  
Arrow functions are defined using the fat arrow (=>) notation.  
**3. Multi-line Strings**: ES6 also provides Multi-line Strings. Users can create multi-line strings by using back-ticks(`).  
**4. Default Parameters:** In ES6, users can provide the default values right in the signature of the functions.  
**5. Template Literals:** ES6 introduces very simple string templates along with placeholders for the variables. The syntax for using the string template is ${PARAMETER} and is used inside of the back-ticked string.

**Q.5 What is the difference between let and const ?  
Ans.** Both are used for declaring variables with block scope. Variables defined with let can be updated but const variables can not be updated.  
let variables can be declared without initialization but const variables should be initialized while declaring.

**Q.6 What is template literals in ES6 and how do you use them?  
Ans. Template Literals** were introduced in ES6. It is very simple string templates along with placeholders for the variables. The syntax for using the string template is ${PARAMETER} and is used inside of the back-ticked string.

**Q.7 What’s difference between map & forEach?  
Ans.**- Map returns a new array whereas forEach does not return a new array.  
- Map is used to transform the elements of an array, whereas forEach is used to loop through the elements of an array.  
- Map can be used with other array methods like filter whereas forEach can not be used.

**Q.8 How can you destructure objects and arrays in ES6?  
Ans.** The destructuring assignment is a feature that came along with ES6. Destructuring is a JavaScript expression that makes it possible to unpack values from arrays, or properties from objects, into distinct variables.  
**Array Destructuring**let introduction = ["Hello", "I" , "am", "Manash"];  
let [greeting, pronoun] = introduction;  
console.log(greeting); //"Hello"  
console.log(pronoun); //"I"  
**Object Destructuring**let person = {name: "Manash", country: "India", job: "Frontend Developer"};  
let {name, country, job} = person;  
console.log(name); //" Manash "  
console.log(country); //" India "  
console.log(job); //”Frontend Developer"

**Q.9 How can you define default parameter values in ES6 functions?  
Ans.** For defining default parameter we can simply assign the value to the parameter at the of defining function. If argument is passed then argument value will be taken otherwise default value will be taken for computation or other task.  
**function name(parameter1, parameter2=value) {  
……  
}**

**Q.10 What is the purpose of the spread operator (...) in ES6?  
Ans.** The spread operator (...) allows us to quickly copy all or part of an existing array or object into another array or object. The spread operator is often used in combination with destructuring.  
**const numbers = [1, 2, 3, 4, 5, 6];  
const [one, two, ...rest] = numbers;  
console.log(one) // 1  
console.log(two) // 2  
console.log(rest) // [3,4,5,6]**