**HW11**

1) Explain about Array forEach() and Array map()

**Array forEach() :** The forEach() method calls a provided function once for each element in an array, in order.  forEach() does not execute the function for array elements without values.

Syntax: *array*.forEach(function(currentValue, index, arr), thisValue)

**Array map() :** The map() method creates a new array with the results of calling a function for every array element. The map() method calls the provided function once for each element in an array, in order. The map() does not change the original array. The map() does not execute the function for array elements without values.

Syntax: *array*.map(function(currentValue, index, arr), thisValue)

2) object and object methods

**Object :** In JavaScript everything is an object. All JavaScript values, except primitives, are objects.

* Booleans can be objects
* Numbers can be objects
* Strings can be objects
* Dates are always objects
* Maths are always objects
* Regular expressions are always objects
* Arrays are always objects
* Functions are always objects
* Objects are always objects

**Object methods:** Methods are **actions** that can be performed on objects. Object properties can be both primitive values, other objects, and functions. An **object method** is an object property containing a **function definition**.

Syntax to create an object method: *methodName : function() { }*

Syntax to access an object method: *objectName.methodName()*

3) Different ways of creating an object

There are different ways to create new objects:

* Define and create a single object, using an object literal.
* Define and create a single object, with the keyword new.
* Define an object constructor, and then create objects of the constructed type.

**Using an Object Literal**

* This is the easiest way to create a JavaScript Object.
* Using an object literal, you both define and create an object in one statement.
* An object literal is a list of name:value pairs inside curly braces {}.

Example:

<html>

<body>

<p>Creating a JavaScript Object.</p>

<p id="demo"></p>

<script>

var person = {firstName:"Srija", lastName:"Reddy", age:24, eyeColor:"brown"};

document.getElementById("demo").innerHTML =

person.firstName + " has " + person.eyeColor + " eyes. " + person.firstName + " is " + person.age;

</script>

</body>

</html>

## Using the JavaScript Keyword new

Example:

<html>

<body>

<p id="demo"></p>

<script>

var person = new Object();

person.firstName = "Srija";

person.lastName = "Reddy";

person.age = 24;

person.eyeColor = "brown";

document.getElementById("demo").innerHTML =

person.firstName + " is " + person.age + " years old.";

</script>

</body>

</html>

## Using an Object Constructor:

Example:

## <html>

## <body>

## <p id="demo"></p>

## <script>

## function person(first, last, age, eye) {

## this.firstName = first;

## this.lastName = last;

## this.age = age;

## 

## }

## var myFather = new person("Srinivas", "Reddy", 50);

## var myMother = new person("Suneela", "Reddy", 48);

## document.getElementById("demo").innerHTML =

## "My father is " + myFather.firstName + ". My mother is " + myMother.firstName;

## </script>

## </body>

## </html>

# 4) JavaScript Object Properties

Properties are the values associated with a JavaScript object. A JavaScript object is a collection of unordered properties. Properties can usually be changed, added, and deleted, but some are read only.

## Syntax for accessing the property of an object is:

## *objectName.property (*person.age*)*

## *or*

## *objectName*["*property*"] (person["age"])

## or

## *objectName*[*expression*] (x = "age"; person[x])

## Adding New Properties:

## We can add new properties to an existing object by simply giving it a value.

## Example:

## <html>

## <body>

## <p>You can add new properties to existing objects.</p>

## <p id="demo"></p>

## <script>

## var person = {

## firstname:"Srija",

## lastname:"Reddy",

## age:24,

## eyecolor:"brown"

## };

## person.nationality = "Indian";

## document.getElementById("demo").innerHTML =

## person.firstname + " is " + person.nationality + ".";

## </script>

## </body>

## </html>

## Deleting Properties:

The **delete** keyword deletes a property from an object. The delete keyword deletes both the value of the property and the property itself. The delete operator is designed to be used on object properties. It has no effect on variables or functions. The delete operator should not be used on predefined JavaScript object properties. It can crash your application.

## 5) Prototype in object

JavaScript objects inherit the properties of their prototype. The delete keyword does not delete inherited properties, but if you delete a prototype property, it will affect all objects inherited from the prototype.