**ASSESSMENT THEORY**

**1.Closures:**

 A closure gives access to an outer function's scope from an inner function. In JavaScript, closures are created every time a function is created, at function creation time.Usually,it is can be understood as function along with its lexical scope.

Example:

function x() {

      var a=7;

      function y() {

        console.log(a);

      }

      y();

     }

        x();

Here, we will get the output as 7. Because the function looks for the value in it’s lexical scope.

**2.Shallow copy and deep copy:**

Shallow Copy:

In shallow copy we will have only one instance of object shared by multiple reference variables.

 var employee = {

  id: "102",

  empName: "Shreya",

  empAddress: "Hyderabad",

  empSalary: 50000

}

console.log("Employee=> ", employee);

var newEmployee = employee; // Shallow copy

console.log("New Employee=> ", newEmployee);

console.log("---------After modification----------");

newEmployee.ename = "Manasa";

console.log("Employee=> ", employee);

console.log("New Employee=> ", newEmployee);

From the above example, it is seen that when the name of newEmployee is modified, it is also reflected for the old employee object because both of them have same memory location. This is known as shallow copy.

Deep Copy:

In shallow copy there is same memory location allocated but in deep copy all the members of the old object are allocated separate memory location for the new object and then assigned the copied members to the new object. Because of this, both the objects are independent of each other and in case of any changes to any one the other is not affected.

JSON.stringify((empobj));

(This converts objects to string)

JSON.parse -> This converts string to object.

  id: "102",

  empName: "Shreya",

  empAddress: "Hyderabad",

  empSalary: 50000

}

console.log("=========Deep Copy========");

var newEmployee = JSON.parse(JSON.stringify(employee));

console.log("Employee=> ", employee);

console.log("New Employee=> ", newEmployee);

console.log("---------After modification---------");

newEmployee.ename = "Manasa";

newEmployee.salary = 70000;

console.log("Employee=> ", employee);

console.log("New Employee=> ", newEmployee);

Here the new object is created using the JSON.parse() and JSON.stringify().Because of this the creation of newEmployee will not affect the old employee.

**3.Destructuring an object in Javascript:**

The object destructuring is a useful JavaScript feature to extract properties from objects and bind them to variables.

let empObj = {

        firstName: "Shreya",

        lastName: "Tr",

      };

      let { firstName: fName, lastName: lName } = empObj;

      console.log(fName);

      console.log(lName);

Here, we can observe the object destructuring can assign values to variables declared using  let.The firstName is assigned to fName, the lastName is assigned to lName.

This is known as object destructuring.

**4.Arrow Functions:**

Arrow functions help in writing a code shortly i.e. with the help of it if a function has only one statement and returns value it can be typed in a single statement.

Arrow functions are denoted by =>

Example:

var mul=(a,b)=>a\*b;

      console.log(mul(10,20));

**5.Higher order functions in javascript:**

A “higher-order function” is a function that accepts functions as parameters and/or returns a function.

1)For each:

The .forEach() method executes a callback function on each of the elements in an array in order.

|  |
| --- |
|  |

 let marks = [4o, 50, 60, 70, 80, 90];

marks.forEach((ele) => console.log(ele));

2)Map:

Let us take the above marks list used in .forEach.It takes action on every array element.

let newMarks = marks.map((ele) => ele + 5);

console.log(newMarks);

Now, in this +5 will be done to every element in the array.

3)Filter:

If we want to return an element based on a prediction or particular condition filter method can be used.

let averages = [45, 57, 93, 28, 36, 29, 37, 13, 26, 28, 75, 57, 90];

let first = averages.filter((ele) => ele >= 60);

console.log(first);

In this example, all the students who secured marks above 60 are declared as first.So,we used filter method here.

4)Reduce:

This method calculates the total of each marks.

let totalMarks = marks.reduce((sum, ele) => {

  sum = sum + ele;

  return sum;

}, 0);

**6.Event looping:**

An event loop is something that pulls stuff out of the queue and places it onto the function execution stack whenever the function stack becomes empty.

The event loop got its name because of how it's usually implemented, which usually resembles:

while (queue.waitForMessage()) {

queue.processNextMessage();

}

queue.waitForMessage() waits synchronously for a message to arrive (if one is not already available and waiting to be handled).

**7.Building blocks of an angular application:**

The basic building blocks of an angular application are directives, modules, decorators, components, services, dependency injection, pipes, and templates,data binding,routers etc.

Modules,components and routing are 3 main aspects which are important.ng modules are also known as modules.

**8.Advantages of angular in web app development:**

* It is very useful to build single page applications.
* It is a very powerful framework and also has real time testing.
* Everything is organized properly if angular is used.
* It is two way binding.
* Angular creates templates using HTML which helps in easy declarations.

**9.Single page application and it’s advantages:**

Single page application:

A single page application usually known as SPA,is a web app which uses only a single HTML page and only a part of the page instead of the entire page gets updated with every click of the mouse.

Advantages:

* The loading time is less in single page applications.
* The user experience can be improved by SPAs.
* A feature rich application can be built well with the help of SPAs.
* Uses less internet bandwith as there will be only one page loading.

**10.Different data bindings in angular:**

1)One way binding:

This binding uses string interpolation i.e. {{ }}

Model to view is used in this. If there is something in typescript file then it can be displayed in HTML template by using {{ }}.

2)Two way binding:

[( )] are used in 2 way binding.

It is from model to view and view to model.(vice-versa)

If any changes are made in view it can be seen in model and vice versa.

3)Event Binding:

( ) are used in event binding.

It is from view to model it means HTML view to typecsript file.

4)Property Binding:

Square brackets [ ] are used in this binding.

It is also from model to view.

5)Attribute Binding:

[attrib.attributeName] is used in attribute binding.

It is from model to view.