CMR
INSTITUTE OF

TECHNOLOGY

Internal Assessment Test 3 Answer Key – Oct. 2024									
Sub:	Web Technologies					Sub Code:	22MCA24	Branch:	MCA
Date:	15/09/2024	Duration:	90 min's	Max Marks:	50	Sem	II		OBE

Q1) What is jquery? Why jquery ? Explain the syntax of jQuery with suitable example

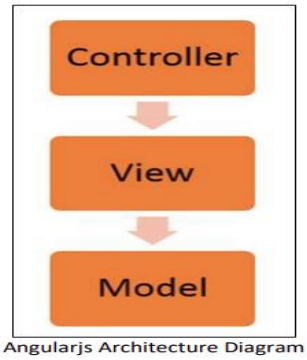
What is jQuery
• jQuery is a JavaScript library designed to simplify HTML DOM tree traversal and manipulation, as well as event handling, CSS animation, and Ajax.
• jQuery is a lightweight, "write less, do more", JavaScript library.
• Using raw JavaScript can result in dozens of lines of code.
• The creators of jQuery specifically created the library to make common tasks trivial.
• The real power in this jQuery statement comes from the selector, an expression for identifying target elements on a page that allows us to easily identify and grab the elements we need.

Why jQuery?
 Using raw JavaScript can result in dozens of lines of code for each of these tasks.
 The creators of jQuery specifically created the library to make common tas ks trivial. For example, designers will use JavaScript to "zebra-stripe" tables— highlighting every other row in a table with a contrasting color—taking up to 10 lines of code or more. Here's how we accomplish it using jQuery:
$("table tr:nth-child(even)").addClass("striped");
 jQuery statements to make your pages come alive.
 The real power in this jQuery statement comes from the selector, an expression for identifying target elements on a page that allows us to easily identify and grab the elements we need

jQuery provides a simple means to trigger the execution of code once the DOM tree, but not external image resources, has loaded. The formal syntax to define such code is as follows:
Syntax
```
$(document).ready(function() {
//jquery statements
});
```
Example
```
$(document).ready(function() {
$("table tr:nth-child(even)").addClass("even");
});
```
First, we wrap the document instance with the jQuery() function, and then we apply the ready() method, passing a function to be executed when the document is ready to be manipulated.
We called that the formal syntax for a reason; a shorthand form used much more frequently is as follows:
```
$(function() {
$("table tr:nth-child(even)").addClass("even");
});
```
By passing a function to $(), we instruct the browser to wait until the DOM has fully loaded (but only the DOM) before executing the code. Even better, we can use this technique multiple times within the same HTML document, and the browser will execute all of the functions we specify in the order that they are declared within the page.

Q2) What is angular JS? Discuss the architecture of angular JS

- AngularJS is an open source Model-View-Controller framework which is similar to the JavaScript framework.
- Angular JS framework is used for developing mostly Single Page applications.
- This framework has been developed by a group of developers from Google itself. Because of the sheer support of Google and ideas from a wide community forum, the framework is always kept up to date. Also, it always incorporates the latest development trends in the market.
- The Controller represents the layer that has the business logic. User events trigger the functions which are stored inside your controller. The user events are part of the controller.
- Views are used to represent the presentation layer which is provided to the end users
- Models are used to represent your data. The data in your model can be as simple as just having primitive declarations. For example, if you are maintaining a student application, your data model could just have a student id and a name. Or it can also be complex by having a structured data model. If you are maintaining a car ownership application, you can have structures to define the vehicle itself in terms of its engine capacity, seating capacity, etc.



Angularjs Architecture Diagram

Q3) Develop jQuery program to implement hide(), show(), fadeIn() and fadeout()

```
<!DOCTYPE HTML>
<html xmlns="http://www.w3.org/1999/xhtml">
    <head>
        <title>MCA Department</title>
        <script type="text/javascript" src="jquery-1.2.1.js"></script>
        <script type="text/javascript">
```

```
                $(document).ready(function() {
                        $("#hide").click(function(){
                                $("p").hide(1000);
                        });

                        $("#show").click(function(){
                                $("p").show(1000);
                        });
                        $("#toggle").click(function(){
                                $("p").toggle(1000);
                        });

                        $("#fadein").click(function(){
                                $("p").fadeIn(1000);
                        });

                        $("#fadeout").click(function(){
                                $("p").fadeOut(1000);
                        });
                });
        </script>
    </head>
    <body>
        <button id="hide">Hide</button>
        <button id="show">show</button>
        <button id="fadeout">fadeout</button>
        <button id="fadein">fadein</button>
        <p>This is a paragraph </p>
        <p>This is a paragraph </p>
    </body>
</html>
```

Q4) What are directives in angular JS? Explain following directive with examples?

i) ng-model ii) ng-bind iii) ng-init iv) ng-repeat

App Directive

```
ng-app= " "
```

The app directive defines the area of AngularJS application. The syntax of app directive is **ng-app = " "**; In here the **ng** is the namespace of AngularJS and **app** is the application area of Angular JS.

Model Directive

```
ng-model = "data"
```

The model directive is used to bind the inputted value from HTML controls (input, checkbox and select etc.) to application data. The **ng-model = "data"** is the syntax of model directive. Let`s take an example for better understanding.

Example 2.2

```
<!DOCTYPE html>
<html >
<head>
   <title>AngularJSfor beginners</title> <script src="js\angular.min.js">
</script>
</head>
<body>
<div ng-app=""> <p>User Name: <br> <input type="text" ng-model =
"Username"></p> </div>
</body>
</html>
```

Bind Directive

```
<p>ng-bind = "data"</p>
```

The bind directive is used to bind the data value to an html element ; the syntax of bind directive is ** ng-bind = "data" **. Let`s take an example for better understanding.

Example 2.3

```
<!DOCTYPE html>
<html >
<head>
   <title>AngularJSfor beginners</title> <script src="js\angular.min.js">
</script> </head>
<body>
<div ng-app=""> <p>User Name: <br> <input type="text" ng-model =
"Username"></p> <p ng-bind ="Username"></p> </div>
</body>
</html>
```

Open the notepad and paste the above mentioned code with .html extension, and type username "Ray Yao" in the input box.

Init Directive

```
ng-init = "data = 'value'"
```

The init directive is used to initialize the data with a value. The syntax of init directive is **ng-init = "data = 'value'"**. Let`s take an example for better understanding.

Example 2.4

```
<!DOCTYPE html>
<html >
<head>
   <title>AngularJSfor beginners</title> <script src="js\angular.min.js">
</script> </head>
<body>
<div ng-app=""    ng-init="Username= 'Andy Smith' "> <p>User Name:
<input type="text" ng-model = "Username"></p> <p ng-
bind="Username"></p> </div>
</body>
</html>
```

Repeat Directive

```
ng-repeat = "variable in array"
```

The repeat directive works like a loop. The **ng-repeat** directive repeats to get the value of an array.

Example 2.5

```
<html >
<head>
   <title>AngularJSfor beginners</title> <script src="js\angular.min.js">
</script> <meta http-equiv="Content-Type" content="text/html; charset=UTF-8"
/> </head>
<body>
<div ng-app=""  ng-init = "ColorName = ['Pink', 'Red', 'Green', 'Blue',
'Black', 'White', 'Yellow', 'Gray']"> <p style="color:green; font-
weight:bold">Colours Name:</p> <ol>
    <li ng-repeat ="x in ColorName"> <p ng-bind="x"></p> </li>
</ol>
</div>
</body>
</html>
```

Q5) What is an event? List the common events found in jQuery. Develop a jQuery program to implement mouseover() event

All the different visitors' actions that a web page can respond to are called events.

An event represents the precise moment when something happens.

Examples:

- moving a mouse over an element
- selecting a radio button

- clicking on an element

Mouse Events	Keyboard Events	Form Events	Document/Window Events
click	keypress	submit	load
dblclick	keydown	change	resize
mouseenter	keyup	focus	scroll
mouseleave		blur	unload

```
<!DOCTYPE HTML>
<html xmlns="http://www.w3.org/1999/xhtml">
	<head>
		<title>MCA Department</title>
		<script type="text/javascript" src="jquery-1.2.1.js"></script>
		<script type="text/javascript">
			$(document).ready(function() {

				$("p").mouseover(function(){
				  $(this).css("background-color", "green");
				});


			});
		</script>
	</head>
	<body>
		<p> Paragraph </p>

	</body>
</html>
```

Q6) what is filter, explain following with example

i) uppercase ii)lowercase iii)orderby iv) currency

Uppercase filter

Value | uppercase

The uppercase filter changes the text to upper case. Suppose a user writes a text in lower case (e.g. ray) or title case (e.g. Ray) or in mixed case (e.g. rAy or RaY or rAY etc.), and you want the upper case result, then you will have to use upper case filter.

Example 3.1

```
<!DOCTYPE html>
<html >
<head>
   <title>AngularJSfor beginners</title> <script src="js\angular.min.js">
</script> <meta http-equiv="Content-Type" content="text/html; charset=UTF-8"
/> </head>
<body>
<h3>Using Upper Case Filter</h3> <div ng-app=""  ng-init="Username=
'ray' "> <p>User Name: <input type="text" ng-model = "Username"></p>
<p style="color:red" ng-bind="Username | uppercase"></p> </div>
</body>
</html>
```

Lowercase filter

Value | lowercase

The lowercase filter changes the text to lower case. Suppose a user writes a text in upper case (e.g. RAY YAO) or title case (e.g. Ray Yao) or in mixed case (e.g. rAy or RaY or rAY etc.), and you want the lower case result, then you will have to use lower case filter.

Example 3.2

```
<html >
<head>
   <title>AngularJSfor beginners</title> <script src="js\angular.min.js">
</script> </head>
<body>
<h3>Using Lower Case Filter</h3> <div ng-app=""  ng-init="Username=
'Ray YAO' "> <p>User Name: <input type="text" ng-model="Username">
</p> <p style="color:red" ng-bind="Username | lowercase"></p> </div>
</body>
</html>
```

OrderBy filter

OrderBy filer is used to display values in ascending order or descending order.
The syntax of “orderBy” looks like this:

```
Value | orderBy: ’value’    //for ascending order
Value | orderBy: ’-value’   //for descending order
```

Let`s take an example for better understanding.

Example 3.3

```
<!DOCTYPE html>
<html >
<head>
   <title>AngularJSfor beginners</title> <script src="js\angular.min.js">
</script> </head>
<body>
<h1>Using OrderBy filter</h1> <div ng-app="" ng-init="StudentsResult=
[{name: 'Tienq', marks:81},          {name: 'Svbrf', marks:70},
{name: 'Yaito', marks:90},          {name: 'Pewfn', marks:63}, {name:
'Riet', marks:98}]"> <table border="1" > <tr>
<th>Student Name</th> <th>Mathematics’ Result</th> </tr>
<tr ng-repeat="x in StudentsResult | orderBy:'-marks' "> <td ng-
bind="x.name "></td> <td ng-bind="x.marks "></td> </tr>
</table>
</div>
</body>
</html>
```

Currency filter

Value | currency

The currency filter is used to display the result in currency format.

Example 3.5

```
<!DOCTYPE html>
<html >
<head>
   <title>AngularJSfor beginners</title> <script src="js\angular.min.js">
</script> <meta http-equiv="Content-Type" content="text/html; charset=UTF-8"
/> </head>
<body>
<h1>Using Currency filter</h1> <div ng-app="" ng-init =
"Employees_Monthly_Salary=[{name: 'Jay', salary:8100}, {name: 'Sdwt',
salary:7000},          {name: 'Hao', salary:9000}, {name: 'Luoe',
salary:6300}, {name: 'Fin', salary:9800}]"> <table border="1" > <tr>
<th>Employee Name</th> <th>Employee Salary</th> </tr>
<tr ng-repeat="x in Employees_Monthly_Salary  "> <td ng-bind="x.name
"></td> <td ng-bind="x.salary | currency "></td> </tr>
</table>
</div>
</body>
</html>
```

Q7) Explain following methods with examples

i)wrapInner() ii) wrap() iii) append() iv)appendTo() v)html and text()

Command syntax: wrap

`wrap(wrapper)`

Wraps the elements of the matched set with the passed HTML tags or a clone of the passed element.

Parameters

`wrapper` (String|Element) The opening and closing tags of the element with which to wrap each element of the matched set, or an element to be cloned and server as the wrapper.

Returns

The wrapped set.

Example

To wrap each link with the class surprise in a with the class hello, we write

```
$("a.surprise").wrap("<div class='hello'></div>")
```

When multiple elements are collected in a matched set, the wrap() method operates on each one individually. If we'd rather wrap all the elements in the set as a unit, we can use the wrapAll() method instead:

sometimes we may not want to wrap the elements that are in a matched set, but rather their contents. For just such cases, the wrapInner() method is available:

Command syntax: wrapInner

`wrapInner(wrapper)`

Wraps the contents, to include text nodes , elements of the matched set with the passed HTML tags or a clone of the passed element.

Parameters

`wrapper` (String|Element) The opening and closing tags of the element with which to wrap each element of the matched set, or an element to be cloned and server as the wrapper.

Returns

The wrapped set

Moving and copying elements

To add content to the end of existing content, the append() command is available.

Command syntax: append

`append(content)`

Appends the passed HTML fragment or elements to the content in all matched elements.

Parameters

`content` (String|Element|Object) A string, element, or wrapped set to append to the elements of the wrapped set. See the following description for details.

Returns

The wrapped set.

This function accepts a string containing an HTML fragment, a reference to an existing or newly created DOM element, or a jQuery wrapped set of elements.

Consider the following simple case:

```
$('p').append('<b>some text<b>');
```

This statement appends the HTML fragment created from the passed string to the end of the existing content of all <p> elements on the page.

If we want to move or copy an element from one place to another, a simpler approach uses the appendTo() command, which allows us to grab an element and move it somewhere else in the DOM.

Command syntax: appendTo

`appendTo(target)`

Moves all elements in the wrapped set to the end of the content of the specified target(s).

Parameters

`target` (String|Element) A string containing a jQuery selector or a DOM element. Each element of the wrapped set will be appended to that location. If more than one element matches a string selector, the element will be copied and appended to each element matching the selector.

Returns

The wrapped set.

Example

```
$('#flower').appendTo('#targets p')
```

Replacing HTML or text content

html() command, which allows us to retrieve the HTML contents of an element when used without parameters or, to set its contents when used with a parameter.

Command syntax: html

`html()`

Obtains the HTML content of the first element in the matched set.

Parameters

`none`

Returns

The HTML content of the first matched element. The returned value is identical to accessing the `innerHTML` property of that element.

Command syntax: html

`html(text)`

Sets the passed HTML fragment as the content of all matched elements

Parameters

`text` (String) The HTML fragment to be set as the element content

Returns

The wrapped set

We can also set or get only the text contents of elements. The text()command, when used without parameters, returns a string that's the concatenation of all text.

Command syntax: text

`text()`

Concatenates all text content of the wrapped elements and returns it as the result of the command

Parameters

`none`

Returns

The concatenated string

Q8) What is scope and how to define method in controller explain with example.

What is Scope?

Scope is a JavaScript object which contains model data.

```
function ($scope) {  }
```

$scope is a parameter of JavaScript function which is called by a controller.
Let`s take an example for understanding.

Example 7.2

```
<script>
function ($scope) {
   $scope.firstNumber = 23;
   $scope.secondNumber = 63;
}
</script>
```

Explanation: In above example, the **$scope** is the parameter of the function ($scope) { }. $scope is an object in AngularJS.

$scope.**firstNumber and **$scope.secondNumber are models used in HTML. Model data is accessed by the $scope object. We assign values to the model with following formula: "$scope.property = value".

How to define Controller?

The **ng-controller** directive is used to define the Controller. We know that Controller is a JavaScript object which contains JavaScript function and properties. The syntax of Controller is as following:

```
<div ng-app=""  ng-controller="controllerName">
```

Example 7.1

```
<html>
<script src= "js\angular.min.js"></script> <body>
<div ng-app="Calculation" ng-controller="myController"> First Number:
<input type="number" ng-model="firstNumber"><br> Second Number:
<input type="number" ng-model="secondNumber"><br> <br>
Sum: {{firstNumber + secondNumber}}
</div>
<script>
var app = angular.module('Calculation', [ ]); app.controller('myController',
function($scope) {
   $scope.firstNumber = 4; $scope.secondNumber = 8; });
</script>
</body>
</html>
```

Q9) Explain following

i)setting attribute ii) removing attribute iii) fetching attribute iv)manipulating element property

Command syntax: wrap

`wrap(wrapper)`

Wraps the elements of the matched set with the passed HTML tags or a clone of the passed element.

Parameters

`wrapper` (String|Element) The opening and closing tags of the element with which to wrap each element of the matched set, or an element to be cloned and server as the wrapper.

Returns

The wrapped set.

Example

To wrap each link with the class surprise in a with the class hello, we write

```
$("a.surprise").wrap("<div class='hello'></div>")
```

When multiple elements are collected in a matched set, the wrap() method operates on each one individually. If we'd rather wrap all the elements in the set as a unit, we can use the wrapAll() method instead:

sometimes we may not want to wrap the elements that are in a matched set, but rather their contents. For just such cases, the wrapInner() method is available:

Command syntax: wrapInner

`wrapInner(wrapper)`

Wraps the contents, to include text nodes , elements of the matched set with the passed HTML tags or a clone of the passed element.

Parameters

`wrapper` (String|Element) The opening and closing tags of the element with which to wrap each element of the matched set, or an element to be cloned and server as the wrapper.

Returns

The wrapped set

Moving and copying elements

To add content to the end of existing content, the append() command is available.

Command syntax: append

`append(content)`

Appends the passed HTML fragment or elements to the content in all matched elements.

Parameters

`content` (String|Element|Object) A string, element, or wrapped set to append to the elements of the wrapped set. See the following description for details.

Returns

The wrapped set.

This function accepts a string containing an HTML fragment, a reference to an existing or newly created DOM element, or a jQuery wrapped set of elements. Consider the following simple case:

```
$('p').append('<b>some text<b>');
```

This statement appends the HTML fragment created from the passed string to the end of the existing content of all <p> elements on the page.

If we want to move or copy an element from one place to another, a simpler approach uses the appendTo() command, which allows us to grab an element and move it somewhere else in the DOM.

Command syntax: appendTo

`appendTo(target)`

Moves all elements in the wrapped set to the end of the content of the specified target(s).

Parameters

`target` (String|Element) A string containing a jQuery selector or a DOM element. Each element of the wrapped set will be appended to that location. If more than one element matches a string selector, the element will be copied and appended to each element matching the selector.

Returns

The wrapped set.

Example

```
$('#flower').appendTo('#targets p')
```

Replacing HTML or text content

html() command, which allows us to retrieve the HTML contents of an element when used without parameters or, to set its contents when used with a parameter.

Command syntax: html

`html()`

Obtains the HTML content of the first element in the matched set.

Parameters

`none`

Returns

The HTML content of the first matched element. The returned value is identical to accessing the `innerHTML` property of that element.

Command syntax: html

`html(text)`

Sets the passed HTML fragment as the content of all matched elements

Parameters

`text` (String) The HTML fragment to be set as the element content

Returns

The wrapped set

We can also set or get only the text contents of elements. The text()command, when used without parameters, returns a string that's the concatenation of all text.

Command syntax: text

`text()`

Concatenates all text content of the wrapped elements and returns it as the result of the command

Parameters

`none`

Returns

The concatenated string

Q10) What is a Angular js service? Explain any three of them with the code snippets

Angular Service Service is a function or an object, which is used to provide with a specified action. In AngularJS, there are about 30 builtin services, such as $http, $location, $interval and $timeout.

The $location service has methods which return information about the location of the current web page:

```
<div ng-app="myApp" ng-controller="myCtrl">
<h1> Location Service</h1>
<h3>URL: {{myUrl}}</h3>
<h3>Port: {{currentPort}}</h3>
<h3>Protocol: {{protocol}}</h3>
<script>
var app = angular.module('myApp', []);
app.controller('myCtrl', function($scope, $location) {
$scope.myUrl = $location.absUrl();
$scope.currentPort = $location.port();
$scope.protocol = $location.protocol();
});
</script>
</div>
```

The $http service is one of the most common used services in AngularJS applications. The service makes a request to the server, and lets your application handle the response.

```
<h1> http Service</h1>
<h1>{{myResponse}}</h1>
<script>
var app = angular.module('myApp', []);
app.controller('myCtrl', function($scope, $http) {
```

```
$http.get("welcome.html").then(function (response) {
$scope.myResponse = response.data;
});
});
</script>
```

The $timeout service of AngularJS is functionally similar to the ‘window.setTimeout’ object of vanilla JavaScript. This service allows the developer to set some time delay before the execution of the function.

```
<h1> timeout Service</h1>
<h1>{{myText}}</h1>
<script>
var app = angular.module('myApp', []);
app.controller('myCtrl', function($scope, $timeout) {
$scope.myText = "Hello World!";
$timeout(function () {
$scope.myText = "How are you today?";
}, 5000);
});
</script>
```