SNS COLLEGE OF ENGINEERING

Kurumbapalayam (Po), Coimbatore – 641 107

AN AUTONOMOUS INSTITUTION

Approved by AICTE, New Delhi and Affiliated to Anna University, Chennai

INTERNAL ASSESSMENT EXAMINATION – I

VI Semester

B.E., COMPUTER SCIENCE AND TECHNOLOGY

19TS601 FULL STACK DEVELOPMENT

Regulations 2019

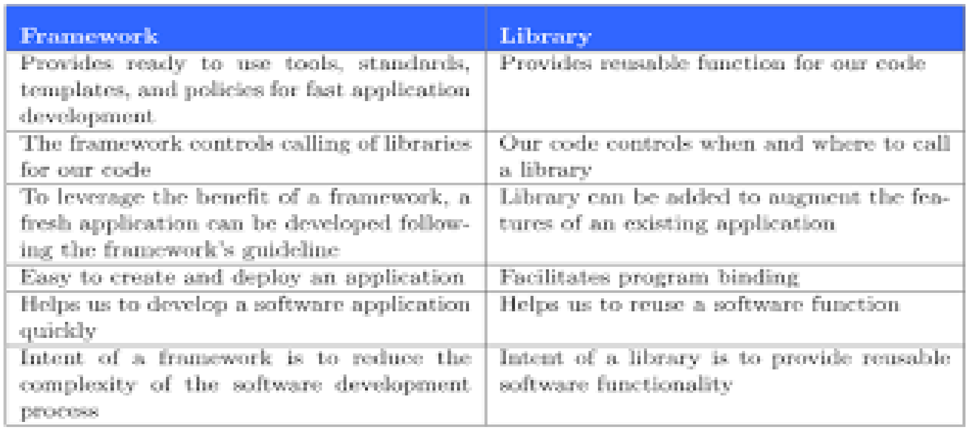
Answer Key
Part A

Q.No Question M

1. **Explain the benefits of MVC**

The MVC pattern enhances application maintainability and scalability by separating concerns into model, view, and controller 2

2. **Compare framework and library.** (any 3 points)



Framework	Library
Provides ready to use tools, standards, templates, and policies for fast application development	Provides reusable function for our code
The framework controls calling of libraries for our code	Our code controls when and where to call a library
To leverage the benefit of a framework, a fresh application can be developed following the framework's guideline	Library can be added to augment the features of an existing application
Easy to create and deploy an application	Facilitates program binding
Helps us to develop a software application quickly	Helps us to reuse a software function
Intent of a framework is to reduce the complexity of the software development process	Intent of a library is to provide reusable software functionality

2

3. **Write a JS program to get the current date.**

- Using Date().toDateString() method
- Getting the day, month, and year using their respective methods 2

Example

```
let date = new Date().toDateString();
console.log(date);
```

4. List out the limits of Angular JS **(any 4 points)**

- Outdated
- Long learning curve
- Limited mobile support
- Dependency on javascript only large file size
- Limited seo support and
- Less support from third party library.

2

5. Define AJAX- $http

Asynchronous JavaScript and XML (AJAX) is a combination of web application development technologies that make web applications more responsive to user interaction. 2

Whenever your users interact with a web application, such as when they click buttons or checkmark boxes, the browser exchanges data with the remote server.

PART B

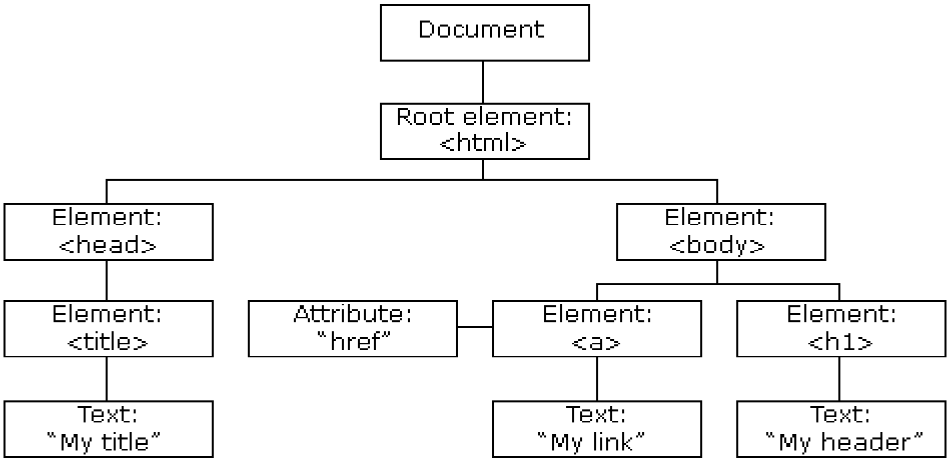
6. (a) **Explain the structure of the DOM tree in detail.** 13

DOM, or Document Object Model, is a programming interface that represents structured documents like HTML and XML as a tree of objects.

It defines how to access, manipulate, and modify document elements using scripting languages like JavaScript

.

The HTML DOM model is constructed as a tree of Objects:



(b) 7

Analyze the functionalities of document and resource loading in detail.

HTML document loads whenever a web client (Browser) makes a request. This process involves three important objects, which are:
Client: This makes a request for the document. It is basically the Web browser
Server: That serves the Document to the client
HTTP Protocol: The communication standard for Web

The following events happen in the loading of the document:

- The client makes a request to the server with the hosting for www.example.com
- The server responds with the default web page (Ex: index.html)

- The Client receives the HTML document through the HTTP protocol
- The Client contains an HTML parser, Which is an algorithm that parses the HTML code
- The HTML parser converts the HTML document received from the server into a Tree called a Document Object Model (DOM Tree)
- The client also makes the request to the URL present within the document like hyperlinks, stylesheets, or JS scripts, and the other resources are loaded also.
- The client contains a rendering engine that displays the final HTML DOM.
- Events during the document loading process:
- Some events in javascript represent the loading of the document with the help of which we can take actions in the script while the document is loading.

There are 4 important events in the document lifecycle which can be accessed with Javascript :
load event: When all the resources are loaded
domcontentloaded event: When the DOM tree is built and ready to create the interface
unload: When the document unloads
beforeunload: Same as unload but it confirms once from the user before unloading

(ii)Analyze the different browser events. 6

JavaScript Events Table

Event Attribute	Description
onclick	Triggered when an element is clicked.
onmouseover	Fired when the mouse pointer moves over an element.
onmouseout	Occurs when the mouse pointer leaves an element.
onkeydown	Fired when a key is pressed down.
onkeyup	Fired when a key is released.
onchange	Triggered when the value of an input element changes.
onload	Occurs when a page has finished loading.
onsubmit	Fired when a form is submitted.

onfocus	Occurs when an element gets focus.
onblur	Fired when an element loses focus.

7.a) Explain in detail about ngModel directive in detail with example.

The ngModel directive is a directive that is used to bind the values of the HTML controls (input, select, and textarea) or any custom form controls, and stores the required user value in a variable and we can use that variable whenever we require that value.

It also is used during form validations. The various form input types (text, checkbox, radio, number, email, URL, date, datetime-local time, month, week) can be used with the ngModel directive. This directive is supported by <input>, <select> & <textarea>.

Syntax:

<element ng-model="">
Content...
</element> 13

The ngModel directive stores a variable by reference, not value. Usually in binding inputs to models that are objects (e.g. Date) or collections (e.g. arrays). The phone object created has many fields which are used for validation purposes.

The following is the list of classes for validation purposes:

ng-touched: It shows that field has been touched.
ng-untouched: It shows that field has not been touched yet.
ng-valid: It specifies that the field content is valid.
ng-invalid: It specifies that the field content is not valid.
ng-dirty: It illustrates that the field has been modified.
ng-pristine: It represents that the field has not been modified yet.

(b) **Summarize the concept of Controller method & how the Controller can be implemented in an external.** 13

AngularJS controllers play a significant role in AngularJS applications. All the AngularJS application mainly relies on the controllers to control the flow of data in that application.

Basically, it controls the data of AngularJS applications and the controller is a Javascript object, created by a standard JavaScript object constructor.

The ng-controller directive defines the application controller.

In AngularJS, a controller is defined by a Javascript construction function, which is used in AngularJS scope and also the function $scope) is defined when the controller is defining and it returns the concatenation of the $scope.firstname and $scope.lastname.

Syntax:

<element ng-controller="expression">
Contents...

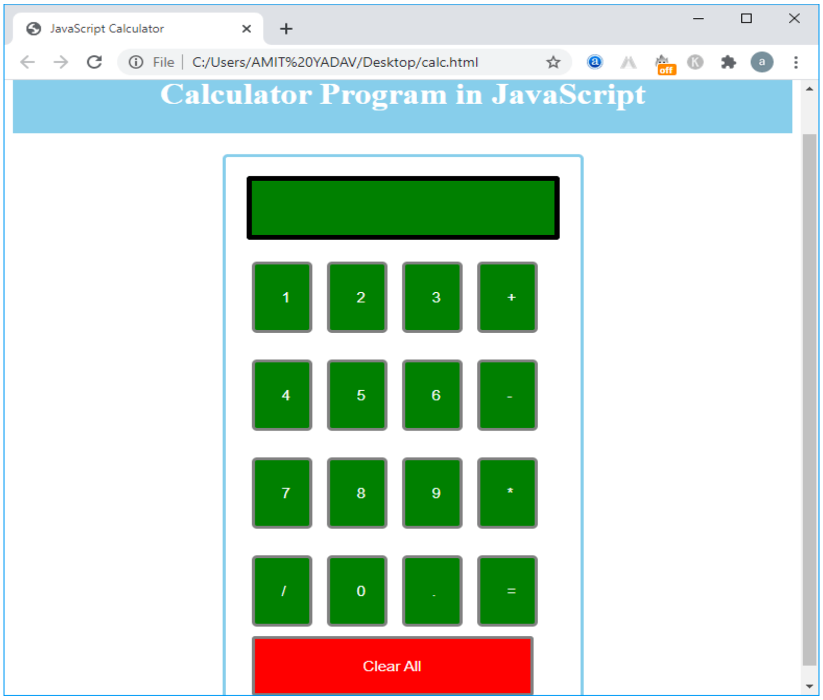
```
</element>
```

8. (a) **Construct a Calculator that performs some basic arithmetic operation using Java script code.** 14

```
<!DOCTYPE html>
<html lang = "en">
<head>
<title> JavaScript Calculator </title>
<style>
h1 {
   text-align: center;
   padding: 23px;
   background-color: skyblue;
   color: white;
   }
#clear{
width: 270px;
border: 3px solid gray;
   border-radius: 3px;
   padding: 20px;
   background-color: red;
}
.formstyle
{
width: 300px;
height: 530px;
margin: auto;
border: 3px solid skyblue;
border-radius: 5px;
padding: 20px;
}
input
{
width: 20px;
background-color: green;
color: white;
border: 3px solid gray;
   border-radius: 5px;
   padding: 26px;
   margin: 5px;
   font-size: 15px;
}
#calc{
width: 250px;
border: 5px solid black;
   border-radius: 3px;
   padding: 20px;
   margin: auto;
}
</style>
</head>
<body>
<h1> Calculator Program in JavaScript </h1>
```

```
<div class= "formstyle">
<form name = "form1">
    <!-- This input box shows the button pressed by the user in calculator. -->
  <input id = "calc" type ="text" name = "answer"> <br> <br>
  <!-- Display the calculator button on the screen. -->
  <!-- onclick() function display the number prsses by the user. -->
  <input type = "button" value = "1" onclick = "form1.answer.value += '1' ">
  <input type = "button" value = "2" onclick = "form1.answer.value += '2' ">
  <input type = "button" value = "3" onclick = "form1.answer.value += '3' ">
   <input type = "button" value = "+" onclick = "form1.answer.value += '+' ">
  <br> <br>
  <input type = "button" value = "4" onclick = "form1.answer.value += '4' ">
  <input type = "button" value = "5" onclick = "form1.answer.value += '5' ">
  <input type = "button" value = "6" onclick = "form1.answer.value += '6' ">
  <input type = "button" value = "-" onclick = "form1.answer.value += '-' ">
  <br> <br>
  <input type = "button" value = "7" onclick = "form1.answer.value += '7' ">
  <input type = "button" value = "8" onclick = "form1.answer.value += '8' ">
  <input type = "button" value = "9" onclick = "form1.answer.value += '9' ">
  <input type = "button" value = "*" onclick = "form1.answer.value += '*' ">
  <br> <br>
  <input type = "button" value = "/" onclick = "form1.answer.value += '/' ">
  <input type = "button" value = "0" onclick = "form1.answer.value += '0' ">
    <input type = "button" value = "." onclick = "form1.answer.value += '.' ">
  <input type = "button" value = "=" onclick = "form1.answer.value = eval(form
1.answer.value) ">    <br>
  <!-- Display the Cancel button and erase all data entered by the user. -->
  <input type = "button" value = "Clear All" onclick = "form1.answer.value = ' '
" id= "clear" >    <br>   </form>
</div>
</body>
</html>
```

Output

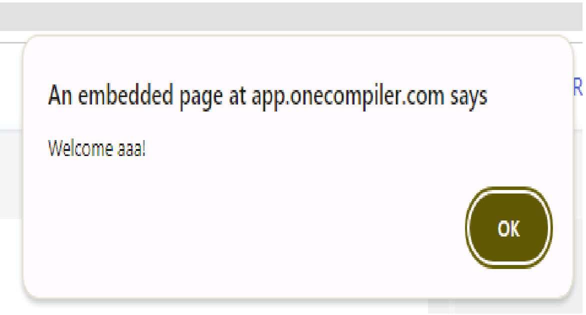

JavaScript Calculator
C:/Users/AMIT%20YADAV/Desktop/calc.html
Calculator Program in JavaScript
1
2
3
+
4
5
6
-
7
8
9
*
/
0
.
=
Clear All

(b) **Create a webpage to accept username and password. Write a javascript program to validate it. Display the alert message “welcome<<username>>” on successful login** 14

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Login Page</title>
  <style>
    body {
      font-family: Arial, sans-serif;
      display: flex;
      justify-content: center;
      align-items: center;
      height: 100vh;
      background-color: #f0f0f0;
    }
    .login-form {
      background-color: white;
      padding: 20px;
      border-radius: 8px;
      box-shadow: 0 0 10px rgba(0, 0, 0, 0.1);
    }
    .login-form h2 {
      margin-bottom: 20px;
    }
    .login-form label {
      display: block;
      margin-bottom: 8px;
    }
    .login-form input {
      width: 100%;
      padding: 8px;
      margin-bottom: 10px;
      border: 1px solid #ccc;
      border-radius: 4px;
    }
    .login-form button {
      width: 100%;
      padding: 10px;
      background-color: #28a745;
      border: none;
      border-radius: 4px;
      color: white;
      font-size: 16px;
      cursor: pointer;
    }
    .login-form button:hover {
      background-color: #218838;
    }
  </style>
</head>
```

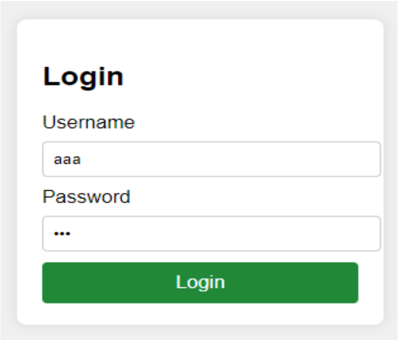
```
<body>
   <div class="login-form">
      <h2>Login</h2>
      <form id="loginForm" onsubmit="return validateForm()">
         <label for="username">Username</label>
         <input type="text" id="username" name="username" required>
         <label for="password">Password</label>
         <input type="password" id="password" name="password" required>
         <button type="submit">Login</button>
      </form>
   </div>
   <script>
      function validateForm() {
         var username = document.getElementById("username").value;
         var password = document.getElementById("password").value;
         // Simple validation: ensure both fields are filled
         if (username === "" || password === "") {
            alert("Both username and password are required.");
            return false;
         }
         // On successful validation, display the welcome message
         alert("Welcome " + username + "!");
         return true;
      }
   </script>
</body>
</html>
```

OUTPUT



An embedded page at app.onecompiler.com says

Welcome aaa!



Login

Username: aaa

Password: •••