1.a Develop a program using AngularJS to display the values for different data types

```
<html>
<head>
<title>different data types</title>
<script src="js/angularmin.js" type="text/javascript"></script>
</head>
<body>
<div ng-app="myapp" ng-controller="myctrl">
        studentname:<input type="text" ng-model="name"/><br>
        rollno:<input type="number" ng-model="rno"/>
        total:<input type="number" ng-model="total"/>
</div>
        <script>
            var app=angular.module('myapp',[]);
        app.controller('myctrl',function($scope)
        {
        $scope.name = "john";
        $scope.rno=26001;
        $scope.total=460;
         });
          </script>
            </body>
    </html>
```

1.b Develop a program using AngularJS to Basic Arithmetic Operators

```
<!DOCTYPE html >
<html ng-app="dashboard">
<head>
<title>AngularJS</title>
<script type="text/javascript" src="./js/angular.min.js"></script>
</head>
<body ng-controller="myController">
<h1>Welcome to AngularJS Example2</h1>
<table border='1'>
```

```
<tr>
<td colspan='2'>
<h2>Arithmetic Operation</h2>
</td>
</tr>
<tr>
<td>Enter First Operand</td>
<td><input type="number" ng-model="number1"></td>
</tr>
<tr>
<td>Enter Second Operand</td>
<td><input type="number" ng-model="number2"></td>
</tr>
<tr>
<td>Select Operator</td>
<td><select ng-model="operation"
ng-options="item for item in operations">
</select></td>
</tr>
<tr>
<td>Result is</td>
<td>{{result}}</td>
<tr>
<td><input type="button" ng-click="AddMe()" value="Result"></td>
</tr>
<table>
</body>
<script>
        var dashboard = angular.module("dashboard", []);
        dashboard.controller('myController', function($scope) {
        $scope.operations = [ '+', '-', '*', '/', '%' ];
        $scope.AddMe = function() {
        var num1 = $scope.number1;
        var num2 = $scope.number2;
```

```
switch ($scope.operation) {
case '+':
$scope.result = parseInt(num1) + parseInt(num2);
break;
case '-':
$scope.result = parseInt(num1) - parseInt(num2);
break;
case '*':
$scope.result = parseInt(num1) * parseInt(num2);
break;
case '/':
$scope.result = parseInt(num1) / parseInt(num2);
break;
case '%':
$scope.result = parseInt(num1) % parseInt(num2);
break;
default:
$scope.message = "Invalid Selection ";
break;
}
};
});
</script>
</html>
```

2. Demonstrate a shopping list using various built-in functions in Array

```
<html>
<head>
<title>shopping cart</title>
<script src="js/angularmin.js" type="text/javascript"></script>
</head>
<body>
    <div ng-app="myShoppingList" ng-controller="myCtrl">
    <ul>
      <li ng-repeat="x in products">
```

```
                {{x}}<span ng-click="removeItem($index)">&times;</span>
              </li>
            </ul>
        <input ng-model="addMe">
         <button ng-click="addItem()">Add</button>
         <p>{{errortext}}</p>
        </div>
<script>
        var app = angular.module("myShoppingList", []);
        app.controller("myCtrl", function($scope) {
        $scope.products = ["Milk", "Bread", "Cheese"];
        $scope.addItem = function () {
        $scope.products.push($scope.addMe);
          }
        $scope.removeItem = function (x) {
        $scope.products.splice(x, 1);
          }
        });
</script>
</body>
</html>
```

3. Change the date format using ng-model

```
        <html>
        <head>
        <title>Change date format</title>
        <script src="js/angularmin.js" type="text/javascript"></script>
        </head>
        <body>
        <div ng-app="gfg">
        <div ng-controller="dateCtrl" class="container">
         <script>
                angular.module('gfg', [])
               .controller('dateCtrl', function ($scope) {
               $scope.firstDate = new Date();
```

```
$scope.secondDate = "2020-05-20";
    })
.directive('date', function (dateFilter) {
 return {
  require: 'ngModel', link: function (scope, elm, attrs, ctrl) {
  var dateFormat =attrs['date'] || 'yyyy-MM-dd';
 ctrl.$formatters.unshift
 (
 function (modelValue )
 {
  return dateFilter(
   modelValue, dateFormat);
    });
  }
};
 })
 </script>
 <p>
 <center>
 <h1 style="color: green">
      Change the date format
 </h1>
<label for=" ">
Standard format for India
</label>
<input type="text" date='dd-MM-yyyy'
 ng-model="secondDate"
 class="form-control" />
<p class="text-primary">
<label for=" ">
Standard format for Korea
</label>
</p>
{{secondDate}}
```

```
</center>
</div>
</div>
</body>
</html>
```

4. Illustrate the use of Filter

```
<html>
<head>
<title> use of Filter</title>
<script src="angularmin.js" type="text/javascript"></script>
</head>
<body>
        <div ng-app="myApp" ng-controller="namesCtrl">
        <ul>
        <li ng-repeat="x in names | filter : 'i'">
           {{ x }}
         </li>
        </ul>
        </div>
<script>
angular.module('myApp', []).controller('namesCtrl', function($scope) {
   $scope.names = [
      'abbi',
      'jafrin',
      'sangeetha',
      'hanniya',
      'rifa',
      'sajuna' ];
});
</script>
<p>This example displays only the names containing the letter "i".</p>
 </body>
</html>
```

5. Apply Mouse events

```
<html>
<head>
<title>Apply mouse event</title>
<script src="js/angularmin.js" type="text/javascript"></script>
</head>
<body>
<div ng-app="">
<p>Total Number of Clicks: {{ ClickCount }}</p>
<button ng-click = "ClickCount = ClickCount + 1">Click Me!</button>
<p>Total Number of Double Clicks: {{ ClickCount1 }}</p>
<button ng-dblclick= "ClickCount1 = ClickCount1 + 1">Click Me!</button>
</div>
</body>
</html>
```

5. Apply keyboard Events

```
<html>
<head>
<title>keyboard event</title>
<script src="js/angularmin.js" type="text/javascript"></script>
</head>
<body>
<h4>ngkeypress Example</h4>
<div ng-app="app" ng-controller="HomeController">
<input ng-keypress="keyPress()">
<br> Key Press count: {{countKeyPress}}
</div>
<script>
var app = angular.module("app", []);
app.controller ("HomeController", function($scope)
{
   $scope.countKeyPress = 0;
   $scope.keyPress = function()
   {
```

```
        $scope.countKeyPress = $scope.countKeyPress + 1;
    }
});
</script>
</body>
</html>
```

6. Slide Toggle Animation with Angular

```
<html lang="en">
<head>
<meta charset="utf-8">
<title>slideToggle demo</title>
<style>
 p {
     width: 400px;
      }
 </style>
<script src="https://code.jquery.com/jquery-3.5.0.js"></script>
<script src="js/angularmin.js" type="text/javascript"></script>
</head>
<body>
<button>Toggle</button>
   <p>
      <em>lucky for your life</em>
   </p>
   <script>
$( "button" ).click(function() {
$( "p" ).slideToggle( "slow" );
});
 </script>
</body>
</html>
```

7. Simple calculator

```
<html>
<head>
<title>Simple calculator</title>
<script src="js/angularmin.js" type="text/javascript"></script>
</head>
<body>
<h1>AngularJS calculator</h1>
<div ng-app="CalculatorApp" >
<div class="formstyle">
    <form name="form1">
      <input id="calc" type="text" name="answer"> <br> <br>
      <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(form1.answer.value) ">
      <br><br />
```

```
            <input type="button" value="Clear All" onclick="form1.answer.value = ' ' "
        id="clear">
            <br>
    </form>
      </div>
      </body>
    </html>
```

8. validate the username and password

```
    <html>
    <head>
    <title>Login form</title>
    <script src="js/angularmin.js" type="text/javascript"></script>
    </head>
    <body>
    <form class="form-horizontal" name="loginForm"
      novalidate ng-app="formValidationApp" ng-controller="ValidationCtrl">
    <div class="form-heading">
    <h2>Sign In</h2>
    </div>
    <div class="form-group"
        ng-class="{'has-error': loginForm.email.$invalid &&
        loginForm.email.$dirty, 'has-success': loginForm.email.$valid }">
      <input type="email" class="form-control" placeholder="Email"
          name="email" ng-model="user.email" required/>
      <p class="help-block" ng-if="loginForm.email.$invalid &&
          loginForm.email.$dirty">Please Enter a valid email address</p>
      </div>
      <div class="form-group" ng-class="{'has-error':
          loginForm.password.$invalid &&
     loginForm.password.$dirty, 'has-success': loginForm.password.$valid}">
      <input type="password" class="form-control" placeholder="Password"
       name="password" ng-model="user.password" ng-minLength="8" required/>
      <p class="help-block" ng-if="loginForm.password.$invalid &&
       loginForm.password.$dirty">Please Enter at least 8 characters</p>
```

```
  </div>
  <button class="btn btn-default pull-right"
    ng-disabled="loginForm.emails.$invalid ||
   loginForm.password.$invalid">Sign In</button>
</form>
      <script>
        angular.module('formValidationApp', [])
.controller('ValidationCtrl', function (){
})
</script>
</body>
</html>
```

10. Develop an application based on Node.js to demonstrate Lookup () functions on DNS

```
const dns = require('dns');
// Function to lookup a domain
function lookupDomain(domain) {
dns.lookup(domain, (err, address, family) => {
      if (err) {
        console.error(`Error resolving domain: ${err.message}`);
      } else {
        console.log(`Address for ${domain}: ${address}`);
        console.log(`IP Family: IPv${family}`);
            }
       });
        }
lookupDomain('google.com');
lookupDomain('yahoo.com');
lookupDomain('nodejs.org');
```

11. Develop an application based on Node.js to display OS details using Utility module

```
const os = require('os');
// Function to display OS details
function displayOSDetails() {
console.log("Operating System Details:");
// Get the OS platform
console.log(`Platform: ${os.platform()}`);
 // Get the OS architecture (32-bit or 64-bit)
console.log(`Architecture: ${os.arch()}`);
 // Get the hostname of the machine
console.log(`Hostname: ${os.hostname()}`);
// Get the system's uptime in seconds
 console.log(`System Uptime: ${os.uptime()} seconds`);
 // Get the current user information
 const userInfo = os.userInfo();
 console.log(`User Info:`);
 console.log(`  Username: ${userInfo.username}`);
  console.log(`  Home Directory: ${userInfo.homedir}`);
    }
// Call the function to display OS details
 displayOSDetails();
```

12. Develop an application based on Node.js to file operations

```
const fs = require('fs');
const path = require('path');
// Define file path
const filePath = path.join(__dirname, 'example.txt');
// Create and write to a new file
fs.writeFile(filePath, 'Hello, this is a test file!\n', (err) => {
        if (err) {
        return console.log('Error creating file:', err);
          }
  console.log('File created and data written!');
  // Read the content of the file
  fs.readFile(filePath, 'utf8', (err, data) => {
          if (err) {
        return console.log('Error reading file:', err);
          }
console.log('File content:', data);
// Append data to the file
fs.appendFile(filePath, 'Appending some more text.\n', (err) => {
        if (err) {
         return console.log('Error appending to file:', err);
                }
  console.log('Data appended to the file!');
  // Read the updated file content
  fs.readFile(filePath, 'utf8', (err, updatedData) => {
          if (err) {
        return console.log('Error reading file:', err);
                }
   console.log('Updated file content:', updatedData);
     // Rename the file
```

```
            const newFilePath = path.join(__dirname, 'new_example.txt');
            fs.rename(filePath, newFilePath, (err) => {
                if (err) {
                    return console.log('Error renaming file:', err);
                }
                console.log('File renamed successfully!');
               // Delete the renamed file
                fs.unlink(newFilePath, (err) => {
                    if (err) {
                        return console.log('Error deleting file:', err);
                    }
                    console.log('File deleted successfully!');
                });
            });
        });
    });
});
```