



Bharati Vidyapeeth's
**Institute of Management & Information
Technology**

C.B.D. Belapur, Navi Mumbai 400614

Vision:

Providing high quality, innovative and value-based education in information technology to build competent professionals.

Mission

M1. Technical Skills:-To provide solid technical foundation theoretically as well as practically capable of providing quality services to industry.

M2. Development: -Department caters to the needs of students through comprehensive educational programs and promotes lifelong learning in the field of computer Applications.

M3. Ethical leadership:-Department develops ethical leadership insight in the students to succeed in industry, government and academia.

CERTIFICATE

This is to certify that the journal is the work of

Mr. Sainath Ravi Machha Roll No. 32 of MCA

(Sem- 1 Div:A) Batch: A2 for the academic year 2023 - 2024

Subject Code: MCAL14

Subject Name: Web Technology Lab

Subject-in-charge

Date:

Principal

External Examiner

Date:

INDEX

Sr No.	Topic	Sign
1	Nodejs Module	
1.1	Create an application to demonstrate Node.js Modules.	
2	Events	
2.2	Create an application to demonstrate various Node.js Events.	
2.3	Implement all Methods of EventEmitter class.	
	Create an application to demonstrate Node.js Functions	
3	File System and HTTP Server	
3.1	Create an HTTP Server and perform operations on it.	
3.2	Using File Handling demonstrate all basic file operations (Create, write, read, delete)	
4	MySQL database connectivity.	
4.1	Create an application to establish a connection with the MySQL database and perform basic database operations on it.	
5	AngularJs	
5.1	Write a program in AngularJs of expression for operators and variables .	
5.2	Write a program in AngularJs of expression contains any two data type.	
5.3	Write a program in AngularJs of expression for arithmetic operators which will produce the result based on the type of operands.	
5.4	Write a program in AngularJs which demonstrates handling click event of a button	
5.5	Write a program in AngularJs for scope object where controller available to the HTML elements and its child elements	
5.6	Write a program in AngularJs demonstrates multiple controllers.	
5.7	Write a program in AngularJs to demonstratesng-init directive for string, number, array, and object.	
5.8	Write a program in AngularJs to demonstrates ng-if, ng-readonly, and ng- disabled directives.	
5.9	Write a program in AngularJs for currency filter to person salary.	
6.0	Write a program in AngularJs demonstrates Date filter.	
6.1	Write a program in AngularJs upper case and lowercase filter	
6.2	Write a program in AngularJs to demonstrates mouse event	

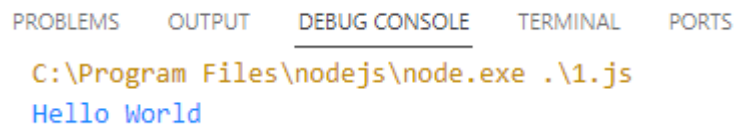
1. Nodejs Module

1.1 Create an application to demonstrate Node.js Modules.

1. Hello World

```
console.log("Hello World");
```

Output:

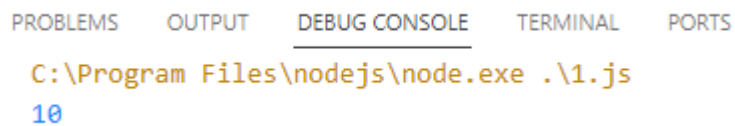


A screenshot of the Visual Studio Code output window. The 'DEBUG CONSOLE' tab is selected. The command prompt shows the execution of a Node.js script: 'C:\Program Files\nodejs\node.exe .\1.js'. The output of the script is 'Hello World'.

2. Multiplication

```
function myfun(x,y)
{
return x*y;
}
console.log (myfun(2,5));
```

Output:

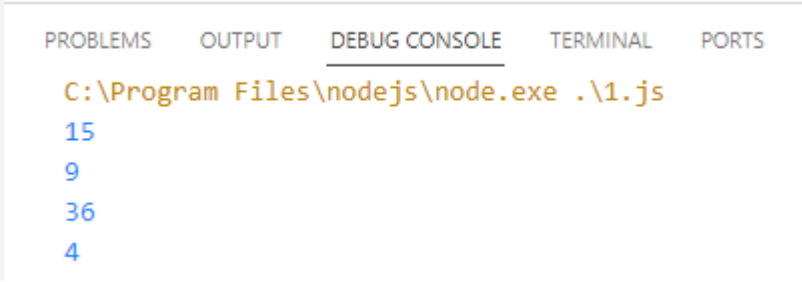


A screenshot of the Visual Studio Code output window. The 'DEBUG CONSOLE' tab is selected. The command prompt shows the execution of a Node.js script: 'C:\Program Files\nodejs\node.exe .\1.js'. The output of the script is '10'.

3. Fibonacci

```
function myfun(num1,num2)
{
  console.log(num1+num2);
  console.log(num1-num2);
  console.log(num1*num2);
  console.log(num1/num2);
}
myfun(12,3);
```

Output:



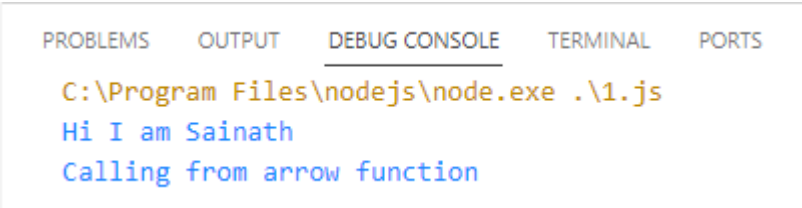
The screenshot shows the VS Code interface with the 'OUTPUT' tab selected. The command prompt shows the execution of a JavaScript file. The output consists of four lines of numbers: 15, 9, 36, and 4.

```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS
C:\Program Files\nodejs\node.exe .\1.js
15
9
36
4
```

4. Callback Function

```
const message=function(){ console.log("Hi I am Sainath");
}
setTimeout(message,3000);
setTimeout(()=>{console.log("Calling from arrow function");
},3000);
```

Output:



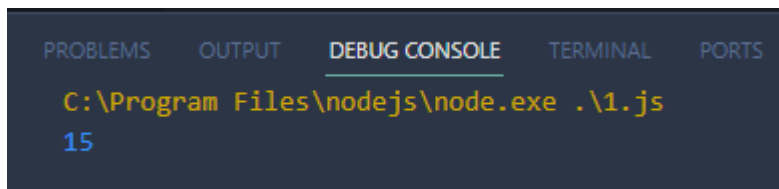
The screenshot shows the VS Code interface with the 'OUTPUT' tab selected. The command prompt shows the execution of a JavaScript file. The output consists of two lines of text: 'Hi I am Sainath' and 'Calling from arrow function'.

```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS
C:\Program Files\nodejs\node.exe .\1.js
Hi I am Sainath
Calling from arrow function
```

5. Javascript callback

```
function displayresult(some)
{
    console.log(some);
}
function calculate(x,y,mycallback)
{
    let sum=x+y;
    mycallback(sum);
}
calculate(5,10,displayresult);
```

Output:

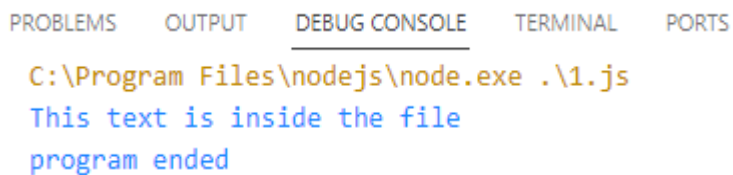


The screenshot shows the VS Code interface with the 'DEBUG CONSOLE' tab selected. The output displays the file path 'C:\Program Files\nodejs\node.exe .\1.js' followed by the number '15' on a new line.

6. Block Code

```
var fs=require('fs');
var data=fs.readFileSync('input.txt');
console.log(data.toString());
console.log("program ended")
```

Output:



The screenshot shows the VS Code interface with the 'DEBUG CONSOLE' tab selected. The output displays the file path 'C:\Program Files\nodejs\node.exe .\1.js' followed by two lines of text: 'This text is inside the file' and 'program ended' on separate lines.

7. Non Block code

```
var fs=require('fs');
fs.readFile("input.txt",function(err,data){
    if(err)
    {
        return console.error(err);
    }
    console.log(data.toString());
});
console.log("program ended");
```

Output:

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
C:\Program Files\nodejs\node.exe .\1.js
program ended
This text is inside the file
```

2. Events

2.1 Create an application to demonstrate various Node.js Events.

```
// step 1 importing event
const events = require("events");

// step 2 creating an Event emitter object
const EventEmitter = new events.EventEmitter();

//write a function of event 1
function listner1() {
    console.log("Event received by Listner 1");
}

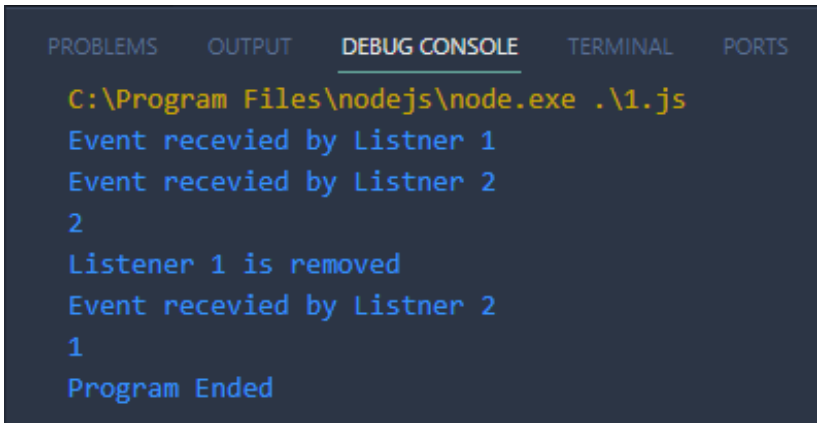
//write a function of event 2
function listner2() {
    console.log("Event received by Listner 2");
}

// step 3 adding listener through addlistener or on
EventEmitter.addListener("write", listner1);
EventEmitter.on("write", listner2);

// step 4 emitting event
EventEmitter.emit("write");
console.log(EventEmitter.listenerCount("write"));

// step 5 removing listener
EventEmitter.removeListener("write", listner1);
console.log("Listener 1 is removed");
EventEmitter.emit("write");
console.log(EventEmitter.listenerCount("write"));
console.log("Program Ended");
```

Output:



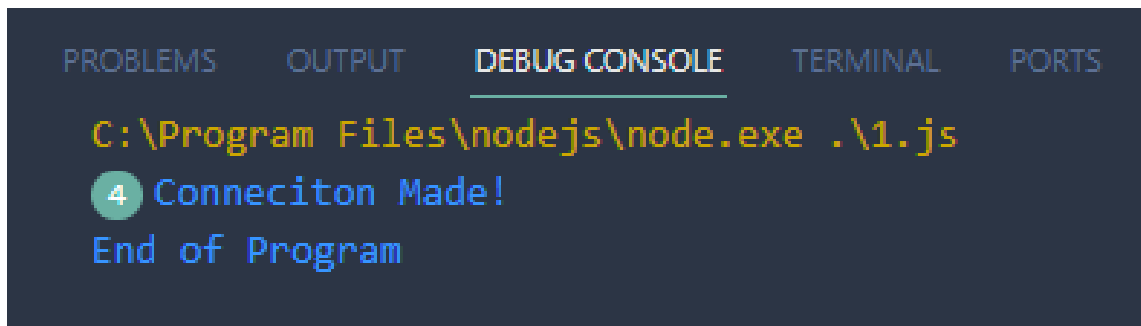
```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS

C:\Program Files\nodejs\node.exe .\1.js
Event received by Listner 1
Event received by Listner 2
2
Listener 1 is removed
Event received by Listner 2
1
Program Ended
```

2.2 Implement all Methods of EventEmitter class.

```
const events = require("events");
const EventEmitter = new events.EventEmitter();
eventEmitter.on("connection", handleConnectionEvent);
eventEmitter.emit("connection");
eventEmitter.emit("connection");
eventEmitter.emit("connection");
eventEmitter.emit("connection");
function handleConnectionEvent() {
    console.log("Conneciton Made!");
}
console.log("End of Program");
```

Output:



```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS

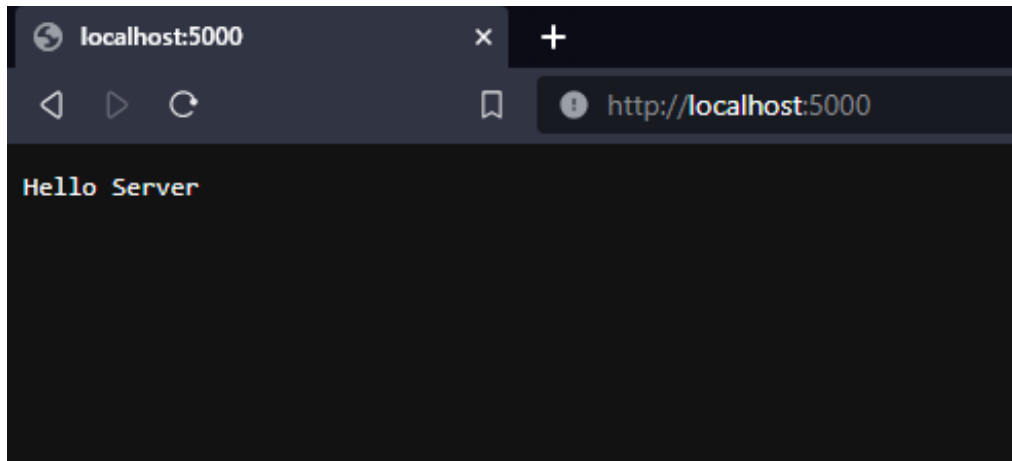
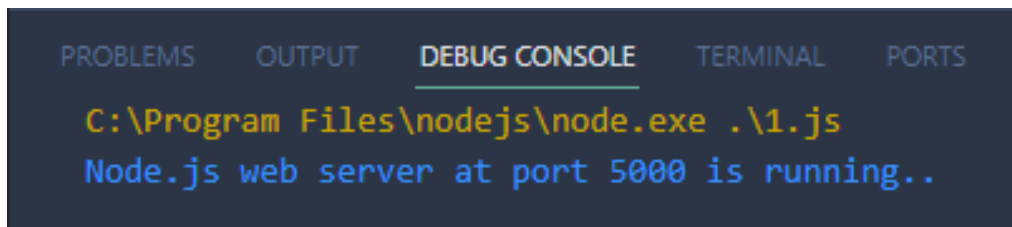
C:\Program Files\nodejs\node.exe .\1.js
4 Conneciton Made!
End of Program
```


2.3 Create an application to demonstrate Node.js Functions.

```
// understand http request module
var http = require('http'); // 1 - Import Node.js core module
var server = http.createServer(function (req, res) { // 2 - creating
server

    //handle incomming requests here..
    res.write("Hello Server");
    res.end();
});
server.listen(5000); //3 - listen for any incoming requests
console.log('Node.js web server at port 5000 is running..')
```

Output:

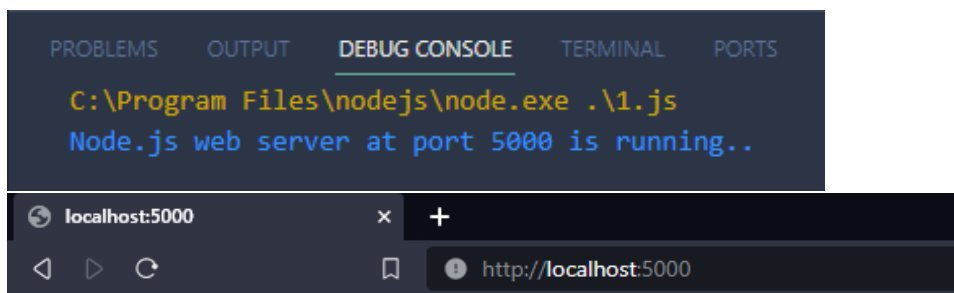


3. File System and HTTP Server

3.1 Create an HTTP Server and perform operations on it.

```
//understand routing in http module
var http = require('http'); // Import Node.js core module
var server = http.createServer(function (req, res) { //create web server
  if (req.url == '/') { //check the URL of the current request
    // set response header
    res.writeHead(200, { 'Content-Type': 'text/html' });
    // set response content
    res.write('<html><body><p>This is home Page.</p></body></html>');
    res.end();
  }
  else if (req.url == "/student") {
    res.writeHead(200, { 'Content-Type': 'text/html' });
    res.write('<html><body><p>This is student Page.</p></body></html>');
    res.end();
  }
  else if (req.url == "/admin") {
    res.writeHead(200, { 'Content-Type': 'text/html' });
    res.write('<html><body><p>This is admin Page.</p></body></html>');
    res.end();
  }
  else
    res.end('Invalid Request!');
});
server.listen(5000); //6 - listen for any incoming requests
console.log('Node.js web server at port 5000 is running..')
```

Output:



This is home Page.

3.2 Using File Handling demonstrate all basic file operations (Create, write, read, delete)

```
//Writing file
var fs = require('fs');

fs.writeFile('test.txt', 'Hello World!', function (err) {
  if (err)
    console.log(err);
  else
    console.log('Write operation complete.');
```



```
});

fs.readFile('test.txt', function (err, data) {
  if (err) throw err;
  console.log(data.toString());
});

fs.unlink('test.txt', function () {

  console.log('delete operation complete.');
```

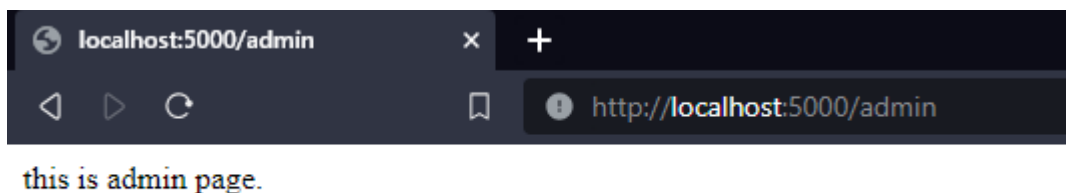


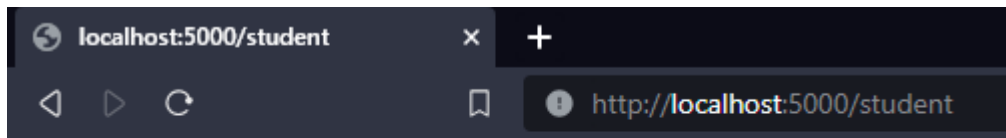
```
});
```

Output:

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
C:\Program Files\nodejs\node.exe .\1.js
node.js web server at port 5000 is running
```



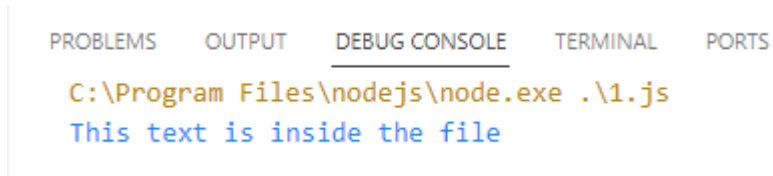


this is student page.

>> Reading a file

```
var fs = require('fs');
fs.readFile('input.txt', function (err, data) {
  if (err) throw err; console.log(data.toString());
});
```

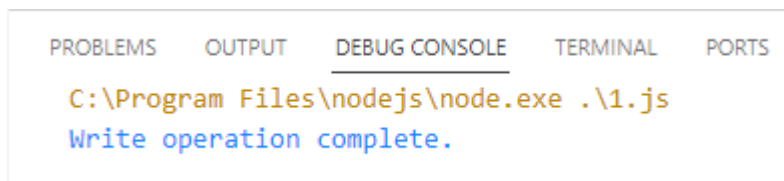
Output:

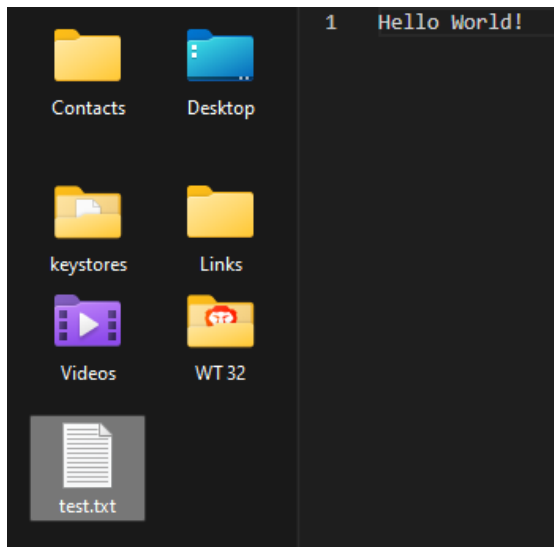


>>Write to File

```
var fs = require('fs');
fs.writeFile('test.txt', 'Hello World!', function (err) {
  if (err)
    console.log(err); else
    console.log('Write operation complete.');
```

Output:





>>Update the file

```
var fs = require('fs');

fs.appendFile('test.txt', ' Hey Hello', function (err) {
  if (err) throw err;
  console.log('Updated!');
});
```

Output:

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL

C:\Program Files\nodejs\node.exe .\1.js

Updated!

```
1   Hello World! Hey Hello
```

>> Rename

```
var fs = require('fs');

fs.rename('input.txt', 'myrenamedfile.txt', function (err) {
  if (err) throw err;
  console.log('File Renamed!');
});
```

Output:

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL

C:\Program Files\nodejs\node.exe .\1.js
File Renamed!

>>Delete the file

```
var fs = require('fs');

fs.unlink('myrenamedfile.txt', function (err) {
  if (err) throw err;
  console.log('File deleted!');
});
```

Output:

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL

C:\Program Files\nodejs\node.exe .\1.js
File deleted!

>> Buffer

```
var fs = require('fs');
fs.open('test.txt', 'r', function (err, fd) {
  if (err) {
    return console.error(err);
  }
  var buffr = new Buffer(10240); fs.read(fd, buffr, 0, buffr.length, 0,
function (err, bytes) {
  if (err) throw err; if (bytes > 0) {
    console.log(buffr.slice(0, bytes).toString());
  }
  fs.close(fd, function (err) {
    if (err) throw err;
  });
});
});
```

Output:

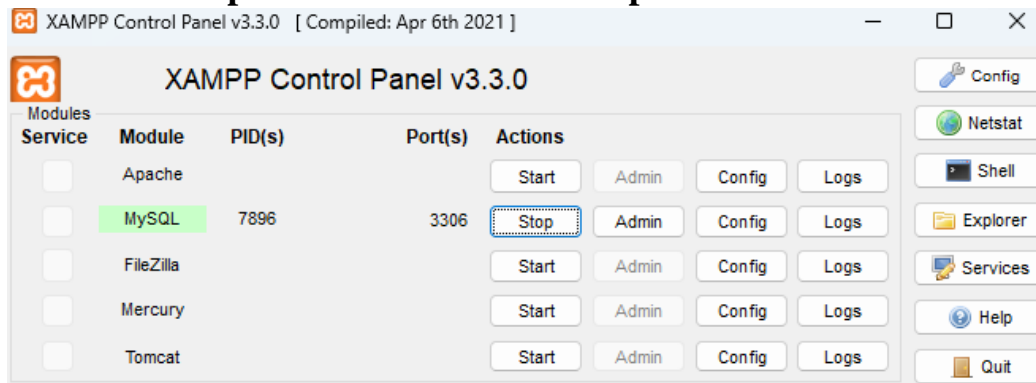
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS SEARCH ERROR COMMENTS

C:\Program Files\nodejs\node.exe .\1.js
(node:11516) [DEP0005] DeprecationWarning: Buffer() is deprecated due to security and usability issues. Please use the Buffer.allocUnsafe() method, Buffer.allocUnsafeSlow(), or Buffer.from() methods instead.
(Use `node --trace-deprecation ...` to show where the warning was created)
Hello World! Hey Hello

js:8.1

4. MySQL database connectivity.

4.1 Create an application to establish a connection with the MySQL database and perform basic database operations on it.



```
Command Prompt
Microsoft Windows [Version 10.0.22621.1105]
(c) Microsoft Corporation. All rights reserved.

C:\Users\Student>node -v
v18.12.1

C:\Users\Student>npm -v
8.19.2

C:\Users\Student>npm install mysql

up to date, audited 12 packages in 732ms

found 0 vulnerabilities

C:\Users\Student>init -y
'init' is not recognized as an internal or external command,
operable program or batch file.

C:\Users\Student>npm init -y
Wrote to C:\Users\Student\package.json:

{
  "dependencies": {
    "mysql": "^2.18.1"
  },
  "name": "student",
  "version": "1.0.0",
  "main": "index.js",
  "scripts": {
    "test": "echo \"Error: no test specified\" && exit 1"
  },
  "keywords": [],
  "author": "",
  "license": "ISC",
  "devDependencies": {},
  "description": ""
}
```

Creating database

```
var mysql = require('mysql');
var con = mysql.createConnection({
  host: "localhost", user: "root",
  password: ""
});
con.connect(function (err) {
  if (err) throw err; console.log("connected!");
  con.query("create database mydb1", function (err, result) {
    if (err) throw err;
    console.log("Database created");
  });
});
```

Output:

```
Node.js v18.12.1
PS C:\Users\Student\database> node demo.js
Connected!
Database create
```

Creating Table

```
var mysql = require('mysql');
var con = mysql.createConnection({
  host: "localhost",
  user: "root",
  password: "", database: "mydb1"
})

con.connect(function (err) {
  if (err) throw err;
  console.log("Connected!");
  var sql = "CREATE TABLE cutomers11 (name VARCHAR(255),address
  VARCHAR(255))";
  con.query(sql, function (err, result) {
    if (err) throw err; console.log("Table creates");
  });
});
```

Output:

```
PS C:\Users\Student\database> node db01
Connected!
Table creates
```


Primary Key

```
var mysql = require('mysql');
var con = mysql.createConnection({
  host: "localhost",
  user: "root",
  password: "", database: "mydb01"
})

con.connect(function (err) {
  if (err) throw err;
  console.log("Connected!");
  var sql = "CREATE TABLE cutomers01 (id int auto_increment primary
key,name VARCHAR(255),address VARCHAR(255))";
  con.query(sql, function (err, result) {
    if (err) throw err; console.log("Table creates");
  });
});
```

Output:

```
PS C:\Users\Student\database> node db01.js
Connected!
Table creates
```

Connectivity

```
var mysql = require('mysql');
var con = mysql.createConnection({
  host: "localhost",
  user: "root",
  password: "", database: "mydb01"
})

con.connect(function (err) {
  if (err) throw err;
  console.log("Connected!");
});
```

Output:

```
PS C:\Users\Student\database> node connecttivity.js
Connected!
```

Insertion

```
var mysql = require('mysql');
var con = mysql.createConnection({
  host: "localhost", user: "root",
  password: "", database: "mydb1"
```

```

});
con.connect(function (err) {
  if (err) throw err; console.log("connected!");
  var sql = "Insert INTO customers21 (name,address) VALUES ('company
inc','Highway 32')";
  var sql = "Insert INTO customers21 (name,address) VALUES ('company
ltd','Highway Glory')";
  var sql = "Insert INTO customers21 (name,address) VALUES
('citypride','under the sky')";
  con.query(sql, function (err, result) {
    if (err) throw err;
    console.log("1 record inserted");
  });
});
});

```

Output:

```

PS C:\Users\Student\database> node insert.js
Connected!
1 record inserted

```

Reading Data

```

var mysql = require('mysql');
var con = mysql.createConnection({
  host: "localhost",
  user: "root",
  password: "",
  database: "mydb01"
});

con.connect(function (err) {
  if (err) throw err;
  console.log("Connected!");
  var sql = "INSERT INTO cutomers01 (name, address) VALUES ('Company
st','Highway 05')";
  var sql = "INSERT INTO cutomers01 (name, address) VALUES ('Company
pvi','Highway NH06')";
  var sql = "INSERT INTO cutomers01 (name, address) VALUES ('Service
Inc','High street')";
  con.query(sql, function (err, result) {
    if (err) throw err;
    console.log("1 record inserted");
  });
});
});

```

Output:

```
PS C:\Users\Student\database> node show.js
[
  RowDataPacket { id: 1, name: 'Company Inc', address: 'Highway 84' },
  RowDataPacket { id: 2, name: 'Service Inc', address: 'High street' },
]
```

Update

```
var mysql = require('mysql');
var con = mysql.createConnection({
  host: "localhost",
  user: "root",
  password: "", database: "mydb01"
});

con.connect(function (err) {
  if (err) throw err;
  var sql = "UPDATE cutomers01 SET address = 'canyon' WHERE address = 'Highway 84'";
  con.query(sql, function (err, result) {
    if (err) throw err;
    console.log(result.affectedRows + "record(s) updated");
  });
});
```

Output:

```
PS C:\Users\Student\database> node update.js
0record(s) updated
```

After Update

```
PS C:\Users\Student\database> node show.js
[
  RowDataPacket { id: 1, name: 'Company Inc', address: 'canyon' },
  RowDataPacket { id: 2, name: 'Service Inc', address: 'High street' },
]
```

Delete

```
var mysql = require('mysql');
var con = mysql.createConnection({
  host: "localhost",
  user: "root",
  password: "", database: "mydb01"
});

con.connect(function (err) {
  if (err) throw err;
  var sql = "DELETE FROM cutomers01 WHERE address = 'High street'";
  con.query(sql, function (err, result) {
    if (err) throw err;
```

```
        console.log("Number of records deleted: " + result.affectedRows);  
    });  
});
```

Output:

```
PS C:\Users\Student\database> node show.js  
[ RowDataPacket { id: 1, name: 'Company Inc', address: 'canyon' } ]  
█
```

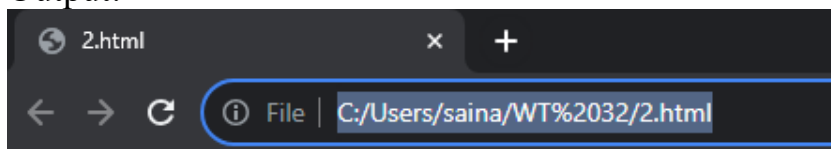
```
PS C:\Users\Student\database> node delete  
Number of records deleted: 4  
█
```

5. AngularJs

5.1 Write a program in AngularJs of expression for operators and variables.

```
<!DOCTYPE html>
<html >
<head>
  <script
src="https://ajax.googleapis.com/ajax/libs/angularjs/1.8.2/angular.min.js"></s
cript>
</head>
<body >
  <h1>AngularJS Expression Demo:</h1>
  <div ng-app>
    2 + 2 = {{2 + 2}} <br />
    2 - 2 = {{2 - 2}} <br />
    2 * 2 = {{2 * 2}} <br />
    2 / 2 = {{2 / 2}}
  </div>
</body>
</html>
```

Output:



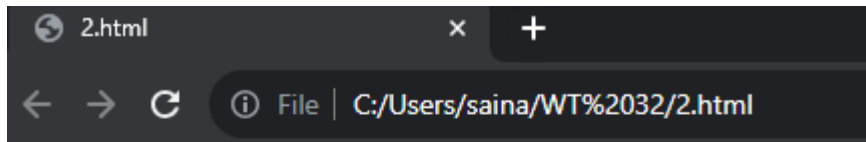
AngularJS Expression Demo:

2 + 2 = 4
2 - 2 = 0
2 * 2 = 4
2 / 2 = 1

5.2 Write a program in AngularJs of expression contains any two data type.

```
<html >
<head>
  <script
src="https://ajax.googleapis.com/ajax/libs/angularjs/1.8.2/angular.min.js"></s
cript>
</head>
<body >
  <h1>AngularJS Expression Demo:</h1>
  <div ng-app>
    {{ "Hello World" }}<br />
    {{ 100 }}<br />
    {{ true }}<br />
    {{ 10.2 }}
  </div>
</body>
</html>
```

Output:



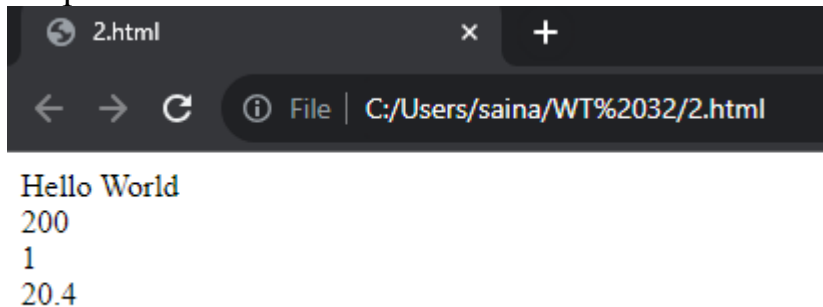
AngularJS Expression Demo:

Hello World
100
true
10.2

5.3 Write a program in AngularJs of expression for arithmetic operators which will produce the result based on the type of operands

```
<!DOCTYPE html>
<html >
<head>
  <script
src="https://ajax.googleapis.com/ajax/libs/angularjs/1.8.2/angular.min.js"></s
cript>
</head>
<body >
  <div ng-app>
    {{ "Hello" + " World" }}<br />
    {{ 100 + 100 }}<br />
    {{ true + false }}<br />
    {{ 10.2 + 10.2 }}<br />
  </div>
</body>
</html>
```

Output:



5.4 Write a program in AngularJs which demonstrates handling click event of a button

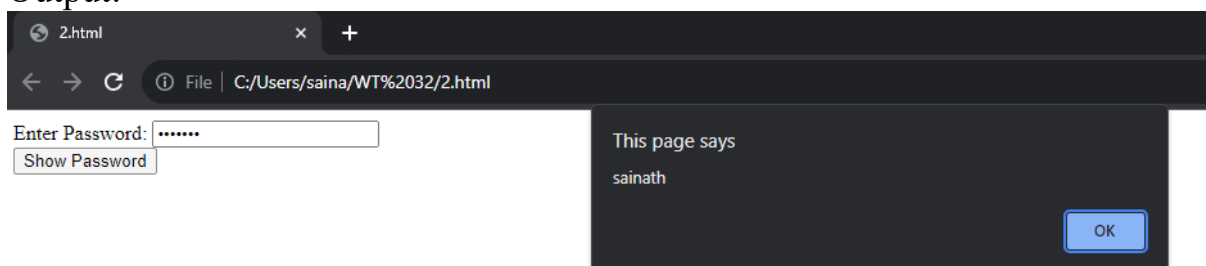
```
<!DOCTYPE html>
<html >
<head>
  <script
src="https://ajax.googleapis.com/ajax/libs/angularjs/1.8.2/angular.min.js"></s
cript>
</head>
<body ng-app="myApp">
  <div ng-controller="myController">
    Enter Password: <input type="password" ng-model="password" /> <br
/>

    <button ng-click="DisplayMessage(password)">Show Password</button
</div>
<script>
  var myApp = angular.module('myApp', []);

  myApp.controller("myController", function ($scope, $window) {

    $scope.DisplayMessage = function (value) {
      $window.alert(value)
    }
  });
</script>
</body>
</html>
```

Output:



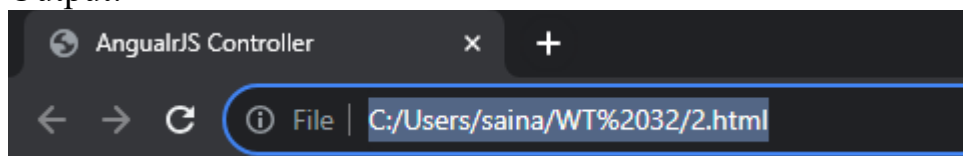
5.5 Write a program in AngularJs for scope object where controller available to the HTML elements and its child elements

```
<!DOCTYPE html>
<html>
<head>
  <title>AngularJS Controller</title>
  <script
src="https://ajax.googleapis.com/ajax/libs/angularjs/1.8.2/angular.min.js"></s
cript>
</head>
<body ng-app="myNgApp">
  <div ng-controller="parentController">
    Message: {{message1}}
    <div ng-controller="childController">
      Parent Message: {{message1}} </br>
      Child Message: {{message2}}
    </div>
    Child Message: {{message2}}
  </div>
  <script>
    var ngApp = angular.module('myNgApp', []);

    ngApp.controller('parentController', function ($scope) {
      $scope.message1 = "This is parentController";
    });

    ngApp.controller('childController', function ($scope) {
      $scope.message2 = "This is childController";
    });
  </script>
</body>
</html>
```

Output:



Message: This is parentController
Parent Message: This is parentController
Child Message: This is childController
Child Message:

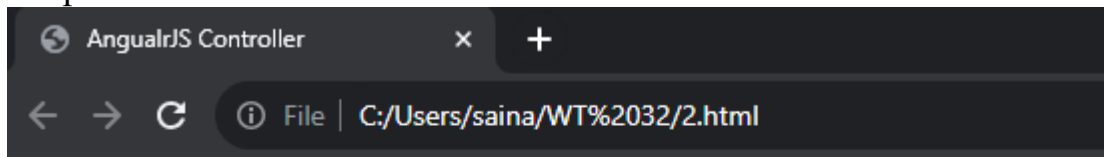
5.6 Write a program in AngularJs demonstrates multiple controllers.

```
<!DOCTYPE html>
<html>
<head>
  <title>AngularJS Controller</title>
  <script
src="https://ajax.googleapis.com/ajax/libs/angularjs/1.8.2/angular.min.js"></s
cript>
</head>
<body ng-app="myNgApp">
  <div id="div1" ng-controller="myController">
    Message: {{message}} <br />
    <div id="div2">
      Message: {{message}}
    </div>
  </div>
  <div id="div3">
    Message: {{message}}
  </div>
  <div id="div4" ng-controller="anotherController">
    Message: {{message}}
  </div>
  <script>
    var ngApp = angular.module('myNgApp', []);

    ngApp.controller('myController', function ($scope) {
      $scope.message = "This is myController";
    });

    ngApp.controller('anotherController', function ($scope) {
      $scope.message = "This is anotherController";
    });
  </script>
</body>
</html>
```

Output:

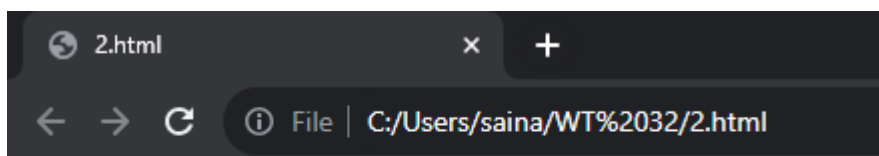


Message: This is parentController
Parent Message: This is parentController
Child Message: This is childController
Child Message:

5.7 Write a program in AngularJs to demonstratesng-init directive for string, number, array, and object.

```
<!DOCTYPE html>
<html >
<head>
  <script
src="https://ajax.googleapis.com/ajax/libs/angularjs/1.8.2/angular.min.js"></s
cript>
</head>
<body >
  <div ng-app ng-init="greet='Hello World!'; amount= 100; myArr = [100,
200]; person = { firstName:'Steve', lastName : 'Jobs'}">
    {{amount}}    <br />
    {{myArr[1]}}  <br />
    {{person.firstName}}
  </div>
</body>
</html>
```

Output:

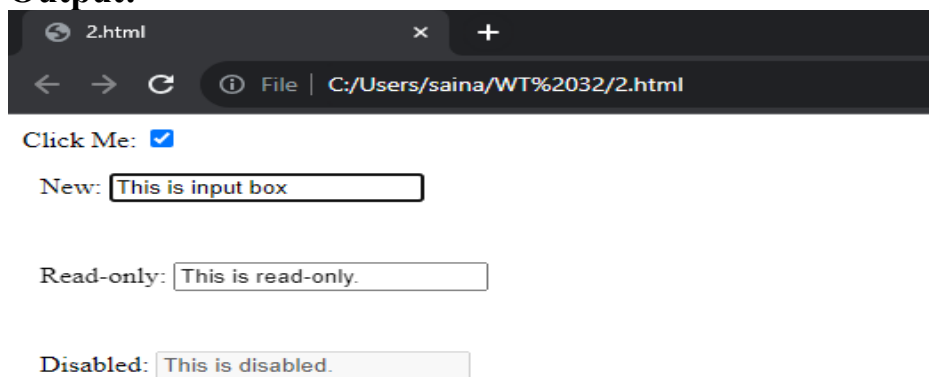


100
200
Steve

5.8 Write a program in AngularJs to demonstrates ng-if, ng-readonly, and ng-disabled directives.

```
<!DOCTYPE html>
<html>
<head>
  <script
src="https://ajax.googleapis.com/ajax/libs/angularjs/1.8.2/angular.min.js"></s
cript>
  <style>
    div {
      width: 100%;
      height: 50px;
      display: block;
      margin: 15px 0 0 10px;
    }
  </style>
</head>
<body ng-app ng-init="checked=true" >
  Click Me: <input type="checkbox" ng-model="checked" /> <br />
  <div>
    New: <input ng-if="checked" type="text" />
  </div>
  <div>
    Read-only: <input ng-readonly="checked" type="text" value="This is
read-only." />
  </div>
  <div>
    Disabled: <input ng-disabled="checked" type="text" value="This is
disabled." />
  </div>
</body>
</html>
```

Output:

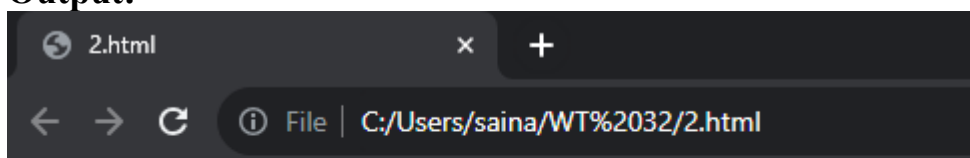


5.9 Write a program in AngularJs for currency filter to person salary.

```
<!DOCTYPE html>
<html >
<head>
  <script
src="https://ajax.googleapis.com/ajax/libs/angularjs/1.8.2/angular.min.js"></s
cript>
</head>
<body ng-app="myApp">
  <div ng-controller="myController">
    Default currency: {{person.salary | currency}} <br />
    Custom currency identifier: {{person.salary | currency:'Rs.'}} <br />
    No Fraction: {{person.salary | currency:'Rs.':0}} <br />
    Fraction 2: <span ng-bind="person.salary| currency:'GBP':2"></span>
  </div>
  <script>
    var myApp = angular.module('myApp', []);

    myApp.controller("myController", function ($scope) {
      $scope.person = { firstName: 'James', lastName: 'Bond', salary:
100000}
    });
  </script>
</body>
</html>
```

Output:

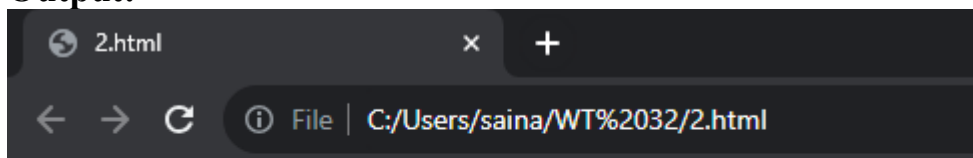


Default currency: \$100,000.00
Custom currency identifier: Rs.100,000.00
No Fraction: Rs.100,000
Fraction 2: GBP100,000.00

6.0 Write a program in AngularJs demonstrates Date filter.

```
<!DOCTYPE html>
<html >
<head>
  <script
src="https://ajax.googleapis.com/ajax/libs/angularjs/1.8.2/angular.min.js"></s
cript>
</head>
<body ng-app>
  <div ng-init="person.DOB = 323234234898">
    Default date: {{person.DOB| date}} <br />
    Short date: {{person.DOB| date:'short'}} <br />
    Long date: {{person.DOB | date:'longDate'}} <br />
    Year: {{person.DOB | date:'yyyy'}} <br />
  </div>
</body>
</html>
```

Output:

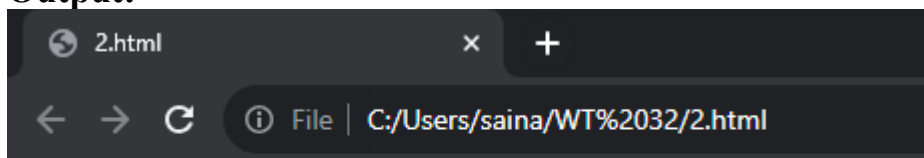


Default date: Mar 30, 1980
Short date: 3/30/80 8:47 AM
Long date: March 30, 1980
Year: 1980

6.1 Write a program in AngularJs upper case and lowercase filter

```
<!DOCTYPE html>
<html >
<head>
  <script
src="https://ajax.googleapis.com/ajax/libs/angularjs/1.8.2/angular.min.js"></s
cript>
</head>
<body ng-app>
  <div ng-init="person.firstName='Sainath';person.lastName='Machha'">
    Lower case: {{person.firstName + ' ' + person.lastName | lowercase}} <br
/>
    Upper case: {{person.firstName + ' ' + person.lastName | uppercase}}
  </div>
</body>
</html>
```

Output:



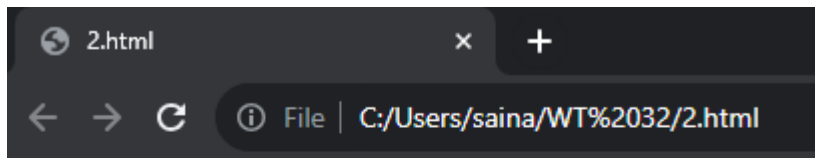
Lower case: sainath machha
Upper case: SAINATH MACHHA

6.2 Write a program in AngularJs to demonstrates mouse event

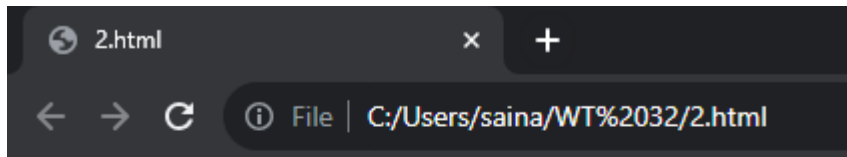
```
<!DOCTYPE html>
<html>
<head>
  <script
src="https://ajax.googleapis.com/ajax/libs/angularjs/1.8.2/angular.min.js"></s
cript>
  <style>
    .redDiv {
      width: 100px;
      height: 100px;
      background-color: red;
      padding:2px 2px 2px 2px;
    }

    .yellowDiv {
      width: 100px;
      height: 100px;
      background-color: yellow;
      padding:2px 2px 2px 2px;
    }
  </style>
</head>
<body ng-app>
  <div ng-class="{redDiv: enter, yellowDiv: leave}" ng-
mouseenter="enter=true;leave=false;" ng-
mouseleave="leave=true;enter=false">
    Mouse <span ng-show="enter">Enter</span> <span ng-
show="leave">Leave</span>
  </div>
</body>
</html>
```

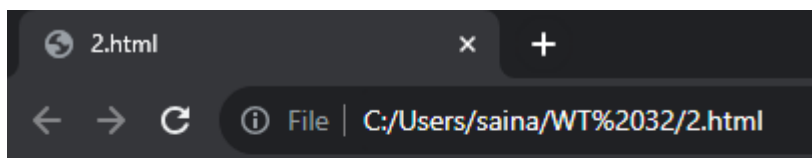
Output:



Mouse



Mouse Enter



Mouse Leave