ATMA RAM SANATAN DHARMA COLLAGE

UNIVERSITY OF DELHI

INTERNET TECHNOLOGY

Name - Suraj Rai

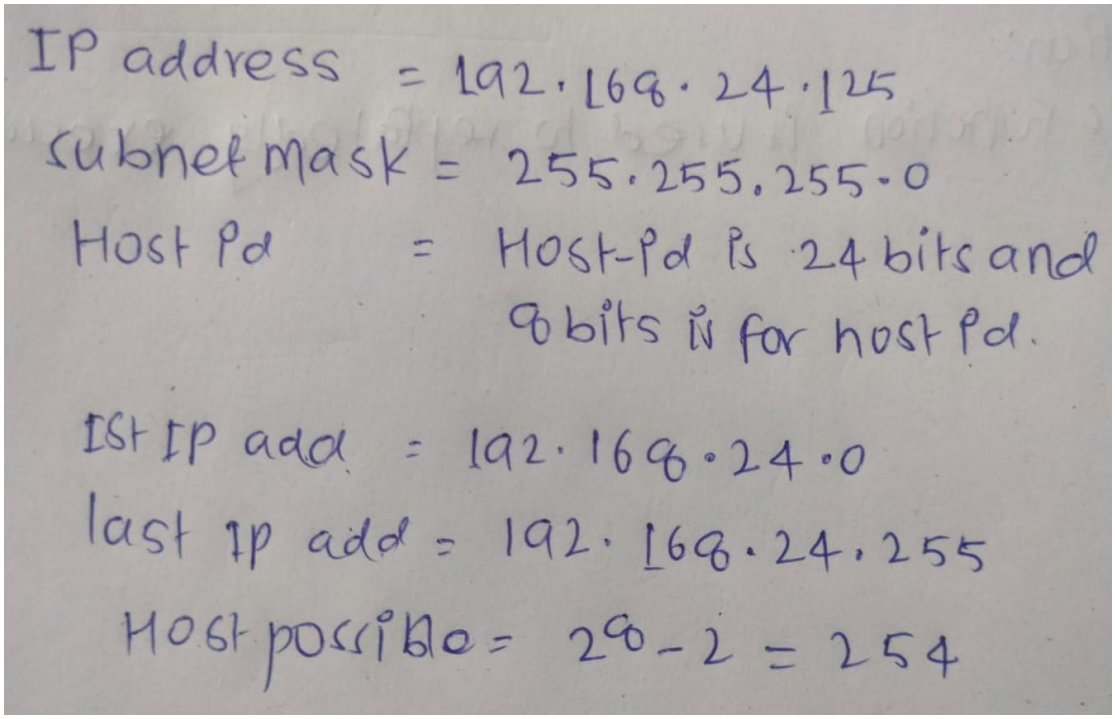
Roll no.: - 18102

Course- BSc(H) Computer Science

Submitted to - Mrs Oma Ojha

Q1. Display your systems IP Address, Subnet mask using ipconfig, and find out the network
address and the maximum number of systems possible on your network and range of IP addresses available to these systems.

```
C:\Users\Gautam>ipconfig

Windows IP Configuration


Ethernet adapter Ethernet:

   Media State . . . . . . . . . . . : Media disconnected
   Connection-specific DNS Suffix  . :

Wireless LAN adapter Local Area Connection* 1:

   Media State . . . . . . . . . . . : Media disconnected
   Connection-specific DNS Suffix  . :

Wireless LAN adapter Local Area Connection* 2:

   Media State . . . . . . . . . . . : Media disconnected
   Connection-specific DNS Suffix  . :

Wireless LAN adapter WiFi:

   Connection-specific DNS Suffix  . :
   IPv6 Address. . . . . . . . . . . : 2409:40d6:1001:ae55:1322:cf4b:5d7e:eb2f
   Temporary IPv6 Address. . . . . . : 2409:40d6:1001:ae55:8811:fd22:9985:776e
   Link-local IPv6 Address . . . . . : fe80::b1c:b9ad:ec76:4a6c%16
   IPv4 Address. . . . . . . . . . . : 192.168.24.125
   Subnet Mask . . . . . . . . . . . : 255.255.255.0
   Default Gateway . . . . . . . . . : fe80::2ca8:6aff:fe0b:20d9%16
                                       192.168.24.245
```



Q2. With help of ping, check if you are connected to other systems of your network and find
the route to connect to that system using tracert. List all the processes which are using ports
for TCP protocol

```
C:\Users\Gautam>ping 192.168.24.96

Pinging 192.168.24.96 with 32 bytes of data:
Reply from 192.168.24.96: bytes=32 time=210ms TTL=64
Reply from 192.168.24.96: bytes=32 time=83ms TTL=64
Reply from 192.168.24.96: bytes=32 time=74ms TTL=64
Reply from 192.168.24.96: bytes=32 time=68ms TTL=64

Ping statistics for 192.168.24.96:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 68ms, Maximum = 210ms, Average = 108ms

C:\Users\Gautam>tracert 192.168.24.96

Tracing route to 192.168.24.96 over a maximum of 30 hops

  1    57 ms     5 ms     4 ms  192.168.24.96

Trace complete.
```

All the processes which are using ports for TCP protocol:-
C:\Users\Gautam>netstat -a
Active Connections
Proto Local Address Foreign Address State
TCP 0.0.0.0:80 DESKTOP-257TP51:0 LISTENING
TCP 0.0.0.0:135 DESKTOP-257TP51:0 LISTENING
TCP 0.0.0.0:443 DESKTOP-257TP51:0 LISTENING
TCP 0.0.0.0:445 DESKTOP-257TP51:0 LISTENING
TCP 0.0.0.0:3000 DESKTOP-257TP51:0 LISTENING
TCP 0.0.0.0:3306 DESKTOP-257TP51:0 LISTENING
TCP 0.0.0.0:5040 DESKTOP-257TP51:0 LISTENING
TCP 0.0.0.0:5432 DESKTOP-257TP51:0 LISTENING
TCP 0.0.0.0:5500 DESKTOP-257TP51:0 LISTENING
TCP 0.0.0.0:27121 DESKTOP-257TP51:0 LISTENING
TCP 0.0.0.0:49664 DESKTOP-257TP51:0 LISTENING
TCP 0.0.0.0:49665 DESKTOP-257TP51:0 LISTENING
TCP 0.0.0.0:49666 DESKTOP-257TP51:0 LISTENING
TCP 0.0.0.0:49667 DESKTOP-257TP51:0 LISTENING
TCP 0.0.0.0:49668 DESKTOP-257TP51:0 LISTENING
TCP 0.0.0.0:49669 DESKTOP-257TP51:0 LISTENING
TCP 127.0.0.1:3306 DESKTOP-257TP51:54668 ESTABLISHED
TCP 127.0.0.1:5500 DESKTOP-257TP51:55255 ESTABLISHED
TCP 127.0.0.1:54107 DESKTOP-257TP51:0 LISTENING
TCP 127.0.0.1:54121 DESKTOP-257TP51:0 LISTENING
TCP 127.0.0.1:54668 DESKTOP-257TP51:3306 ESTABLISHED

TCP 127.0.0.1:55255 DESKTOP-257TP51:5500 ESTABLISHED
TCP 192.168.24.125:139 DESKTOP-257TP51:0 LISTENING
TCP [::]:80 DESKTOP-257TP51:0 LISTENING
TCP [::]:135 DESKTOP-257TP51:0 LISTENING
TCP [::]:443 DESKTOP-257TP51:0 LISTENING TCP [::]:445 DESKTOP-257TP51:0 LISTENING
TCP [::]:3000 DESKTOP-257TP51:0 LISTENING
TCP [::]:3306 DESKTOP-257TP51:0 LISTENING
TCP [::]:5432 DESKTOP-257TP51:0 LISTENING
TCP [::]:27121 DESKTOP-257TP51:0 LISTENING
TCP [::]:49664 DESKTOP-257TP51:0 LISTENING
TCP [::]:49665 DESKTOP-257TP51:0 LISTENING
TCP [::]:49666 DESKTOP-257TP51:0 LISTENING
TCP [::]:49667 DESKTOP-257TP51:0 LISTENING
TCP [::]:49668 DESKTOP-257TP51:0 LISTENING
TCP [::]:49669 DESKTOP-257TP51:0 LISTENING
TCP [::1]:3000 DESKTOP-257TP51:55188 ESTABLISHED
TCP [::1]:55188 DESKTOP-257TP51:3000 ESTABLISHED
TCP [2409:40d6:1001:ae55:8811:fd22:9985:776e]:54193
[2606:4700:90c2:883b:1064:554:6810:e095]:https ESTABLISHED TCP
[2409:40d6:1001:ae55:8811:fd22:9985:776e]:54324 bingforbusiness:https ESTABLISHED
TCP [2409:40d6:1001:ae55:8811:fd22:9985:776e]:54788 g2600-140f-d800-02a6-0000-0000-
0000-
1011:https CLOSE_WAIT TCP
[2409:40d6:1001:ae55:8811:fd22:9985:776e]:54863 g2600-140f-0005-0690-0000-0000-
0000-
356e:https CLOSE_WAIT TCP
[2409:40d6:1001:ae55:8811:fd22:9985:776e]:54869 g2600-140f-0005-0000-0000-0000-
17d9-
6e93:https CLOSE_WAIT TCP
[2409:40d6:1001:ae55:8811:fd22:9985:776e]:55261 [64:ff9b::142a:491a]:https TIME_WAIT
TCP [2409:40d6:1001:ae55:8811:fd22:9985:776e]:55262 [64:ff9b::7512:e8c8]:https
TIME_WAIT
TCP [2409:40d6:1001:ae55:8811:fd22:9985:776e]:55268 whatsapp-cdn6-shv-02-del1:https
TIME_WAIT TCP
[2409:40d6:1001:ae55:8811:fd22:9985:776e]:55271
[2405:200:1630:ff91:face:b00c:3333:7020]:https TIME_WAIT TCP
[2409:40d6:1001:ae55:8811:fd22:9985:776e]:55272 whatsapp-cdn6-shv-01-del2:https
TIME_WAIT
TCP [2409:40d6:1001:ae55:8811:fd22:9985:776e]:55275 whatsapp-cdn6-shv-01-del2:https
ESTABLISHED TCP
[2409:40d6:1001:ae55:8811:fd22:9985:776e]:55276
[2405:200:1630:ff91:face:b00c:3333:7020]:https ESTABLISHED TCP
[2409:40d6:1001:ae55:8811:fd22:9985:776e]:55279 [64:ff9b::14c6:778f]:https ESTABLISHED
TCP [2409:40d6:1001:ae55:8811:fd22:9985:776e]:55280 si-in-f188:5228 ESTABLISHED
TCP [2409:40d6:1001:ae55:8811:fd22:9985:776e]:55281 si-in-f188:5228 ESTABLISHED
TCP [2409:40d6:1001:ae55:8811:fd22:9985:776e]:55283 del12s03-in-x01:https
ESTABLISHED
UDP 0.0.0.0:5050 *:*
UDP 0.0.0.0:5353 *:*
UDP 0.0.0.0:5353 *:*
UDP 0.0.0.0:5353 *:*

UDP 0.0.0.0:5355 *:*
UDP 0.0.0.0:50469 *:*
UDP 0.0.0.0:52236 *:*
UDP 0.0.0.0:53470 *:*
UDP 0.0.0.0:54084 *:*
UDP 0.0.0.0:56310 *:*
UDP 0.0.0.0:60746 *:* UDP 0.0.0.0:62040 *:*
UDP 0.0.0.0:63541 *:*
UDP 0.0.0.0:63957 *:*
UDP 0.0.0.0:64444 *:*
UDP 0.0.0.0:65068 *:*
UDP 127.0.0.1:1900 *:*
UDP 127.0.0.1:49664 *:*
UDP 127.0.0.1:51454 *:*
UDP 192.168.24.125:137 *:*
UDP 192.168.24.125:138 *:*
UDP 192.168.24.125:1900 *:*
UDP 192.168.24.125:51453 *:*
UDP [::]:5353 *:*
UDP [::]:5353 *:*
UDP [::]:5355 *:*
UDP [::]:50469 *:*
UDP [::]:52236 *:*
UDP [::]:53470 *:*
UDP [::]:54084 *:*
UDP [::]:56310 *:*
UDP [::]:60746 *:*
UDP [::]:62040 *:*
UDP [::]:63541 *:*
UDP [::]:63957 *:*
UDP [::]:64444 *:*
UDP [::]:65068 *:*
UDP [::1]:1900 *:*
UDP [::1]:51452 *:*
UDP [fe80::b1c:b9ad:ec76:4a6c%16]:546 *:*
UDP [fe80::b1c:b9ad:ec76:4a6c%16]:1900 *:*
UDP [fe80::b1c:b9ad:ec76:4a6c%16]:51451 *:*

Q3. Create an HTML page that shows information about you, your course, hobbies, address,
and your plans. Use CSS for styling of HTML page so that looks nice.

Css: -

```
*{
    font-family: system-ui, -apple-system, BlinkMacSystemFont, 'Segoe UI',
    Roboto, Oxygen, Ubuntu, Cantarell, 'Open Sans', 'Helvetica Neue', sansserif;
    }
    body
    {
    background: linear-gradient(60deg,rgb(127, 120, 226), rgb(233, 57, 63));
    }
    #heading
    {
```

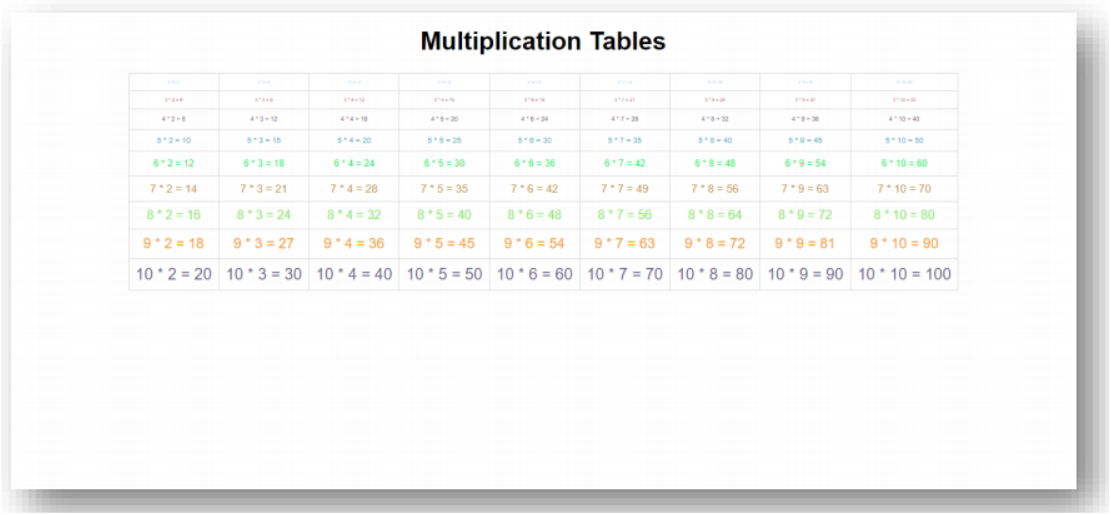
```
    text-align: center; }
    #info
    {
    border: 2px solid black;
    border-radius: 10px;
    padding-left: 25px;
    transition: all 3ms ease-in-out;
    margin-left: 35vw;
    margin-right: 35vw; }
    #info:hover
    {
    color: rgb(255, 115, 0);
    box-shadow: 10px 5px 5px rgb(128, 128, 128); }
```

Html:-

```
<!DOCTYPE html>
 <html lang="en">
     <head><meta charset="UTF-8">
         <meta name="viewport" content="width=device-width, initial-scale=1.0">
         <title>Practical 3</title> <link rel="stylesheet" href="deo.css">
</head>
 <body>
    <h1 id="heading">SURAJ RAI </h1>
    <div id="info">
         <h2 id="heading2">Bsc.(H) Computer Science</h2>
         <h3 class="heading3">Hobbies</h3>
         <ol>
            <li>CODDING</li>
             <li>PLAYING BADMINTON</li>
             <li>FIGHTING</li>
</ol>
<h3 class="heading3">Address</h3>
 <p>VARANASI UP</p>
  <h3 class="heading3">Plans</h3> <p>TM.TECH</p>
</div>
</body>
</html>
```

Q4. Create an HTML page with the sole purpose to show multiplication tables of 2 to 10 (row-wise) created by JavaScript. Initially, the page is blank. With help of setInterval function
print a row every 5 seconds in different colors and increasing font size. Use clearInterval() function to stop the given task.

```
Css & HTML: -
<!DOCTYPE html> <html lang="en"> <head><meta charset="UTF-8"> <meta name="viewport"
content="width=device-width, initial-scale=1.0"> <title>Practical 4</title> <style>
body {
font-family: Arial, sans-serif;
text-align: center;
margin: 20px; }
table {
margin: 20px auto;
border-collapse: collapse;
width: 80%; }
th, td {
border: 1px solid #dddddd;
padding: 8px;
text-align: center; }
</style>
</head> <body><h1>Multiplication Tables</h1> <table id="multiplicationTable">
</table> <script>
let currentRow = 2;
let intervalId;
function printRow() {
const table = document.getElementById('multiplicationTable');
const newRow = table.insertRow();
for (let i = 2; i <= 10; i++) {
const cell = newRow.insertCell();
const result = currentRow * i;
cell.textContent = `${currentRow} * ${i} = ${result}`; }
newRow.style.fontSize = `${currentRow * 2}px`;
newRow.style.color = getRandomColor();
currentRow++;
if (currentRow > 10) {
clearInterval(intervalId);
} }
function getRandomColor() {
const letters = '0123456789ABCDEF';
let color = '#';
for (let i = 0; i < 6; i++) {
color += letters[Math.floor(Math.random() * 16)];
}
return color; }
intervalId = setInterval(printRow, 5000);
</script>
</body>
</html>
```



Q5. Explain setInterval function and setTimeout function with the help of an example

SetInterval() Method

The **setInterval** Javascript method is used to call a function repeatedly at a specified interval of time. This time interval at which the function will be called is provided by the user in milliseconds

SetTimeout() Method

The **setTimeout** Javascript method is used to call a function after a certain period of time. The time after which the function will be called is given by the user in milliseconds.

```
<!DOCTYPE html>

<html>

<head>

   <title>setInterval() method</title>

</head>

<body>

 <button onclick="setInterval(demo, 5000);"> Click here</button>

<script>

      function demo() {alert('Here is an example of setInterval Javascript':  }   </script>

</body>

</html>
```

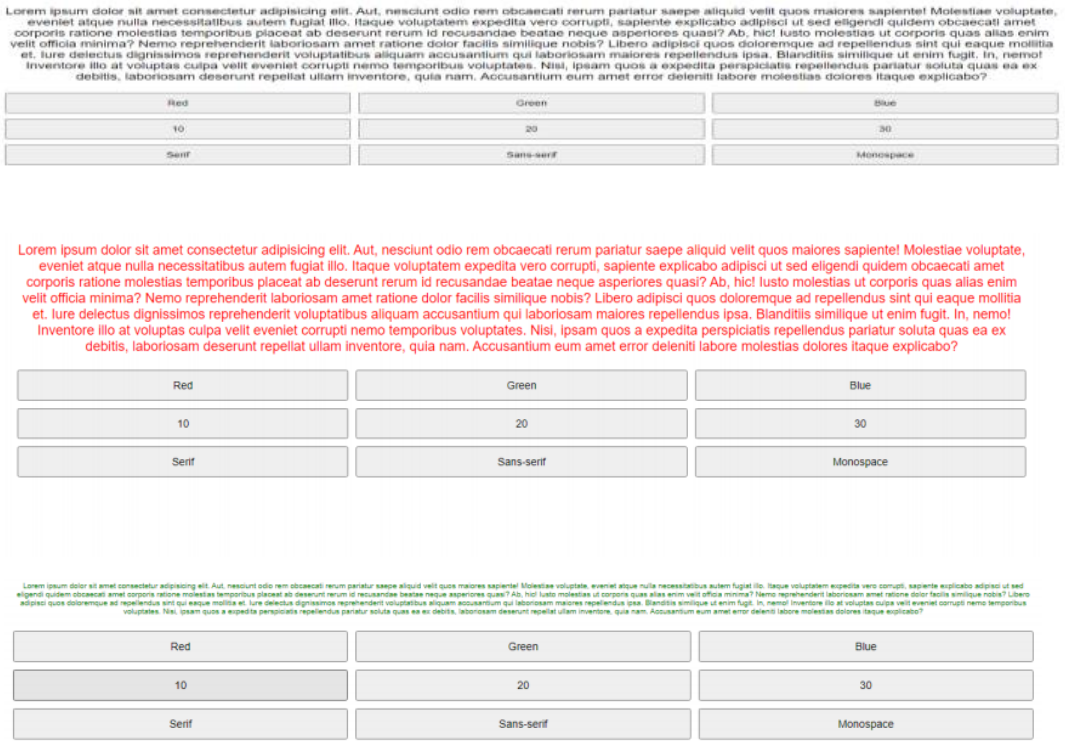
Q6. Create an HTML page with a paragraph written on it and under which 9 buttons are placed in a 3X3 grid. The first row is for buttons labeled with colors names Red, Green, and
Blue, the second row with numbers 10, 20, 30, and the third row with different font names. Click event of each of the buttons should make the appropriate change in the style of paragraph.

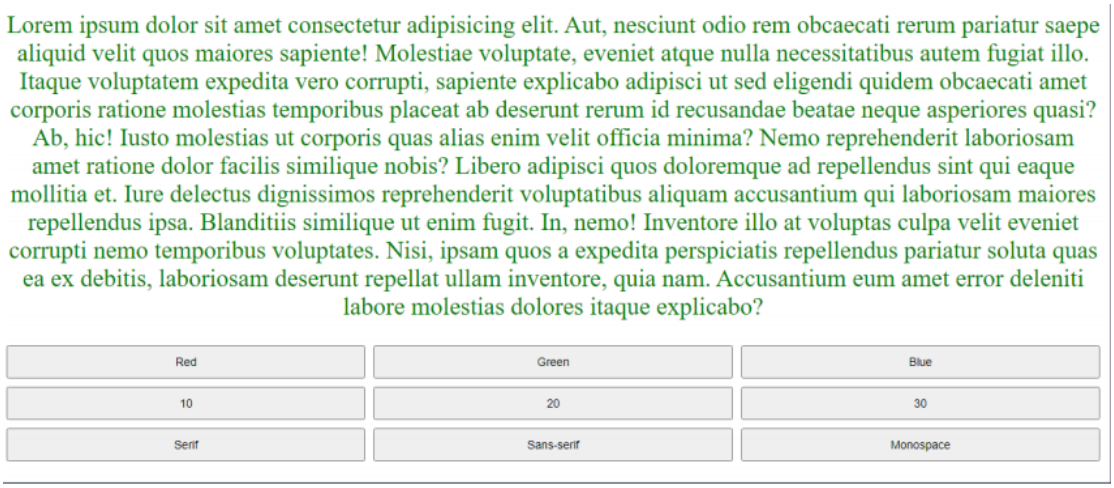
HTML & CSS: -

```
<!DOCTYPE html> <html lang="en"> <head><meta charset="UTF-8"> <meta name="viewport"
content="width=device-width, initial-scale=1.0"> <title>Practical 6</title> <style>
    body {
    font-family: Arial, sans-serif;
    text-align: center;
    margin: 20px; }p {
    font-size: 18px; }
    .button-grid {
    display: grid;
    grid-template-columns: repeat(3, 1fr);
    gap: 10px;
    margin-top: 20px; }button {
    padding: 10px;
    font-size: 14px;
    cursor: pointer; }
    </style>
    </head> <body><p id="paragraph">Lorem ipsum dolor sit amet consectetur adipisicing elit.
    Aut, nesciunt odio rem obcaecati rerum pariatur saepe aliquid velit quos
    maiores sapiente! Molestiae voluptate, eveniet atque nulla necessitatibus
    autem fugiat illo.
    Itaque voluptatem expedita vero corrupti, sapiente explicabo adipisci ut
    sed eligendi quidem obcaecati amet corporis ratione molestias temporibus
    placeat ab deserunt rerum id recusandae beatae neque asperiores quasi? Ab,
    hic!
    Iusto molestias ut corporis quas alias enim velit officia minima? Nemo
    reprehenderit laboriosam amet ratione dolor facilis similique nobis? Libero
    adipisci quos doloremque ad repellendus sint qui eaque mollitia et.
    Iure delectus dignissimos reprehenderit voluptatibus aliquam accusantium
    qui laboriosam maiores repellendus ipsa. Blanditiis similique ut enim fugit.
    In, nemo! Inventore illo at voluptas culpa velit eveniet corrupti nemo
    temporibus voluptates.
    Nisi, ipsam quos a expedita perspiciatis repellendus pariatur soluta quas
    ea ex debitis, laboriosam deserunt repellat ullam inventore, quia nam.
    Accusantium eum amet error deleniti labore molestias dolores itaque
    explicabo?</p> <div class="button-grid"> <button onclick="changeColor('red')">Red</button>
<button onclick="changeColor('green')">Green</button> <button
onclick="changeColor('blue')">Blue</button> <button onclick="changeFontSize(10)">10</button>
<button onclick="changeFontSize(20)">20</button> <button
onclick="changeFontSize(30)">30</button> <button
onclick="changeFontFamily('serif')">Serif</button> <button onclick="changeFontFamily('sans-
serif')">Sans-serif</button> <button onclick="changeFontFamily('monospace')">Monospace</button>
    </div> <script>
    function changeColor(color) {
    document.getElementById('paragraph').style.color = color; }
    function changeFontSize(size) {
    document.getElementById('paragraph').style.fontSize = size + 'px'; }
    function changeFontFamily(font) {
    document.getElementById('paragraph').style.fontFamily = font; }
    </script>
    </body>
```

```
    </html>
```

Output:_



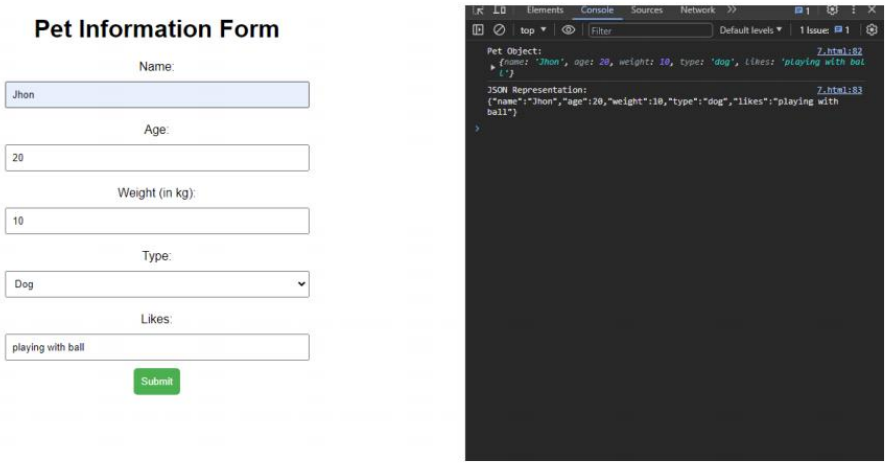


O7. Create a form that takes data about a pet. The form must be well designed and should
accept the pet's name, age, weight, type, and what it likes most. At the submission of this form create a Pet object in JavaScript filled with these values and log that object and equivalent JSON on the console.

```
<!DOCTYPE html> <html lang="en"> <head><meta charset="UTF-8"> <meta name="viewport"
content="width=device-width, initial-scale=1.0"> <title>Pet Information Form</title> <style>
    body {
    font-family: Arial, sans-serif;
    text-align: center;
    margin: 20px; }
    form {
    max-width: 400px;
    margin: 0 auto; }
```

```
    label {
    display: block;
    margin: 10px 0; }
    input, select {
    width: 100%;
    padding: 8px;
    margin-bottom: 10px;
    box-sizing: border-box; }
    button {
    padding: 10px;
    background-color: #4CAF50;
    color: white;
    border: none;
    border-radius: 5px;
    cursor: pointer; }
    </style>
    </head> <body><h1>Pet Information Form</h1> <form id="petForm"> <label
for="name">Name:</label> <input type="text" id="name" name="name" required> <label
for="age">Age:</label> <input type="number" id="age" name="age" required> <label
for="weight">Weight (in kg):</label> <input type="number" id="weight" name="weight" required>
<label for="type">Type:</label> <select id="type" name="type" required> <option
value="dog">Dog</option> <option value="cat">Cat</option> <option value="bird">Bird</option>
    </select> <label for="likes">Likes:</label> <input type="text" id="likes" name="likes"
required> <button type="button" onclick="submitForm()">Submit</button>
    </form> <script>
    function submitForm() {
    const name = document.getElementById('name').value;
    const age = parseInt(document.getElementById('age').value);
    const weight =
    parseFloat(document.getElementById('weight').value);
    const type = document.getElementById('type').value;
    const likes = document.getElementById('likes').value;
    const petObject = {
    name: name,
    age: age,
    weight: weight,
    type: type,
    likes: likes
    };
    console.log("Pet Object:", petObject);
    console.log("JSON Representation:", JSON.stringify(petObject));
    }
    </script>
    </body>
    </html>
```

Output: -



Q8. Store JSON data of few pets that you created in previous practical in a JSON file (copy
from console output of previous program to a .json file). Using AJAX, load data from the file
and display it in a presentable way using HTML and CSS.

```
JSON file: - [ {
    "name": "Buddy",
    "age": 3,
    "weight": 10,
    "type": "dog",
    "likes": "playing fetch"
    },
    {
    "name": "Whiskers",
    "age": 2,
    "weight": 5,
    "type": "cat",
    "likes": "sunbathing"
    },
    {
    "name": "Tweety",
    "age": 1,
    "weight": 0.1,
    "type": "bird",
    "likes": "singing"
    } ]
    HTML & CSS: -
    <!DOCTYPE html> <html lang="en"> <head><meta charset="UTF-8"> <meta name="viewport"
content="width=device-width, initial-scale=1.0"> <title>Pet Information</title> <style>
    body {
    font-family: Arial, sans-serif;
    text-align: center;
    margin: 20px; }
    #petContainer {
    display: flex;
    justify-content: space-around;
    flex-wrap: wrap;
    margin-top: 20px; }
    .petCard {
    border: 1px solid #ddd;
    border-radius: 8px;
```

```
padding: 10px;
margin: 10px;
width: 200px;
box-shadow: 0 4px 8px rgba(0, 0, 0, 0.1);
}
.petCard h3 {
color: #333; }
.petCard p {
margin: 5px 0; }
</style>
</head> <body><h1>Pet Information</h1> <div id="petContainer"></div> <script>
const xhr = new XMLHttpRequest();
xhr.onreadystatechange = function() {
if (this.readyState === 4 && this.status === 200) {
const pets = JSON.parse(this.responseText);
displayPets(pets);
}
};
xhr.open("GET", "8.json", true);
xhr.send();
function displayPets(pets) {
const petContainer = document.getElementById('petContainer');
pets.forEach(pet => {
const petCard = document.createElement('div');
petCard.classList.add('petCard');
const petName = document.createElement('h3');
petName.textContent = pet.name;
const petDetails = document.createElement('p');
petDetails.innerHTML = `Age: ${pet.age} years<br>Weight:
${pet.weight} kg<br>Type: ${pet.type}<br>Likes: ${pet.likes}`;
petCard.appendChild(petName);
petCard.appendChild(petDetails);
petContainer.appendChild(petCard);
});
}
</script>
</body>
</html>
```

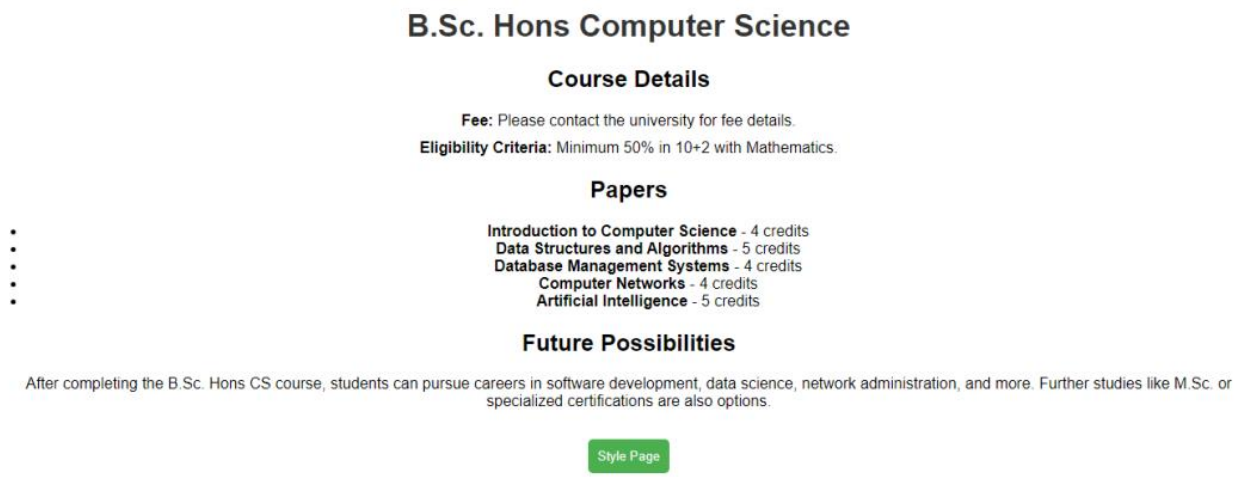
Output: -

Q9. Create a plain HTML page for B.Sc. Hons CS course, mentioning details like fee, eligibility criteria, papers with names and credits, and future possibilities after the course. A
button for styling should be there at bottom of the page. On clicking on this button JavaScript
should redesign the complete page using jQuery in a nice presentable way.

```
<!DOCTYPE html> <html lang="en"> <head><meta charset="UTF-8"> <meta name="viewport"
content="width=device-width, initial-scale=1.0"> <title>B.Sc. Hons CS Course</title> <style>
    body {
    font-family: Arial, sans-serif;
    text-align: center;
    margin: 20px; }
    h1 {
    color: #333;
    }p {
    margin: 10px 0; }
    button {
    padding: 10px;
    background-color: #4CAF50;
    color: white;
    border: none;
    border-radius: 5px;
    cursor: pointer;
    margin-top: 20px; }
    </style>
    </head> <body><h1>B.Sc. Hons Computer Science</h1> <h2>Course Details</h2>
<p><strong>Fee:</strong> Please contact the university for fee
    details.</p> <p><strong>Eligibility Criteria:</strong> Minimum 50% in 10+2 with
    Mathematics.</p> <h2>Papers</h2> <ul><li><strong>Introduction to Computer Science</strong>
- 4 credits</li> <li><strong>Data Structures and Algorithms</strong> - 5 credits</li>
<li><strong>Database Management Systems</strong> - 4 credits</li> <li><strong>Computer
Networks</strong> - 4 credits</li> <li><strong>Artificial Intelligence</strong> - 5
credits</li>
    </ul> <h2>Future Possibilities</h2> <p>After completing the B.Sc. Hons CS course, students
can pursue careers
    in software development, data science, network administration, and more.
    Further studies like M.Sc. or specialized certifications are also options.</p> <button
id="styleButton">Style Page</button> <script src="https://code.jquery.com/jquery-
3.6.4.min.js"></script> <script> $(document).ready(function() {
    $('#styleButton').click(function() {
    $('body').css({
    'font-family': 'Verdana, sans-serif',
    'background-color': '#f4f4f4',
    'padding': '20px'
    });
    $('h1, h2').css({
```

```
'color': '#333',
'border-bottom': '2px solid #333',
'padding-bottom': '10px'
});
$('p').css({
'margin': '10px 0',
'line-height': '1.6'
});
$('button').css({
'background-color': '#008CBA',
'color': 'white'
});
});
});
</script>
</body>
</html>
```

Output: -



After click the button

B.Sc. Hons Computer Science

Course Details

Fee: Please contact the university for fee details.

Eligibility Criteria: Minimum 50% in 10+2 with Mathematics.

Papers

- **Introduction to Computer Science** - 4 credits
- **Data Structures and Algorithms** - 5 credits
- **Database Management Systems** - 4 credits
- **Computer Networks** - 4 credits
- **Artificial Intelligence** - 5 credits

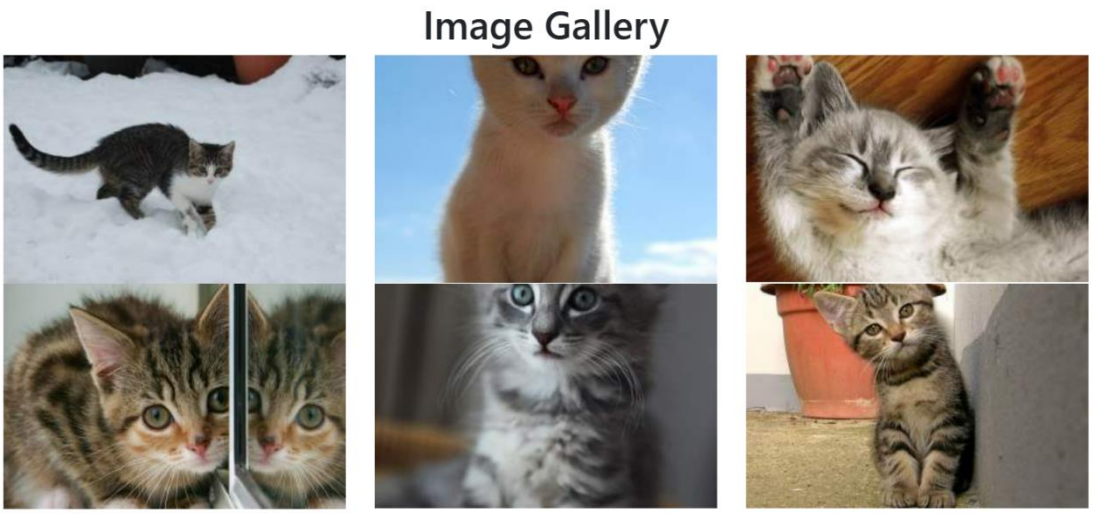
Future Possibilities

After completing the B.Sc. Hons CS course, students can pursue careers in software development, data science, network administration, and more. Further studies like M.Sc. or specialized certifications are also options.

Style Page

Q10. Create an HTML page for an image gallery, which shows the use of BOOTSTRAP to
rearrange and resize its contents on resizing the browser.

```
<!DOCTYPE html> <html lang="en">
     <head>
        <meta charset="UTF-8">
         <meta name="viewport" content="width=device-width, initial-scale=1,
    shrink-to-fit=no"> <title>Bootstrap Image Gallery</title> <link rel="stylesheet"
    href="https://stackpath.bootstrapcdn.com/bootstrap/4.5.2/css/bootstrap.min.css
    "> <style>
    body {
    padding: 20px; }
    .gallery img {
    width: 100%;
    height: auto; }
    </style>
    </head> <body><div class="container"> <h1 class="text-center">Image Gallery</h1> <div
class="row gallery"> <div class="col-md-4"> <img src="https://placekitten.com/300/200"
alt="Image 1">
    </div> <div class="col-md-4">
    <img src="https://placekitten.com/301/200" alt="Image 2">
    </div> <div class="col-md-4"> <img src="https://placekitten.com/302/200" alt="Image 3">
    </div> <div class="col-md-4"> <img src="https://placekitten.com/303/200" alt="Image 4">
    </div> <div class="col-md-4"> <img src="https://placekitten.com/304/200" alt="Image 5">
    </div> <div class="col-md-4"> <img src="https://placekitten.com/305/200" alt="Image 6">
    </div>
    </div>
    </div> <script src="https://code.jquery.com/jquery-3.5.1.slim.min.js"></script> <script
    src="https://cdn.jsdelivr.net/npm/@popperjs/core@2.5.2/dist/umd/popper.min.js"
    ></script> <script
    src="https://stackpath.bootstrapcdn.com/bootstrap/4.5.2/js/bootstrap.min.js">

    </script>
    </body>
    </html>
```



Q11. Create an HTTP server using Node.js, which handles requests on port 10000, or a free
port beyond 10000. Modify the server in such a way that opening localhost:10000 will display
“Hello world, This is my Node.js server” on browser.

```
const http = require('http');
const server = http.createServer((Request, Response) =>{
Response.writeHead(200, {"Content-type": 'text/plain'});
Response.end('Hello world, This is my Node.js server\n');
});
const port = 10000;
server.listen(port, ()=>{
console.log(`Server is runnint at ${port}`);
});
```

Output: -

```
PS D:\College\Practical file\IT> node 11.js
Server is runnint at 10000
```

Q12. Create index.html file containing two forms for SignIn and SignUp. Submitting SignIn
form should search for credentials in mysql database using server created in previous practical. On successful signin, a welcome page should be displayed. Submitting SignUp form
should insert new entry for credentials in mysql database using server created in previous
practical. On successful signup, user should be returned back to index.html (use Node.js for
database connectivity with mysql)

Index.html

```
<!DOCTYPE html>
<html lang="en">
    <head>
        <meta charset="UTF-8">
         <meta name="viewport" content="width=device-width, initial-scale=1.0">
          <title>SignIn / SignUp

          </title>
</head>
 <body>
    <h1>SignIn</h1>
<form action="/signin" method="post">
    <label for="username">Username:</label>
     <input type="text" id="username" name="username" required>
```

```
      <br> <label for="password">Password:</label>
       <input type="password" id="password" name="password" required>
        <br> <button type="submit">SignIn</button>
</form> <h1>SignUp</h1> <form action="/signup" method="post">
     <label for="newUsername">Username:</label>
     <input type="text" id="newUsername" name="username" required>
      <br>
       <label for="newPassword">Password:</label>
        <input type="password" id="newPassword" name="password" required>
         <br> <button type="submit">SignUp</button>
</form>
</body>
</html>
```

Welcom.html: -

```
<!DOCTYPE html>
<html lang="en">
<head>
<meta charset="UTF-8">
<meta name="viewport" content="width=device-width, initial-scale=1.0">
<title>Welcome</title>
</head>
<body>
<h1>Welcome to the System!</h1>
</body>
</html>
```

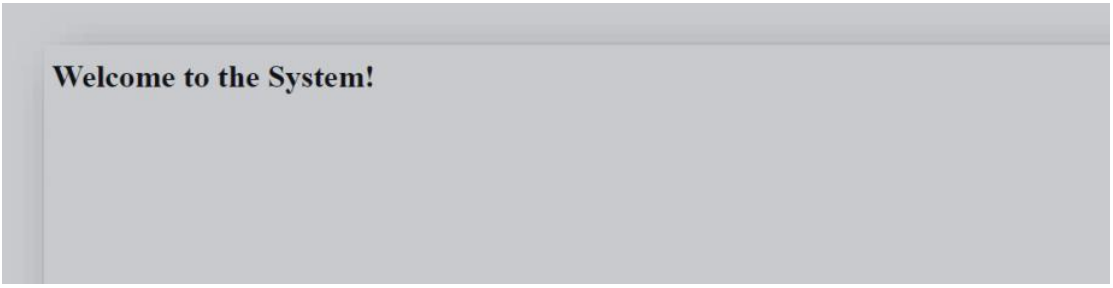
Node.js

```
const http = require('http');
const fs = require('fs');
const url = require('url');
const mysql = require('mysql2');
const path = require('node:path/win32');
const PORT = 3000;
const db = mysql.createConnection({
host: '127.0.0.1',
user: 'root',
password: '',
database: 'IT_Project',
});
db.connect((err) => {
if (err) {
console.error('Error connecting to MySQL:', err);
return; }
console.log('Connected to MySQL');
});
const server = http.createServer((req, res)=>{
const parsedUrl = url.parse(req.url, true);
const pathname = parsedUrl.pathname;
if (req.method === 'GET') {
if(pathname === '/') {
fs.readFile('index.html', (err, data)=>{
if(err) console.log(err);
else{
res.writeHead(200, {'Content-type' : 'text/html'});
res.end(data);
}
})
} }
else if (req.method === 'POST') {
```

```
if (pathname === '/signin' || pathname === '/signup') {
let body = '';
req.on('data', (chunk) => {
body += chunk.toString();
})
req.on('end', () => {
const formData = new URLSearchParams(body.toString());
const username = formData.username;
const password = formData.password;
if (pathname === '/signin') {
const query = 'SELECT * FROM users WHERE USERNAME = ? AND
PASSWORD = ?';
db.query(query, [username, password], (err, result)=>{
if (err) console.log(err);
else {
if (result > 0) {
fs.readFile('welcome.html', (err, data) => {
if(err) console.log(err);
else {
res.writeHead(200, {'content-type' :
'text/html'});
res.end(data);
}
})
} }
})
}
else if (pathname === '/signup') {
const query = 'INSERT INTO users (USERNAME, PASSWORD)
VALUES (?, ?)';
db.query(query, [username, password], (err) => {
if (err) console.log(err);
else {
fs.readFile('welcome.html', (err, data) => {
if (err) console.log(err);
else {
res.writeHead(200, {'content-type' :
'text/html'});
res.end(data);
}
})
}
})
}
else {
confirm.log('error');
}
})
} }
});
server.listen(PORT, () => {
console.log(`Server is running at http://localhost:${PORT}/`);
});
```

Output

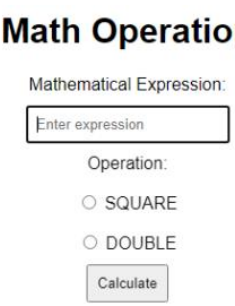
```
PS D:\College\Practical file\IT> node server.js
Server is running at http://localhost:3000/
Connected to MySQL
```

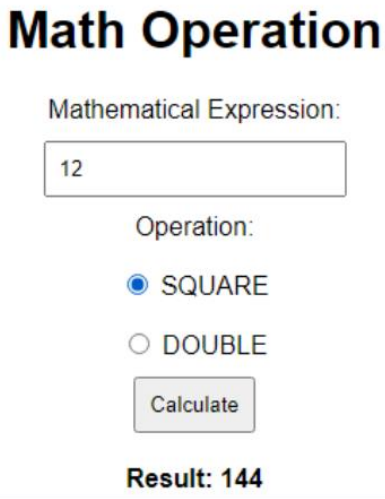


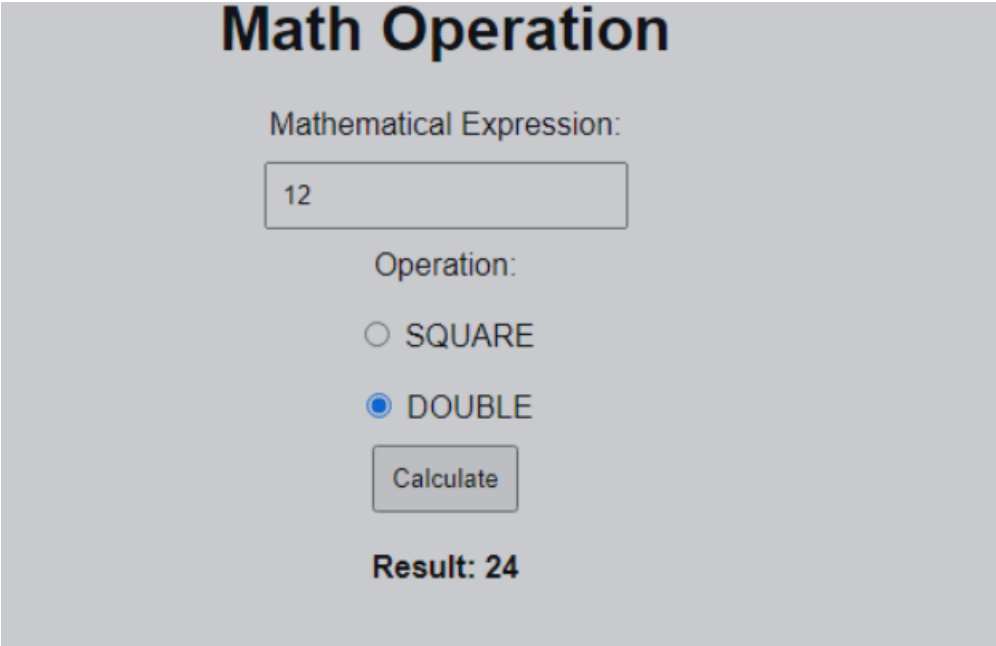
Q13 (a). Create an HTML page with one input field, one radio button and a text field for display. The first input field will take a mathematical expression as input. The two radio buttons will be displayed as SQUARE and DOUBLE. The user selects whichever option, the
result of the mathematical expression as entered by the user, will be squared or doubled and
the corresponding answer should be displayed in the text field.

```
<!DOCTYPE html> <html lang="en"> <head><meta charset="UTF-8"> <meta name="viewport"
content="width=device-width, initial-scale=1.0"> <title>Math Operation</title>
<style>
body {
font-family: Arial, sans-serif;
text-align: center;
margin: 20px; }
label {
display: block;
margin-top: 10px; }
input, select, button {
margin-top: 10px;
padding: 8px;
box-sizing: border-box; }
#result {
margin-top: 20px;
```

```
font-weight: bold; }
</style>
</head> <body><h1>Math Operation</h1> <label for="expression">Mathematical Expression:</label>
<input type="text" id="expression" placeholder="Enter expression"
required> <label>Operation:</label> <label><input type="radio" name="operation" value="square">
SQUARE</label> <label><input type="radio" name="operation" value="double"> DOUBLE</label>
<button onclick="performOperation()">Calculate</button> <div id="result"></div> <script>
function performOperation() {
const expression = document.getElementById('expression').value;
const operation =
document.querySelector('input[name="operation"]:checked');
let result;
if (operation) {
if (operation.value === 'square') {
result = Math.pow(eval(expression), 2);
} else if (operation.value === 'double') {
result = eval(expression) * 2; }
document.getElementById('result').innerText = `Result:
${result}`;
} else {
alert('Please select an operation.');
} }
</script>
</body>
</html>
```







Q13 (b) Create a form that takes data from a customer. The form must be well designed and
should accept the customer's FirstName, LastName, Age, Birthday and FoodPreferences. At
the submission of this form, create a Customer object in JavaScript using the above values
and an equivalent JSON object. Print both these objects on the console. Using AJAX, displays
the data of two customers in a presentable way

```
<!DOCTYPE html> <html lang="en"> <head><meta charset="UTF-8"> <meta name="viewport"
content="width=device-width, initial-scale=1.0"> <title>Customer Information Form</title>
<style>
body {
font-family: Arial, sans-serif;
text-align: center;
margin: 20px; }
form {
max-width: 400px;
margin: 0 auto; }
label {
display: block;
margin: 10px 0; }
input, select {
width: 100%;
padding: 8px;
margin-bottom: 10px;
box-sizing: border-box;
}
button {
padding: 10px;
background-color: #4CAF50;
color: white;
border: none;
border-radius: 5px;
cursor: pointer; }
#customerData {
margin-top: 20px; }
```

```
#customerData div {
border: 1px solid #ddd;
border-radius: 8px;
padding: 10px;
margin: 10px 0;
text-align: left; }
</style>
</head> <body><h1>Customer Information Form</h1> <form id="customerForm"> <label
for="firstName">First Name:</label> <input type="text" id="firstName" name="firstName"
required> <label for="lastName">Last Name:</label> <input type="text" id="lastName"
name="lastName" required> <label for="age">Age:</label> <input type="number" id="age"
name="age" required> <label for="birthday">Birthday:</label> <input type="date" id="birthday"
name="birthday" required> <label for="foodPreferences">Food Preferences:</label> <select
id="foodPreferences" name="foodPreferences" required> <option
value="vegetarian">Vegetarian</option> <option value="non-vegetarian">Non-Vegetarian</option>
</select> <button type="button" onclick="submitForm()">Submit</button>
</form>
<div id="customerData"></div> <script>
function submitForm() {
const firstName = document.getElementById('firstName').value;
const lastName = document.getElementById('lastName').value;
const age = parseInt(document.getElementById('age').value);
const birthday = document.getElementById('birthday').value;
const foodPreferences =
document.getElementById('foodPreferences').value;
const customerObject = {
firstName: firstName,
lastName: lastName,
age: age,
birthday: birthday,
foodPreferences: foodPreferences
};
console.log("Customer Object:", customerObject);
console.log("JSON Representation:",
JSON.stringify(customerObject));
displayCustomerData(customerObject);
}
function displayCustomerData(customer) {
const customerDataDiv = document.getElementById('customerData');
const customerDiv = document.createElement('div');
customerDiv.innerHTML = `
<strong>Customer Information:</strong><br>
<strong>First Name:</strong> ${customer.firstName}<br>
<strong>Last Name:</strong> ${customer.lastName}<br>
<strong>Age:</strong> ${customer.age}<br>
<strong>Birthday:</strong> ${customer.birthday}<br>
<strong>Food Preferences:</strong> ${customer.foodPreferences} `;
customerDataDiv.appendChild(customerDiv);
}
</script>
</body>
</html>
```

Customer Information Form

First Name:

suraj

Last Name:

rai

Age:

22

Birthday:

25-07-2002

Food Preferences:

Non-Vegetarian

Submit

Customer Information Form

First Name:

suraj

Last Name:

rai

Age:

22

Birthday:

25-07-2002

Food Preferences:

Non-Vegetarian

Submit

Customer Information:
First Name: suraj
Last Name: rai
Age: 22
Birthday: 2002-07-25
Food Preferences: non-vegetarian