

Web Programming Lab

(AI & DS 259)

Submitted To:

Dr. Neha Singh

Assistant Professor

Department of Information Technology

Submitted By:

Student Name: Saksham kochhar

Roll no: 01114811921

Semester: 3

Group: AI DS 1



Maharaja Agrasen Institute of Technology, PSP Area, Sector – 22,
Rohini, New Delhi – 110086

WEB PROGRAMMING LAB

PRACTICAL RECORD

PAPER CODE : AIDS-259
Name of the student : Saksham kochhar
University roll number : 01114811921
Branch : Artificial Intelligence and Data Science
Section/Group : AIDS 1

PRACTICAL DETAILS

Exp. No	Experiment Name	Date of Performance	Date of Checking	Remarks	Mark
1	Create a web page that covers your CV using various HTML Tags (UL, OL, Table, etc).	11/10/22	18/10/22		
2	Create a webpage that displays brief details of various Programming Languages using various types of CSS.	18/10/22	6/12/22		
3	Create a webpage using JavaScript and HTML to demonstrate Simple Calculator Application.	1/11/22	6/12/22		
4	Create a web page covering the basic CRUD operations (Create, Read, Update, Delete) that implements To-do/Grocery lists using JavaScript and HTML	22/11/22	13/12/22		
5	(a) WAP to show the usage of external javascript file. (b) WAP to show the usage of comments for single line and multiline. (c) WAP to print the numbers from 1 to 10 using for loop.	29/11/22	13/12/22		
6	(a) WAP to print the table of a number entered by the user using for loop. (b) WAP to calculate the sum of first 10 numbers by using while loop. (c) WAP to print the Fibonacci series by using functions.	6/12/22	13/12/22		
7	(a) WAP to add two numbers by using parameterized function. (b) WAP to display a message on status bar. (c) WAP to show the usage of window object.	13/12/22	20/12/22		

Exp. No	Experiment Name	Date of Performance	Date of Checking	Remarks	Mark
8	(a) WAP to show the usage of onclick, onmouseover and onmouseout events. (b) WAP to display the current day and time. (c) WAP to show the usage of all the date object functions.	13/12/22	20/12/22		
9	(a) WAP to show the usage of math object. (b) WAP to show the usage of string object functions: (i)indexOf (ii),lastIndexOf (iii)substr (iv)bold (v)italics (vi)sup (vii)sub (viii)charAt (ix)match (x)replace (c) WAP to show the usage of font tag in javascript.	20/12/22	27/12/22		
10	(a) Perform The Following Operations Using JavaScript: Addition, Subtraction, Division And Multiplication By Taking The Input From The User. (b) Perform swapping of two numbers by using temporary variable.	27/12/22	27/12/22		

EXPERIMENT-1

Create a web page that covers your CV using various HTML Tags (UL, OL, Table, etc).

CODE

```
<html>
<head>
<title>RESUME</title>
</head>
<body bgcolor="lightblue">
<div class="right">
<div class="image">

</div>
<div class="left">
<div class="CONTACT">
<h2>CONTACT</h2>
<p><b>Name : </b>Saksham Kochhar</p>
<p><b>Email id : </b>sakshamkochhar987@gmail.com</p>
<p><b>Phone No : </b>9711333881</p>
<p><b>LinkedIn : </b><a href="https://www.linkedin.com/in/saksham-kochhar-
a29645227">https://www.linkedin.com/in/saksham-kochhar-a29645227</a></p>
</div>
<div class="LANGUAGE">
<h2>LANGUAGE</h2>
<ul><li>English</li>
<li>Hindi</li>
<li>Freench</li>
</ul>
</div>
<div class="SKILLS">
<h2>SKILLS</h2>
<ul><li><b>Programing Language : </b>Python</li>
<li><b>Frontend : </b>HTML</li>
</ul></div>
<div class="HOBBIES">
<h2>HOBBIES</h2>
<ul><li>Dancing</li>
<li>Listening Music</li>
<li>table tennis</li>
```

</div>

<div class="STRENGTHS">

<h2>STRENGTHS</h2>

Determined

Time Management

Leadership

Patient

</div>

<div class "EDUCATION">

<h2>EDUCATION</h2>

<table border="2">

<caption><h4>Qualifications</h4></caption>

<tr><th>SCHOOL / COLLEGE</th>

<th>PASSING YEAR</th>

<th>PERCENTAGE</th></tr>

<tr><td>Delhi International Public School</td>

<td>2021</td>

<td>96%</td></tr>

<tr><td>MAHARAJA AGRASEN INSTITUTE OF TECHNOLOGY (SEM-1)</td>

<td>2022</td>

<td>97%</td></tr>

</tbody></table></div>

<div class="WORK">

<h2>WORK EXPERIENCE</h2>

founder member of dataxlect

</div>

</body>

</head>

</html>

←↻📄 File | C:/Users/HP/OneDrive/Desktop/resume.html

A🔍🌟🔖🔗👤⋮


CONTACT

Name : Saksham Kochhar

Email id : sakshamkochhar987@gmail.com

Phone No : 9711333881

LinkedIn : <https://www.linkedin.com/in/saksham-kochhar-a29645227>



LANGUAGE

- English
- Hindi
- Freench

SKILLS

- **Programming Language :** Python
- **Frontend :** HTML

HOBBIES

- Dancing
- Listening Music
- table tennis

STRENGTHS

- Determined
- Time Management
- Leadership
- Patience

EDUCATION

Qualifications

SCHOOL / COLLEGE	PASSING YEAR	PERCENTAGE
Delhi International Public School	2021	96%
MAHARAJA AGRASEN INSTITUTE OF TECHNOLOGY (SEM-1)	2022	97%

WORK EXPERIENCE

- founder member of dataxlect

EXPERIMENT-2

Create a webpage that displays brief details of various Programming Languages using various types of CSS.

CODE

```
<html>
<head>
  <meta charset="utf-8">
  <title>CSS Website</title>
  <link rel="stylesheet" href="styles.css">
  <link rel="preconnect" href="https://fonts.googleapis.com">
  <link rel="preconnect" href="https://fonts.gstatic.com" crossorigin>
  <link
href="https://fonts.googleapis.com/css2?family=Alkalami&family=Montserrat:ital,w
ght@0,100;0,700;0,800;0,900;1,600&display=swap" rel="stylesheet">
</head>
<body>
  <h1 class="Main">Coding Languages</h1>
  <ol>
    <div class="python">
      <li>Python</li>
      
    </div>
    <hr>
    <div class="cpp">
      <li>C++</li>
      
    </div>
    <br>
    <br>
    <br>
    <hr>
    <div class="js">
      <li>Java script</li>
      
    </div>
  </ol>
</body>
</html>
```

```
.Main{
text-align: center;
font-family: Alkalami;
font-size: 100px
}
body{
background-color: gray;
}
```

```
.list{
font-weight: bolder;
font-size: 25px;
padding-left: 20px;
}
```

```
.cpp{
height: 200px;
width: 200px;
```

```
}
```

```
.python{
color: blue;
font-size: large;
font-size: 40px;
}
```

```
.cpp{
color: red;
font-size: large;
font-size: 40px;
}
```

```
.js1{
height: 300px;
width: 300px;
}
```

```
.js{
color: green;
font-size: large;
font-size: 40px;
}
```




EXPERIMENT-3

Create a webpage using JavaScript and HTML to demonstrate Simple Calculator Application.

CODE

```
<html>
  <head>
    <meta charset="UTF-8" />
    <meta http-equiv="X-UA-Compatible" content="IE=edge" />
    <meta name="viewport" content="width=device-width, initial-scale=1.0" />
    <link rel="stylesheet" href="index.css">
    <title>Calculator App</title>
  </head>
  <body>
    <main>
      <div class="container">
        <div id="display"></div>
        <table>
          <tr>
            <th colspan="2" onclick="allClear()">AC</th>
            <th onclick="del()">DEL</th>
            <td></td>
          </tr>
          <tr>
            <td>1</td>
            <td>2</td>
            <td>3</td>
            <td>*</td>
          </tr>
          <tr>
            <td>4</td>
            <td>5</td>
            <td>6</td>
            <td>+</td>
          </tr>
          <tr>
            <td>7</td>
            <td>8</td>
            <td>9</td>
            <td>-</td>
          </tr>
```

```

        <tr>
            <td>.</td>
            <td>0</td>
            <th colspan="2" onclick="calc()">=</th>
        </tr>
    </table>
</div>
</main>
<script src="index.js"></script>
</body>
</html>

```

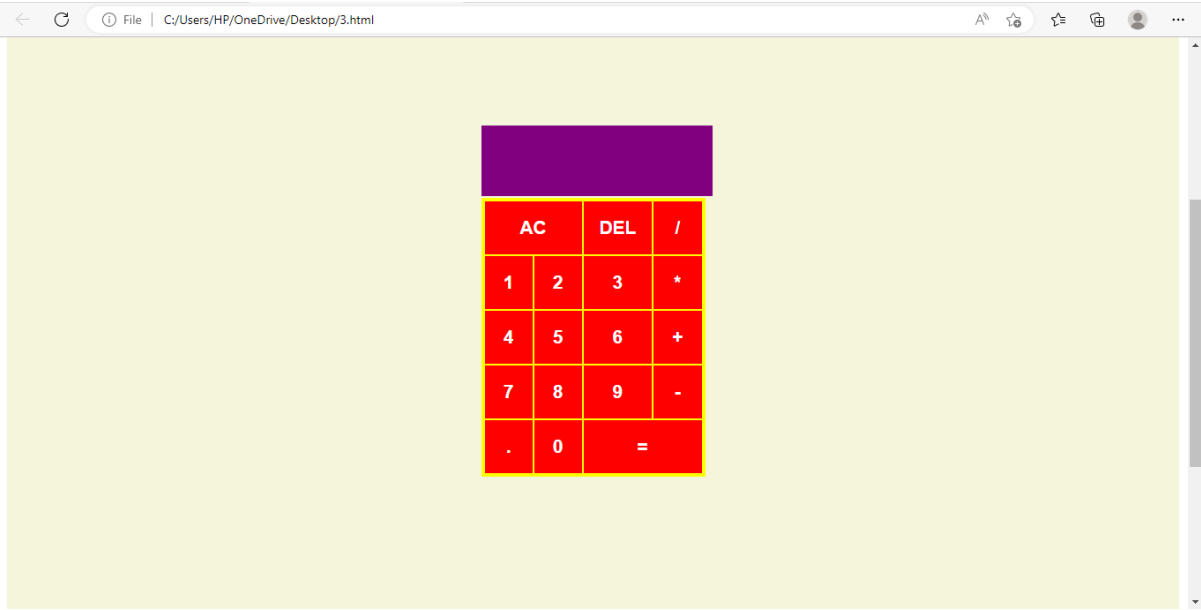
```

* {
    padding: 2;
    margin: 2;
    font-family: arial;
    box-sizing: border-box;
}
main {
    display: flex;
    align-items: center;
    justify-self: center;
    height: 200vh;
    background-color: beige;
    background-size: 250%;
    background-position: left;
}
@keyframes changeBG {
    from {
        background-position: left;
    }
    to {
        background-position: right;
    }
}
.container {

```

```
width: 262px;
margin: 0 auto;
vertical-align: center;
}
#display {
display: flex;
background-color: purple;
width: 262px;
height: 80px;
padding: 10px;
color: #fff;
align-items: flex-end;
justify-content: flex-end;
font-size: 1.5rem;
}
table {
background-color: yellow;
}
tr {
column-gap: 2;
row-gap: 2;
}
td,th {
font-weight: bold;
padding: 18px;
color: #fff;
width: 60px;
text-align: center;
margin: 0;
font-size: 1.3rem;
cursor: pointer;
background-color: red;
background-size: 250%;
background-position: right;
}
```

```
"use strict";
const display = document.querySelector("#display");
const td = document.querySelectorAll("td");
function allClear() {
  display.innerText = "";
}
function del() {
  let text = display.innerText;
  text = text.substring(0, text.length - 1);
  display.innerText = text;
}
for (let num of td) {
  num.addEventListener("click", () => {
    display.innerText += num.innerText;
  });
}
let str = "String";
let str2 = str.substring(0, str.length - 1);
console.log(str);
console.log(str2);
function calc() {
  if (display.innerText === "") {
    display.innerText === "";
  } else {
    return (display.innerText = eval(display.innerText));
  }
}
```



EXPERIMENT-4

Create a web page covering the basic CRUD operations (Create, Read, Update, Delete) that implements To-do list using JavaScript and HTML

CODE

```
<html>
  <head>
    <link rel="stylesheet" href="style.css">
    <title>to-do list</title>
    <body>
      <div id="myDIV" class="header">
        <h2>My To Do List</h2>
        <input type="text" id="myInput" placeholder="Title...">
        <span onclick="newElement()" class="addBtn">Add</span>
      </div>
      <ul id="myUL">
        <li>Complete assignments</li>
        <li>Meet friends</li>
        <li>File making</li>
        <li>Buy JAVA subscription</li>
      </ul>
      <script src="to.js"></script>
    </body>
  </head>
</html>
```

```
* {
  box-sizing: border-box;
}
ul {
  margin: 12px;
  padding: 10px;
}
```

```
ul li {
  cursor: pointer;
  position: relative;
  padding: 12px 8px 12px 40px;
  background: antiquewhite;
  font-size: 18px;
  transition: 0.2s;
  -webkit-user-select: none;
  -moz-user-select: none;
  -ms-user-select: none;
  user-select: none;
}
ul li:nth-child(odd) {
  background: #f9f9f9;
}
ul li:hover {
  background: #ddd;
}
ul li.checked {
  background: #888;
  color: #fff;
  text-decoration: line-through;
}
ul li.checked::before {
  content: "";
  position: absolute;
  border-color: #fff;
  border-style: solid;
  border-width: 0 2px 2px 0;
  top: 10px;
  left: 16px;
  transform: rotate(45deg);
  height: 15px;
  width: 7px;
}
.close {
  position: absolute;
  right: 0;
  top: 0;
  padding: 12px 16px 12px 16px;
}
.close:hover {
  background-color: #f44336;
```



```

    color: white;
}
.header {
    background-color: indianred;
    padding: 30px 40px;
    color: white;
    text-align: center;
}
.header:after {
    content: "";
    display: table;
    clear: both;
}
input {
    margin: 0;
    border: none;
    border-radius: 0;
    width: 75%;
    padding: 10px;
    float: left;
    font-size: 16px;
}
.addBtn {
    padding: 10px;
    width: 25%;
    background: #d9d9d9;
    color: #555;
    float: left;
    text-align: center;
    font-size: 16px;
    cursor: pointer;
    transition: 0.3s;
    border-radius: 0;
}
.addBtn:hover {
    background-color: #bbb;
}

```

```

var myNodelist = document.getElementsByTagName("LI");
var i;
for (i = 0; i < myNodelist.length; i++) {

```

```

var span = document.createElement("SPAN");
var txt = document.createTextNode("\u00D7");
span.className = "close";
span.appendChild(txt);
myNodeList[i].appendChild(span);
}
var close = document.getElementsByClassName("close");
var i;
for (i = 0; i < close.length; i++) {
  close[i].onclick = function() {
    var div = this.parentElement;
    div.style.display = "none";
  }
}
var list = document.querySelector('ul');
list.addEventListener('click', function(ev) {
  if (ev.target.tagName === 'LI') {
    ev.target.classList.toggle('checked');
  }
}, false);
function newElement() {
  var li = document.createElement("li");
  var inputValue = document.getElementById("myInput").value;
  var t = document.createTextNode(inputValue);
  li.appendChild(t);
  if (inputValue === "") {
    alert("You must write something!");
  } else {
    document.getElementById("myUL").appendChild(li);
  }
  document.getElementById("myInput").value = "";
  var span = document.createElement("SPAN");
  var txt = document.createTextNode("\u00D7");
  span.className = "close";
  span.appendChild(txt);
  li.appendChild(span);
  for (i = 0; i < close.length; i++) {
    close[i].onclick = function() {
      var div = this.parentElement;
      div.style.display = "none";
    }
  }
}
}

```

My To Do List

Title...

Add

- Complete assignments ×
- ✓ Meet friends ×
- File making ×
- Buy JAVA subscription ×

EXPERIMENT-5

- (a) WAP to show the usage of external javascript file.
- (b) WAP to show the usage of comments for single line and multiline.
- (c) WAP to print the numbers from 1 to 10 using for loop.

THEORY

DATA TYPES

Data can come in many different forms, or what we term *types*

Some programming languages are strongly typed languages. In these languages, whenever we use a piece of data we need to explicitly state what sort of data we are dealing with, and use of that data must follow strict rules applicable to its type. For example, we can't add a number and a word together.

JavaScript, on the other hand, is a weakly typed language and a lot more forgiving about how we use different types of data. When we deal with data, we often don't need to specify what type of data it is; JavaScript will work it out for itself.

Numerical Data

Numerical data comes in two forms:

- Whole numbers, such as 145, which are also known as *integers*. These numbers can be positive or negative and can span a very wide range: -2^{53} to 2^{53} .
- Fractional numbers, such as 1.234, which are also known as *floating-point* numbers. Like integers, they can be positive or negative, and they also have a massive range.

Text Data

Another term for one or more characters of text is a *string*. We tell JavaScript that text is to be treated as text and not as code simply by enclosing it inside quote marks ("). For example, "Hello World" and "A" are examples of strings that JavaScript will recognize. You can also use the single quote marks (')

JavaScript has a lot of other special characters, which can't be typed in but can be represented using the escape character in conjunction with other characters to create *escape sequences*. The principle behind this is similar to

that used in HTML. For example, more than one space in a row is ignored in HTML, so we represent a space by ` `. Similarly, in JavaScript there are instances where we can't use a character directly but must use an escape sequence. The following table details some of the more useful escape sequences:

Escape Sequences	Character Represented
<code>\b</code>	Backspace
<code>\f</code>	Form feed
<code>\n</code>	New line
<code>\r</code>	Carriage return
<code>\t</code>	Tab
<code>\'</code>	Single quote
<code>\"</code>	Double quote
<code>\\</code>	Backslash
<code>\xNN</code>	NN is a hexadecimal number that identifies a character in the Latin-1 character set.

Boolean Data

The use of yes or no, positive or negative, and true or false is commonplace in the "real" world. The idea of true and false is also fundamental to digital computers; they don't understand maybes, only true and false. In fact, the concept of "yes or no" is so useful it has its own data type in JavaScript: the *Boolean* data type. The Boolean type has two possible values: true for yes and false for no.

Variables—Storing Data in Memory

Data can be stored either permanently or temporarily.

We will want to keep important data, such as the details of a person's bank account, in a permanent store

However, there are other cases where we don't want to permanently store data, but simply want to keep a temporary note of it.

Each variable is given a name so that you can refer to it elsewhere in your code. These names must follow certain rules.

Declaring Variables and Giving Them Values

Before you can use a variable, you should declare its existence to the computer using the var keyword. This warns the computer that it needs to reserve some memory for your data to be stored in later. To declare a new variable called my First Variable you would write
var my First Variable;

Note that the semicolon at the end of the line is not part of the variable name, but instead is used to indicate to JavaScript the end of a statement. This line is an example of a JavaScript statement. Once declared, a variable can be used to store any type of data.

CODE

```
<html>
<head>
  <title>javascript demo</title>
  <script src="123.js"></script>
</head>
<body bgcolor="lightgrey"style="position:relative; top:200px;font-weight: bold;
font-family: Times New Roman;
font-size: 25px;
border: 10px solid rgb(9, 54, 159);
height: 500px;
text-align: center;">
</body>
</html>
```

```
document.write("Hello" + "<br><br><br>");
document.write("Two types of comments" + "<br>");
document.write("Single line comment:-//this is single line comment" + "<br>");
document.write("Multi line comment:-/*this is multi line comment*/" + "<br><br>");
```

```
var i;  
for(i=1;i<=10;i++)  
{ document.write(i + "<br>"); }
```

EXPERIMENT-6

- (a) WAP to print the table of a number entered by the user using for loop.
- (b) WAP to calculate the sum of first 10 numbers by using while loop.
- (c) WAP to print the Fibonacci series by using functions.

THEORY

What's a Function?

A function is a piece of code that sits dormant until it is referenced or called upon to do its "function". In addition to controllable execution, functions are also a great time saver for doing repetitive tasks.

Instead of having to type out the code every time you want something done, you can simply call the function multiple times to get the same effect. This benefit is also known as "code reusability".

Example Function in JavaScript

A function that does not execute when a page loads should be placed inside the *head* of your HTML document. Creating a function is really quite easy. All you have to do is tell the browser you're making a function, give the function a name, and then write the JavaScript like normal.

Looping—The **for** and **while** Statements

Looping means repeating a block of code while a condition is true. This is achieved in JavaScript using two statements, the **while** statement and the **for** statement.

The JavaScript *For Loop* resembles the **for** loop you may have seen in many other programming languages. It is used when you need to do a set of operations many times, with an increment of some kind after each run through the block of code.

Often when you write code, you want the same block of code to run over and over again in a row. Instead of adding several almost equal lines in a script we can use loops to perform a task like this.

In JavaScript, there are different kinds of loops:

- **for** - loops through a block of code a specified number of times
- **while** - loops through a block of code while a specified condition is true
- **do...while** - also loops through a block of code while a specified condition is true
- **for...in** - loops through the properties of an object

JavaScript For Loop Explained

There are four important aspects of a JavaScript *for loop*:

1. The *counter* variable is something that is created and usually used only in the *for loop* to count how many times the *for loop* has *looped*. *i* is the normal label for this counter variable and what we will be using.
2. The conditional statement. It is what decides whether the *for loop* continues executing or not. This check usually includes the *counter* variable in some way.
3. The counter variable is incremented after every loop in the *increment* section of the *for loop*.
4. The code that is executed for each loop through the *for loop*.

The While Loop

While Loops execute a block of code as long as a specified condition is true.

The while loop loops through a block of code while a specified condition is true.

The while loop is an advanced programming technique that allows you to do something over and over *while* a conditional statement is true. Although the general uses of the *while loop* are usually a bit complex.

JavaScript While Loop Explained

There are two key parts to a JavaScript *while loop*:

1. The conditional statement which must be *True* for the *while loop's* code to be executed.
2. The *while loop's* code that is contained in curly braces "{ and }" will be executed if the

condition is *True*.

When a *while loop* begins, the JavaScript interpreter checks if the condition statement is true. If it is, the code between the curly braces is executed. At the end of the code segment "}", the *while loop* loops back to the condition statement and begins again.

If the condition statement is always *True*, then you will never exit the *while loop*, so be **very careful** when using while loops!

CODE

```
<html>
<head>
<title>NUMBERS</title>
<link rel="stylesheet" href="5a.css">
</head>
<body bgcolor="lightblue">
  <div id="display">
    <div id="div1">
      The table is:-
      <script type="text/JavaScript">
        var n=parseInt(prompt("Enter the number:-")); var res;
        document.write("The table is:-"+"<br>");
        for(i=1;i<=10;++i)
        {
          res=i*n;
          document.write(n+"*" +i+"="+res+"<br>"); }
      </script>
    </div>
    <div id="div2">
      Sum of first 10 natural numbers will be:-
      <script type="text/JavaScript">
        var i=1;
        var res=0;
        while(i<=10)
        {
          res=res+i;
          i++; }
      </script>
    </div>
  </div>
```

```

        for(b=1;b<=9;b++)
        {
            document.write(b+""); } document.write("10=
"+res+"<br><br><br><br><br>");
        </script>
    </div>
    <div id="div3">
        Fibonacci series will be:-
        <script type="text/JavaScript">
            function fibbo(y)
            {
                var a=0;
                var b=1;
                var c;
                document.write(a+"<br>"); document.write(b+"<br>");
                for(i=2;i<y;i++) {
                    c=a+b;
                    a=b;
                    b=c; document.write(c+"<br>");
                }
            }
            var n=parseInt(prompt("Enter limit:-")); fibbo(n);
        </script>
    </div>
</div>
</body>
</html>

```

```

#display {
    font-weight: bold;
    font-family: Times New Roman;
    font-size: 25px;
    border: 10px solid rgb(150, 97, 203);
    text-align: center;
    flex-direction: row;
}
#div1{background-color:#7ff9ae;
    width:350px;height:350px;
}
#div2{background-color:#76bbfc;

```

```
width:350px;height:350px;}
#div3{background-color:#ff7f8e;
width:350px;height:350px;}
```

EXPERIMENT-7

(a) WAP to add two numbers by using parameterized function.

THEORY

Function parameters are the names listed in the function definition.

Function arguments are the real values passed to (and received by) the function.

Parameter Rules

JavaScript function definitions do not specify data types for parameters.

JavaScript functions do not perform type checking on the passed arguments.

JavaScript functions do not check the number of arguments received.

CODE

```
<html>
<head>
<script>
function add(a, b) {
a=10;
b=10;
var c=parseInt(a)+parseInt(b);
return c;
}
</script>
</head>
```

```
<body bgcolor="lightpink">
<div id="div1" style = "position:relative; top:250px;background-
color:lightgreen;"><center>
<script type="text/JavaScript">
var x,y;
var d=add(x, y);
document.write("The sum of two numbers is : "+d);
</script>
</center>
</body>
</html>
```

OUTPUT



(b) WAP to display a message on status bar.

THEORY

The status bar appears below the grid and holds components that typically display information about the data in the grid.

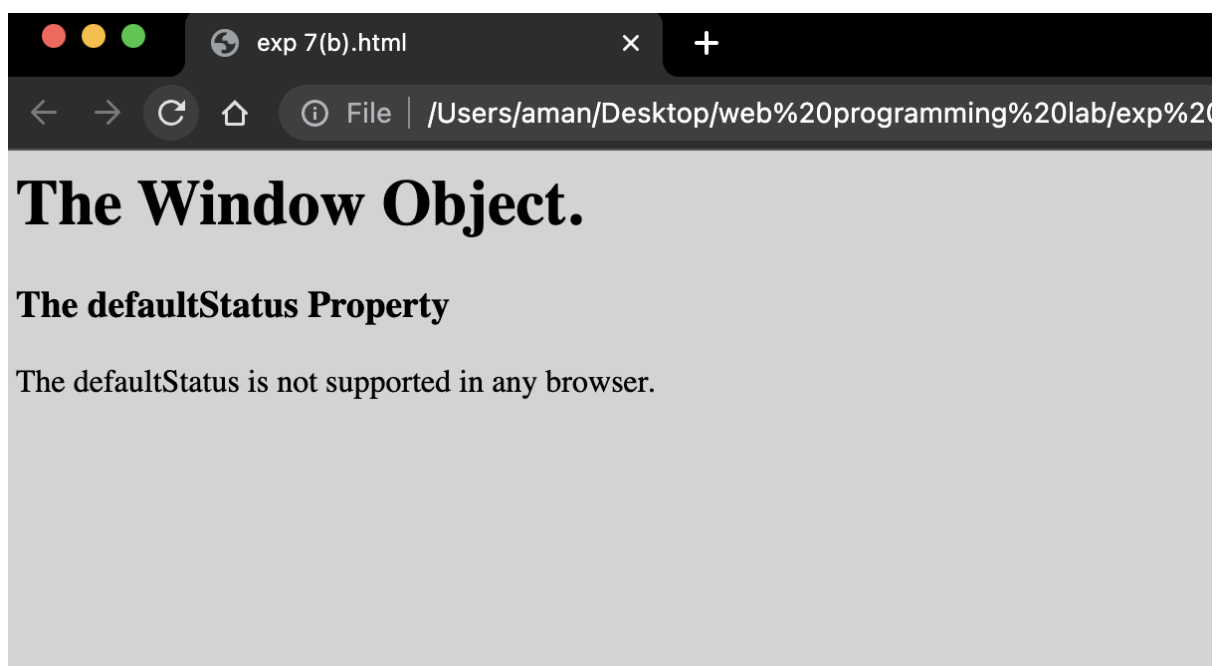
Within the Status Bar you can specify which Status Bar Panels you want to display.

Status Bar Panels allow you to add your own components to the grid's Status Bar. Use this when the provided status bar panels do not meet your requirements.

CODE

```
<html>
<head>
<body bgcolor="lightgrey">
<h1>The Window Object.</h1>
<p><h3>The defaultStatus Property</h3></p>
<p>The defaultStatus is not supported in any browser.</p>
<script>
window.defaultStatus="Hello and Welcome";
</script>
</body>
</head>
</html>
```

OUTPUT



(c) WAP to show the usage of window object.

THEORY

The **window** object is supported by all browsers. It represents the browser's window.

All global JavaScript objects, functions, and variables automatically become members of the window object.

Global variables are properties of the window object.

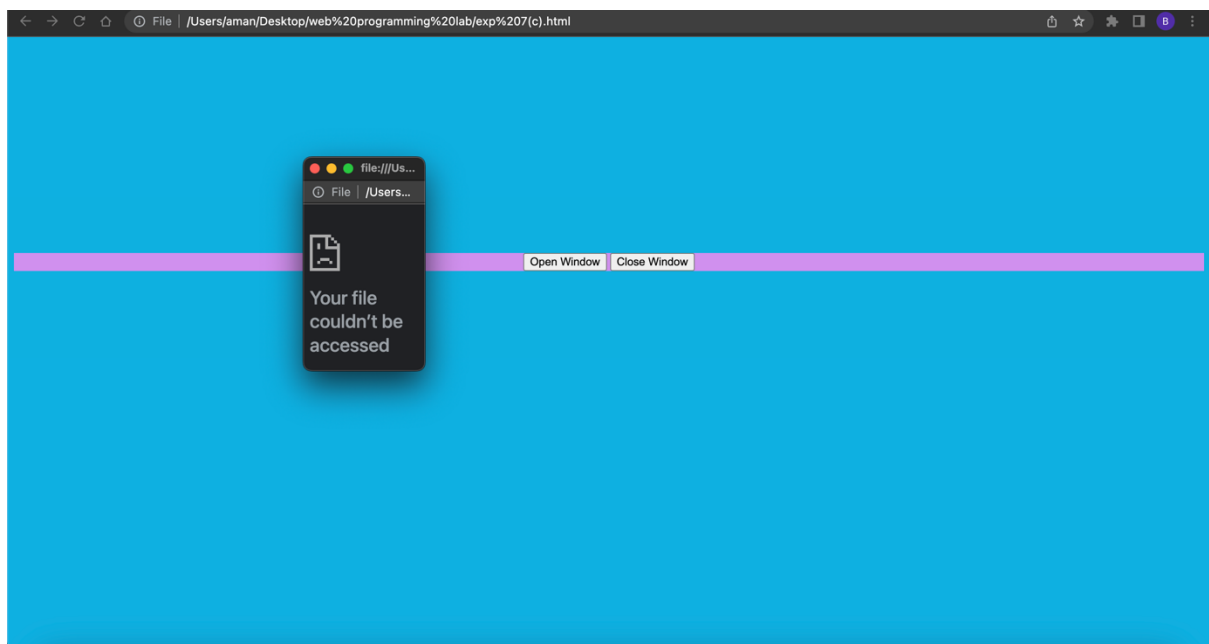
Global functions are methods of the window object.

CODE

```
<html>
<head>
<script>
function openwin(){ m=window.open("w.html", "_blank", "height=200,width=100"); }
function closewin() {
m.close();} </script>
</head>
<body bgcolor="seablue">
<div id="div1" style = "position:relative;top:250px;background-color:rgb(208, 144,
238);"><center>
<input type="button" value="Open Window" onclick="openwin()">
```

```
<input type="button" value="Close Window" onclick="closewin()">
</center>
</body>
</html>
```

OUTPUT



EXPERIMENT-8

(a) WAP to show the usage of onclick, onmouseover and onmouseout events.

THEORY

Reacting to Events

A JavaScript can be executed when an event occurs, like when a user clicks on an HTML element.

onclick Event :

This is the most frequently used event type which occurs when a user clicks mouse left button. You can put your validation, warning etc against this event type.

To execute code when a user clicks on an element, add JavaScript code to an HTML event attribute

The onmouseover and onmouseout Events

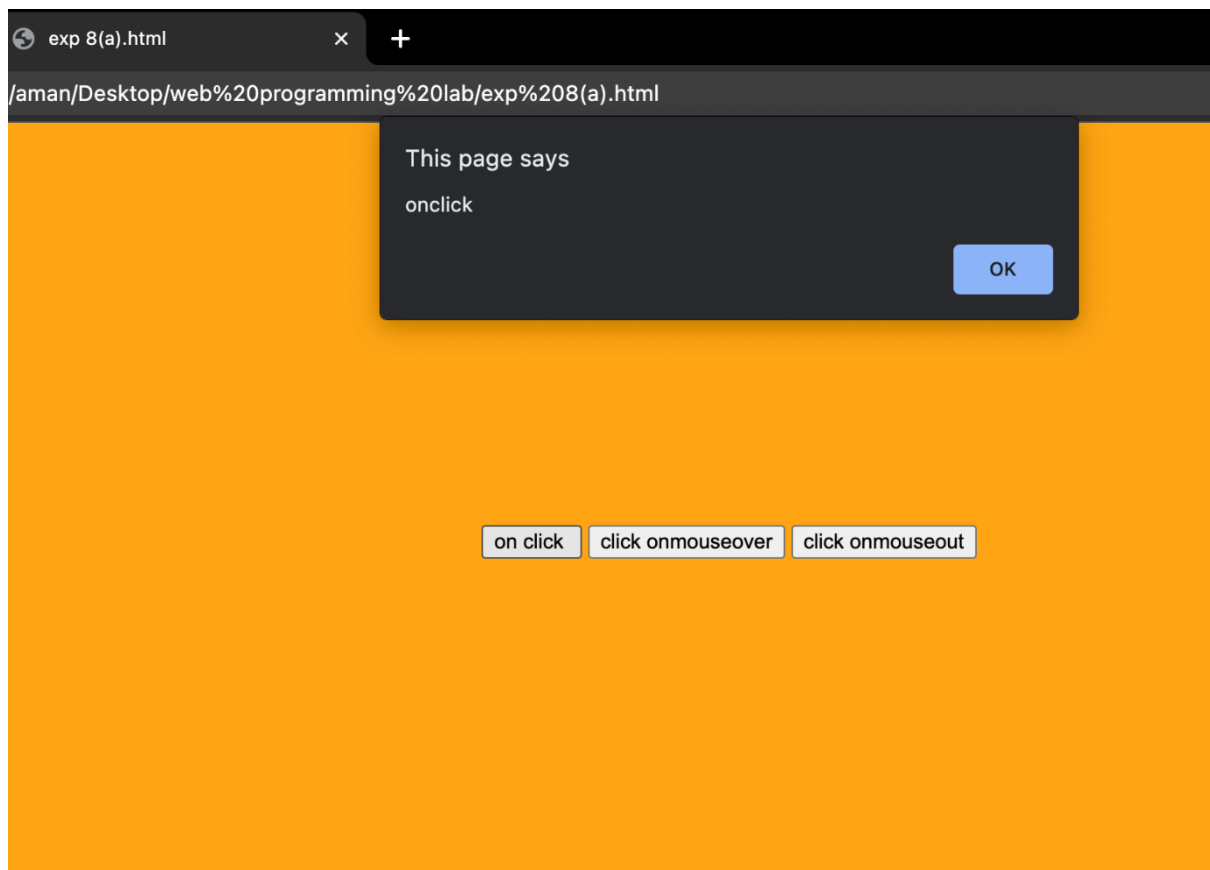
The onmouseover and onmouseout events can be used to trigger a function when the user mouses over, or out of, an HTML element.

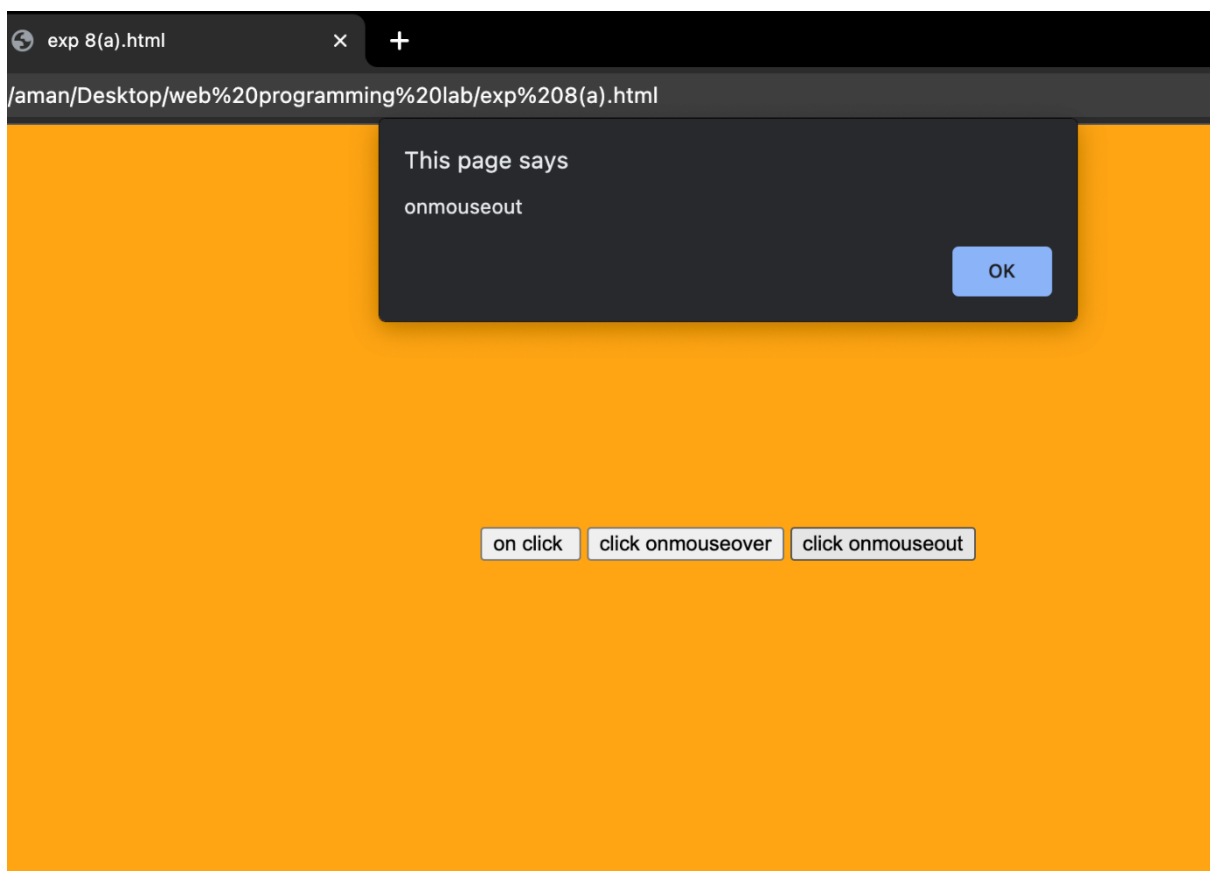
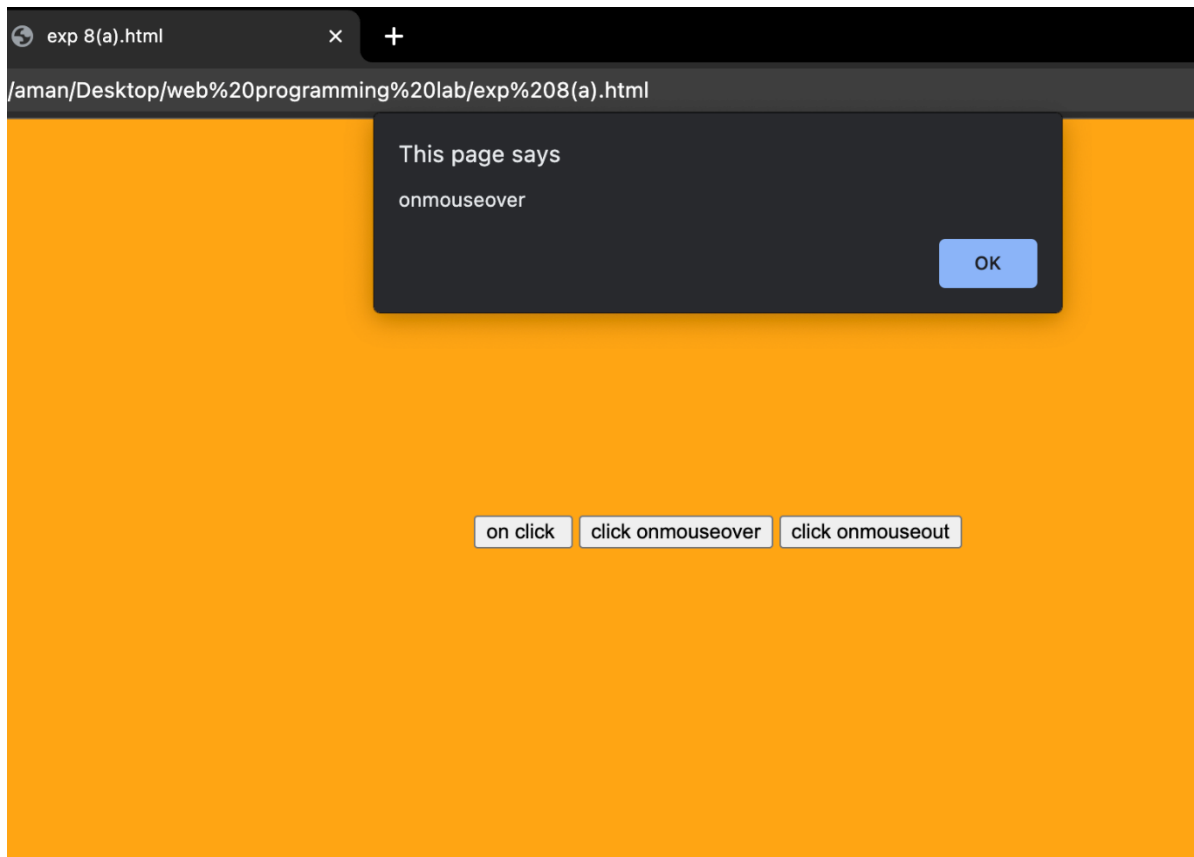
OnMouseOver, as the name suggests, will execute the code when the mouse passes over the link. The onMouseOver will execute a piece of code when the mouse moves away from the link. They are used in exactly the same way as onClick.

CODE

```
<html>
<head>
<script>
function f1(){ alert("onclick");}
function f2(){ alert("onmouseover"); }
function f3(){ alert("onmouseout"); }
</script>
</head>
<body bgcolor="orange" style="position:relative; top:250px;">
<center>
<input type="button" onclick="f1()" value="on click ">
<input type="button" onmouseover="f2()" value="click onmouseover">
<input type="button" onmouseout="f3()" value="click onmouseout">
</center>
</body>
</html>
```

OUTPUT





(b) WAP to display the current day and time.

THEORY

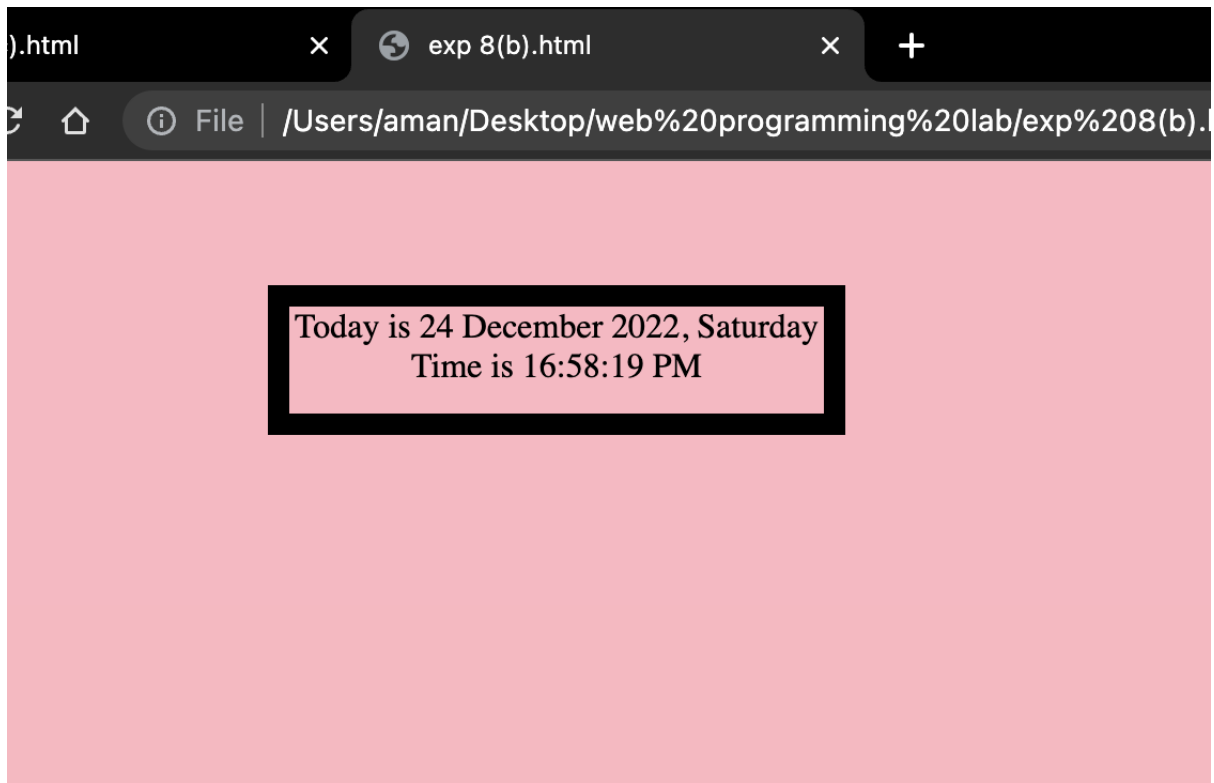
In JavaScript, we can easily get the current date or time by using the new Date() object. By default, it uses our browser's time zone and displays the date as a full text string, such as "Fri Jun 17 2022 10:54:59 GMT+0500 (Indian Standard Time)" that contains the current date, time, and time zone.

CODE

```
<html>
<body bgcolor="lightpink" style="position:relative;left:200px;top:50px;border:10px
solid;width:250px;
height:50px;">
<center>
<script>
var d=new Date(); var
days=["Sunday","Monday","Tuesday","Wednesday","Thursday","friday","Saturday"];
```

```
var
month=["January","February","March","April","May","June","July","August","September","October","November","December"];
document.write("Today is "+d.getDate()+" "+month[d.getMonth()]+" "+d.getFullYear()+" "+days[d.getDay()]+<br>");
if(d.getHours()<=12)
document.write("Time is "+d.getHours()+":"+d.getMinutes()+":"+d.getSeconds()+" AM"+<br>");
else
document.write("Time is "+d.getHours()+":"+d.getMinutes()+":"+d.getSeconds()+" PM"+<br>");
</script>
</center>
</body>
</html>
```

OUTPUT



(c) WAP to show the usage of all the date object functions.

THEORY

Date object

The Date object is used to work with dates and times. The Date object is useful when you want to display a date or use a timestamp in some sort of calculation. In Java, you can either make a Date object by supplying the date of your choice, or you can let JavaScript create a Date object based on your visitor's system clock. It is usually best to let JavaScript simply use the system clock.

Get the JavaScript Time

The Date object has been created, and now we have a variable that holds the current date! To get the information we need to print out, we have to utilize some or all of the following functions:

- getTime() - Number of milliseconds since 1/1/1970 @ 12:00 AM
- getSeconds() - Number of seconds (0-59)
- getMinutes() - Number of minutes (0-59)
- getHours() - Number of hours (0-23)
- getDay() - Day of the week(0-6). 0 = Sunday, ... , 6 = Saturday
- getDate() - Day of the month (0-31)
- getMonth() - Number of month (0-11)
- getFullYear() - The four digit year (1970-9999)

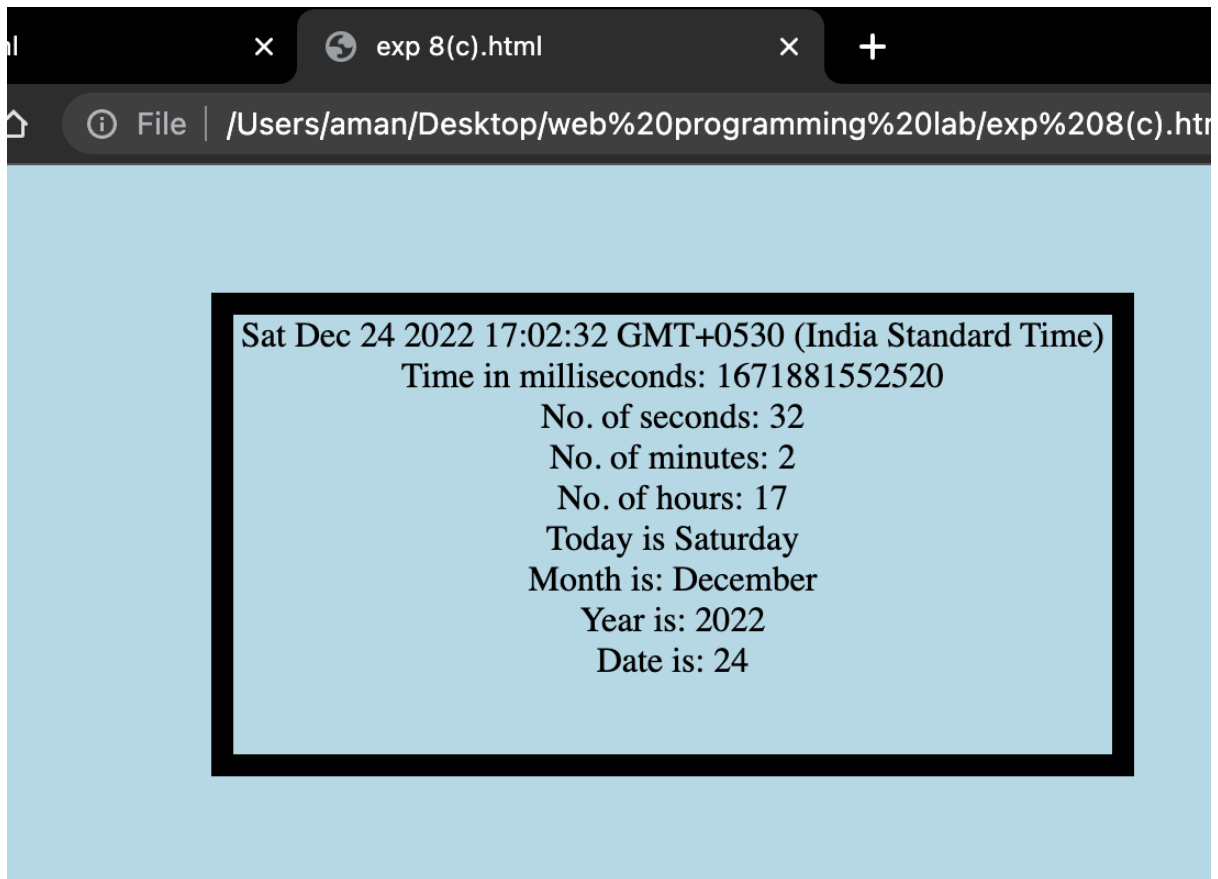
CODE

```
<html>
<body bgcolor="lightblue" style="position:relative;left:200px;top:50px;border:10px
solid;width:400px;
height:200px;">
<center>
<script>
var
days=["Sunday","Monday","Tuesday","Wednesday","Thursday","friday","Saturday"];
var
month=["January","February","March","April","May","June","July","August","Septe
mber","Oct ober","November","December"];
var d=new Date();
document.write(d+"<br>");
document.write("Time in milliseconds: "+d.getTime()+"<br>");
document.write("No. of seconds: "+d.getSeconds()+"<br>");
document.write("No. of minutes: "+d.getMinutes()+"<br>");
```



```
document.write("No. of hours: "+d.getHours()+"<br>");
document.write("Today is "+days[d.getDay()+"<br>");
document.write("Month is: "+month[d.getMonth()+"<br>");
document.write("Year is: "+d.getFullYear()+"<br>");
document.write("Date is: "+d.getDate()+"<br>");
</script>
</center>
</body>
</html>
```

OUTPUT



EXPERIMENT-9

(a) WAP to show the usage of math object.

THEORY

Math Object

The Math object allows you to perform mathematical tasks.

The Math object includes several mathematical constants and methods.

Syntax for using properties/methods of Math:

```
var x=Math.PI;  
var y=Math.sqrt(16);
```

Note: Math is not a constructor. All properties and methods of Math can be called by using Math as an object without creating it.

Mathematical Constants

JavaScript provides eight mathematical constants that can be accessed from the Math object. These are: E, PI, square root of 2, square root of 1/2, natural log of 2, natural log of 10, base-2 log of E, and base-10 log of E.

You may reference these constants from your JavaScript like this:

```
Math.E Math.PI Math.SQRT2 Math.SQRT1_2 Math.LN2 Math.LN10 Math.LOG2E  
Math.LOG10E
```

Mathematical Methods

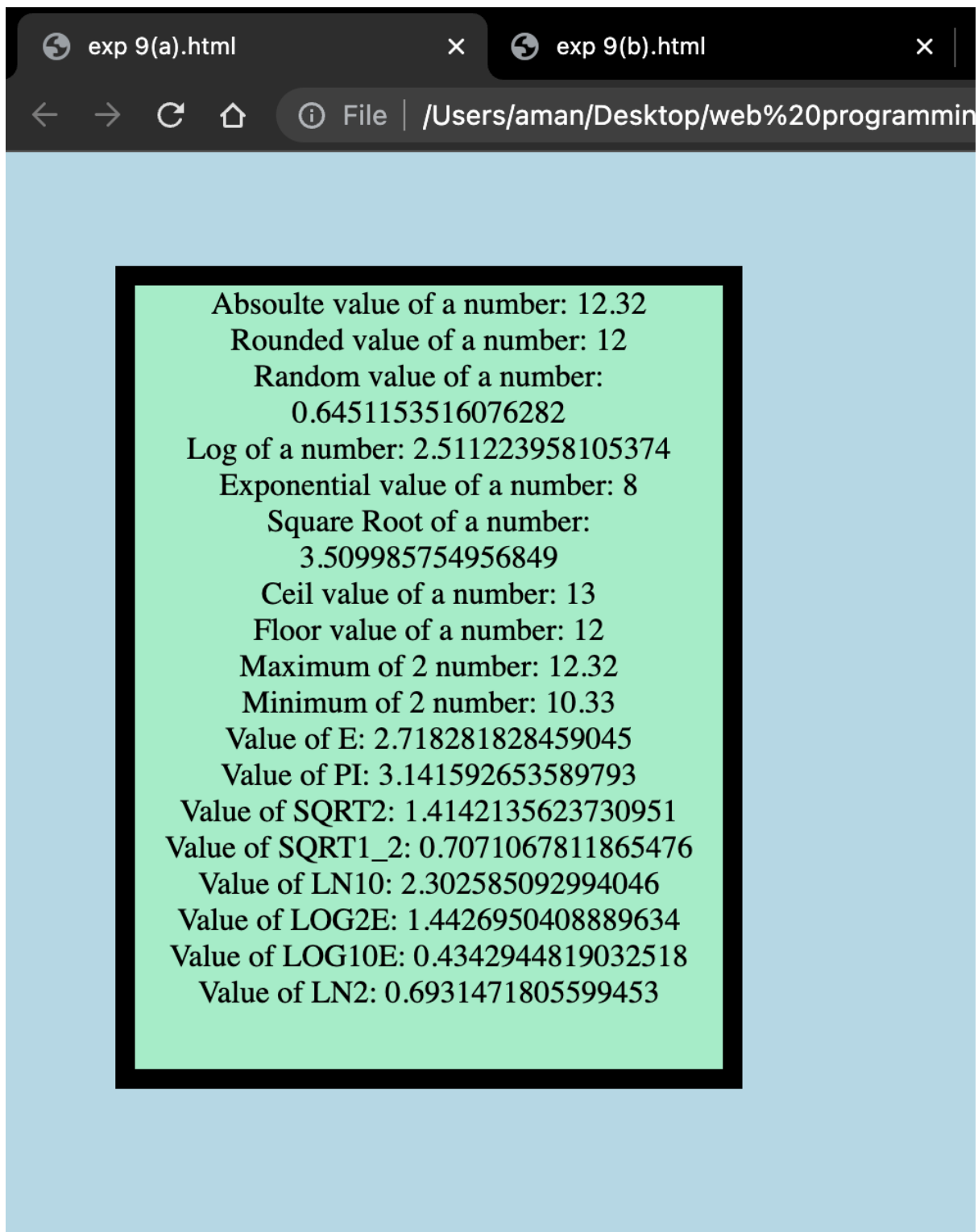
In addition to the mathematical constants that can be accessed from the Math object there are also several methods available.

CODE

```
<html>
```

```
<body bgcolor="lightblue">
<div id="div1" style = "position:relative; left:50px; top:50px;background-
color:rgb(144, 238, 197);border:10px solid;
width:300px;height:400px;">
<center>
<script>
var a=12.32;
var b=10.33;
document.write("Absoulte value of a number: "+Math.abs(a)+"<br>");
document.write("Rounded value of a number: "+Math.round(a)+"<br>");
document.write("Random value of a number: "+Math.random(a)+"<br>");
document.write("Log of a number: "+Math.log(a)+"<br>");
document.write("Exponential value of a number: "+Math.pow(2,3)+"<br>");
document.write("Square Root of a number: "+Math.sqrt(a)+"<br>");
document.write("Ceil value of a number: "+Math.ceil(a)+"<br>");
document.write("Floor value of a number: "+Math.floor(a)+"<br>");
document.write("Maximum of 2 number: "+Math.max(a,b)+"<br>");
document.write("Minimum of 2 number: "+Math.min(a,b)+"<br>");
document.write("Value of E: "+Math.E+"<br>");
document.write("Value of PI: "+Math.PI+"<br>");
document.write("Value of SQRT2: "+Math.SQRT2+"<br>");
document.write("Value of SQRT1_2: "+Math.SQRT1_2+"<br>");
document.write("Value of LN10: "+Math.LN10+"<br>");
document.write("Value of LOG2E: "+Math.LOG2E+"<br>");
document.write("Value of LOG10E: "+Math.LOG10E+"<br>");
document.write("Value of LN2: "+Math.LN2+"<br>");
</script>
</center>
</body>
</html>
```

OUTPUT



(b) WAP to show the usage of string object functions:

(i)indexOf (ii),lastIndexOf (iii)substr (iv)bold (v)italics (vi)sup (vii)sub (viii)charAt (ix)match (x)replace

THEORY

String Objects

Like most objects, String objects need to be created before they can be used. To create a String object, we can write

```
var string1 = new String("Hello");
```

```
var string2 = new String(123);
```

```
var string3 = new String(123.456);
```

However, as we have seen, we can also declare a string primitive and use it as if it were a String object, letting JavaScript do the conversion to an object for us behind the scenes. For example

```
var string1 = "Hello";
```

The charAt() Method—Selecting a Single Character from a String

The charAt() method takes one parameter: the index position of the character you want in the string. It then returns that character. charAt() treats the positions of the string characters as starting at 0, so the first character is at index 0, the second at index 1, and so on.

For example, to find the last character in a string, we could use the code

```
var myString = prompt("Enter some text","Hello World!");
```

```
var theLastChar = myString.charAt(myString.length - 1);
```

```
document.write("The last character is " + theLastChar);
```

The indexOf() and lastIndexOf() Methods—Finding a String Inside Another String

```
<script language="JavaScript" type="text/javascript">
var myString = "Hello paul. How are you Paul";
var foundAtPosition;
foundAtPosition = myString.indexOf("Paul");
alert(foundAtPosition);
```

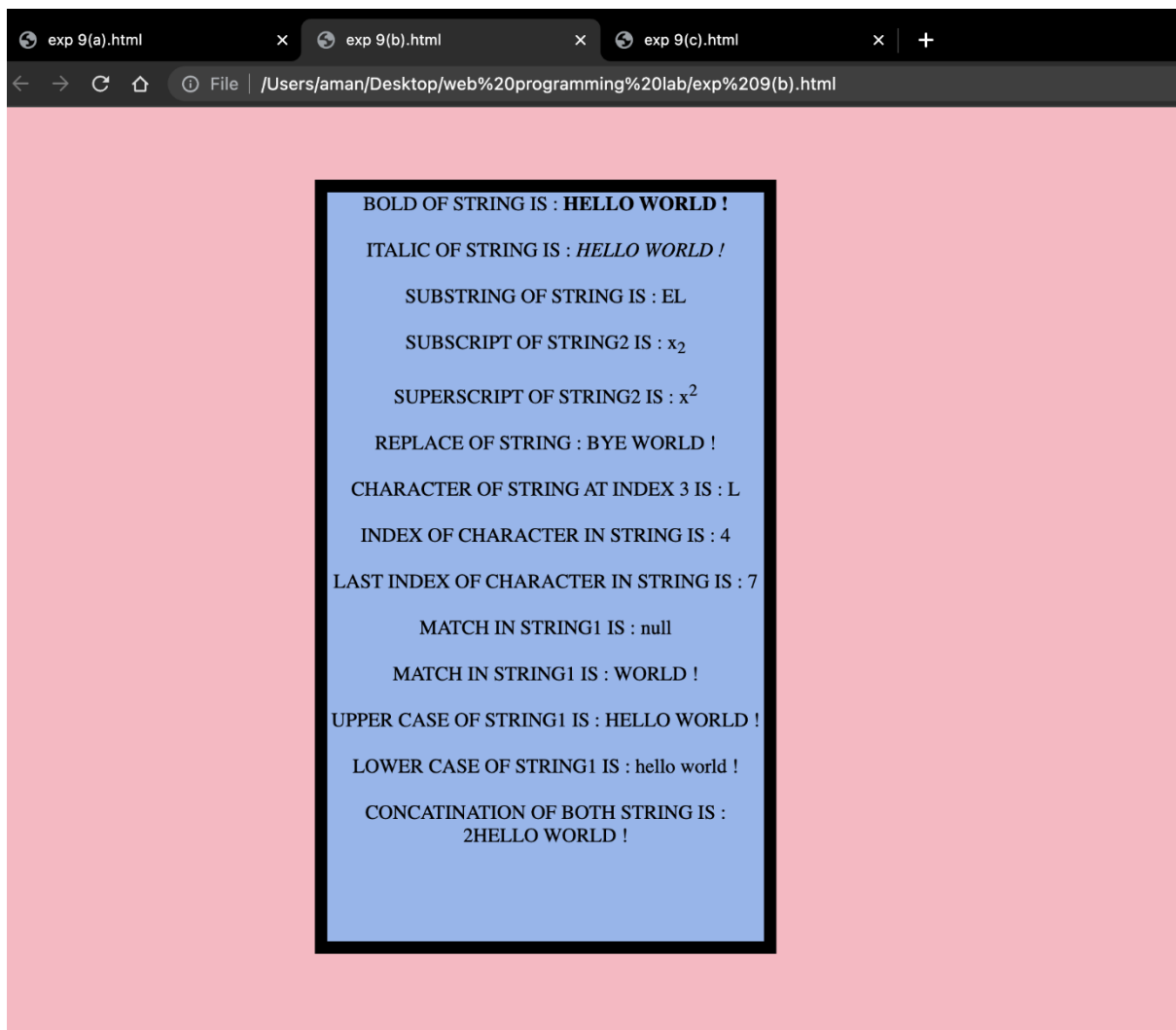
The match() Method

The match() method is very similar to the search() method, except that instead of returning the position where a match was found, it returns an array. Each element of the array contains the text of each match that is found.

CODE

```
<html>
<body bgcolor="lightpink">
<div id="div1" style = "position:relative; left:250px; top:50px;background-
color:rgb(144, 182, 238);border:10px solid;
width:350px;height:600px;">
<center>
<script>
var str=new String("HELLO WORLD !");
var str2="2";
document.write("BOLD OF STRING IS : "+str.bold()+"<br><br>");
document.write("ITALIC OF STRING IS : "+str.italics()+"<br><br>");
document.write("SUBSTRING OF STRING IS : "+str.substring(1,3)+"<br><br>");
document.write("SUBSCRIPT OF STRING2 IS : "+ "x"+str2.sub()+"<br><br>");
document.write("SUPERScript OF STRING2 IS : "+ "x"+str2.sup()+"<br><br>");
document.write("REPLACE OF STRING : "+str.replace("HELLO","BYE")+"<br><br>");
document.write("CHARACTER OF STRING AT INDEX 3 IS :
"+str.charAt(3)+"<br><br>");
document.write("INDEX OF CHARACTER IN STRING IS :
"+str.indexOf("O")+"<br><br>");
document.write("LAST INDEX OF CHARACTER IN STRING IS : "+str.lastIndexOf
("O")+"<br><br>");
document.write("MATCH IN STRING1 IS : "+str.match("WORLD2")+"<br><br>");
document.write("MATCH IN STRING1 IS : "+str.match("WORLD !")+"<br><br>");
document.write("UPPER CASE OF STRING1 IS : "+str.toUpperCase()+"<br><br>");
document.write("LOWER CASE OF STRING1 IS : "+str.toLowerCase()+"<br><br>");
document.write("CONCATINATION OF BOTH STRING IS :
"+str2.concat(str)+"<br><br>");
</script>
</center>
</body>
</html>
```

OUTPUT



(c) WAP to show the usage of font tag in javascript.

THEORY

The font tag is used **to change the color, size, and style of a text**. The base font tag is used to set all the text to the same size, color and face. Example: In this example, we have used the tag with a font size as 5.

CODE

```
<html>
<body bgcolor="skyblue">
<div id="div1" style = "position:relative; left:250px; top:50px;background-
color:rgb(144, 182, 238);border:5px solid;
width:200px;height:100px;">
<center>
<script>
document.write("<font face='Chalkboard SE' color='red' size='15'>
SAKSHAM</font>");
</script>
</center>
</body>
</html>
```

OUTPUT

SAKSHAM

EXPERIMENT-10

(a) Perform The Following Operations Using JavaScript: Addition, Subtraction, Division And Multiplication By Taking The Input From The User.

THEORY

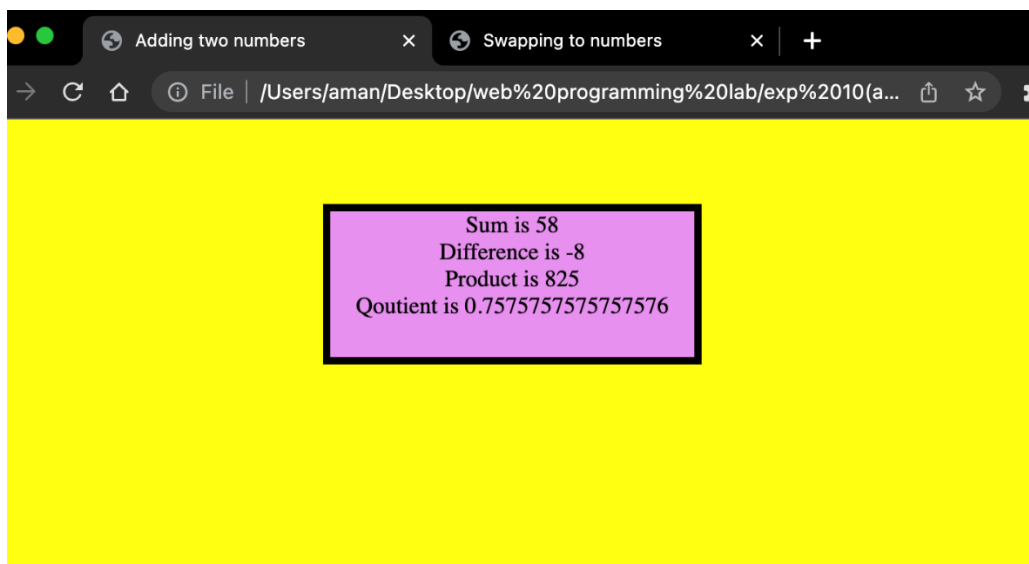
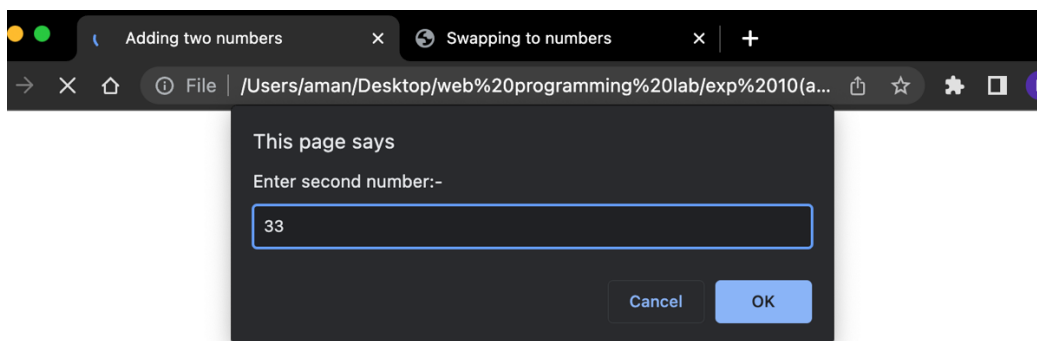
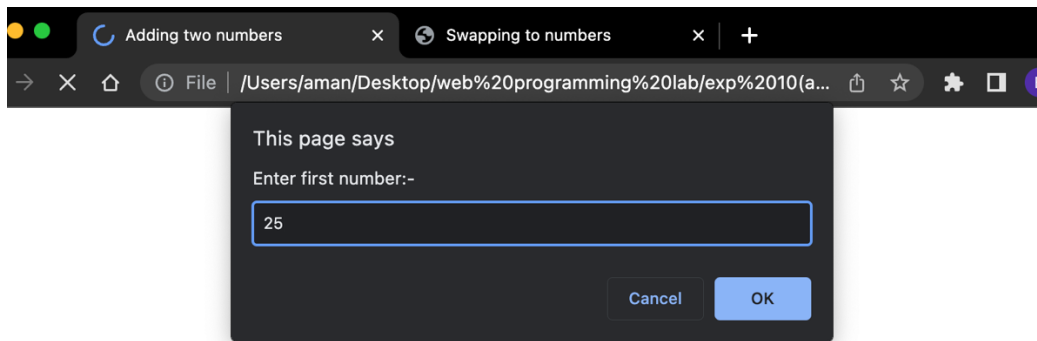
Arithmetic operators perform arithmetic on numbers (literals or variables).

Operator	Description
+	Addition
-	Subtraction
*	Multiplication
**	Exponentiation (ES2016)
/	Division
%	Modulus (Remainder)

CODE

```
<html>
<head>
<title>Adding two numbers</title>
</head>
<body bgcolor="yellow">
<div id="div1" style = "position:relative; left:250px; top:50px;background-
color:rgb(230, 144, 238);border:5px solid;
width:250px;height:100px;">
<center>
<script type="text/JavaScript">
var a=parseInt(prompt("Enter first number:-"));
var b=parseInt(prompt("Enter second number:-"));
document.write("Sum is "+(a+b)+"<br>");
document.write("Difference is "+(a-b)+"<br>");
document.write("Product is "+(a*b)+"<br>");
document.write("Qoutient is "+(a/b)+"<br>");
</script>
</center>
</body>
</html>
```

OUTPUT



(b) Perform swapping of two numbers by using temporary variable.

THEORY

In this program, the temp variable is assigned the value of the first variable. Then, the value of the first variable is assigned to the second variable. Finally, the temp (which holds the initial value of first) is assigned to second. This completes the swapping process.

CODE

```
<html>
<head>
<title>Swapping to numbers</title>
</head>
<body bgcolor="blue">
<div id="div1" style = "position:relative; left:250px; top:50px;background-
color:rgb(176, 172, 176);border:5px solid;
width:250px;height:50px;">
<center>
<script type="text/JavaScript">
var a=parseInt(prompt("Enter first number:-"));
var b=parseInt(prompt("Enter second number:-"));
var temp;
temp=a;
a=b;
b=temp;
document.write("The new first no. is " +a+"<br>");
document.write("The new second no. is " +b+"<br>");
</script>
</center>
</body>
</html>
```

OUTPUT

