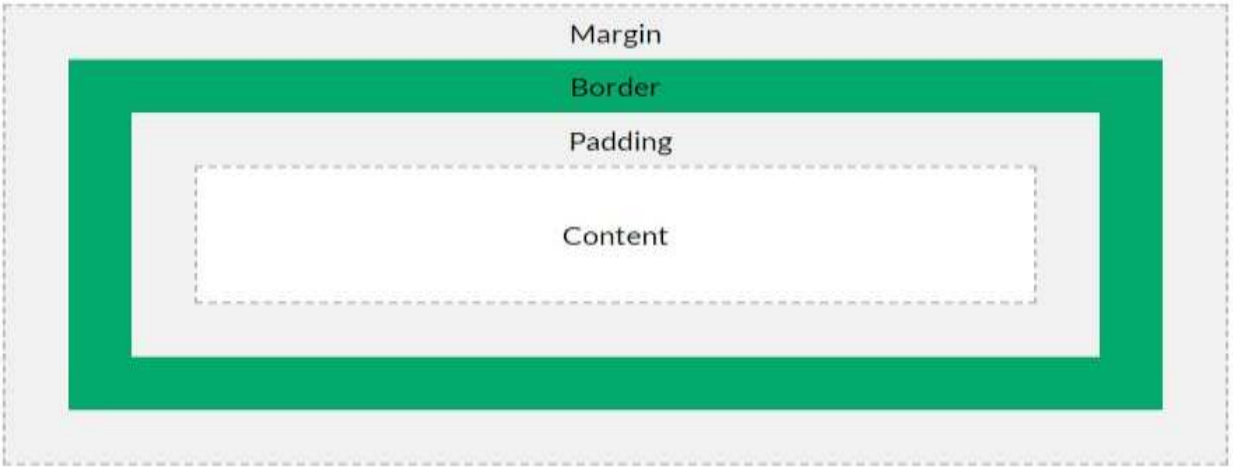
3c. Write a program to apply Box model.

Description:

- In CSS, the term "box model" is used when talking about design and layout.
- The CSS box model is essentially a box that wraps around every HTML element. It consists of: margins, borders, padding, and the actual content. The image below illustrates the box model:



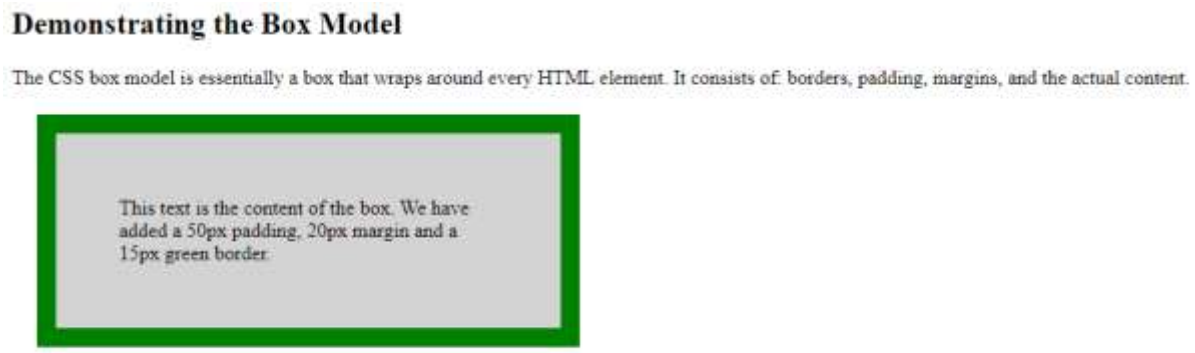
- Explanation of the different parts:
- **Content** - The content of the box, where text and images appear
- **Padding** - Clears an area around the content. The padding is transparent
- **Border** - A border that goes around the padding and content
- **Margin** - Clears an area outside the border. The margin is transparent
- The box model allows us to add a border around elements, and to define space between elements.

Program:

```
<!DOCTYPE html>

<html>

<head>

<style>

div {
```

```
  background-color: lightgrey;
  width: 300px;
  border: 15px solid green;
  padding: 50px;
  margin: 20px;
}
</style>
</head>
<body>
<h2>Demonstrating the Box Model</h2>
<p>The CSS box model is essentially a box that wraps around every HTML element. It consists of: borders, padding, margins, and the actual content.</p>
<div>This text is the content of the box. We have added a 50px padding, 20px margin and a 15px green border. </div>
</body>
</html>
```

Output:



3d. Write a CSS rule that places a background image halfway down the page, tilting it horizontally. The image should remain in place when the user scrolls up or down.

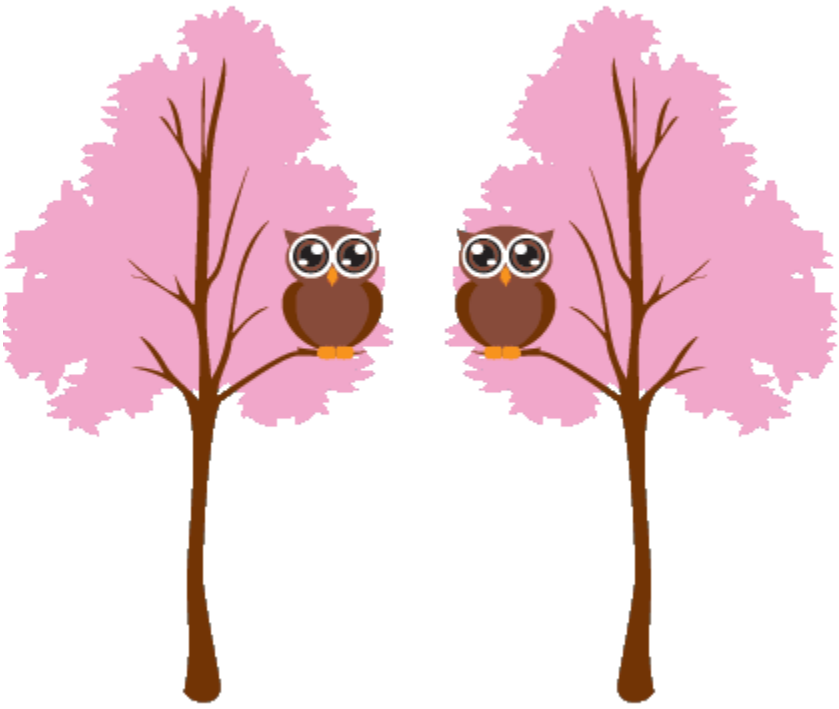
Program:

```
<!DOCTYPE html>
<html>
<head>
<style>
img {
  -webkit-box-reflect: right 20px;
}
</style>
</head>
<body>
<h1>CSS Image Reflection</h1>
<p>Show the reflection below the image, with a 20 pixels offset:</p>
<img src="./images/img_tree.gif">
</body>
</html>
```

Output:

CSS Image Reflection

Show the reflection below the image, with a 20 pixels offset:



3e. Write a program using the following terms related to CSS:

i. font-size ii. font-weight iii. font-strech

iv. text-decoration v. text-transformation vi. text-alignment

Description:

Font Properties in CSS:

- Using a font that is easy to read is important.
- The font adds value to your text. It is also important to choose the correct color and text size for the font.
- In CSS there are five generic font families:
- **Serif** fonts have a small stroke at the edges of each letter. They create a sense of formality and elegance.
- **Sans-serif** fonts have clean lines (no small strokes attached). They create a modern and minimalistic look.
- **Monospace** fonts - here all the letters have the same fixed width. They create a mechanical look.
- **Cursive** fonts imitate human handwriting.
- **Fantasy** fonts are decorative/playful fonts.
- All the different font names belong to one of the generic font families
- In CSS, we use the font-family property to specify the font of a text.
- The font-family property should hold several font names as a "fallback" system, to ensure maximum compatibility between browsers/operating systems.
- Start with the font you want, and end with a generic family (to let the browser pick a similar font in the generic family, if no other fonts are available).
- The font names should be separated with comma.
- font-family: *family-name*|*generic-family*|initial|inherit;
- Font-family:Georgia|Palatino Linotype|Book Antiqua|Times New Roman|Arial|Helvetica|Arial Black|Impact|Lucida Sans Unicode|Tahoma|Verdana|Courier New|Lucida Console|initial
- In a small-caps font, all lowercase letters are converted to uppercase letters.
- However, the converted uppercase letters appears in a smaller font size than the original uppercase letters in the text.
- The font-variant property specifies whether or not a text should be displayed in a small-caps font.
- font-variant: normal|small-caps|initial|inherit;
- The font-variant-caps property controls the usage of alternate glyphs for capital letters.
- font-variant-caps: normal|small-caps|all-small-caps|petite-caps|all-petite-caps|unicase|titling-caps|initial|inherit|unset;
- The font-size property sets the size of a font.
- font-size:medium|xx-small|x-small|small|large|x-large|xx-large|smaller|larger|*length*|%|initial|inherit;
- The font-style property specifies the font style for a text.

- font-style: normal|italic|oblique|initial|inherit;
- The font-weight property sets how thick or thin characters in text should be displayed.
- font-weight: normal|bold|bolder|lighter|*number*|initial|inherit;

Text Properties in CSS:

- CSS has a lot of properties for formatting text
- The color property is used to set the color of the text. The color is specified by:
- a color name - like "red"
- a HEX value - like "#ff0000"
- an RGB value - like "rgb(255,0,0)"
- Look at CSS Color Values for a complete list of possible color values.
- The default text color for a page is defined in the body selector.
- Text-color/color:colorname/rgb/hex/hsl|initial|inherit;
- The text-align property is used to set the horizontal alignment of a text.
- A text can be left or right aligned, centered, or justified.
- When the text-align property is set to "justify", each line is stretched so that every line has equal width, and the left and right margins are straight (like in magazines and newspapers)
- Text-align: left|right|center|justify|initial|inherit;
- The text-decoration property is used to set or remove decorations from text.
- The value text-decoration: none; is often used to remove underlines from links
- The other text-decoration values are used to decorate text
- Text-decoration: *text-decoration-line text-decoration-color text-decoration-style*|initial|inherit;
- The text-decoration-line property sets the kind of text decoration to use (like underline, overline, line-through).
- Text-decoration-line:none|underline|overline|line-through|initial|inherit;
- The text-decoration-style property sets the style of the text decoration (like solid, wavy, dotted, dashed, double).

text-decoration-style: solid|double|dotted|dashed|wavy|initial|inherit;

- The text-transform property is used to specify uppercase and lowercase letters in a text.
- It can be used to turn everything into uppercase or lowercase letters, or capitalize the first letter of each word
- Text-transform:none|capitalize|uppercase|lowercase|initial|inherit;
- The text-indent property is used to specify the indentation of the first line of a text
- Text-indent:px|pts|cm|em|initial|inherit;
- The direction property specifies the text direction/writing direction within a block-level element.
- direction: ltr|rtl|initial|inherit;
- The text-overflow property specifies how overflowed content that is not displayed

should be signaled to the user. It can be clipped, display an ellipsis (...), or display a custom string.

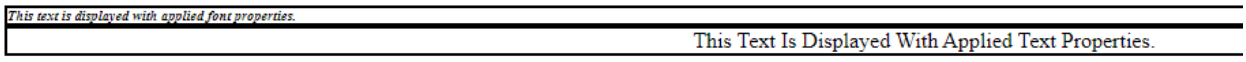
- Both of the following properties are required for text-overflow:
- text-overflow: clip|ellipsis|*string*|initial|inherit;
- The letter-spacing property is used to specify the space between the characters in a text.
- Letter spacing can be given either a positive or –ve value.
- Letter-spacing:+-normal|px|pts|cm|em|initial|inherit;
- The line-height property is used to specify the space between lines as floating point value
- Line-height:normal|value|length|initial|inherit;
- The word-spacing property is used to specify the space between the words in a text
- word-spacing: normal|px|pts|cm|em|inherit;
- The white-space property specifies how white-space inside an element is handled.
- White-space: normal|nowrap|pre|pre-line|pre-wrap|initial|inherit;
- The unicode-bidi property is used together with the direction property to set or return whether the text should be overridden to support multiple languages in the same document.
- unicode-bidi: normal|embed|bidi-override|initial|inherit;
- The vertical-align property sets the vertical alignment of an element.
- vertical-align: baseline|*length*|sub|super|top|text-top|middle|bottom|text-bottom|initial|inherit;
- The text-shadow property adds shadow to text.
- In its simplest use, you only specify the horizontal shadow (2px) and the vertical shadow (2px)
- text-shadow: *h-shadow v-shadow blur-radius color*|none|initial|inherit;

Program:

```
<html>

<head>

<style>

.font{

font-family:Times;

font-style:oblique;

font-weight:900;

font-size:65%;

font-stretch:expanded;
```

```
border:2px solid;
}
.text{
text-align:center;
text-decoration:overline;
text-transform:capitalize;
border:2px solid;
}
</style>
</head>
<body>
<section class="font">
This text is displayed with applied font properties.
</section>
<div class="text">
This text is displayed with applied text properties.
</div>
</body>
</html>
```

Output:



4a Write a JavaScript program, that makes use of different objects (predefined – Array, String, Math, Date, RegExp and User-defined). Note: Use atleast 5 properties and methods from each object.

Description:

- **JavaScript array** is an object that represents a collection of similar type of elements.
- There are 3 ways to construct array in JavaScript
- By array literal
- By creating instance of Array directly (using new keyword)
- By using an Array constructor (using new keyword)
- JavaScript array literal
- The syntax of creating array using array literal is given below:
- var arrayname=[value1,value2.....valueN];
- As you can see, values are contained inside [ ] and separated by , (comma).
- JavaScript Array directly (new keyword)
- The syntax of creating array directly is given below:
- var arrayname=new Array();
- Here, **new keyword** is used to create instance of array.
- JavaScript array constructor (new keyword)
- Here, you need to create instance of array by passing arguments in constructor so that we don't have to provide value explicitly.
- The length Property
- The length property of an array returns the length of an array (the number of array elements).

- Syntax: arrayname.length
- Accessing the First and Last Array Elements
- Syntax: arrayname[0]//for first element
- Syntax:arrayname[arrsyname.length-1];//for last element
- We can use the Array.forEach() function to traverse array elements
- Methods in Math object are used for manipulation of numbers and to perform any common mathematical calculations.
- It contains many rounding methods like floor value, ceil value, round value and many trigonometric functions like sin, cos and tan functions and other functions like max, min etc…
- The web content is to be displayed on the web page in string form.
- Java script provides many string functions to process these string objects.
- A string is a collection of objects;these may include any kind of special characters, digits, normal characters.
- The String object is used to manipulate a stored piece of text.
- String manipulation can be done & generate HTML markup methods
- There are 2 ways to create string in JavaScript
- By string literal
- By string object (using new keyword)
- 1) By string literal
- The string literal is created using double quotes. The syntax of creating string using string literal is given below:
- var stringname="string value";
- By string object (using new keyword)
- The syntax of creating string object using new keyword is given below:
- var stringname=new String("string literal");
- Here new keyword is used to create instance of a string.

- The **JavaScript date** object can be used to get year, month and day.
- You can display a timer on the webpage by the help of JavaScript date object.
- You can use different Date constructors to create date object.
- It provides methods to get and set day, month, year, hour, minute and seconds.
- You can use 4 variant of Date constructor to create date object.
- Date()
- Date(milliseconds)
- Date(dateString)
- Date(year, month, day, hours, minutes, seconds, milliseconds)
- A regular expression is a sequence of characters that forms a **search pattern**.
- When you search for data in a text, you can use this search pattern to describe what you are searching for.
- A regular expression can be a single character, or a more complicated pattern.
- Regular expressions can be used to perform all types of **text search** and **text replace** operations.
- Syntax
- /*pattern*/*modifiers*;
- In JavaScript, regular expressions are often used with the two **string methods**: search() and replace().
- The search() method uses an expression to search for a match, and returns the position of the match.
- The replace() method returns a modified string where the pattern is replaced.
- The test() method is a RegExp expression method.
- It searches a string for a pattern, and returns true or false, depending on the result.
- The exec() method is a RegExp expression method.
- It searches a string for a specified pattern, and returns the found text as an object.
- If no match is found, it returns an empty *(null)* object.

- toString() returns the string representation of regular expression.

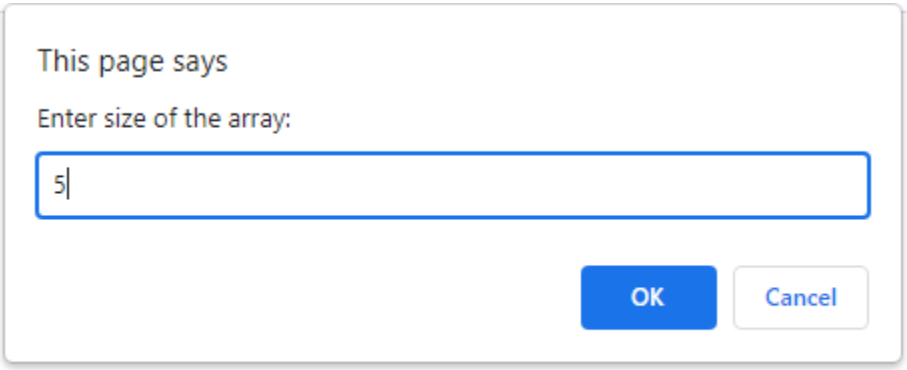
Programs:

Arraydemo.html

```
<html>
<head>
<script type="text/javascript">
var arr = [];
var size = window.prompt("Enter size of the array:");
for(var a=0; a<size; a++)
{
    arr[a] = parseInt(prompt('Enter array Element ' + (a+1)));
}
document.writeln("Array generated is:"+arr);
arr.push(10);
document.writeln("Array after pushing is:"+arr);<br>
arr.pop();
document.writeln("Array after popping is:"+arr);
arr.shift();
document.writeln("Array after shifting is:"+arr);
delete arr[2];
document.writeln("Array after deleting is:"+arr);
arr.splice(3,0,3,4,5);
document.writeln("Array after splicing is:"+arr);
var a1=arr.slice(1,3);
document.writeln("Array created by splicing is:"+a1);
```

```
document.writeln("Length of the array is:"+arr.length);
Array.prototype.myprot = function() {
  this.sort();
};
arr.myprot();
document.writeln("Array after sorting is:"+arr);
</script>
</head>
</html>
```

Output:



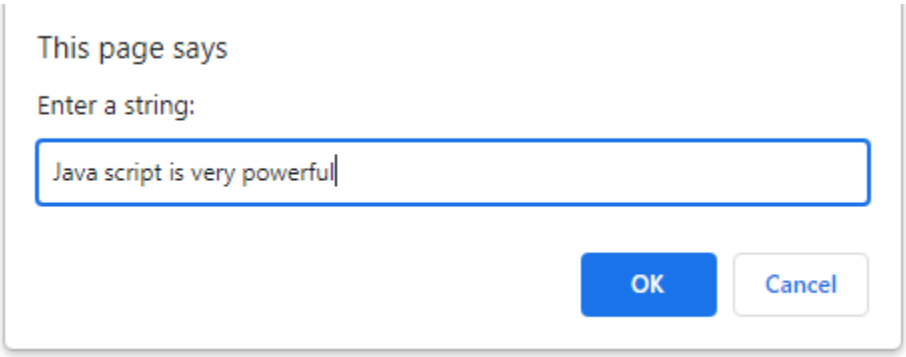
Array generated is:1,2,3,4,5 Array after pushing is:1,2,3,4,5,10 Array after popping is:1,2,3,4,5 Array after shifting is:2,3,4,5 Array after deleting is:2,3,,5 Array after splicing is:2,3,,3,4,5,5 Array created by splicing is:3, Length of the array is:7 Array after sorting is:2,3,3,4,5,5,

Stringdemo.html:

```
<html>
<head>
<script type="text/javascript">
var s="";
var s = window.prompt("Enter a string:");
```

```
document.writeln("String generated is:"+s);
document.writeln("Character at position 1 is:"+s.charAt(1));
document.writeln("Index of a is:"+s.indexOf('a'));
document.writeln("Pattern matched at:"+s.match("Java"));
var s1=s.slice(1,3);
document.writeln("String created by splicing is:"+s1);
var s2=s.substr(1,3);
document.writeln("String created by slicing is:"+s2);
document.writeln("Length of the string is:"+s.length);
</script>
</head>
</html>
```

Output:



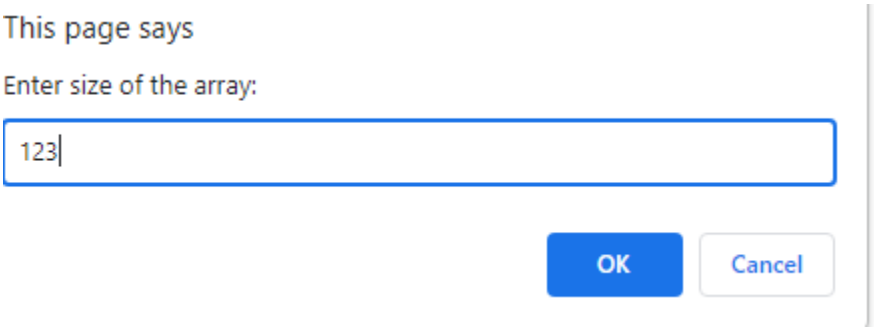
String generated is:Java script is very powerful Character at position 1 is a Index of a is:1 Pattern matched at:Java String created by splicing is:av String created by slicing is:ava Length of the string is:28

Mathdemo.html:

```
<html>
<head>
```

```
<script type="text/javascript">
var n = parseInt(window.prompt("Enter a number:"));
document.writeln("Ceil Value is:"+Math.ceil(n));
document.writeln("Floor value is:"+Math.floor(n));
var l=Math.random();
document.writeln("random value is:"+l);
document.writeln("sine value is:"+Math.sin(n));
document.writeln("Tan value is:"+Math.tan(n));
document.writeln(Log N base 2 value is"+Math.LN2);
document.writeln(Log N base 10 value is"+Math.LN10);
document.writeln(base 2 logarithm of E value is"+Math.LOG2E);
document.writeln(base 10 logarithm of E value is"+Math.LOG10E);
</script>
</head>
</html>
```

Output:



Datedemo.html:

```
<html>
<head>
<script type="text/javascript">
```

```
 var dt = new Date( "April 11, 2022 08:30:00");
document.writeln(dt);
dt.setDate(10);
document.writeln("Changed date is:"+dt);
document.writeln("Hours are:"+dt.getHours());
document.writeln("Year is:"+dt.getFullYear());
document.writeln("Day is:"+dt.getDay());
</script>
</head>
</html>
```

Output:

Mon Apr 11 2022 08:30:00 GMT+0530 (India Standard Time) Changed date is:Sun Apr 10 2022 08:30:00 GMT+0530 (India Standard Time) Hours are:8 Year is:2022 Day is:0

Regularexpressiondemo.html:

```
<html>
  <head>
    <title>JavaScript RegExp exec Method</title>
  </head>
  <body>
    <script type = "text/javascript">
      var str = "Javascript is an interesting scripting language";
      var re = new RegExp( "script", "g" );
      var result1 = re.exec(str);
      document.write("Test 1 - returned value : " +  result1);
      var re1 = new RegExp( "script", "g" );
```

```
        var result2 = re1.test(str);
        document.write("Test 2 - returned value : " +  result2);
var re2 = new RegExp( "script", "g" );
        var patt = new RegExp("Hello World", "g");
  var res = patt.toString();
        document.write("Test 3 - returned value : " +  res);
      </script>
   </body>
</html>
```

Output:

```
Test 1 - returned value : scriptTest 2 - returned value : trueTest 3 - returned value : /Hello World/g
```

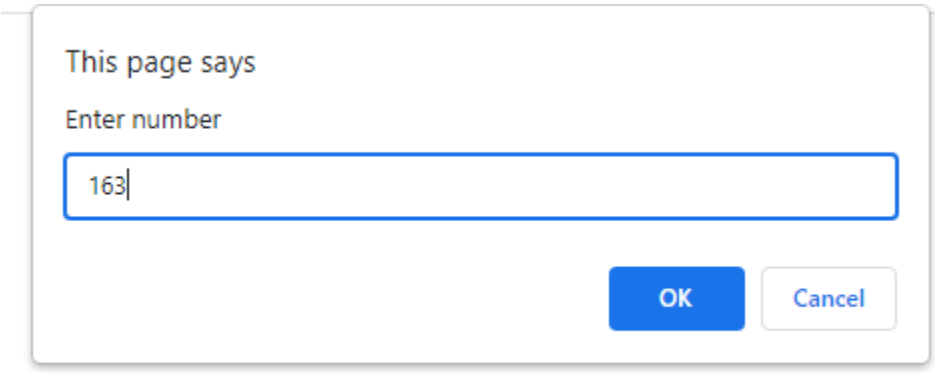
4b Write a javascript to display the denomination of the amount deposited in the bank in terms of 100's, 50's, 20's, 10's, 5's, 2's & 1's. (Eg: If deposited amount is Rs.163, the output should be 1-100's, 1-50's, 1- 10's, 1-2's & 1- 1's)

Program:

```
<html>
<head><title>Display the Denomination</title></head>
<body><script>
a=prompt("Enter number"," ");
n=parseInt(a);
h=Math.floor(n/100);
n=n%100;
f=Math.floor(n/50);
n=n%50;
tw=Math.floor(n/20);
n=n%20;
t=Math.floor(n/10);
n=n%10;
fi=Math.floor(n/5);
n=n%5;
two=Math.floor(n/2);
n=n%2;
one=Math.floor(n/1);
```

```
document.write("Hundreds="+h+"<br>Fifties="+f+"<br>Twenties="+tw+"<br>Tens="+
t+"<br>Fives="+fi+"<br>Twos="+two+"<br>Ones="+one);

</script>

</body></html>
```

Output:



Hundreds=1
Fifties=1
Twenties=0
Tens=1
Fives=0
Twos=1
Ones=1

4c Write a JavaScript that takes a number from one text field in the range of 0-999 and display it in other text field in words. If the number is out of range, it should show “out of range” and if it is not a number, it should show “not a number” message in the result box.

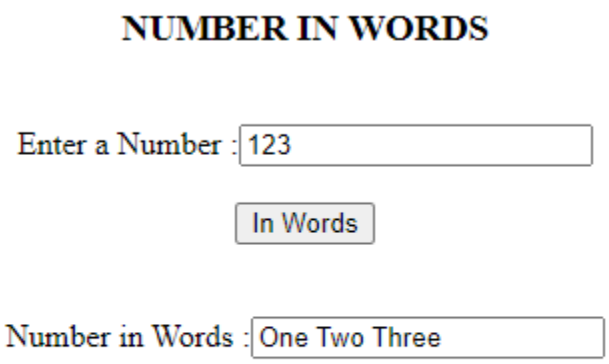
Program:

```
<html>
<head>
<title>Number in words</title>
<script language="javascript">
function convert()
{
var num=document.forms["frm1"].num.value;
document.forms["frm1"].words.value="";
if(isNaN(num))
{
alert("Not a Number");
}
else if (num<0 || num>999)
{
alert("Out of Range");
}
else
{
```

```
var len=num.length;
var words="";
for(var i=0;i<len;i++)
{
var n=num.substr(i,1);
switch(n)
{
case '0':words+="Zero ";break;
case '1':words+="One ";break;
case '2':words+="Two ";break;
case '3':words+="Three ";break;
case '4':words+="Four ";break;
case '5':words+="Five ";break;
case '6':words+="Six ";break;
case '7':words+="Seven ";break;
case '8':words+="Eight ";break;
default:words+="Nine ";
}
}
document.forms["frm1"].words.value=words;
}
}
</script>
</head>
<body>
```

```
<form name="frm1">

<center><h3>NUMBER IN WORDS</h3></center>

<br/>

<center>Enter a Number :<input type="text" name="num"</input><br/></center>

<br/>

<center><input type="button"  name="inwords" value="In Words"
onclick="convert()"></input></center>

<br/><br/><center>Number in Words :<input type="text"
name="words"</input></center>

<br/>

</form>

</body>

</html>
```

Output:



4d Write a JavaScript program, that makes use of class, object, method, constructor for student information system.

Program:

```
<html>
  <meta charset="utf-8"/>
<body> Demonstrating creation of objects with the help of class</br>
   <script type="text/javascript">


        class Student{
      name;
      getStudentName(){
        document.write(this.name);
      }


     }


     const student1 = new Student();
     student1.name="Arjun";
     const student2 = new Student();
     student2.name="Jagadeesh";
     const student3 = new Student();
```

```
    student3.name="Lakshay";
    student1.getStudentName();
    student2.getStudentName();
    student3.getStudentName();

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

Output:

Demonstrating creation of objects with the help of class
ArjunJagadeeshLakshay

5a Design a HTML having a text box and four buttons named Factorial, Fibonacci, Prime, and Palindrome. When a button is pressed an appropriate javascript function should be called to display

i. **Factorial of that number**
ii. **ii. Fibonacci series up to that number**
iii. **iii. Prime numbers up to that number**
iv. **iv. Is it palindrome or not**

Program:

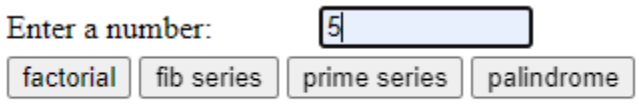
```
<html>
<head>
<title>Javascript Demo</title>
<script type="text/javascript">
function fact(){
var n=parseInt(document.getElementById("n").value);
var fact=1;
if(n==0){
fact=1;
}
else{
for(i=1;i<=n;i++){
fact*=i;
}
```

```
}
document.write("Factorial value is "+fact);
}
function fib(){
var n=parseInt(document.getElementById("n").value);
var n1 = 0,  n2 = 1, next_num, i;
document.write( "Fibonacci Series: ");
for ( i = 1; i <= n; i++)
{  document.write (" <br> " +  n1);
   next_num = n1 + n2;
   n1 = n2;
   n2 = next_num;
}
}
function prime(){
var n=parseInt(document.getElementById("n").value);
var store  = [], i, j, primes = [];
   for (i = 2; i <= n; ++i)
   {
      if (!store [i])
        {
          primes.push(i);
          for (j = i << 1; j <= n; j += i)
          {
             store[j] = true;
```

```
            }
        }
    }
    document.write(primes);
}
function pal(){
var n=parseInt(document.getElementById("n").value);
var rem, temp, final = 0;
temp = n;
while(n>0)
{
rem = n%10;
n = parseInt(n/10);
final = final*10+rem;
}
if(final==temp)
{
document.write("The inputed number is Palindrome");
}
else
{
document.write("The inputted number is not palindrome");
}
}
</script>
```

```
</head>
<body>
<table align="center">
<caption="Formdemo">
<form name="f1" method="post">
<tr>
<td>Enter a number: </td>
<td><input type="text" id="n" name="num" size="10" maxlength="15" required></td>
</tr>
<tr>
<td colspan=4><input type="submit" value="factorial" onclick="fact()">
<input type="submit" value="fib series" onclick="fib()">
<input type="submit" value="prime series" onclick="prime()">
<input type="submit" value="palindrome" onclick="pal()"></td>
</tr>
<tr>
</body>
</html>
```

Output:



Factorial value is 120

Fibonacci Series:
0
1
1
2
3

2,3,5

Enter a number: 121

factorial | fib series | prime series | palindrome

The inputed number is Palindrome

5b Write JavaScript programs on Event Handling

i. **Open a Window from the current window**
ii. **Change color of background at each click of button or refresh of a page**
iii. **Display calendar for the month and year selected from combo box**
iv. **On Mouse over event**

Description:

Even	Attrib	Meaning	Associated Tags
blur	Onblur	Losing focus	<button> , <input> <textarea>,<a>
change	Onchange	On occurrence of Change	<input> , <textarea>,<select>
click	Onclick	When user clicks m button	<input> .<a>
dbclick	Ondbclick	When user double clicks mouse button	<input> .<a>, <button>
focus	Onfocus	When user acquires input focus	<a>, <input> , <textarea>,<select>
keyup	Onkeyup	When user releases key from keyboard	Form elements
keydown	Onkeydow	When user presses k Down	Form elements
keypress	Onkeypres	When user presses k	Form elements
mousedo	Onmoused	When user clicks le mouse button	Form elements
mouseup	Onmouseu	When user releases mouse button	Form elements
mouseme	Onmousem	When user moves mouse	Form elements
mouseou	Onmouseo	User moves mouse away from some element	Form elements
mouseov	Onmouseo	User moves mouse away over some element	Form elements
load	Onload	After getting docum Loaded	<body>
reset	Onreset	When reset button	<form>

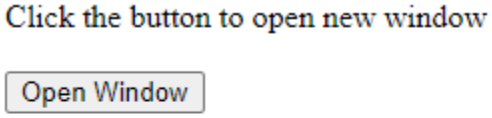
		clicked	
submit	Onsubmit	Submit button clicke	<form>
select	Onselect	On selection	<input>, <textarea>
unload	Onunload	User exits the docun	<body>

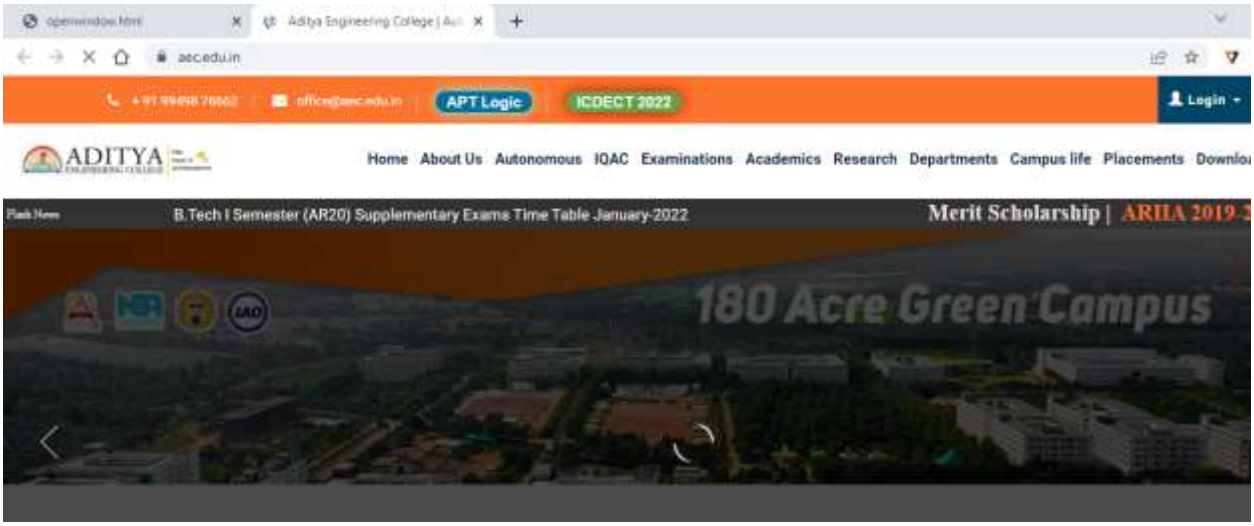
Programs:

Openwindow.html:

```
<html>

<body>

<script>

function openWindow() {

window.open('https://www.aec.edu.in');

}

</script>

Click the button to open new window <br><br>

<button onclick="openWindow()"> Open Window </button>

</body>

</html>
```

Output:



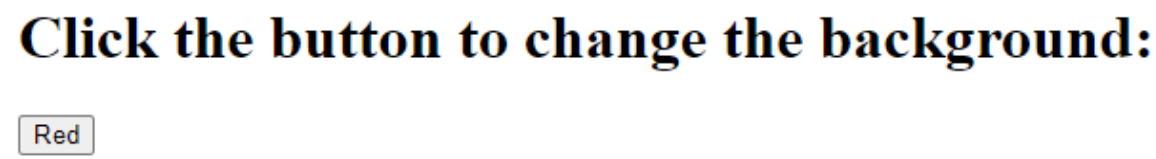


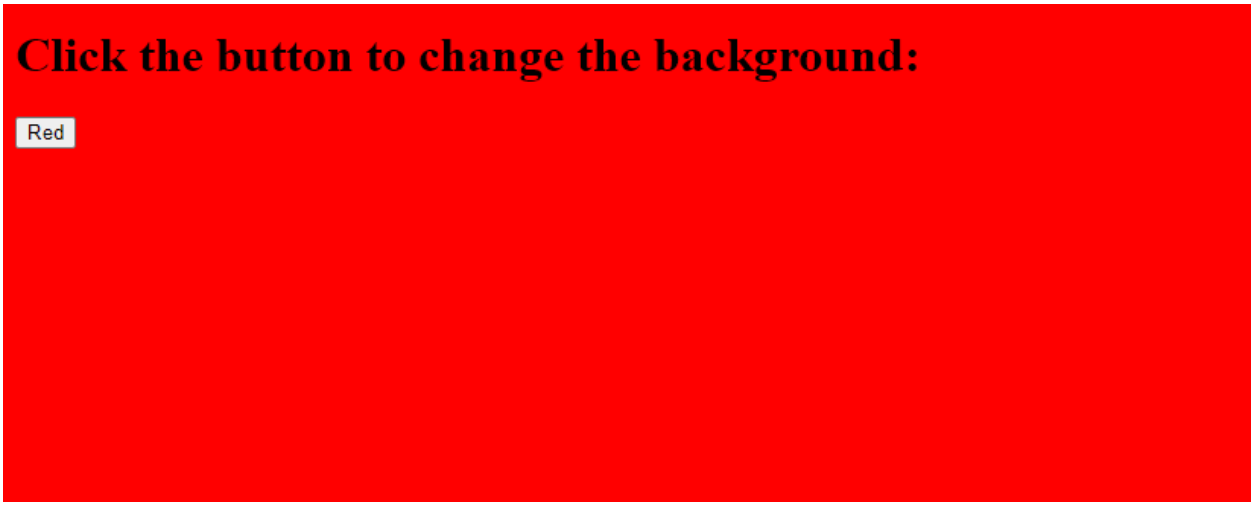
Onclickbutton.html:

```
<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="utf-8" />
    <title>JavaScript change background color</title>
    <script>
      function changeBackgroundRed() {
        document.body.style.background = "red";
      }
    </script>
  </head>
  <body>
    <h1>Click the button to change the background:</h1>
    <button onclick="changeBackgroundRed();">Red</button>
  </body>
```

</html>

Output:





Onrefreshdemo.html:

<html>

<head>

<script type="text/javascript">

var color = new Array();

color[0] = "#CC99FF";

color[1] = "#FF99CC";

color[2] = "#FF9999";

color[3] = "#FFCC99";

color[4] = "#FFFF99";

```
color[5] = "#CCFF99";
color[6] = "#99FF99";
color[7] = "#99FFCC";
color[8] = "#66FFFF";
color[9] = "#66CCFF";
function changeColor()
{
  var randomColor = Math.floor(Math.random() * color.length);
  document.body.style.backgroundColor = color[randomColor];
}
</script>
</head>
<body onload="changeColor()">
</body>
</html>
```

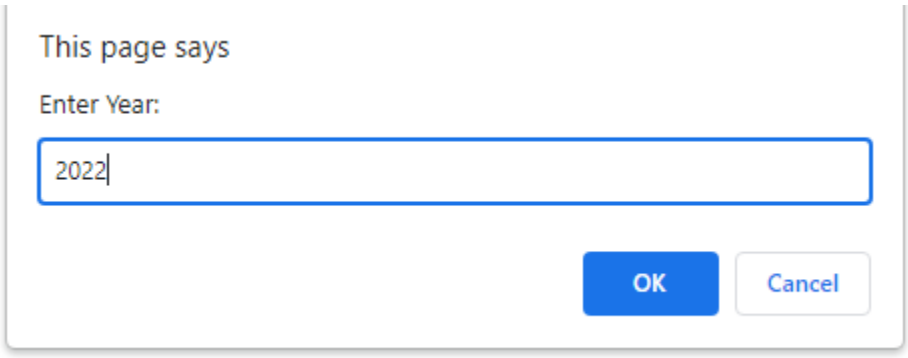
Output:

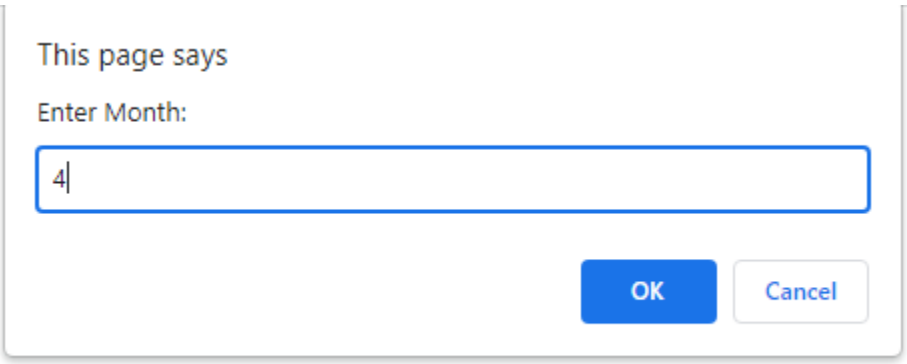
Calendardemo.html:

```
<!DOCTYPE HTML>
<html>

<head>
  <style>
    table {
      border-collapse: collapse;
    }

    td,
    th {
      border: 1px solid black;
      padding: 3px;
      text-align: center;
    }

    th {
      font-weight: bold;
      background-color: #E6E6E6;
    }
  </style>
</head>
<body>
  <div id="calendar"></div>
  <script>
    function createCalendar(elem, year, month) {
      let mon = month - 1; // months in JS are 0..11, not 1..12
      let d = new Date(year, mon);
      let table = '<table><tr><th>MO</th><th>TU</th><th>WE</th><th>TH</th><th>F
R</th><th>SA</th><th>SU</th></tr><tr>';
      for (let i = 0; i < getDay(d); i++) {
        table += '<td></td>';
      }
      while (d.getMonth() == mon) {
        table += '<td>' + d.getDate() + '</td>';
        if (getDay(d) % 7 == 6) {
          table += '</tr><tr>';
        }
        d.setDate(d.getDate() + 1);
      }
      if (getDay(d) != 0) {
        for (let i = getDay(d); i < 7; i++) {
          table += '<td></td>';
        }
      }
```

```
    table += '</tr></table>';
    elem.innerHTML = table;
  }
  function getDay(date) {
    let day = date.getDay();
    if (day == 0) day = 7;
    return day - 1;
  }
  var year=parseInt(prompt("Enter Year:"));
  var month=parseInt(prompt("Enter Month:"));
  createCalendar(calendar, year, month);
</script>
</body>
</html>
```

Output:





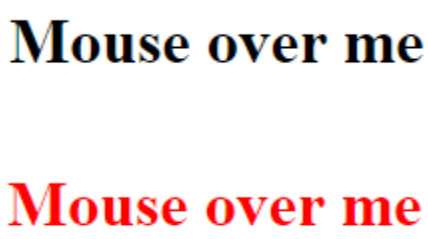
MO	TU	WE	TH	FR	SA	SU
				1	2	3
4	5	6	7	8	9	10
11	12	13	14	15	16	17
18	19	20	21	22	23	24
25	26	27	28	29	30	

Onmouseoverdemo.html:

Program:

```
<!DOCTYPE html>
<html>
<body>
<h1 id="demo">Mouse over me</h1>
<script>
document.getElementById("demo").addEventListener("mouseover", mouseOver);
document.getElementById("demo").addEventListener("mouseout", mouseOut);
function mouseOver() {
  document.getElementById("demo").style.color = "red";
}
function mouseOut() {
  document.getElementById("demo").style.color = "black";
}
</script>
</body>
</html>
```

Output:



5c Write a JavaScript to validate the following fields in a registration page created in Experiment 2

i. **Name (start with alphabet and followed by alphanumeric and the length should not be less than 6 characters)**
ii. **Password (it allows alphanumeric, special symbols and should not be less than 6 characters)**
iii. **E-mail (should not contain invalid email addresses)**

Program:

```
<html>

<head>

<title>Formdemo</title>

<script type="text/javascript">

function validate(){

var name=document.getElementById("un").value;

var pass=document.getElementById("pwd").value;

var email=document.getElementById("em").value;

var re = /^\w+([\.-]?\w+)*@\w+([\.-]?\w+)*(\.\w{2,3})+$/;

if(name.test(/^[a-zA-Z0-9]/)==false){

window.alert("Please enter a valid name");

}

else if(name.length<6){

window.alert("Username must be atleast 6 characters long");

}

else if(pass.test(/^(?=.*\d)(?=.*[a-z])(?.=*[A-Z])(?.=*[a-zA-Z0-
9])(?.*\s).{6,20}$/)==false){

window.alert("Password should have 1 lowercase, 1 uppercase, 1 digit, 1 special
character and it should be greater than 6 characters");
```

```
}
else if (re.test(email)==false) {
        alert("Please enter a valid email");

}

}
</script>
</head>
<body>
<table align="center">
<caption="Formdemo">
<form name="f1" method="post">
<tr>
<td>Username: </td>
<td><input type="text" id="un" name="uname" size="10" maxlength="15" required></td>
</tr>
<tr>
<td>Password:</td>
<td><input type="password" id="pwd" name="pwd" size="10" maxsize="15" required></td>
</tr>
<tr>
<td>Email </td>
<td><input type="text" id="em" name="uname" required></td>
```

```
</tr>
<tr>
<td>Address:</td>
<td><textarea name="ta" rows="5" cols="10"></textarea></td>
</tr>
<tr>
<td>Languages known:</td>
<td><input type="checkbox" name="c1" value="Te" checked>Telugu</br>
<input type="checkbox" name="c2" value="Ta">Tamil</br>
<input type="checkbox" name="c3" value="Hi">Hindi</br>
<input type="checkbox" name="c4" value="En">English</br>
</td>
</tr>
<tr>
<td>Programming Proficiency:</td>
<td>
<input type="radio" name="r1" value="C">C</br>
<input type="radio" name="r1" value="Java">Java</br>
<input type="radio" name="r1" value="Python" checked>Python</br>
<input type="radio" name="r1" value="PHP">PHP</br>
</td>
</tr>
<tr>
<td>Willing to work at:</td>
<td>
```

```
<select name="s1" multiple>
<option value="Mu">Mumbai</option>
<option value="ch" label="Chennai">Chennai</option>
<option value="Hy">Hyderbad</option>
<option value="viz" selected>Vizag</option>
</select>
</td>
</tr>
<tr>
<td colspan="2"><input type="submit" value="submit" onclick="validate()">
<input type="reset" value="reset">
</td>
</tr>
</form>
</table>
</body>
</html>
```