

# BACHELOR OF COMPUTER APPLICATION LAB MANUAL

1st Semester



Prepared By

**Pure and Applied Science Dept.**

Computer Application

## MIDNAPORE CITY COLLEGE

```
hidden.bs.select", function() {  
    $.closest(".ub_select_el").ub_  
    console.log("hidden")  
}, function() {  
    $.on("shown.bs.select", function() {  
        $.addClass("shown")  
        console.log("shown")  
    });  
    $.find("select");  
    $.container  
    function(f) {  
        $.first();  
    }  
});
```

## INSTRUCTIONS TO STUDENTS

- Before entering the lab, the student should carry the following things (MANDATORY)

1. Identity card issued by the college.
2. Class notes
3. Lab observation book
4. Lab Manual
5. Lab Record

- Student must sign in and sign out in the register provided when attending the lab session without fail.
- Come to the laboratory in time. Students, who are late more than 10 min., will not be allowed to attend the lab.
- Students need to maintain 80% attendance in lab if not a strict action will be taken.
- All students must follow a Dress Code while in the laboratory.
- Foods, drinks are NOT allowed.
- All bags must be left at the indicated place.
- Refer to the lab staff if you need any help in using the lab.
- Respect the laboratory and its other users.
- Workspace must be kept clean and tidy after experiment is completed.
- Read the Manual carefully before coming to the laboratory and be sure about what you are supposed to do.
- Do the experiments as per the instructions given in the manual.
- Copy all the programs to observation which are taught in class before attending the lab session.
- Students are not supposed to use floppy disks, pen drives without permission of lab- in charge.
- Lab records need to be submitted on or before the date of submission.

***SEC01: WEB DESIGNING  
LABORATORY MANUAL  
(Course Code: BCASEC01P)***

## ***HTML***

### ***Hyper Text Markup Language (HTML):***

- ✓ The language used to develop web pages is called Hyper Text Markup Language (HTML).
- ✓ HTML is a combination of both hypertext and markup language.
- ✓ HTML describes the structure of a Web page.
- ✓ HTML consists of a series of elements.
- ✓ HTML elements label pieces of content such as "this is a heading", "this is a paragraph", "this is a link", etc.
- ✓ HTML is specified as TAGS in an HTML document (i.e., the Web page).
- ✓ HTML is the language interpreted by a Browser.
- ✓ HTML was created by Tim Berners-Lee in 1991. The first-ever version of HTML was HTML 1.0, but the first standard version was HTML 2.0, published in 1995. and the latest version is HTML 5. We can save HTML files with an extension .html or .htm.
- ✓ Web Pages are also called HTML documents HTML is a set of special codes that can be embedded in text to add formatting and linking information.

### ***Hyper Text:***

Hyper Text simply means Text within Text. A text has a link within it, is a hypertext. Whenever you click on a link which brings you to a new webpage, you have clicked on a hypertext. Hyper Text is a way to link two or more web pages (HTML documents) with each other.

### ***Markup Language:***

Markup Language is a language that is interpreted by the browser and it defines the elements within a document using tags. It is human-readable, which means that markup files use common words rather than the complicated syntax of programming languages.

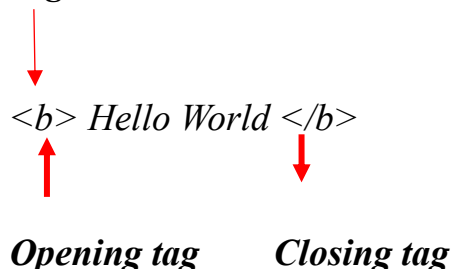
### ***Web Page:***

A web page is a document which is commonly written in HTML and translated by a web browser. A web page can be identified by entering an URL. A Web page can be of the static or dynamic type. *With the help of HTML only, we can create static web pages.*

### ***Elements and Tags:***

HTML uses predefined **tags and elements** which tell the browser how to properly display the content. Remember to include closing tags. If omitted, the browser applies the effect of the opening tag until the end of the page.

### Tag element



### HTML `<html>` Tag:

The `<html>` tag in HTML is used to define **the root of HTML documents**. The `<html>` tag tells the browser that it is an HTML document. It is the second outer container for everything that appears in an HTML document followed by the `<!DOCTYPE>` tag. The `<html>` tag requires a **starting and end tag**.

**Syntax:**     `<html> HTML Contents... </html>`

**Example:**

```

<!DOCTYPE html>
<!-- html tag starts here -->
<html>
  <body>
    <h1>Midnapore City College</h1>
    <h2> <html>Tag</h2>
  </body>
</html>
<!-- html tag ends here -->

```

### HTML tags can be of two types:

#### Paired Tags:

A tag is said to be a paired tag if it, along **with a companion** tag, flanks the text. For example, the `<B>` tag is a paired tag. The `<B>` tag with its companion tag `</B>` causes the text contained between them to be rendered in bold. The effect of other paired tags is applied only to the text they contain.

In paired tags, the **first tag** (`<B>`) is often called the **opening tag** and the **second tag** (`</B>`) is called **the closing tag**.

The opening tag activates the effect and the closing tag turns the effect off.

### ***Singular Tags:***

The second type of tag is the singular or stand-alone tag. A stand-alone tag ***does not have a companion*** tag. For example, <br> tag will insert a line break. This tag does not require any companion tag.

**Note:** Some HTML elements have no content (like the <br> element). These elements are called empty elements. Empty elements do not have an end tag!

### ***HTML Elements:***

An HTML element is a collection of ***start and end tags with the content inserted in between them.***

**Syntax:**     <tagname > Contents... </tagname>

**HTML Element:** The HTML element consists of 3 parts.

- ✓ **Opening tag:** It is used to tell the browser where the content material starts.
- ✓ **Closing tag:** It is used to tell the browser where the content material ends.
- ✓ **Content:** It is the actual content material inside the opening and closing tags.

### ***Essential Tags***

HTML contains four essential tags that form the basic structure of any webpage or HTML file:

1. <html></html>
2. <head></head>
3. <title></title>
4. <body></body>

Now let us discuss each tag one by one:

#### **1. <!DOCTYPE html>**

It is also known as ***document type*** and should be included in an ***HTML file***. It actually tells the browser that this is an HTML document. It is ***not a tag or an element but it is information.***

**Syntax:**                     <!DOCTYPE html>

## 2. <html></html>

This tag marks the *beginning and ending* of the HTML document and whatever code is present in between these tags totally gets considered by the browser. Also, it tells the browser that the document is an HTML document. All the other tags in between these tags only get considered by the browser.

**Syntax:**                *<html> Content </html>*

## 3. <head></head>

This tag stores the data which actually *doesn't appear on the webpage* but it gives more information about the webpage. Or in other words, this tag is used to define the *head part of the document* which contains the information related to the webpage. It also contains tags like, <title>, <meta>, <link>, <style>, etc.

**Syntax:**            *<head> <title> Title of the Webpage </title></head>*

## 4. <title> </title>

This tag stores the *title/name* of the web page. Whatever title/content is given in this tag, the content appears on the tab when opened by the browser. It is described in the head tag.

**Syntax:**                *<title> Title of the Webpage </title>*

## 5. <body></body>

This tag is used to display all the information or data, i.e, text, images, hyperlinks videos, etc., on the webpage to the user. Here, all the content like text, images, hyperlinks videos, etc., are enclosed between this tag.

**Syntax:**                *<body> Content </body>*



## HTML Page Structure

```
<!DOCTYPE html>    ← Tells version of HTML
<html>             ← HTML Root Element

<head>             ← Used to contain page HTML metadata
  <title>Page Title</title> ← Title of HTML page
</head>

<body>             ← Hold content of HTML
  <h2>Heading Content</h2> ← HTML heading tag
  <p>Paragraph Content</p> ← HTML paragraph tag
</body>

</html>
```

Some other HTML tags are:

### 1. <!-- comment -->

This tag is used to add comments in the HTML codes. These comments help the program to understand the code. The content inside the comment tag doesn't visible on the browser.

**Syntax:**                <!--Write comments here -->

### 2. <meta>

These meta tags are used inside the head tag and they making describe the metadata i.e., data about data. These tags are useful in *search engine optimization* which means when users search for our websites the chances that the browser *recommends our webpage becomes high which leads to an increase in traffic over the webpage*. A single HTML document can contain multiple tags.

**Syntax:**                <meta attribute-name="value">



### 3. <link rel = "stylesheet" href= "file.css">

This tag is used to include external style sheets. Use this tag when you don't want to include CSS in the HTML document. To make it more simple we make a CSS file with the code and include this file in the link tag.

**Syntax:**     <link rel = "stylesheet" href= "file.css" >

### 4. <script></script>

It is used for including javascript code. The external javascript can also be linked using the **src** attribute in the opening script tag. It can be included in the head or body tag.

**Syntax:**     <script>script content</script>

### 5. Heading:

HTML provides six types of headings, i.e., H1, H2, H3, H4, H5, and H6. Here, H1 is the highest-level heading and H6 is the lower-level heading. These headings are used to highlight the important topics.

**Syntax:**

<h1> content </h1>

<h2> content </h2>

<h3> content </h3>

<h4> content </h4>

<h5> content </h5>

<h6> content </h6>

### **HTML Editors:**

HTML editor is a software used for writing code in HTML, which is used for structuring and creating websites. Even though codes can be written from scratch using a normal text editor, HTML editors provide a great deal of ease to the developers by ensuring hassle-free coding.

#### ***When you should use an HTML Editor?***

- ✓ Developers prefer to use HTML editors when they want to have a full control over their code and easily create their websites.
- ✓ HTML editors are of great importance to the users who don't have much knowledge of HTML, as of now, and also those who need to generate source codes quickly.
- ✓ HTML editors are highly beneficial for the sake of convenience as they successfully conceal and correct the developers' part of minor errors by syntax correction, auto-completion, simple editing, etc.

#### ***Advantages of using HTML Editors:***

- ✓ They are of great benefit since they allow the users to easily check their syntax, insert commonly used HTML tags and structures and also provide auto-completion.
- ✓ The code generated through an HTML editor can be translated to other languages such as XML, JavaScript, etc. For example-NVU editor provides this translation functionality.
- ✓ Website development can be very exhausting and cumbersome. With the help of online HTML editors, it is possible to create websites with ease and at a faster rate.
- ✓ HTML editors provide full control to the developer, hence helping him to delve deeper into the source code and find the hidden intricacies.
- ✓ HTML editors provide an amicable and aesthetic designing experience.

#### ***Types of HTML Editors:***

There are broadly two types of HTML Editors:

- **Textual HTML Editor**
- **WYSIWYG HTML Editor**

## 1. *Textual HTML Editor*

These are text-based editors where the developers can write their codes and compile them. The code appears in the same manner we write it, thus it requires basic knowledge of HTML. Some of these editors also provide features of making a project, managing all the files related to the web, etc.

Examples of HTML Text editors include-Notepad++, VSCode, Sublime Text.

## 2. *WYSIWYG HTML Editor*

‘What you see is what you get’ is its full form. WYSIWYG are editors that provide the preview of the output of the source code i.e., as it would appear on a browser. There is a drag and drop feature available in most of them that cases the handling. It does not require any hardcore knowledge of HTML, thus enabling non-technical to easily develop websites.

Examples include-Adobe Dreamweaver, Amaya, BlueGriffon, Visual Studio, Sublime Text3 etc.

## **Deprecated Tags**

### ***Deprecated Tags:***

The deprecated tags or attributes are those attributes which are replaced by some other attributes. The tag or attributes deprecated when the same attributes is achieved by some other way. They are considered outdated and may not be supported in modern browsers or future versions of HTML.

### ***HTML Deprecated Tag:***

***Complete list of deprecated tags are given below:***

<b><i>TAGS</i></b>	<b><i>DESCRIPTION</i></b>	<b><i>Alternate Tags</i></b>
applet tag	Specify an applet	object tag
basefont tag	Specify a basefont	font style sheets
center tag	Use to specify a centered Text	text-align:center
dir tag	Specify a directory list	ul tag
embed tag	Embed an application to HTML document	object tag

font tag	Used to specify font text, size and color	font-family, font-size, color
isindex tag	Specify a single-line input field	form tag
menu tag	Specify a menu list	ul tag
plaintext tag	Specify a plaintext	pre tag
s tag	Specify a strike through text	text-decoration
strike tag	Specify a strike through text	text-decoration
u tag	Specify underlined text	text-decoration
xmp tag	Specify preformatted text	pre tag

**HTML Deprecated Attributes:** There are some attributes which are deprecated from HTML4. Some of these attributes are given below:

Attribute	Description	Alternate Attributes
hspace	specify the horizontal space around the element	padding attribute
align attribute	Used to specify the positioning of an element	text-align, vertical-align
alink attribute	Specify color for selected link	active attribute
background attribute	Specify background image	background-image
bgcolor attribute	Specify background color	background-color
bgcolor attribute	Specify background color	background-color
border attribute	Used to specify border width of an element	border-width
height attribute	Specify height of body tag	padding attribute
language attribute	Specify scripting language being used	type attribute
link attribute	Specify default color of links in the document	link attribute
nowrap attribute	Prevent the text from wrapping within that table cell	white-space
vlink attribute	Specify the color of visited links	visited attribute
type attribute	Specify the type of list in li tag	list-style-type
vspace attribute	Specify the amount of whitespace or padding that should appear above or below an element	padding attribute

## **Tags and Attributes**

### **HTML Tags:**

Tags are the starting and ending parts of an HTML element. They begin with < symbol and end with > symbol. Whatever written inside < and > are called tags.

**Example:** <h1> </h1>

### **HTML elements:**

Elements enclose the contents in between the tags. They consist of some kind of structure or expression. It generally consists of a start tag, content and an end tag. Where, <b> is the starting tag and </b> is the ending tag.

**Example:** <h2>This is the content. </h2>

### **HTML Attributes:**

It is used to define the character of an HTML element. It always placed in the opening tag of an element. It generally provides additional styling (attribute) to the element.

**Example:** <p align="center">This is paragraph. </p>

<b>HTML Tags</b>	<b>HTML Elements</b>	<b>HTML Attributes</b>
HTML tags are used to hold the HTML element.	HTML element holds the content.	HTML attributes are used to describe the characteristic of an HTML element in detail.
HTML tag starts with < and ends with >	Whatever written within a HTML tag are HTML elements.	HTML attributes are found only in the starting tag.
HTML tags are almost like keywords where every single tag has unique meaning.	HTML elements specifies the general content.	HTML attributes specify various additional properties to the existing HTML element.

## Text Styles and Text Arrangements

In HTML, you can style text using various elements and CSS (Cascading Style Sheets) properties. Here are some common text styles and arrangements in HTML:

### Font Styles:

#### ➤ **<b> and <strong>: Makes text bold.**

- The HTML `<b>` element defines **bold text**, without any extra importance.

**Example**                      `<b>This text is bold</b>`

- The HTML `<strong>` element defines text with strong importance. The content inside is typically displayed in **bold**.

**Example**                      `<strong>This text is strong </strong>`

#### ➤ **<i> and <em>: Renders text in italics.**

- The HTML `<i>` element defines a part of text in an alternate voice or mood. The content inside is typically displayed in **italic**.

**Example**                      `<i>This text is italic</i>`

- The HTML `<em>` element defines emphasized text. The content inside is typically displayed in italic.

**Example**                      `<em>This text is emphasized</em>`

#### ➤ **<u>: Underlines text.**

- The `<u>` tag represents some text that is unarticulated and styled differently from normal text, such as **misspelled words or proper names** in Chinese text. The content inside is typically displayed with an **underline**.

**Example**                      `<u>mispeled</u>`

➤ **<s>: Renders text with a strikethrough.**

- The <s> tag specifies text that is ***no longer correct***, accurate or relevant. The text will be displayed with a ***line through it***.
- The <s> tag should not be used to define deleted text in a document, use the <del> tag for that.

**Example**

Mark up text that is ***no longer correct***:

```
<p> <s>Only 50 tickets left! </s> </p>
```

```
<p>SOLD OUT! </p>
```

- The <del> tag defines text that has been ***deleted from a document***. Browsers will usually ***strike a line through deleted text***.

**Example**

A text with a deleted part, and a new, inserted part:

```
<p>My favourite color is <del>blue</del> <ins>red</ins>!</p>
```

➤ **<mark>: Highlights text.**

- The <mark> tag defines text that should ***be marked or highlighted***.

**Example**

Highlight parts of a text:

```
<p>Do not forget to buy <mark>milk</mark> today.</p>
```

➤ **<sub>: Displays text as subscript.**

The <sub> tag defines subscript text. Subscript text appears half a character ***below the normal line***, and is sometimes rendered in a smaller font. Subscript text can be used for chemical formulas, like H<sub>2</sub>O.

**Example**

Subscript text:

```
<p>This text contains <sub>subscript</sub> text.</p>
```

➤ **<sup>: Displays text as superscript.**

The <sup> tag defines superscript text. Superscript text appears half a character ***above the normal line***, and is sometimes rendered in a smaller font. Superscript text can be used for footnotes, like WWW<sup>[1]</sup>.

**Example**

Superscript text:

```
<p>This text contains <sup>superscript</sup> text.</p>
```



➤ **<span>: A <span> element which is used to color a part of a text:**

- The <span> tag is an inline container used to mark up a part of a text, or a part of a document.
- The <span> tag is easily styled by CSS or manipulated with JavaScript using the class or id attribute.

**Example:**      <p>My mother has <span style="color:blue; font-weight:bold">blue</span> eyes.

**Text Alignment:**

- CSS properties like text-align can be used to align text within an element.
- We can change the alignment of the text using the text-align property. We can align the text in the center, Left, Right.

Property	Description	Values	Example
text-align	Specifies the horizontal alignment of text or block of text	left(Default)/right/center/justify	text-align: right

Value	Description
✓ left	The text will align to the left
✓ right	The text will align to the right
✓ center	The text will align to the center

**Example:**

<h1 style="text-align:center;">Centered Heading</h1>

**or**

<h1 align="center"> Centered Heading </h1>

**Font Properties:**

- CSS properties like font-family, font-size, font-weight, and font-style can be used to control the font of text.

**Example:** `<p style="font-family: Arial; font-size: 16px; font-weight: bold; font-style: italic;">Custom font styles</p>`

**Note:** The `<font>` tag was used in HTML 4 to specify the font face, font size, and color of text.

The **`<font>` tag** in HTML plays an important role in the web page to create an attractive and readable web page. 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.

**Syntax:** `<font attribute = "value"> Content </font>`

**font Size:** This attribute is used to adjust the size of the text in the HTML document using a font tag with the size attribute. The range of size of the font in HTML is from 1 to 7 and the default size is 3.

**Syntax:** `<font size="number">`

**Example:** `<font size="2">Midnapore City College</font>`

**Font Type:** Font type can be set by using face attribute with font tag in HTML document. But the fonts used by the user need to be installed in the system first.

**Syntax:** `<font face="font_family">`

**Example:** `<font size="2" face="Times New Roman"|"Verdana"|"Comic sans MS">Midnapore City College</font>`

**Font Color:** Font color is used to set the text color using a font tag with the color attribute in an HTML document. Color can be specified either with its name or with its hex code.

**Syntax:** `<font color="color_name|hex_number|rgb_number">`

**Example:** `<font color="green"|#999099|"rgb(0, 153, 0)"> Midnapore City College</font>`

### **Text Decoration:**

CSS properties like text-decoration can be used to add decorations to text, such as underline, overline, or line-through.

**Example:** `<p style="text-decoration: underline;">Underlined text</p>`

### **Line Spacing:**

You can control the spacing between lines of text using the line-height property.

**Example:** `<p style="line-height: 1.5;">This is text with increased line spacing. </p>`

### **Text Shadow:**

The text-shadow property allows you to add a shadow effect to text.

**Example:** `<h1 style="text-shadow: 2px 2px red;">Text with shadow</h1>`

### **Text Transformation:**

CSS properties like text-transform can be used to change the capitalization of text, making it uppercase, lowercase, or capitalize the first letter of each word.

**Example:** `<p style="text-transform: uppercase;">Uppercase text</p>`

## **Text Effects**

### **Exposure to Various Tags (DIV, MARQUEE, NOBR, DFN, HR, LISTING, Comment, IMG)**

#### **DIV Tag:**

- ✓ The div tag is known as **Division tag**.
- ✓ The div tag is used in HTML to make divisions of content in the web page like (text, images, header, footer, navigation bar, etc).
- ✓ Div tag has **both open (<div>) and closing (</div>)** tag and it is mandatory to close the tag.
- ✓ Div tag is **Block level tag**.
- ✓ It is a **generic container tag**.
- ✓ It is used to group various tags of HTML so that sections can be created and styles can be applied to them.
- ✓ Every div tag will **start from a new line**, and not the same line.

**Syntax:** `<div>--- contents ---</div>`

**Example:**

```
<div>I am in div 1</div>
<div>It's 2nd div</div>
```

#### **MARQUEE Tag:**

- ✓ The Marquee HTML tag is a **non-standard HTML** element which is used to scroll an image or text horizontally or vertically.
- ✓ In simple words, you can say that it scrolls the image or text **up, down, left or right** automatically.

- ✓ Marquee tag was first introduced in early versions of Microsoft's Internet Explorer.

**Syntax:**            `<marquee>--- contents ---</marquee>`

**Example:**        `<marquee>This is an example of html marquee </marquee>`

Marquee's element contains several attributes that are used to control and adjust the appearance of the marquee.

<i>Attribute</i>	<i>Description</i>
behavior	It facilitates user to set the behavior of the marquee to one of the three different types: scroll, slide and alternate.
direction	defines direction for scrolling content. It may be left, right, up and down.
width	defines width of marquee in pixels or %.
height	defines height of marquee in pixels or %.
hspace	defines horizontal space in pixels around the marquee.
vspace	defines vertical space in pixels around the marquee.
scrolldelay	defines scroll delay in seconds.
scrollamount	defines scroll amount in number.
loop	defines loop for marquee content in number.
bgcolor	defines background color. It is now <i>deprecated</i> .

### **HTML Scroll Marquee:**

It is a by default property. It is used to scroll the text from *right to left, and restarts at the right side* of the marquee when it is reached to the *end of left side*. After the completion of loop text disappears.

**Example:**

```
<marquee width="100%" behavior="scroll" bgcolor="yellow">
```

This is an example of a scroll marquee...

```
</marquee>
```

### **HTML Slide Marquee:**

In slide marquee, all the contents to be scrolled will slide the entire length of marquee but *stops at the end to display* the content permanently.

**Example:**

```
<marquee width="100%" behavior="slide" bgcolor="yellow">
```

This is an example of a slide marquee...

```
</marquee>
```

**HTML Alternate Marquee:**

It scrolls the text from right to left and goes back left to right.

```
<marquee width="100%" behavior="alternate" bgcolor="pink">
```

This is an example of a alternate marquee...

```
</marquee>
```

**Nested marquee example:**

```
<marquee width="400px" height="100px" behavior="alternate" style="border: 2px solid red">
```

```
<marquee behavior="alternate">
```

Nested marquee...

```
</marquee>
```

```
</marquee>
```

**NOBR Tag**

- ✓ The HTML NOBR <nobr> tag is applied on text to *not break a single line into multiple lines* for users to scroll down to see the whole content. This element must be used in HTML.
- ✓ When text goes outside the screen, the browser will immediately break the text to the next line. If we use the <nobr> tag then it will not permit the browser to break the line.
- ✓ The <nobr> tag is a non-standard element. It works in some browsers but its use is discouraged and can be removed at any time.
- ✓ Use the CSS white-space property instead.

**Note:** <nobr> tag is not supported in html5.

**Syntax:** <nobr> Statement </nobr>

**Attribute:** This tag doesn't contain any attribute.

**Example:** <h2>Nobr Tag Example</h2>

<nobr>Codingtag is the E-learning website covering all aspects of technical and nontechnical tutorials including advanced programming, web Development languages, current affairs and technical interviews question and Answers on C, C++, Python, PHP, CSS, AngularJS, MongoDB and on all latest trending technologies. </nobr>

**DFN Tag**

HTML <dfn> tag also called as HTML definition tag. It is used to represent the term which is defined within context of definition phrase or sentence in an HTML document. The defining instance term usually the first term in a document.

If a term is contained within the <dfn> element then browser understands that nearby text is the definition of the term.

**Syntax:**     <dfn>Content..... </dfn>

**Example 1:** <p><dfn>Midnapore City College</dfn> is a portal for mcc.</p>

**Example 2:** Using title attribute of the <dfn> tag.

```
<p>
    <dfn title=" Midnapore City College ">MCC</dfn>is a portal for
    mcc.
</p>
```

**Example 3:** Using title attribute of the <abbr> tag inside the <dfn> element.

```
<p>
<dfn>
    <abbr title=" Midnapore City College ">MCC</abbr>

</dfn> is a portal for mcc.
</p>
```

**HR Tag**

The <hr> tag in HTML stands for horizontal rule and is used to insert a *horizontal rule or a thematic break* in an HTML page to *divide or separate document sections*. The <hr> tag is an empty tag, and it does not require an end tag.

**Tag Attributes:** The table given below describe the <hr> tag attributes. *These attributes are not supported in HTML5:*

<i>Attribute</i>	<i>Value</i>	<i>Description</i>
<i>align</i>	<i>Left, center right</i>	<i>Used to specify the alignment of the horizontal rule.</i>
<i>noshade</i>	<i>noshade</i>	<i>Used to specify the bar without shading effect.</i>
<i>size</i>	<i>pixels</i>	<i>Used to specify the height of the horizontal rule.</i>
<i>width</i>	<i>pixels</i>	<i>Used to specify the width of the horizontal rule.</i>

**Syntax:**     <hr> ...

**Example:**

```
<p>Normal horizontal line.</p>
<!--HTML hr tag is used here-->
<hr>
<p>Horizontal line with height of 30 pixels</p>
<hr size="30" >
<p>Horizontal line with height of 30 pixels and noshade.</p>
<hr size="30" noshade>
```

## LISTING Tag

### Lists and their Types

HTML Lists are used to specify lists of information. All lists may contain one or more list elements. There are three different types of HTML lists:

1. Ordered List or Numbered List (ol)
2. Unordered List or Bulleted List (ul)
3. Description List or Definition List (dl)

#### HTML Ordered List or Numbered List:

In the ordered HTML lists, all the list items are marked with numbers by default. It is known as numbered list also. The ordered list starts with <ol> tag and the list items start with <li> tag.

**Syntax:**

```
<ol>
  <li>Item1</li>
  <li>Item2</li>
  <li>Item3</li>
</ol>
```

**Attributes:**

- **compact:** It defines the list should be compacted (compact attribute is not supported in HTML5. Use CSS instead.).
- **reversed:** It defines that the order will be descending.
- **start:** It defines from which number the order will start.
- **type:** It defines which type(1, A, a, I, and i) of the order you want in your list of numeric, alphabetic, or roman numbers.



**Example:** This example illustrates the use of the reverse attribute, control list counting & type attribute.

```
<p>reversed attribute</p>
<ol reversed>
  <li>HTML</li>
  <li>CSS</li>
  <li>JS</li>
</ol>
```

```
<p>start attribute</p>
<ol start="5">
  <li>HTML</li>
  <li>CSS</li>
  <li>JS</li>
</ol>
```

```
<p>type attribute</p>
<ol type="i">
  <li>HTML</li>
  <li>CSS</li>
  <li>JS</li>
</ol>
```

**Nested ordered list, a nested ordered list is a list that has a list inside another list.**

```
<ol>
  <li>Coffee</li>
  <li>Tea
    <ol>
      <li>Black tea</li>
      <li>Green tea</li>
    </ol>
  </li>
  <li>Milk</li>
</ol>
```

### **The HTML Unordered List:**

An unordered list starts with the “ul” tag. Each list item starts with the “li” tag. The list items are marked with bullets i.e small black circles by default.

**Syntax:** <ul> list of items </ul>

**Attribute:** This tag contains two attributes which are listed below:

- **compact:** It will render the list smaller.

- **type:** It specifies which kind of marker is used in the list.

**Note:** The <ul> attributes are not supported by HTML5.

**Example:** This example describes the unordered list.

```
<h2>Grocery list</h2>
<ul>
  <li>Bread</li>
  <li>Eggs</li>
  <li>Milk</li>
  <li>Coffee</li>
</ul>
```

**HTML unordered list has various list item markers:**

**Example 1:** The *Disc* can be used to set the list item marker to a *bullet i.e default*.

```
<h2>Unordered List with Disc Bullets</h2>

<ul style="list-style-type: disc">
  <li>MCC</li>
  <li>Sudo</li>
  <li>Gfg</li>
  <li>Gate</li>
  <li>Placement</li>
</ul>
```

**Example 2:** The *Circle* can be used to set the list item marker to a *circle*.

```
<h2>Unordered List with Circle Bullets</h2>

<ul style="list-style-type: circle">
  <li>MCC</li>
  <li>Sudo</li>
  <li>Gfg</li>
  <li>Gate</li>
  <li>Placement</li>
</ul>
```

**Example 3:** The *Square* can be used to set the list item marker to a *square*.

```
<h2>Unordered List with Square Bullets</h2>

<ul style="list-style-type: square">
  <li>MCC</li>
```

```
<li>Sudo</li>
<li>Gfg</li>
<li>Gate</li>
<li>Placement</li>
</ul>
```

**Example 4:** It's *none* that can be used to set the list item marker with *no mark*.

**<h2>Unordered List with No Bullets</h2>**

```
<ul style="list-style-type: none">
  <li>MCC</li>
  <li>Sudo</li>
  <li>Gfg</li>
  <li>Gate</li>
  <li>Placement</li>
</ul>
```

**Example:** Nested Unordered List, It is used to nest the list items ie., a list inside another list.

**<h2>Nested Unordered List</h2>**

```
<ul>
  <li>DSA</li>
  <ul>
    <li>Array</li>
    <li>Linked List</li>
    <li>stack</li>
    <li>Queue</li>
  </ul>
  <li>Web Technologies</li>
  <ul>
    <li>HTML</li>
    <li>CSS</li>
    <li>JavaScript</li>
  </ul>
  <li>Aptitude</li>
  <li>Gate</li>
  <li>Placement</li>
</ul>
```

### **HTML Description List:**

A description list is a list of terms, with a description of each term. *The <dl> tag defines the description list, the <dt> tag defines the term name, and the <dd> tag describes each term.*

#### **Syntax:**

```
<dl> Contents... </dl>
```

**Example:** This example describes the HTML Description List.

```
<h2>A Description List</h2>
<dl>
  <dt>Coffee</dt>
  <dd>- 500 gms</dd>
  <dt>Milk</dt>
  <dd>- 1 ltr Tetra Pack</dd>
</dl>
```

### **Comment Tag**

The comment tag is used to insert comments in the source code. Comments are not displayed in the browsers.

You can use comments to explain your code, which can help you when you edit the source code at a later date. This is especially useful if you have a lot of code.

**Syntax:**     <!-- Comments here -->

**Types of HTML Comments:** There are three types of comments in HTML which are:

- Single-line comment
- Multi-lines comment
- Using <comment> tag

**Single-line comment:** Single line comment is given inside the

( <!-- comment --> ) tag.

**Multi-line comment:** Multiple lines can be given by the syntax (<!-- -->), Basically it's the same as we used in single line comment, difference is half part of the comment (" --> "), is appended where the intended comment line ends.

**Using <comment> tag:** There used to be an HTML <comment> tag, but currently it is not supported by any modern browser.

## IMG Tag

### Attributes of Image Tag

HTML **<img>** tag is used to add image inside webpage/website. Nowadays website does not directly add images to a web page, as the images are linked to web pages by using the **<img>** tag which holds space for the image.

**Syntax:**      <img src="" alt="" width="" height="">

**Attributes:** The **<img>** tag has following attributes.

- ✓ **src:** It is used to specify the path to the image.
- ✓ **alt:** It is used to specify an alternate text for the image. It is useful as it informs the user about what the image means and also due to any network issue if the image cannot be displayed then this alternate text will be displayed.
- ✓ **crossorigin:** It is used to import images from third-party sites that allow cross-origin access to be used with canvas.
- ✓ **height:** It is used to specify the height of the image.
- ✓ **width:** It is used to specify the width of the image.
- ✓ **ismap:** It is used to specify an image as a server-side image map.
- ✓ **usemap:** It is used to specify an image as a client-side image map.
- ✓ **sizes:** It is used to specify image sizes for different page layouts.
- ✓ **srcset:** It is used to specify a list of image files to use in different situations.
- ✓ **loading:** It is used to specify whether a browser should defer loading of images until some conditions are met or load an image immediately.
- ✓ **longdesc:** It is used to specify a URL to a detailed description of an image.
- ✓ **referrerpolicy:** It is used to specify which referrer information to use when fetching an image i.e. *no-referrer*, *no-referrer-when-downgrade*, *origin*, *origin-when-cross-origin*, *unsafe-url*.

### Example:

```

```

### Image Maps:

In image mapping/maps an image is specified with certain *set of coordinates inside the image* which act as *hyperlink areas to different destinations*.

The HTML **<map>** tag defines an image map. An image map is an image with clickable areas. The areas are defined with *one or more <area>* tags.

***Elements required in Mapping an Image:***

There are three basic html elements which are required for creating a mapped image.

1. **Map:** It is used to create a map of the image with clickable areas.
2. **Image:** It is used for the image source on which mapping is done.
3. **Area:** It is used within the map for defining clickable areas.

***Example:***

```

    <map name="workmap">
<area shape="rect" coords="34,44,270,350" alt="Computer"
href="computer.htm">
<area shape="rect" coords="290,172,333,250" alt="Phone" href="phone.htm">
    <area shape="circle" coords="337,300,44" alt="Cup of coffee"
href="coffee.htm">
    </map>
```

***Background Image on a HTML element:***

To add a background image on an HTML element, use the HTML style attribute and the CSS background-image property:

***Example:******Add a background image on a web page:***

```
<body style="background-image:url(mcc.jpg);">
    <h2>Background Image</h2>
</body>
```

***Or***

If you want the entire page to have a background image in CSS background properties you must specify the background image on the <body> element:

***Example:***

Add a background image for the entire page:

```
<html>
<head>
<style>
    body {
        background-image: url(mcc.jpg);
```

```
    }  
</style>  
</head>  
<body>  
</body>  
</html>
```

**Note:** To avoid the background image from repeating itself, set the background-repeat property to no-repeat.

### Example

```
<style>  
    body {  
        background-image: url(mcc.jpg);  
        background-repeat: no-repeat;  
    }  
</style>
```

## Color and Background of Web Pages

In HTML, we can change the color of the background of a webpage using the following different ways:

1. Using bgcolor attribute
2. Using an Inline style attribute
3. Using internal CSS

### 1. Using bgcolor attribute

HTML provides various styles and attributes to make changes to the documents as per the user's needs. Following is an HTML code that shows the use of **bgcolor** attribute:

**Syntax:**     <Body bgcolor=" ">

**Example:**

```
<body bgcolor="green" >  
    <h1>Hello reader my name is sanjoy Welcome to MCC</h1>  
</body>
```



## 2. Using an Inline style attribute

If we want to change the color of a background of a web page using an inline style attribute, which are given below.

**Syntax:**     <body style="background-color: ">

**Example:**

```
<body style="background-color:green">
    <h1>Hello reader my name is sanjoy Welcome to MCC</h1>
</body>
```

## 3. Using internal CSS

If we want to change the color of a background of a web page using an internal cascading stylesheet, which are given below.

**Syntax:**

```
<Head>
    <style>
        Body
        {
            background-color: color_name;
        }
    </style>
</Head>
```

**Example:**

```
<Head>
    <style>
        Body
        {
            background-color: red;
        }
    </style>
</Head>
```

**<Body>**

This page helps you to understand how to change the background color of a web page.

**</Body>**

### ***Hypertext, Hyperlink and Hypermedia***

***Hypertext:*** Hypertext is a cross referencing tool which connects the links to other text using hyperlinks. Hypertext is non-linear and multi sequential and it is different from our normal text. By the help of hypertext one organized way is achieved to present information. This makes the user to move from one part of the information to another part of the information which is in same page or any other page. It makes the documentation simple by providing a way of easily accessible to the end user.

***Hypermedia:*** Hypermedia is the extension of Hypertext which includes multiple forms of media such as text, graphics, audio or video etc rather than only text based like hypertext. It provides a facility to connect the web pages to create a network with multimedia elements with a simple click for a better multimedia experience. Hypermedia allows links to be integrate in multimedia elements like images and videos and when we click on that it takes us to that page.

***Hyperlink:*** The hyperlink contains the URL of the webpages. In a general way, a hyperlink is referenced when a hypertext navigated. These hyperlinks are hidden under the text, image, graphics, audio, video, and gets highlighted once we hover the mouse over it. To activate the hyperlink, we click the hypermedia, which ends up within the opening of the new document. It establishes the connection between the knowledge units, usually known as the target document and therefore the alternate name for the hyperlink is anchor or node.

### ***Comparison Between the Hypertext and Hypermedia:***

<b><i>Features</i></b>	<b><i>Hypertext</i></b>	<b><i>Hypermedia</i></b>
<b><i>Definition</i></b>	Hypertext is the text that connects to other text blocks in the same or a distinct document.	Hypertext is an extension of hypermedia, which is not just text-based.

<b><i>Involvement</i></b>	It involves only text.	It involves images, video, graphics, audio, etc.
<b><i>User Experience</i></b>	The usage of hypertext encourages the user to move across the document and also from one page to another.	Hypermedia is more attractive to users than hypertext since it allows for greater mobility.
<b><i>Application</i></b>	Users may easily switch between documents by clicking on the hypertext or goto links.	It expands the capabilities of hypertext and allows users to move to another page by clicking text or other multimedia.
<b><i>Relation</i></b>	It is a part of hypermedia.	It comes in the superior-level entity.
<b><i>Method</i></b>	It is a non-linear way.	It is a linear way.
<b><i>Link</i></b>	Only the text becomes a component of the link in this case.	It is an improved version of hypertext in which, in addition to text, other multimedia becomes a part of the link.

### ***Comparison Between the Hypertext and Hyperlink:***

<b><i>Hypertext</i></b>	<b><i>Hyperlink</i></b>
Hypertext contains the Non-linear linking of the text with some other information.	In Hyperlinks the references are used in the hypertext or with other hypermedia.
Hypertext involves only text.	Hyperlink involves Text, media, audio, video, images, and graphics.
Hypertext directed information only generates the related information.	Hyperlink directed link could contain some unrelated information.
Hypertext contains Hyperlink.	Hyperlink contains the comprised of the URLs.
Hypertext associate with the keywords.	Hyperlink associate with the anchor tags.

## ***Links, Anchors and URLs, Links to External Documents***

### ***HTML Links - Hyperlinks***

- ✓ HTML links are hyperlinks.
- ✓ You can click on a link and jump to another document.
- ✓ When you move the mouse over a link, the mouse arrow will turn into a little hand.

**Note:** *A link does not have to be **text**. A link can be an **image** or any other HTML element!*

The HTML **anchors tag** <a> tag defines a hyperlink. It has the following syntax:

**Syntax:** <a href="url">link text</a>

The most important attribute of the <a> element is the **href** attribute, which indicates the link's destination.

The *link text* is the part that will be visible to the reader.

Clicking on the link text, will send the reader to the specified URL address.

#### ***Example:***

```
<a href="https://mcconline.org.in">Visit Midnapore City College</a>
```

### ***HTML Links - The target Attribute***

By default, the linked page will be displayed in the current browser window. To change this, you must specify another target for the link.

The target attribute specifies where to open the linked document.

The target attribute can have one of the following values:

- **\_self** - Default. Opens the document in the same window/tab as it was clicked
- **\_blank** - Opens the document in a new window or tab
- **\_parent** - Opens the document in the parent frame
- **\_top** - Opens the document in the full body of the window

### ***Links to External Documents and internal document:***

In HTML, you can create links to both external documents (such as other web pages or files hosted on external servers) and internal documents (pages or resources within the same website). Here's how to create both types of links:

#### ***Links to External Documents:***

To create links to external documents, you can use the "a" (anchor) element with the "href" attribute specifying the URL of the external resource. Here's an example:

```
<a href="https://www.example.com">Visit Example Website</a>
```

In this example, the link text "Visit Example Website" will take the user to the external website "https://www.example.com" when clicked.

#### ***Links to Internal Documents:***

To create links to internal documents within your website, you can use the "a" element with the "href" attribute, specifying a relative path to the internal document. Relative paths are used to reference files within the same website directory. Here's an example:

```
<a href="about.html">Learn About Us</a>
```

In this case, when the user clicks the "Learn About Us" link, they will be directed to the internal webpage "about.html."

*Remember to replace the example URLs and file names with the actual external URLs and internal document paths you want to link to. Additionally, you can use **absolute URLs for external links** and **relative paths for internal links** based on your website's structure.*

#### ***Absolute URLs vs. Relative URLs:***

- ✓ Using an **absolute URL** (a full web address) in the href attribute.
- ✓ A local link (a link to a page within the same website) is specified with a **relative URL** (without the "https://www" part)

#### ***Example:***

```
<h2>Absolute URLs</h2>
```

```
<p><a href="https://www.google.com/">Google</a></p>
```

```
<h2>Relative URLs</h2>
```

```
<p><a href="first.html">HTML Images</a></p>
```

### ***HTML Links - Use an Image as a Link***

To use an image as a link, just put the `<img>` tag inside the `<a>` tag:

#### ***Example:***

```
<a href="StructureOfHTML.htm">  
  
</a>
```

### ***HTML Links - Create Bookmarks***

#### ***Create a Bookmark in HTML:***

- ✓ Bookmarks can be useful if a web page is very long.
- ✓ To create a bookmark - first create the bookmark, then add a link to it.
- ✓ When the link is clicked, the page will scroll down or up to the location with the bookmark.

#### ***Example:***

First, use the `id` attribute to create a bookmark:

```
<h2 id="C4">Chapter 4</h2>
```

Then, add a link to the bookmark ("Jump to Chapter 4"), from within the same page:

#### ***Example:***

```
<a href="#C4">Jump to Chapter 4</a>
```

## Footnote and eMailing

### Creating Footnotes:

To create footnotes in HTML, you typically use anchor tags and the "sup" (superscript) element for the reference number.

#### Example:

```
<p>This is some text with a footnote  
<a href="#footnote1"><sup>[1]</sup></a>.  
</p>  
  
<div id="footnote1">  
    <p><sup>[1]</sup> This is the content of the footnote.</p>  
</div>
```

In this example, a superscript "[1]" is added to the word "footnote1," and a corresponding "div" element with the ID "footnote1" contains the content of the footnote. When a user clicks on the footnote reference, they are taken to the footnote content within the same page.

### Adding Email Links:

To create an email link in HTML, you can use the "a" (anchor) element with the "href" attribute set to "mailto:" followed by the email address.

#### Example:

```
<p>Contact us via email:  
    <a href="mailto:example@email.com">example@email.com</a>  
</p>
```

In this example, when a user clicks the email address, it will open their default email client with a new email addressed to "example@email.com."



### Creating Table

**Tables in HTML:** An HTML table is defined with the `<table>` tag. Each table row is defined with the `<tr>` tag. A table header is defined with the `<th>` tag. By default, table headings are **bold and centered**. A table data/cell is defined with the `<td>` tag.

**Example:**

```
<table border="1">
    <tr>
        <th>Firstname</th>
        <th>Lastname</th>
        <th>Age</th>
    </tr>
    <tr>
        <td>Swastik</td>
        <td>Chakrabarty</td>
        <td>20</td>
    </tr>
    <tr>
        <td>Arun</td>
        <td>Singh</td>
        <td>32</td>
    </tr>
</table>
```

**HTML Table Borders:**

**<table> border Attribute:**

The **HTML <table> border Attribute** is used to *specify the border of a table*. It sets the border around the table cells.

**Syntax:**      <table border="1|0">

**Attribute Values:**

- **1:** It sets the border around the table cells.
- **0:** It removes (not set) the border around the table cells.

***Cellpadding and Cellspacing Attributes:***

There are two attributes called *cellpadding* and *cellspacing* which you will use to adjust the white space in your table cells. The *cellspacing attribute defines space between table cells*, while *cellpadding represents the distance between cell borders* and the content within a cell.

***Example:***

```
<table border="5" cellpadding = "20" cellspacing = "15">
```

```

    <tr>
        <th>Firstname</th>
        <th>Lastname</th>
        <th>Age</th>
    </tr>
    <tr>
        <td>Swastik</td>
        <td>Chakrobarty</td>
        <td>20</td>
    </tr>
    <tr>
        <td>Arun</td>
        <td>Singh</td>
        <td>32</td>
    </tr>
</table>

```

***Colspan and Rowspan Attributes:***

You will use *colspan* attribute if you want to merge two or more columns into a single column. Similar way you will use *rowspan* if you want to merge two or more rows.

***Example:***

```

<table border = "1">
    <tr>
        <th>Column 1</th>
        <th>Column 2</th>
        <th>Column 3</th>
    </tr>
    <tr>
        <td rowspan = "2">Row 1 Cell 1</td>

```

```

        <td>Row 1 Cell 2</td>
        <td>Row 1 Cell 3</td>
    </tr>
    <tr>
        <td>Row 2 Cell 2</td>
        <td>Row 2 Cell 3</td>
    </tr>
    <tr>
        <td colspan = "3">Row 3 Cell 1</td>
    </tr>
</table>

```

### ***Collapsed Table Borders:***

To avoid having double borders like in the example above, set the CSS border-collapse property to collapse.


### ***Example:***

```

<head>
<style>
table, th, td {
    border: 1px solid black;
    border-collapse: collapse;
}
</style>
</head>

```

```

<table style="width:100%">
    <tr>
        <th>Firstname</th>
        <th>Lastname</th>
        <th>Age</th>
    </tr>
    <tr>
        <td>Jill</td>
        <td>Smith</td>
        <td>50</td>
    </tr>
    <tr>

```

```
<td>Eve</td>
<td>Jackson</td>
<td>94</td>
</tr>
<tr>
<td>John</td>
<td>Doe</td>
<td>80</td>
</tr>
</table>
```

### Frame

HTML frames are used to *divide your browser window* into *multiple sections* where each section can load a separate HTML document. A collection of frames in the browser window is known as a *frameset*. The window is divided into frames in a similar way the tables are organized: into rows and columns.

**Creating Frames:** Instead of using body tag, use frameset tag in HTML to use frames in web browser. But this Tag is *deprecated in HTML 5*. The *<frameset>* tag is used to define how to divide the browser. Each frame is indicated by frame tag and it basically defines which HTML document shall open into the frame. To define the *horizontal frames*, use *row attribute* of frame tag in HTML document and to define the *vertical frames* use *col attribute* of frame tag in HTML document.

#### Attributes of Frameset tag:

**cols:** The cols attribute is used to create *vertical frames* in web browser. This attribute is basically used to define the *no. of columns and its size* inside the frameset tag.

The size or width of the column is set in the frameset in the following ways:

#### Use absolute value in pixel:

##### Example:

```
<frameset cols = "300, 400, 300">
```

#### Use percentage value:

##### Example:

```
<frameset cols = "30%, 40%, 30%">
```

#### Use wild card values:

##### Example:

```
<frameset cols = "30%, *, 30%0022">
```

**rows:** The rows attribute is used to create *horizontal frames* in web browser. This attribute is used to define *no of rows and its size* inside the frameset tag. The size of rows or height of each row use the following ways:

**Use absolute value in pixel:**

**Example:**

```
<frameset rows = "300, 400, 300">
```

**Use percentage value:**

**Example:**

```
<frameset rows = "30%, 40%, 30%">
```

**Use wild card values:**

**Example:**

```
<frameset rows = "30%, *, 30%">
```

**Comple code Example:**

```
<frameset rows="50%,50%">
```

```
<frame src="frame1.html">
```

```
    <frameset cols="50%,50%">
```

```
        <frame src="frame2.html" >
```

```
        <frame src="frame3.html" >
```

```
    </frameset>
```

```
</frameset>
```



**Advantages of Frames:**

- ✓ It allows the user to view multiple documents within a single Web page.
- ✓ It loads pages from different servers in a single frameset.
- ✓ The older browsers that do not support frames can be addressed using the tag.

**Disadvantages of Frames:**

There are few drawbacks with using frames, so it's never recommended to use frames in your webpages –

- ✓ Some smaller devices cannot cope with frames often because their screen is not big enough to be divided up.
- ✓ Sometimes your page will be displayed differently on different computers due to different screen resolution.
- ✓ The browser's *back* button might not work as the user hopes.
- ✓ There are still few browsers that do not support frame technology.

**StyleSheet.****HTML Attribute id and class:****Using The id Attribute:**

- ✓ The id attribute specifies a **unique id** for an HTML element. The value of the id attribute **must be unique** within the HTML document.
- ✓ The id attribute is used to point to a specific style declaration in a style sheet. It is also used by JavaScript to access and manipulate the element with the specific id.
- ✓ write a **hash character (#)**, followed by an id name. Then, define the CSS properties within curly braces {}.

**Syntax:**

```
#h1 {  
    CSS Properties  
}
```

**Example:**

```
<html>  
  <head>  
    <style>  
      #h1 {  
        background-color: lightblue;  
        color: black;  
        padding: 40px;  
        text-align: center;  
      }  
    </style>  
  </head>  
  <body>  
    <h1 id="h1">My Header</h1>  
  </body>  
</html>
```

**Using The class Attribute:**

- ✓ The class attribute is often used to point to a class name in a style sheet.

- ✓ It can also be used by a JavaScript to access and manipulate elements with the specific class name.
- ✓ write a **dot character** (.), followed by a class name. Then, define the CSS properties within curly braces {}.

**Syntax:**

```
#h1 {
  CSS Properties
}
```

**Example:**

```
<html>
  <head>
    <style>
      .h1 {
        background-color: lightblue;
        color: black;
        padding: 40px;
        text-align: center;
      }
    </style>
  </head>
  <body>
    <h1 class="h1">My Header</h1>
  </body>
</html>
```

**CSS (Cascading Style Sheets):**

- ✓ CSS stands for Cascading Style Sheets.
- ✓ CSS describes how HTML elements are to be displayed on screen, paper, or in other media
- ✓ CSS saves a lot of work. It can control the layout of multiple web pages all at once.
- ✓ External stylesheets are stored in CSS files.

**CSS Syntax:**

- ✓ The selector points to the HTML element you want to style.
- ✓ The declaration *block contains one or more declarations separated by semicolons*.
- ✓ Each declaration includes a *CSS property name and a value, separated by a colon*.
- ✓ *Multiple CSS declarations* are separated with *semicolons*, and declaration blocks are surrounded by *curly braces*.

**Example:**

```
<html>
<head>
<style>
    p {
        color: red;
        text-align: center;
    }
</style>
</head>
<body>
<p>Hello World!</p>
<p>These paragraphs are styled with CSS.</p>
</body>
</html>
```

**Types of CSS (Cascading Style Sheet)**

Cascading Style Sheet (CSS) is used to set the style in web pages that contain HTML elements. It sets the background color, font-size, font-family, color, ... etc. properties of elements on a web page.

There are three types of CSS which are given below:

- **Inline CSS**
- **Internal or Embedded CSS**
- **External CSS**

**Inline CSS:**

- ✓ An inline style may be used to apply a unique style for a single element.
- ✓ To use inline styles, add the style attribute to the relevant element. The style attribute can contain any CSS property.



## Example

Inline styles are defined within the "style" attribute of the relevant element:

```
<!DOCTYPE html>
<html>
<body>
<h1 style="color:blue; text-align:center;">This is a heading</h1>
<p style="color:red;">This is a paragraph.</p>
</body>
</html>
```

## External CSS:

- ✓ With an external style sheet, you can change the look of an entire website by changing just one file!
- ✓ Each HTML page must include a reference to the external style sheet file inside the <link> element, inside the head section.

## Example

External styles are defined within the <link> element, inside the <head> section of an HTML page:

```
<!DOCTYPE html>
<html>
<head>
<link rel="stylesheet" href="external.css">
</head>
<body>
<h1>This is a heading</h1>
<p>This is a paragraph.</p>
</body>
</html>
```

- ✓ An external style sheet can be written in any text editor, and must be saved with a **.css extension**.
- ✓ The external .css file should not contain any HTML tags.
- ✓ Here is how the "external.css" file looks:

***external.css file name save***

```
body {  
  background-color: lightblue;  
}
```

```
h1 {  
  color: navy;  
  margin-left: 20px;  
}
```

***Internal CSS***

- ✓ An internal style sheet may be used if one single HTML page has a unique style.
- ✓ The internal style is defined inside the <style> element, inside the head section.

***Example***

Internal styles are defined within the <style> element, inside the <head> section of an HTML page:

```
<!DOCTYPE html>  
<html>  
  <head>  
    <style>  
      body {  
        background-color: linen;  
      }  
  
      h1 {  
        color: maroon;  
        margin-left: 40px;  
      }  
    </style>  
  </head>  
  <body>  
    <h1>This is a heading</h1>  
    <p>This is a paragraph.</p>  
  </body>  
</html>
```

### ***CSS Margins:***

- ✓ The CSS margin properties are used to *create space around elements, outside of any defined borders*.
- ✓ With CSS, you have full control over the margins. There are properties for setting the margin for each side of an element (top, right, bottom, and left).

#### ***Margin - Individual Sides:***

CSS has properties for specifying the margin for each side of an element:

- margin-top
- margin-right
- margin-bottom
- margin-left

#### ***Example:***

```
p {  
  margin-top: 100px;  
  margin-bottom: 100px;  
  margin-right: 150px;  
  margin-left: 80px;  
}
```

### ***CSS Padding:***

The CSS padding properties are used to *generate space around an element's content, inside of any defined borders*.

With CSS, you have full control over the padding. There are properties for setting the padding for each side of an element (top, right, bottom, and left).

#### ***Padding - Individual Sides***

CSS has properties for specifying the padding for each side of an element:

- padding-top
- padding-right
- padding-bottom
- padding-left

#### ***Example:***

```
div {  
  padding-top: 50px;  
  padding-right: 30px;  
  padding-bottom: 50px;  
  padding-left: 80px;  
}
```

## Form

### **HTML Form:**

Form is an HTML element to **collect input data containing interactive controls**. It provides facilities to input text, number, values, email, password, and control fields such as checkboxes, radio buttons, submit buttons, etc., or in other words, form is a container that contains input elements like text, email, number, radio buttons, checkboxes, submit buttons, etc. Forms are generally used when you want to collect data from the user.

### **Syntax:**

```
<form>
    <!--form elements-->
</form>
```

### **The HTML <form> Elements**

The HTML <form> element can contain one or more of the following form elements:

- <input>
- <label>
- <select>
- <textarea>
- <button>
- <fieldset>
- <legend>
- <datalist>
- <output>
- <option>
- <optgroup>

### **The <input> Element:**

- ✓ One of the most used form elements is the <input> element. The <input> element can be displayed in several ways, depending on the type attribute.

### **Example:**

```
<label for="fname">First name:</label>
<input type="text" id="fname" name="fname">
```

### **The <label> Element**

The <label> element defines a label for several form elements.

- ✓ The <label> element is useful for screen-reader users, because the screen-reader will read out loud the label when the user focus on the input element.
- ✓ The <label> element also help users who have difficulty clicking on very small regions (such as radio buttons or checkboxes) - because when the user clicks the text within the <label> element, it toggles the radio button/checkbox. The for attribute of the <label> tag should be equal to the id attribute of the <input> element to bind them together.

### **The <select> Element**

The <select> element defines a drop-down list:

***Example:***

```
<label for="cars">Choose a car:</label>
<select id="cars" name="cars">
  <option value="volvo">Volvo</option>
  <option value="saab">Saab</option>
  <option value="fiat">Fiat</option>
  <option value="audi">Audi</option>
</select>
```

### ***Introduction to JavaScript:***

**JavaScript** is a **lightweight, cross-platform, single-threaded, and interpreted compiled** programming language. It is also known as the scripting language for webpages. It is well-known for the development of web pages, and many non-browser environments also use it.

JavaScript is a **weakly typed language (dynamically typed)**. JavaScript can be used for **Client-side** developments as well as **Server-side** developments. JavaScript is both an imperative and declarative type of language. JavaScript contains a standard library of objects, like **Array**, **Date**, and **Math**, and a core set of language elements like **operators, control structures, and statements**.

- **Client-side:** It supplies objects to control a browser and its **Document Object Model (DOM)**. Like if client-side extensions allow an application to place elements on an HTML form and respond to user events such as **mouse clicks, form input, and page navigation**. Useful libraries for the client side are **AngularJS, ReactJS, VueJS** and so many others.
- **Server-side:** It supplies objects relevant to running JavaScript on a server. For if the server-side extensions allow an application to communicate with a database, and provide continuity of information from one invocation to another of the application, or perform file manipulations on a server. The useful framework which is the most famous these days is **node.js**.

### ***Link JavaScript file to HTML:***

JavaScript can be added to HTML file in two ways:

- **Internal JS:** We can add JavaScript directly to our HTML file by writing the code inside the `<script>` tag. The `<script>` tag can either be placed inside the `<head>` or the `<body>` tag according to the requirement.
- **External JS:** We can write JavaScript code in another files having an **extension.js** and then link this file inside the `<head>` tag of the HTML file in which we want to add this code.

### ***Syntax:***

```
<script>  
  //Code  
</script>
```

### ***Example:***

```
<!DOCTYPE html>
<html >
<head>
  <title>
    JavaScript
  </title>
</head>
<body>
  <script>
    console.log("Welcome to Midnapore City College");
  </script>
</body>
</html>
```

### ***Features of JavaScript:***

Here are a few things that we can do with JavaScript:

- ✓ JavaScript was created in the first place for ***DOM manipulation***. Earlier websites were mostly static, after JS was created dynamic Web sites were made.
- ✓ Functions in JS are objects. They may have properties and methods just like other objects. They can be passed as arguments in other functions.
- ✓ Can handle date and time.
- ✓ Performs Form Validation although the forms are created using HTML.
- ✓ No compiler is needed.

### ***Application of JavaScript:***

***Web Development:*** Adding interactivity and behavior to static sites JavaScript was invented to do this in 1995. By using AngularJS that can be achieved so easily.

***Web Applications:*** With technology, browsers have improved to the extent that a language was required to create robust web applications. When we explore a map in Google Maps then we only need to click and drag the mouse.

***Games:*** The combination of JavaScript and HTML 5 makes JavaScript popular in game development as well.

***Art:*** Artists and designers can create whatever they want using JavaScript to draw on HTML 5 canvas, and make the sound more effective also can be use JavaScript library.

**Machine Learning:** This JavaScript library can be used in web development by using machine learning.

**Mobile Applications:** JavaScript can also be used to build an application for non-web contexts. The features and uses of JavaScript make it a powerful tool for creating mobile applications. This is a Framework for building web and mobile apps using JavaScript.

### ***Limitation of JavaScript:***

**Security risks:** JavaScript can be used to fetch data using AJAX or by manipulating tags that load data such as <img>, <object>, <script>. These attacks are called cross-site script attacks.

**Performance:** JavaScript does not provide the same level of performance as offered by many traditional languages as a complex program written in JavaScript would be comparatively slow.

**Complexity:** To master a scripting language, programmers must have a thorough knowledge of all the programming concepts, core language objects, and client and server-side objects otherwise it would be difficult for them to write advanced scripts using JavaScript.

**Weak error handling and type checking facilities:** It is a weakly typed language as there is no need to specify the data type of the variable. So wrong type checking is not performed by compile.

*Why JavaScript is known as a lightweight programming language?*

JavaScript is considered lightweight due to the fact that it has **low CPU usage**, is **easy to implement**, and has a minimalist syntax. Minimalist syntax as in, has no data types. Everything is treated here as an object. It is very easy to learn because of its syntax similar to C++ and Java.

A lightweight language **does not consume** much of your **CPU's resources**. It doesn't put excess strain on your CPU or RAM. JavaScript runs in the browser even though it has complex paradigms and logic which means it uses fewer resources than other languages.

For example, NodeJS, a variation of JavaScript not only performs faster computations but also uses fewer resources than its counterparts such as Dart or Java.



Moreover, when compared with other programming languages, it has fewer in-built libraries or frameworks, contributing as another reason for the JavaScript being lightweight.

### ***JavaScript Compiled or Interpreted or both?***

JavaScript is both ***compiled and interpreted***. In the earlier versions of JavaScript, it used only the interpreter that executed code line by line and shows the result immediately. But with time the performance became an issue as interpretation is quite slow.

Therefore, in the newer versions of JS, probably after the **V8**, the **JIT compiler** was also incorporated to optimize the execution and display the result more quickly. This JIT compiler generates a bytecode that is relatively easier to code. This bytecode is a set of highly optimized instructions.

The **V8 engine** initially uses an interpreter, to interpret the code. On further executions, the V8 engine finds patterns such as frequently executed functions, and frequently used variables, and compiles them to improve performance.

### ***Difference between Java and JavaScript:***

<b><i>Java</i></b>	<b><i>JavaScript</i></b>
Java is a strongly typed language and variables must be declared first to use in the program. In Java, the type of a variable is checked at compile-time.	JavaScript is a loosely typed language and has a more relaxed syntax and rules.
Java is an object-oriented programming language primarily used for developing complex enterprise applications.	JavaScript is a <a href="#">scripting language</a> used for creating interactive and dynamic web pages.
Java applications can run in any virtual machine (JVM) or browser.	JavaScript code used to run only in the browser, but now it can run on the server via Node.js.
Objects of Java are class-based even we can't make any program in java without creating a class.	JavaScript Objects are prototype-based.
Java program has the file extension ".java" and translates source code into bytecodes which are executed by JVM (Java Virtual Machine).	JavaScript file has the file extension ".js" and it is interpreted but not compiled, every browser has the JavaScript interpreter to execute JS code. If compile time

Java is a Standalone language.	contained within a web page and integrates with its HTML content.
Java has a thread-based approach to concurrency.	JavaScript has an event-based approach to concurrency.
Java supports multithreading, which allows multiple threads of execution to run concurrently within a single program.	JavaScript does not support multithreading, although it can simulate it through the use of web workers.
Java is mainly used for backend	JavaScript is used for the frontend and backend both.
Java is statically typed, which means that data types are determined at compile time.	JavaScript is dynamically typed, which means that data types are determined at runtime.
Java uses more memory	JavaScript uses less memory.
Java requires a Java Development Kit (JDK) to run the code	JavaScript requires any text editor or browser console to run the code

## JavaScript Variables

### *Variables are Containers for Storing Data*

JavaScript Variables can be declared in 4 ways:

- Automatically
- Using var
- Using let
- Using const

### ***Automatically variables:***

Here x, y, and z are undeclared variables.

They are automatically declared when first used:

### ***Example:***

```
<script>
  x = 5;
  y = 6;
  z = x + y;
  console.log(z)
</script>
```

- ***Using var***

### ***Example:***

```
<script>
  var x = 5;
  var y = 6;
```

```
    var z = x + y;  
    console.log(z)  
</script>
```

- **Using let**

*Example:*

```
<script>  
    let x = 5;  
    let y = 6;  
    let z = x + y;  
    console.log(z)  
</script>
```

- **Using const**

*Example:*

```
<script>  
    const x = 5;  
    const y = 6;  
    const z = x + y;  
    console.log(z)  
</script>
```

*When to Use var, let, or const?*

- ✓ Always declare variables.
- ✓ Always use `const` if the value should not be changed.
- ✓ Always use `const` if the type should not be changed (Arrays and Objects).
- ✓ Only use `let` if you can't use `const`.
- ✓ Only use `var` if you MUST support old browsers.

### **JavaScript Identifiers:**

All JavaScript *variables* must be *identified* with *unique names*.

These unique names are called *identifiers*.

Identifiers can be short names (like `x` and `y`) or more descriptive names (age, sum, totalVolume).

The general rules for constructing names for variables (unique identifiers) are:

- Names can contain letters, digits, underscores, and dollar signs.
- Names must begin with a letter.
- Names can also begin with `$` and `_` (but we will not use it in this tutorial).
- Names are case sensitive (`y` and `Y` are different variables).
- Reserved words (like JavaScript keywords) cannot be used as names.

### **JavaScript Let:**

The `let` keyword was introduced in ES6 (2015)

- Variables defined with `let` cannot be **Redeclared**.
- Variables defined with `let` must be **Declared** before use.
- Variables defined with `let` have **Block Scope**.

### ***Cannot be Redeclared:***

Variables defined with `let` ***cannot be redeclared***.

You cannot accidentally redeclare a variable declared with `let`.

***With let you cannot do this:***

#### ***Example:***

```
let x = "John Doe";  
let x = 0;
```

***With var you can:***

#### ***Example:***

```
var x = "John Doe";  
var x = 0;
```

### ***Block Scope:***

JavaScript keywords: `let` and `const`. These two keywords provide **Block Scope** in JavaScript. Variables declared inside a `{ }` block cannot be accessed from outside the block:

#### ***Example***

```
{  
  let x = 2;  
}  
// x can NOT be used here
```

### ***Redeclaring Variables:***

Redeclaring a variable using the `var` keyword can impose problems. Redeclaring a variable inside a block will also redeclare the variable outside the block:

#### ***Example***

```
var x = 10;  
// Here x is 10  
{  
  var x = 2;  
  // Here x is 2  
}  
// Here x is 2
```

Redeclaring a variable using the `let` keyword can solve this problem. Redeclaring a variable inside a block will not redeclare the variable outside the block:

### ***Example***

```
let x = 10;
// Here x is 10
{
  let x = 2;
  // Here x is 2
}
// Here x is 10
```

### ***Operators:***

JavaScript **operators** operate the operands, these are symbols that are used to manipulate a certain value or operand. Operators are used to performing specific mathematical and logical computations on operands.

***JavaScript Operators:*** There are various operators supported by JavaScript.

- ***JS Arithmetic Operators***
- ***JS Assignment Operators***
- ***JS Comparison Operators***
- ***JS Logical Operators***
- ***JS Ternary Operators***
- ***JS Bitwise Operators***
- ***JS typeof Operator***
- ***JS Arithmetic Operators***

These are the operators that operate upon the numerical values and return a numerical value.

<b><i>Operator</i></b>	<b><i>Description</i></b>
+	Addition
-	Subtraction
*	Multiplication
**	Exponentiation
/	Division
%	Modulus (Remainder)
++	Increment
--	Decrement

**Addition (+):** Addition '+' operator performs addition on two operands. This '+' operator can also be used to concatenate (add) strings.

$Y = 5 + 5$  gives  $Y = 10$

$Y = \text{"Sanjoy"} + \text{"Kumar"} + \text{"Barman"}$  gives  $Y = \text{"SanjoyKumarBarman"}$

$Y = \text{"Sanjoy"} + 4 + \text{"Barman"}$  gives  $Y = \text{"Sanjoy4Barman"}$

**Example:**

```
let a = 10 + 20;
```

```
console.log(a);
```

**Subtraction (-):** Subtraction '-' operator performs subtraction on two operands.

$Y = 5 - 3$  gives  $Y = 2$

**Example:**

```
let a = 10 - 20;
```

```
console.log(a);
```

**Multiplication (\*):** Multiplication '\*' operator performs multiplication on two operands.

$Y = 5 * 5$  gives  $Y = 25$

**Example:**

```
let a = 10 * 20;
```

```
console.log(a);
```

**Division (/):** Division '/' operator performs division on two operands (divide the numerator by the denominator).

$Y = 5 / 5$  gives  $Y = 1$

**Example:**

```
let a = 100/20;
```

```
console.log(a);
```

**Modulus (%):** Modulus '%' operator gives a remainder of an *integer* division.  $A \% B$  means remainder (A/B)

$Y = 5 \% 4$  gives  $Y = 1$

**Example:**

```
let a = 10%3;
```

```
console.log(a);
```

**Exponentiation (\*\*):** Exponentiation '\*\*' operator gives the power of the first operator raised to the second operator.

$Y = 5 ** 3$  gives  $Y = 125$

**Example:**

```
let a = 10 ** 20;
```

```
console.log(a);
```

**Increment (++):** Increment '+ +' operator increases an integer value by one.

let  $A = 10$  and  $Y = A ++$  then  $A = 11$ ,  $Y = 10$

if  $A = 10$  and  $Y = ++ A$  then  $A = 11$ ,  $Y = 11$

**Example:**

```
let a = 10;
let y=++a; // let x=a++;
console.log(a+" "+ y);
```

**Decrement (- -):** Decrement ‘- -’ operator decreases an integer value by one.

let A = 10 and Y = A - - then A = 9, Y=10  
if A = 10 and Y = - - A then A = 9, Y=9

**Example:**

```
let a = 10;
let y=--a; // let x=a--;
console.log(a + " " + x);
```

**Unary (+):** Unary ‘+’ is the fastest and preferred way of converting something into a number

+a means a is a positive number

**Example:**

```
let i = 3;
i1 = +i;
console.log(i1)
```

**Negation (-):** Negation ‘-’ operator gives the negation of an operand.

-a means a is a negative number

**Example:**

```
let i = 3;
i1 = -i;
console.log(i1)
```

- **JS Assignment Operators**

The assignment operation evaluates the assigned value. Chaining the assignment operator is possible in order to assign a single value to multiple variables

**Assignment (=):** This operator assigns the right operand value to the left operand.

If A = 10 and Y = A then Y = 10

**Example:**

```
let a = 2;
console.log(a);
```

**Addition Assignment (+=):** Sums up left and right operand values and then assigns the result to the left operand.;

Y += 1 gives Y = Y + 1

**Example:**

```
const b = 3;  
console.log(a = b + 1);
```

**Subtraction Assignment (- =):** It subtracts the right side value from the left side value and then assigns the result to the left operand.

$Y -= 1$  gives  $Y = Y - 1$

**Example:**

```
let b = 3;  
console.log(a = b - 1);
```

**Multiplication Assignment (\*=):** It multiplies a variable by the value of the right operand and assigns the result to the variable.

$Y *= A$  is equivalent to  $Y = Y * A$

**Example:**

```
let b = 3;  
console.log(a = b * 1);
```

**Division Assignment (/ =):** It divides a variable by the value of the right operand and assigns the result to the variable.

$Y /= A$  is equivalent to  $Y = Y / A$

**Example:**

```
const moo = 2;  
console.log(yoo = yoo / moo);
```

**Modules/Remainder Assignment (% =):** It divides a variable by the value of the right operand and assigns the remainder to the variable.

$Y \% = A$  is equivalent to  $Y = Y \% A$

**Example:**

```
Let yoo=4  
console.log(yoo %= 2);
```

**Exponentiation Assignment (\*\* =):** This raises the value of a variable to the power of the right operand.

$Y ** = A$  is equivalent to  $Y = Y ** A$

**Example:**

```
Let yoo=4  
console.log(yoo **= 2);
```

**Left Shift Assignment (<< =):** It moves the specified amount of bits to the left and assigns the result to the variable.

$Y << = A$  is equivalent to  $Y = Y << A$

**Example:**

```
Let yoo=4  
console.log(yoo <<= 2);
```



**Right Shift Assignment ( $\gg =$ ):** It moves the specified amount of bits to the right and assigns the result to the variable.

$Y \gg= A$  is equivalent to  $Y = Y \gg A$

**Example:**

```
Let yoo=4
console.log(yoo >>= 2);
```

**Bitwise AND Assignment ( $\& =$ ):** It does a bitwise AND operation on the operand, and assigns the result to the variable.

$Y \&= b$  is equivalent to  $Y = Y \& A$

**Example:**

```
let y=10
console.log(y &= 2);
```

**Bitwise OR Assignment ( $| =$ ):** It does a bitwise OR operation on the operand, and assigns the result to the variable.

$Y |= A$  is equivalent to  $Y = Y | b$

**Example:**

```
let y=10
console.log(y |= 2);
```

**Bitwise XOR Assignment ( $\wedge =$ ):** It does a bitwise XOR operation on the operand, and assigns the result to the variable.

$Y \wedge= A$  is equivalent to  $Y = Y \wedge A$

**Example:**

```
let y=10
console.log(y ^ = 2);
```

### • JS Comparison Operators and JS Logical Operators:

Comparison and Logical operators are used to test for true or false.

#### Comparison Operators:

Comparison operators are used in logical statements to determine equality or difference between variables or values.

OPERATOR NAME	USAGE	OPERATION
Equality Operator	$a==b$	Compares the equality of two operators
Inequality Operator	$a!=b$	Compares inequality of two operators
Strict Equality Operator	$a===b$	Compares both value and type of the operand

Strict Inequality Operator	<code>a!==b</code>	Compares inequality with type
Greater than Operator	<code>a&gt;b</code>	Checks if the left operator is greater than the right operator
Greater than or equal Operator	<code>a&gt;=b</code>	Checks if the left operator is greater than or equal to the right operator
Less than Operator	<code>a&lt;b</code>	Checks if the left operator is smaller than the right operator
Less than or equal Operator	<code>a&lt;=b</code>	Checks if the left operator is smaller than or equal to the right operator

**Equality (==):** This operator is used to compare the equality of two operands. If equal then the condition is true otherwise false.

**Example:** Below example illustrates the (==) operator in JavaScript.

// Illustration of (==) operator

```
let val1 = 5;
let val2 = '5';
// Checking of operands
console.log(val1 == 5);           //true
console.log(val2 == 5);           //true
console.log(val1 == val2);        //true
// Check against null and boolean value
console.log(0 == false);          //true
console.log(0 == null);           //false
```

**Inequality (!=):** This operator is used to compare the inequality of two operands. If equal then the condition is false otherwise true.

**Example:** Below examples illustrate the (!=) operator in JavaScript.

// Illustration of (!=) operator

```
let val1 = 5;
let val2 = '5';
// Checking of operands
console.log(val1 != 6);           //true
console.log(val2 != '5');         //false
console.log(val1 != val2);        //false
// Check against null and boolean value
console.log(0 != false);          //false
console.log(0 != null);           //true
```

**Strict equality (===):** This operator is used to compare the equality of two operands with type. If both **value and type** are equal then the condition is **true** otherwise **false**.

**Example:** Below examples illustrate the (===) operator in JavaScript.

```
// Illustration of (===) operator

let val1 = 5;
let val2 = '5';
// Checking of operands
console.log(val1 === 6);      //false
console.log(val2 === '5');    //true
console.log(val1 === val2);   //false
// Check against null and boolean value
console.log(0 === false);    //false
console.log(0 === null);     //false
```

**Strict inequality (!==):** This operator is used to compare the inequality of two operands with type. If both value and type are not equal then the condition is true otherwise false.

**Example:** Below examples illustrate the (!==) operator in JavaScript.

```
// Illustration of (!==) operator

let val1 = 5;
let val2 = '5';

// Checking of operands
console.log(val1 !== 6);      //true
console.log(val2 !== '5');    //false
console.log(val1 !== val2);   //true

// Check against null and boolean value
console.log(0 !== false);    //true
console.log(0 !== null);     //true
```

**Greater than (>):** This operator is used to check whether the left-side value is greater than the right-side value. If the value is greater then the condition is true otherwise false.

**Example:** Below examples illustrate the (>) operator in JavaScript.

```
// Illustration of (>) operator

let val1 = 5;
let val2 = "5";
// Checking of operands
console.log(val1 > 0);        //true
console.log(val2 > "10");     //true
console.log(val1 > "10");     //false
console.log(val2 > 0);        //true
```

**Greater than or equal (>=):** This operator is used to check whether the left side operand is greater than or equal to the right side operand. If the value is greater than or equal then the condition is true otherwise false.

**Example:** Below examples illustrate the (>=) operator in JavaScript.

```
// Illustration of (>=) operator
let val1 = 5;
let val2 = "5";

// Checking of operands
console.log(val1 >= 5);           //true
console.log(val2 >= "15");        //true
console.log(val1 >= "5");         //true
console.log(val2 >= 15);          //false
```

**Less than operator (<):** This operator is used to check whether the left-side value is less than the right-side value. If yes then the condition is true otherwise false.

**Example:** Below examples illustrate the (<) operator in JavaScript.

```
// Illustration of (<) operator
let val1 = 5;
let val2 = "5";

// Checking of operands
console.log(val1 < 15);           //true
console.log(val2 < "0");          //false
console.log(val1 < "0");          //false
console.log(val2 < 15);           //true
```

**Less than or equal operator (<=):** This operator is used to check whether the left side operand value is less than or equal to the right side operand value. If yes then the condition is true otherwise false.

**Example:** Below examples illustrate the (<=) operator in JavaScript.

```
// Illustration of (<=) operator
let val1 = 5;
let val2 = "5";

// Checking of operands
console.log(val1 <= 15); //true
console.log(val2 <= "0"); //false
console.log(val1 <= "0"); //false
console.log(val2 <= 15); //true
```

### JS Logical Operators:

There are three types of logical operators in JavaScript:

- **!(NOT):** Converts operator to Boolean and returns flipped value
- **&&(AND):** Evaluates operands and return true only if all are true
- **||(OR):** Returns true even if one of the multiple operands is true

**NOT(!) Operator:** It reverses the Boolean result of the operand (or condition). It Converts the operand to Boolean type i.e, *true/false*

#### Syntax:

```
result = !value; // Can have single argument
```

#### Example:

```
// !(NOT) operator
let a = 1;
console.log(!a);           //false
```

**AND (&&) Operator:** It accepts multiple arguments and it evaluates the operands from left to right. And for each operand, it will first convert it to a Boolean. If the **result is false**, stops and **returns the original value** of that operand. Otherwise, if all were **truthy** it will return the **last truthy** value.

#### Syntax:

```
result = a && b; // Can have multiple arguments.
```

**Example:** The operator checks for the values from left to right and returns the value if the result is false and if the result is true, it will return the last value.

```
// &&(AND) operator
console.log( 0 && 1 );           // 0
console.log( 1 && 3 );           // 3
console.log( null && true );      // null
console.log( 1 && 2 && 3 && 4 );    // 4
```

**OR (||) Operator:** The ‘OR’ operator is somewhat opposite of the ‘AND’ operator. It evaluates the operand from left to right. And for each operand, it will first convert it to a Boolean. If the **result is true**, stops and returns the **original value** of that operand. Otherwise, **if all the values are false**, it will return the **last value**.

#### Syntax:

```
result = a || b;
```

**Example:** The operator checks the values from left to right and if the result is true returns the original value and if false returns the last value of operands.

```
// ||(OR) Operator
console.log( 0 || 1 );           // 1
console.log( 1 || 3 );           // 1
console.log( null || true );     // true
console.log( -1 || -2 || -3 || -4 ); // -1
```

- **JS Ternary Operators**

The “Question mark” or “conditional” operator in JavaScript is a ternary operator that has three operands. It is the simplified operator of if/else.

**Examples:**

Input: let result = (10 > 0) ? true : false;

Output: true

Input: let message = (20 > 15) ? "Yes" : "No";

Output: Yes

**Syntax:**

Condition ? value if true : value if false

- **condition:** Expression to be evaluated which returns a boolean value.
- **value if true:** Value to be executed if the condition results in a true state.
- **value if false:** Value to be executed if the condition results in a false state.

**Characteristics of Ternary Operator**

- The expression consists of three operands: the condition, value if true, and value if false.
- The evaluation of the **condition** should result in either true/false or a Boolean value.
- The **true** value lies between “?” & “:” and is executed if the condition returns true. Similarly, the **false** value lies after “:” and is executed if the condition returns false.

**Example:** Below is an example of the Ternary Operator.

```
let PMarks = 40
```

```
let result = (PMarks > 39) ? "Pass" : "Fail";
```

```
console.log(result);
```

- **JS Bitwise Operators**

The bitwise operators in JavaScript are:

1. **Bitwise AND (&):**

**Syntax:** a & b

**Description:** Sets each bit to 1 if both bits are 1.

**Example:**

```
let a = 5; // In binary: 0101
```

```
let b = 3; // In binary: 0011
```

```
console.log(a & b); // Bitwise AND: 0101 & 0011 = 0001 (Decimal: 1)
```

2. **Bitwise OR (|):**

**Syntax:** a | b

**Description:** Sets each bit to 1 if at least one of the corresponding bits is 1.

**Example:**

```
let a = 5; // In binary: 0101
```

```
let b = 3; // In binary: 0011
```

```
console.log(a | b); // Bitwise OR: 0101 | 0011 = 0111 (Decimal: 7)
```

### 3. **Bitwise XOR (^):**

**Syntax:**  $a \wedge b$

**Description:** Sets each bit to 1 if only one of the corresponding bits is 1.

**Example:**

```
let a = 5; // In binary: 0101
let b = 3; // In binary: 0011
console.log(a ^ b); // Bitwise XOR: 0101 ^ 0011 = 0110 (Decimal: 6)
```

### 4. **Bitwise NOT (~):**

**Syntax:**  $\sim a$

**Description:** Inverts all the bits in the operand.

**Example:**

```
let a = 5; // In binary: 0101
let b = 3; // In binary: 0011
console.log(~a); // Bitwise NOT: ~0101 = 1010 (Decimal: -6)
```

### 5. **Left Shift (<<):**

**Syntax:**  $a \ll b$

**Description:** Shifts the bits of  $a$  to the left by  $b$  positions. The vacant bits on the right are filled with zeros.

**Example:**

```
let a = 5; // In binary: 0101
let b = 3; // In binary: 0011
console.log(a << 1); // Left Shift: 0101 << 1 = 1010 (Decimal: 10)
```

### 6. **Sign-propagating Right Shift (>>):**

**Syntax:**  $a \gg b$

**Description:** Shifts the bits of  $a$  to the right by  $b$  positions. The vacant bits on the left depend on the sign bit (sign-propagating).

**Example:**

```
let a = 5; // In binary: 0101
let b = 3; // In binary: 0011
console.log(a >> 1); // Right Shift: 0101 >> 1 = 0010 (Decimal: 2)
```

### 7. **Zero-fill Right Shift (>>>):**

**Syntax:**  $a \ggg b$

**Description:** Shifts the bits of  $a$  to the right by  $b$  positions. The vacant bits on the left are filled with zeros.

**Example:**

```
let a = 5; // In binary: 0101
let b = 3; // In binary: 0011
console.log(a >>> 1); // Zero-fill Right Shift: 0101 >>> 1 = 0010 (Decimal: 2)
```



## ***JS Data Types:***

In JavaScript, there are several data types that define the kinds of values used in programming. These data types can be broadly categorized into two main groups: primitive types and non-primitive types (also known as reference types).

### ***Primitive Data Types:***

**Number:** The number type in JavaScript contains both integer and floating-point numbers. Besides these numbers, we also have some ‘special-numbers’ in javascript that are: ‘Infinity’, ‘-Infinity’, and ‘NaN’. Infinity basically represents the mathematical ‘∞’. The ‘NaN’ denotes a computational error.

***Example:***

```
let num = 2; // Integer
let num2 = 1.3; // Floating point number
let num3 = Infinity; // Infinity
let num4 = 'something here too'/2; // NaN
```

***String:***

A String in JavaScript is basically a series of characters that are surrounded by quotes. There are three types of quotes in JavaScript, which are:

***Example:***

```
let str = "Hello There";
let str2 = 'Single quotes works fine';
let phrase = `can embed ${str}`;
```

There’s no difference between ‘single’ and “double” quotes in JavaScript. Backticks provide extra functionality as with their help of them we can embed variables inside them.

***Boolean:***

The Boolean type has only two values: true and false. This data type is used to store yes/no values: true means “yes, correct”, and false means “no, incorrect”.

***Example:***

```
let isCoding = true;      // yes
let isOld = false;        // no
```

***NULL:***

The special null value does not belong to any of the default data types. It forms a separate type of its own which contains only the null value:

***Example:***

```
let age = null;
```

The ‘null’ data type basically defines a special value that represents ‘nothing’, ‘empty’, or ‘value unknown’. **Undefined** Just like null, Undefined makes its own type. The meaning of undefined is ‘value is not assigned’.





**Note:-** *Object()* can be called with or without new. Both create a new object.

**Example:**

```
const o = new Object();  
o.foo = 42;  
console.log(o);           // { foo: 42 }
```

- **JS typeof Operator**

In JavaScript, the *typeof operator* returns the data type of its operand in the form of a string. The operand can be any object, function, or variable.

**Syntax:**

```
typeof operand  
  
OR  
  
typeof (operand)
```

**Note:** Operand is an expression representing the object or primitive whose type is to be returned.

The possible types that exist in JavaScript are:

- undefined
- Object
- Boolean
- number
- string
- symbol
- function

**Example:** This example checks the *typeof* of a string, number, and undefined object and returns the value in the console.

```
// "string"  
console.log(typeof ('mukul'))           //string  
  
// "number"  
console.log(typeof 25)                   //number  
  
// "undefined"  
console.log(typeof variable)             //undefined
```

## ***Literals and Type Casting in JavaScript:***

### ***JavaScript Literals:***

JavaScript Literals are the ***fixed value that cannot be changed***, you ***do not need to specify any type of keyword to write literals***. Literals are often used to initialize variables in programming, names of variables are string literals.

A JavaScript Literal can be a ***numeric, string, floating-point value, a boolean value or even an object***.

In simple words, any value is literal, if you write a string ***"mcc"*** is a literal, any number like ***7007*** is a literal, etc.

***Numeric Literals:*** These represent numbers. For instance:

#### ***Example:***

```
let num = 10;           // Integer literal
let floatNum = 3.14;    // Floating-point literal
```

***String Literals:*** These represent textual data enclosed in single or double quotes:

#### ***Example:***

```
let str = 'Hello, World!';           // Single-quoted string literal
let anotherStr = "JavaScript is awesome!"; //Double-quoted string literal
```

***Boolean Literals:*** These represent boolean values, which can be either ***true*** or ***false***:

#### ***Example:***

```
let isTrue = true; // Boolean literal representing true
let isFalse = false; // Boolean literal representing false
```

***Null Literal:*** Represents a null value:

#### ***Example:***

```
let nullValue = null; // Null literal
```

***Undefined Literal:*** Represents a variable that has been declared but not assigned a value:

#### ***Example:***

```
let undefinedValue; // Undefined literal (implicitly assigned)
```

### ***Type Casting in JavaScript:***

**In programming, type conversion is the process of converting data of one type to another.**

***For example: converting String data to Number.***

**There are two types of type conversion in JavaScript.**

- ***Implicit Conversion*** - automatic type conversion
- ***Explicit Conversion*** - manual type conversion

### ***JavaScript Implicit Conversion:***

In certain situations, JavaScript automatically converts one data type to another (to the right type). This is known as implicit conversion.

#### ***Example1: Implicit Conversion to String***

***// numeric string used with + gives string type***

```
let result;  
result = '3' + 2;  
console.log(result)      // 32  
result = '3' + true;  
console.log(result);      // 3true  
result = '3' + undefined;  
console.log(result);      // 3undefined  
result = '3' + null;  
console.log(result);      // 3null
```

***Note:*** When a number is added to a string, JavaScript converts the number to a string before concatenation.

#### ***Example 2: Implicit Conversion to Number***

***// numeric string used with -, /, \*, % results number type***

```
let result;  
result = '4' - '2';  
console.log(result);      // 2  
result = '4' - 2;  
console.log(result);      // 2  
result = '4' * 2;  
console.log(result);      // 8
```

```
result = '4' / 2;  
console.log(result);    // 2
```

**Example 3: Non-numeric String Results to NaN**

**// non-numeric string used with -, /, \* results to NaN**

```
let result;  
result = 'hello' - 'world';  
console.log(result); // NaN  
result = '4' - 'hello';  
console.log(result); // NaN
```

**Example 4: Implicit Boolean Conversion to Number**

**// if boolean is used, true is 1, false is 0**

```
let result;  
result = '4' - true;  
console.log(result); // 3  
result = 4 + true;  
console.log(result); // 5  
result = 4 + false;  
console.log(result); // 4
```

**Note:** JavaScript considers 0 as false and all non-zero number as true. And, if true is converted to a number, the result is always 1.

**Example 5: null Conversion to Number**

**// null is 0 when used with number**

```
let result; result = 4 + null;  
console.log(result); // 4  
result = 4 - null;  
console.log(result); // 4
```

**Example 6: undefined used with number, boolean or null**

**// Arithmetic operation of undefined with number, boolean or null gives NaN**

```
let result;  
result = 4 + undefined;  
console.log(result); // NaN  
result = 4 - undefined;  
console.log(result); // NaN  
result = true + undefined;  
console.log(result); // NaN  
result = null + undefined;
```

```
console.log(result); // NaN
```

### ***JavaScript Explicit Conversion:***

You can also convert one data type to another as per your needs. The type conversion that you do *manually is known as explicit type conversion*. In JavaScript, explicit type conversions are done using built-in methods.

Here are some common methods of explicit conversions.

#### ***1. Convert to Number Explicitly:***

To convert numeric *strings and Boolean* values to *numbers*, you can use *Number()*.

***For example,***

```
let result;  
// string to number  
result = Number('324');  
console.log(result); // 324  
result = Number('324e-1')  
console.log(result); // 32.4  
// boolean to number  
result = Number(true);  
console.log(result); // 1  
result = Number(false);  
console.log(result); // 0
```

***In JavaScript, empty strings and null values return 0.***

***For example,***

```
let result;  
result = Number(null);  
console.log(result); // 0  
let result = Number(' ');  
console.log(result); // 0
```

***If a string is an invalid number, the result will be NaN.***

***For example,***

```
let result;  
result = Number('hello');  
console.log(result); // NaN  
result = Number(undefined);
```

```
console.log(result); // NaN
result = Number(NaN);
console.log(result); // NaN
```

**Note:** You can also generate *numbers from strings* using `parseInt()`, `parseFloat()`, unary operator `+` and `Math.floor()`.

**For example,**

```
let result;
result = parseInt('20.01');
console.log(result); // 20
result = parseFloat('20.01');
console.log(result); // 20.01
result = +'20.01';
console.log(result); // 20.01
result = Math.floor('20.01');
console.log(result); // 20
```

## 2. Convert to String Explicitly

To convert *other data types* to *strings*, you can use either *String()* or *toString()*.

**For example,**

**//number to string**

```
let result;
result = String(324);
console.log(result);           // 324
result = String(2 + 4);
console.log(result);           // 6
```

**//other data types to string**

```
result = String(null);
console.log(result);           // null
result = String(undefined);
console.log(result);           // undefined
result = String(NaN);
console.log(result);           // NaN
result = String(true);
console.log(result);           // true
result = String(false);
console.log(result);           // false
```

**// using toString()**

```
result = (324).toString();
console.log(result);           // 324
```

```
result = true.toString();  
console.log(result);           // true
```

**Note:** String() takes *null and undefined* and converts them to *string*. However, *toString()* gives *error when null are passed*.

### 3. Convert to Boolean Explicitly

To convert *other data types* to a **Boolean**, you can use **Boolean()**.

**In JavaScript, *undefined, null, 0, NaN, ''* converts to false.**

*For example,*

```
let result;  
result = Boolean(' ');  
console.log(result); // false  
result = Boolean(0);  
console.log(result); // false  
result = Boolean(undefined);  
console.log(result); // false  
result = Boolean(null);  
console.log(result); // false  
result = Boolean(NaN);  
console.log(result); // false
```

**All other values give true.**

*For example,*

```
result = Boolean(324);  
console.log(result); // true  
result = Boolean('hello');  
console.log(result); // true  
result = Boolean(' ');  
console.log(result); // true
```

### Type Coercion in JavaScript :

**Type Coercion** refers to the process of *automatic or implicit conversion* of values from one data type to another. This includes conversion from Number to String, String to Number, Boolean to Number etc.



**Example: 1. Number to String Conversion:**

```
var x = 10 + '20';  
var y = '20' + 10;  
var z = true + '10';  
console.log(x);    //1020  
console.log(y);    //2010  
console.log(z);    //true10
```

**2. String to Number Conversion:****Example:**

```
var w = 10 - '5';  
  
var x = 10 * '5';  
var y = 10 / '5';  
var z = 10 % '5';  
console.log(w);    //5  
console.log(x);    //50  
console.log(y);    //2  
console.log(z);    //0
```

**3. Boolean to Number:****Example:**

```
var x = true + 2;  
var y = false + 2;  
console.log(x);    //3  
console.log(y);    //2
```

**JS Functions:**

JavaScript function is a set of statements that take inputs, do some specific computation, and produce output.

A JavaScript function is executed when “something” invokes it (calls it).

**Example 1:** A basic **JavaScript** function, here we create a function that divides the 1st element by the second element.

```
function myFun(g1, g2) {  
    return g1 / g2;  
}  
const value = myFun(8, 2); // Calling the function  
console.log(value);    //4
```

JavaScript allows us to create user-defined functions also. We can create functions in JavaScript using the keyword *“function”*.

**Syntax:** The basic syntax to create a function in JavaScript is shown below.

```
function functionName(Parameter1, Parameter2, ...)  
{  
    // Function body  
}
```

**Function Invocation:**

- Triggered by an event (e.g., a button click by a user).
- When explicitly called from JavaScript code.
- Automatically executed, such as in self-invoking functions.

*Function Definition:*

A function definition is sometimes also termed a function declaration or function statement. Below are the rules for creating a function in JavaScript:

- Every function should begin with the keyword *function* followed by,
- A user-defined function name that should be unique,
- A list of parameters enclosed within parentheses and separated by commas,
- A list of statements composing the body of the function enclosed within curly braces {}.

**Example 2:** This example shows a basic declaration of a function in javascript.

```
function calcAddition(number1, number2) {  
    return number1 + number2;  
}  
  
console.log(calcAddition(6,9)); //15
```

**Function Declaration:** It declares a function with a function keyword. The function declaration must have a function name.

**Syntax:**

```
function fun (param A, param B) {  
    // Set of statements  
}
```

*Function Expression:*

It is similar to a function declaration *without the function name*. Function expressions can be *stored in a variable assignment*.

**Syntax:**

```
let fun= function(paramA, paramB) {  
    // Set of statements  
}
```

**Example 3:** This example explains the usage of the Function expression.

```
let square = function (number) {  
    return number * number;  
};  
let x = square(4);           // x gets the value 16  
console.log(x);              //16
```

*Functions as Variable Values:*

Functions can be used the same way as you use variables.

**Example:**

```
// Function to convert Fahrenheit to Celsius  
function toCelsius(fahrenheit) {  
    return (fahrenheit - 32) * 5/9;  
}  
  
// Using the function to convert temperature  
let temperatureInFahrenheit = 77;  
let temperatureInCelsius = toCelsius(temperatureInFahrenheit);  
let text = "The temperature is " + temperatureInCelsius + " Celsius";  
console.log(text); // The temperature is 25 Celsius
```

*Arrow Function:*

It is one of the most used and efficient methods to create a function in JavaScript because of its comparatively easy implementation. It is a simplified as well as a more compact version of a regular or normal function expression or syntax.

**Syntax:**

```
let function_name = (argument1, argument2 ...) => expression
```

**Example 4:** This example describes the usage of the Arrow function.

```
const fun=(x, y) => x * y;  
document.write(fun(40,5));
```

## ***JS Methods:***

### ***JavaScript Object Methods***

Object Methods in JavaScript can be accessed by using functions. Functions in JavaScript are stored as property values. The objects can also be called without using brackets ().

- In a method, 'this' refers to the owner object.
- Additional information can also be added along with the object method.

**Syntax:**     *objectName.methodName()*

**Properties:** A function may be divided into different property values, which are then combined and returned together.

**Example:** The student function contains the properties:

- name
- class
- section

**Return Value:** It returns methods/functions stored as object properties.

**Example 1:** *This example uses function definition as the property value.*

```
// Object creation
```

```
let student = {  
  name: "Nur",  
  class: "5th",  
  section: "A",  
  
  studentDetails: function () {  
    return this.name + " " + this.class + " " + this.section + " ";  
  }  
};
```

```
// Display object data
```

```
console.log(student.studentDetails()); // Nur 5th A
```

**Example 2:** Using function definition as property value and accessing with additional details.

```
// Object creation
let student = {
  name: "Martin",
  class: "12th",
  section: "A",
  studentDetails: function () {
    return this.name + " " + this.class + " " + this.section + " ";
  }
};
// Display object data
console.log("STUDENT " + student.studentDetails())
// STUDENT Martin 12th A
```

## **JavaScript String Methods**

### *JavaScript String Length*

The `length` property returns the length of a string:

#### **Example**

```
let text = "ABCDEFGHIJKLMNOPQRSTUVWXYZ";
let length = text.length;
document.write(length);
```

### *JavaScript String slice():*

`slice()` extracts a part of a string and returns the extracted part in a new string. The method takes 2 parameters: start position, and end position (end not included).

#### **Example**

```
//Slice out a portion of a string from position 7 to position 13:
let text = "Apple, Banana, Kiwi";
let part = text.slice(7, 13);
document.write(part);    // Banana
```

*Note: JavaScript counts positions from zero. First position is 0. Second position is 1.*

#### **Examples**

//If you omit the second parameter, the method will slice out the rest of the string:

```
let text = "Apple, Banana, Kiwi";
let part = text.slice(7);
```

```
document.write(part);    // Banana, Kiwi
```

//If a parameter is negative, the position is counted from the end of the string:

```
let text = "Apple, Banana, Kiwi";
let part = text.slice(-12);
document.write(part);    // Banana, Kiwi
```

//This example slices out a portion of a string from position -12 to position -6:

```
let text = "Apple, Banana, Kiwi";
let part = text.slice(-12, -6);
document.write(part);    // Banana
```

*JavaScript String substring():*

substring() is similar to slice().

The difference is that start and end values less than 0 are treated as 0 in substring().

### **Example**

```
let str = "Apple, Banana, Kiwi";
let part = str.substring(7, 13);
document.write(part);    // Banana
```

### **Replacing String Content:**

The replace() method replaces a specified value with another value in a string:

### **Example**

```
let text = "Please visit Microsoft!";
let newText = text.replace("Microsoft", "W3Schools");
document.write(newText);    // Please visit W3Schools!
```

*JavaScript String ReplaceAll():*

In 2021, JavaScript introduced the string method replaceAll():

### **Example:**

```
let text = "I love cats. Cats are very easy to love. Cats are very popular.\n";
console.log(text);
text = text.replaceAll("Cats", "Dogs");
text = text.replaceAll("cats", "dogs");
```

```
console.log(text);
```

*Converting to Upper and Lower Case:*

**A string is converted to upper case with `toUpperCase()`:**

**A string is converted to lower case with `toLowerCase()`:**

**JavaScript String `toUpperCase()`**

**Example**

```
let text1 = "Hello World!";
let text2 = text1.toUpperCase();
console.log(text2); // HELLO WORLD!
```

**JavaScript String `toLowerCase()`:**

**Example**

```
let text1 = "Hello World!"; // String
let text2 = text1.toLowerCase(); // text2 is text1 converted to lower
console.log(text2); // hello world!
```

*JavaScript String `concat()`:*

`concat()` joins two or more strings:

**Example:**

```
let text1 = "Hello";
let text2 = "World";
let text3 = text1.concat(" ", text2);
console.log(text3); // Hello World
```

*JavaScript String `trim()`:*

The `trim()` method removes whitespace from both sides of a string:

**Example**

```
let text1 = "   Hello World!   ";
let text2 = text1.trim();
console.log(text2); // Hello World!
```

**JS Events:**

**JavaScript Events:**

The ***change in the state of an object*** is known as an ***Event***. In html, there are various events which represents that some activity is performed by the user or by the browser. When JavaScript code is included in HTML, ***JS react over these***

*events* and allow the execution. This process of *reacting over the events* is called **Event Handling**. Thus, JS handles the HTML events via **Event Handlers**.

**Example:** when a user clicks over the browser, add JS code, which will execute the task to be performed on the event.

Some of the HTML events and their event handlers are:

*Mouse events:*

<b>Event Performed</b>	<b>Event Handler</b>	<b>Description</b>
click	onclick	When mouse click on an element
mouseover	onmouseover	When the cursor of the mouse comes over the element
mouseout	onmouseout	When the cursor of the mouse leaves an element
mousedown	onmousedown	When the mouse button is pressed over the element
mouseup	onmouseup	When the mouse button is released over the element
mousemove	onmousemove	When the mouse movement takes place.

*Click Event Example:*

```
<!doctype html>
<html>
<head>
  <script>
    function hiThere() {
      document.write('Hi there!');
    }
  </script>
</head>

<body>
  <button type="button"
    onclick="hiThere()"
    style="margin-left: 50%;">
    Click me event
  </button>
</body>
</html>
```

*Keyboard events:*



<b><i>Event Performed</i></b>	<b><i>Event Handler</i></b>	<b><i>Description</i></b>
Keydown & Keyup	onkeydown & onkeyup	When the user press and then release the key

### Keydown Event Example:

```

<html>
<head> Javascript Events</head>
<body>
<input type="text" onkeydown="keydown()"/>
<script>
    function keydown()
    {
        document.write("Pressed a down key");
    }
</script>
</body>
</html>

```

<b><i>Event Performed</i></b>	<b><i>Event Handler</i></b>	<b><i>Description</i></b>
focus	onfocus	When the user focuses on an element
submit	onsubmit	When the user submits the form
blur	onblur	When the focus is away from a form element
change	onchange	When the user modifies or changes the value of a form element

*Form events:**Focus Event Example:*

```
<html>
<head> Javascript Events</head>
<body>
<h2> Enter something here</h2>
<input type="text" onfocus="focusevent()"/>
<script>
    function focusevent()
    {
        document.write("This is focusevent");
    }
</script>
</body>
</html>
```

*Onsubmit Event Example:*

```
<!DOCTYPE html>
<html>
<body>
    <form onsubmit="a()">
        <input type="text">
        <input type="submit" value="Submit">
    </form>
<script>
function a() {
    document.write("submitted successfully");
}
</script>
</body>
</html>
```

*Window/Document events*

<b><i>Event Performed</i></b>	<b><i>Event Handler</i></b>	<b><i>Description</i></b>
load	onload	When the browser finishes the loading of the page

unload	onunload	When the visitor leaves the current webpage, the browser unloads it
resize	onresize	When the visitor resizes the window of the browser

*Load event Example:*

```
<html>
<body onload="window.alert('Page successfully loaded');">
<script>
    document.write("The page is loaded successfully");
</script>
</body>
</html>
```

### ***JS Array and Dialog Boxes:***

#### ***JavaScript Array:***

**JavaScript Array** is a *single variable* that is used to store elements of *different data types*. JavaScript arrays are zero-indexed. Javascript Arrays are not associative in nature.

*Declaration of an Array:*

There are basically two ways to declare an array.

#### ***1. Creating an array using array literal:***

```
let arrayName = [value1, value2, ...];
```

#### ***Example:***

```
// Initializing while declaring
```

```
let courses = ["HTML", "CSS", "Javascript", "React"];
```

```
console.log(courses);
```

#### ***2. Creating an array using the JavaScript new keyword:***

```
let arrayName = new Array();
```

#### ***Example:***

```
// Initializing while declaring
```

```
let arr1 = new Array(3)
```

```
arr1[0] = 10
arr1[1] = 20
arr1[2] = 30
console.log("Array 1: ", arr1);
// Creates an array having elements 10, 20, 30, 40, 50
let arr2 = new Array(10, 20, 30, 40, 50);
console.log("Array 2: ", arr2);

// Creates an array of 5 undefined elements
let arr3 = new Array(5);
console.log("Array 3: ", arr3)
// Creates an array with one element
let arr4 = new Array("1BHK","abc");
console.log("Array 4: ", arr4);
```

### ***Accessing Elements of an Array:***

Any element in the array can be accessed using the index number. The index in the arrays starts with 0.

#### ***Example:***

```
const courses = ["HTML", "CSS", "Javascript"];
console.log(courses[0]);
console.log(courses[1]);
console.log(courses[2]);
```

### ***Change elements from a pre-defined array***

**Example:** In the given example, we have changed the value of the first element that is 'CSS' to 'MCC'

```
const courses = ["HTML", "CSS", "Javascript"];
console.log(courses);
courses[1]= "MCC";
console.log(courses);
```

### ***Increase and decrease the length of an array:***

**Example:** In the given example, We have increased and decreased the length of an array using the JavaScript's length property.

```
const courses = ["HTML", "CSS", "Javascript"];
courses.length = 5 // Increasing array length to 5
console.log("Array after increased length: ",courses);
courses.length = 2 // Decreasing array length to 2
console.log("Array after decreased length: ",courses);
```

***We can also update an array after initialization:***

***Example:***

```
const courses = ["HTML", "CSS", "Javascript"];
courses.length = 5 // Increasing array length to 5
console.log("Array after increased length: ", courses);
courses[3] = 'PhP';
courses[4] = 'React';
console.log("Array after initializing: ", courses);
```

***Loop through Javascript Array Elements:***

***Example:*** In the given example, We have looped through the elements of a Javascript array using the for loop:

```
const courses = ["HTML", "CSS", "Javascript"];
for (let i = 0; i < courses.length; i++) {
  console.log(courses[i]);
}
```

***Arrays are Objects:***

***Example:*** In the given example, The Javascript typeof operator returns “object” for arrays.

```
const courses = ["HTML", "CSS", "Javascript"];
console.log(typeof courses);
```

***JS Dialog Boxes:***

Dialogue boxes are a kind of ***popup notification***, this kind of informative functionality is used to ***show success, failure, or any particular/important notification*** to the user.

***JavaScript uses 3 kinds of dialog boxes:***

- ***Alert***
- ***Prompt***
- ***Confirm***

These dialog boxes can be of very much help in making our website look more attractive.

***Alert Box:*** An alert box is used on the website to show a warning message to the user that they have entered the wrong value other than what is required to fill in that position. Nonetheless, an alert box can still be used for friendlier messages. The alert box gives only one button “OK” to select and proceed.

***Example:***

```
<script type="text/javascript">
  function Warning() {
```

```
        alert ("Warning danger you have not filled everything");
        console.log ("Warning danger you have not filled everything");
    }
</script>
```

```
<p>Click the button to check the Alert Box functionality</p>
<form>
    <input type="button" value="Click Me" onclick="Warning();" />
</form>
```

### Confirm box:

A confirm box is often used *if you want the user to verify or accept something*. When a confirm box pops up, the user will have to *click either “OK” or “Cancel”* to proceed. If the user clicks on the **OK button**, the window method confirm() will *return true*. If the user clicks on the **Cancel button**, then confirm() *returns false* and will show null.

#### Example:

```
<script type="text/javascript">
    function Confirmation() {
        var Val = confirm("Do you want to continue ?");
        if (Val == true) {
            console.log(" CONTINUED!");
            return true;
        } else {
            console.log("NOT CONTINUED!");
            return false;
        }
    }
</script>
```

```
<p>Click the button to check the Confirm Box functionality</p>
<form>
    <input type="button" value="Click Me" onclick="Confirmation();" />
</form>
```

### Prompt Box:

A prompt box is often used *if you want the user to input a value before entering a page*. When a prompt box pops up, the user will have to *click either “OK” or “Cancel”* to proceed after entering an input value. If the user clicks the **OK**

**button**, the window method `prompt()` will *return the entered value from the text box*. If the user clicks the **Cancel button**, the window method `prompt()` returns *null*.

**Example:**

```
<script type="text/javascript">
    function Value(){
        var Val = prompt("Enter your name : ", "Please enter your name");
        console.log("You entered : " + Val);
    }
</script>

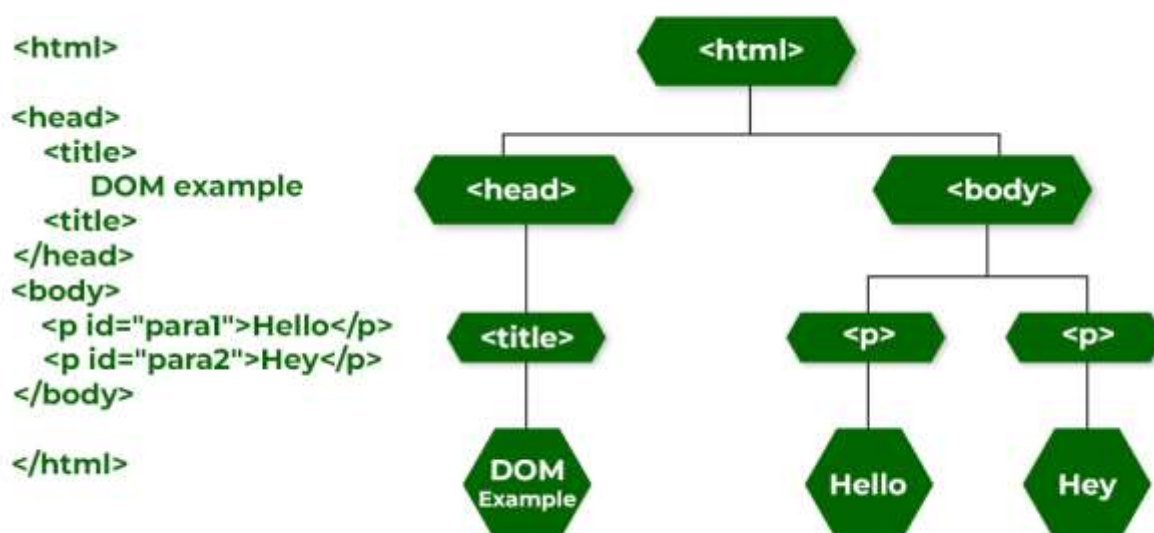
<p>Click the button to check the Prompt Box functionality</p>
<form>
    <input type="button" value="Click Me" onclick="Value();" />
</form>
```

### ***Relating JavaScript to DHTML:***

DHTML stands for Dynamic HTML. Dynamic means that the content of the *web page can be customized or changed according to user inputs* i.e. a page that is interactive with the user. In earlier times, HTML was *used to create a static page*. It only defined the structure of the content that was displayed on the page. With the help of CSS, we can beautify the HTML page by changing various properties like text size, background color, etc. The HTML and CSS could manage to navigate between static pages but couldn't do anything else.

If 1000 users view a page that had their information for eg. Admit card then there was a problem because 1000 static pages for this application build to work. As the *number of users increases, the problem also increases*, and at some point, it *becomes impossible to handle this problem*. To overcome this problem, DHTML came into existence.

DHTML included *JavaScript along with HTML and CSS to make the page dynamic*. This combo made the web pages *dynamic and eliminated the problem of creating static pages for each user*. To integrate JavaScript into HTML, a Document Object Model (DOM) is made for the HTML document. In DOM, the document is represented as nodes and objects which are accessed by different languages like JavaScript to manipulate the document.



### *HTML document include JavaScript:*

The JavaScript document is included in our *html page using the html tag*. *<src> tag is used to specify the source of external JavaScript file*. Following are some of the tasks that can be performed with JavaScript:

- *Performing html tasks*
- *Performing CSS tasks*
- *Handling events*
- *Validating inputs*

### *Example 1: Example to understand how to use JavaScript in DHTML.*

```

<h1>
  MidnaporeCityCollege
</h1>
<p id = "mcc">
  Hello MCC!
</p>
<script>
  document.getElementById("mcc").innerHTML =
  "A bachelor of computer application portal for mcc";
</script>

```

**Explanation:** In the above example, change the text of the paragraph using id. A document is an object of HTML that is displayed in the current window or object of DOM. The `getElementById(id)` gives the element id. The `innerHTML` defines the content within the id element. The id attribute is used to change an



HTML document and its property. Paragraph content changed by document id. **For example** document.getElementById("mcc").style.color = "blue"; It is used to set the paragraph color using the id of elements.

***Example 2: Creating an alert on click of a button.***

```
<h1 id = "para1" >
    MidnaporeCityCollege
</h1>
<input type = "Submit" onclick = "Click()"/>
<script>
    function Click() {
        document.getElementById("para1").style.color = "green";
        window.alert("Color changed to green");
    }
</script>
```

***Dynamically Changing Text, Style, Content:***

***Dynamically Changing Text in JS:***

***Example:***

Dynamically changing text in JavaScript involves accessing an HTML element and modifying its text content using JavaScript. Here's a simple example:

```
<!DOCTYPE html>
<html>
<head>
    <title>Dynamically Changing Text</title>
</head>
<body>
    <h1 id="mcc">Original Text</h1>
    <button onclick="changeText()">Change Text</button>
    <script>
        // JavaScript code to dynamically change text
        function changeText() {
            let dynamicElement = document.getElementById('mcc');
            dynamicElement.textContent = 'New Text'; // Change the text content
        }
    </script>
</body>
</html>
```

In the above example:

- There's an **<h1>** element with an ID of "mcc" that initially contains the text "Original Text."
- Below the heading, there's a button that, when clicked, triggers the **changeText()** function.
- Inside the **changeText()** function, JavaScript fetches the element with the ID "mcc" using **document.getElementById('mcc')**.
- It then **modifies the text content of that element** using **dynamicElement.textContent = 'New Text';**

### ***Dynamically Changing style in JavaScript:***

In JavaScript, you can **dynamically change the style of HTML elements** by accessing their **style properties and modifying them**. There are various ways to achieve this. Here's an example of how you can dynamically change the style of an HTML element:

Let's say you have an HTML element with the ID "myElement" that you want to change dynamically:

```
<h1 id="myElement">This is the element whose style will change.</h1>
<button onclick="changeStyle()">Change Style</button>
<script>
function changeStyle() {
  var element = document.getElementById('myElement');
  // Modify the style properties dynamically
  element.style.color = 'red';
  element.style.backgroundColor = 'yellow';
  // Add or remove other styles as needed...
}
</script>
```

***Minor-1: PC SOFTWARE LABORATORY  
MANUAL  
(Course: BCAMI01P)***

***List of Assignments:******Module – I Using Office with MS-Word***

<b><i>SL. No.</i></b>	<b><i>Experiments</i></b>
<b><i>1.</i></b>	<b><i>Bio-Data</i></b>
<b><i>2.</i></b>	<b><i>Leave letter</i></b>
<b><i>3.</i></b>	<b><i>To create a simple news letter</i></b>
<b><i>4.</i></b>	<b><i>Visiting Card</i></b>
<b><i>5.</i></b>	<b><i>To create a memo for the employee in the company</i></b>
<b><i>6.</i></b>	<b><i>Crate a document on the topic water pollution set the margin, orientation, page color, page border and watermark</i></b>
<b><i>7.</i></b>	<b><i>To create company letter head.</i></b>
<b><i>8.</i></b>	<b><i>Mail Merge</i></b>

***Module — II Working with MS-Excel***

<b><i>SL. No.</i></b>	<b><i>Experiments</i></b>
<b><i>1.</i></b>	<b><i>To Create a MS-Excel worksheet to illustrate sorting.</i></b>
<b><i>2.</i></b>	<b><i>Create a suitable examination data base and find the sum of the marks(total) of each Student and respective class secured by the student</i></b>
<b><i>3.</i></b>	<b><i>Create an electronic spread sheet which shows the sales of different products for 5 years. Create column chart for the following data</i></b>
<b><i>4.</i></b>	<b><i>Create an electronic spread sheet which shows the sales of different products for 5 years. Create pie chart for the following data</i></b>
<b><i>5.</i></b>	<b><i>Create an electronic spread sheet which shows the sales of different products for 5 years. Create bar chart for the following data</i></b>
<b><i>6.</i></b>	<b><i>To Create an employee's salary statement worksheet using MS-Excel</i></b>

***Module – III Working with MS-PowerPoint******SL. No.******Experiments***

1. ***Make a Power point presentation of all the details of the books that you had studied in B.Sc. First Year.***

***Module – IV Working with MS-Access******SL. No.******Experiments***

1. ***Create an Employee database with table Emp (Eno, Ename, Esal, Edept Eloc) and insert any five records. Create a report for the above Emp table of Employee database.***

## ***Module – I Using Office with MS-Word***

### **Word Processing Software:**

The word “word processor” means it processes words with pages and paragraphs.

Word processors are of 3 types which are *electronic, mechanical, and software*.

The word processing software is used to apply the basic *editing and design* and also helps in manipulating the text to your pages whereas the word processor, is a device that provides editing, input, formatting, and output of the given text with some additional features.

It is a type of computer *software application or an electronic device*.

### ***Examples or Applications of a Word Processing Software:***

- *WordPad*
- *Microsoft Word*
- *Lotus word pro*
- *Notepad*
- *WordPerfect (Windows only),*
- *AppleWorks (Mac only),*
- *Work pages*
- *OpenOffice Writer*

### ***Features:***

1. They are stand-alone devices that are dedicated to the function.
2. Their programs are running on general-purpose computers
3. It is easy to use
4. Helps in changing the shape and style of the characters of the paragraphs
5. Basic editing like headers & footers, bullets, numbering is being performed by it.
6. It has a facility for mail merge and preview.

### ***Functions:***

- It helps in Correcting grammar and spelling of sentences
- It helps in storing and creating typed documents in a new way.
- It provides the function of Creating the documents with basic editing, saving, and printing of it or same.
- It helps in Copy the text along with moving deleting and pasting the text within a given document.
- It helps in Formatting text like bold, underlining, font type, etc.
- It provides the function of creating and editing the formats of tables.
- It helps in Inserting the various elements from some other types of software.

**Advantages:**

- It benefits the environment by helping in reducing the amount of paperwork.
- The cost of paper and postage waste is being reduced.
- It is used to manipulate the document text like a report
- It provides various tools like copying, deleting and formatting, etc.
- It helps in recognizing the user interface feature
- It applies the basic design to your pages
- It makes it easier for you to perform repetitive tasks
- It is a fully functioned desktop publishing program
- It is time-saving.
- It is dynamic in nature for exchanging the data.
- It produces error-free documents.
- Provide security to our documents.


**Disadvantages:**

- It does not give you complete control over the look and feel of your document.
- It did not develop out of computer technology.

**MS Word Basics:**

1. Create New File- [Ctrl + N]
2. Open File- [Ctrl + O]
3. Save a File- [Ctrl + S]
4. Close a File- [Alt + F4]
5. Navigate in a File – [F6]
6. Word Interface

**Create New File**

**Method 1:** To open Microsoft Word → click on the Windows Start Button at the bottom left-hand side of the screen or bottom left-hand side on your keyboard.  → Select Word from list → *Select* Blank document → *Create*

**Method 2:** Once Word has opened → Go to the **File** menu (top left) → Select New → *Select* Blank document → *Create*

**Method 3:** To Create a new file short cut key [**Ctrl + N**].

**Method 4:** Windows key + R → Open Run dialog box → Type “winword” → Click Ok → *Select* Blank document → *Create*

**Open File**

**Method 1:** Click the **File** tab → Click **Open** → Select the file you want to open → Click **OK**.

**Method 2:** To Open an existing file shortcut key [**Ctrl + O**] → Select the file you want to open → Click **OK**.

**Save a File**

**Method 1:** Click the **File** tab → Click **Save/Save As** → Select the location where you want to save → Type your file name (.docx) → Click **OK**.

**Method 2:** To Save your file shortcut key [**Ctrl + S**] → Select the location where you want to Save → Type your file name → Click **OK**.

**Method 3:** The shortcut key for Save As is **F12**.

**Close a File**

**Method 1:** To close a document click "X" in the upper right corner of window.

**Method 2:** Click the **File** tab → Click **Close**

**Method 3:** The shortcut key for close file is [**Alt + F4**]



### *Navigate in a File*

**Method 1:** Click **F6** or **Alt** key to navigate your word file.

### *Home Tab:*

#### **Formatting Text: (Under Font Group)**

1. Font
2. Font Style
3. Font Color
4. Font Size
5. Text Highlight Color
6. Clear Formats
7. Change Case
8. Subscript
9. Superscript

#### **Sample Text:**

My college name is Midnapore City College- Font

#### **Font Style (Bold):**

- i) Select the text → Home Tab → Font Group → Select **B**
- ii) Shortcut key is [Ctrl + B]

Example: **My college name is Midnapore City College.**

#### **Font Style (Italic):**

- i) Select the text → Home Tab → Font Group → Select *I*
- ii) Shortcut key is [Ctrl + I]

Example: *My college name is Midnapore City College.*

#### **Font Style (Underline):**

- i) Select the text → Home Tab → Font Group → Select U
- ii) Shortcut key is [Ctrl + U]

Example: My college name is Midnapore City College.

#### **Font Color:**

Select the text that you want to change Colour → **Home** tab → **Font** group choose the arrow next to **Font Color** → Select a Color.

Example: **My college name is Midnapore City College.**

**Font size:**

My college name is Midnapore City College.

My college name is Midnapore City College- Text Highlight Color

My college name is Midnapore City College- Change Case

***Subscript:*** shortcut key (*Ctrl* + =)       $X_2$

***Superscript:*** shortcut key (*Ctrl* + *shift* ++ )       $X^2$

## 1. *Bio-Data*

### ***Procedure:***

***Step 1:*** Open MS Office-MS Word – File – New

***Step 2:*** Go to View- Header and Footer- Type name, mobile number inside the Header

***Step 3:*** Go to Insert- Page Number-select the position bottom of the page and Alignment to Center – Click Ok.

***Step 4:*** Go to Table-Insert-Table- chose Number of Columns 2 and Rows to 1.

***Step 5:*** Enter the name, format it (bold and increase the font size via standard tool Bar). And in the second column type the whole address.

***Step 6:*** Whenever you want to increase the number of columns in the existing row, select that row and go to Table-click Split Cells- enter number of columns- click Ok.

***Step 7:*** In order to decrease the existing column numbers, select that columns and Go to Tables click Merge cells.

***Step 8:*** Finally type the declaration outside the table with your name aligning right side and date to the left side.

### ***Output:***

**Curriculum Vitae****SUBHAM PRATI HAR**

**ADDRESS:** Vill: Ramchandrapur, P.O: Kaithore  
 Dist: Purba Medinipur, Pin: 721429  
**Mob.:** 9020489600  
**Email ID:** subham21@gmail.com

**PERSONAL PROFILE**

**Father Name** : Suman Pratihari  
**Date of Birth** : 12/11/1994  
**Gender** : Male  
**Marital Status** : Single  
**Religion** : Hindu  
**Nationality** : Indian

**CAREER OBJECTIVE**

Develop and promote creativity and high-order thinking skills that increase the performance of the students. To secure a position as a teacher and utilize my dedication to foster quality education required for a student's development.

**ACADMIC QUALIFICATION**

SL. No.	Examination	BOARD/UNIVERSITY	YEAR	PERCENTAGE
1.	M.Sc. in Computer Science	Vidyasagar University	2020	84.75
2.	B.Sc. in Computer Science	Vidyasagar University	2018	65.25
3.	Higher Secondary	W.B.C.H.S.E.	2013	76.40
4.	Secondary	W.B.B.S.E.	2011	70.62

**COMPUTER SKILLS**

**Software Languages:** C, C++, Java, Html, CSS, PHP, Python, R, MATLAB  
**Utility Package:** MS-office

**LANGUAGES**

**English** (Read, Write, Speak), **Bengali** (Read, Write, Speak), **Hindi** (Read, Speak)

**DECLARATION**

I do hereby declare that the statements made in this document are true to the best of my knowledge and belief.

**Place :** Ramchandrapur

**Date :** 02-05-2022

Subham Pratihari  
 Signature

2. *Leave letter*

**Procedure:**

**Step 1:** Open MS-Word by click on START button; go to All Programs, then select Microsoft Office Word 2007.

**Step 2:** To open a new document, Click on Office Button then select New - > Blank Document then click on create option.

**Step 3:** Then select TEXT AREA, and then write Leave Letter as a heading, Select the text, click on bold button to make it bold as ***“LEAVE LETTER”***, and change the font size to 16.

**Step 4:** Then write date and place in a format as follows

DATE: 19/10/2015,

Bhimavaram.

Then Select the text and make it right by clicking on right alignment button



**Step 5:** Then write To address as follows and select this text and make it left by clicking on left alignment button



To

The Principal,

B V Raju College,

Vishnupur,

Bhimavaram.

**Step 6:** Then write Subject according to your letter. And select this text and press tab button for two times.

**Step 7:** Then write the body of the letter according to your letter. And select this text and make it justification by clicking on justify alignment button




**Step 8:** Then write “Thanking you Sir,” select this text and make it to center by clicking on center alignment button



**Step 9:** Now write the “From address” as follows

Yours Faithfully,

T.Rambabu.

Then make it right by clicking on Right alignment button 

**Step 10:** This is the final step in writing leave letter. In this step, we have to save the letter as “leave letter.docx” by selecting “Save” option from Office button. Then a prompt window will ask you to write a file name. Now you have to give the file name and press the save button.

**Output:**

Date:15/10/2015,

Bhimavaram.

To,

The Principal,

B V Raju College,

Vishnupur,

Bhimavaram.

Sub: Requesting for 5 days leave-Reg

Respected Sir,

I T.Shirisha studying B.Tech I year in IT department in your college. As I am going to my home on the occasion of Ugadi festival and also to celebrate my birthday on the next day. So I kindly request you to grant me leave for 5 days i.e., 24/3/2012-28/3/2012.

Thanking You Sir,

Yours Faithfully,

T.Rambabu,

B.Sc, I year.

### 3. To create a simple news letter

**Procedure:**

**Step 1:** Open MS Office-MS Word – File – New - Type the heading

**Step 2:** Whenever you want to change the number of columns then go to

**Step 3:** Insert – Break - Select the section break type as continuous - Click OK

**Step 4:** Go to Format – Column - select the number of columns u want and click ok. Type news and whenever you need curser in the next column then go to Insert - Break- now select Column Break – click Ok.

**Step 5:** If you want picture to be inserted then go to Insert- Picture-From file- and browse for the required picture/file-then click Insert

**Step 6:** Format the text by changing the font size and color by selecting the required text and chose font size, style and color in the formatting tool bar below the menu.

**Step 7:** Formatting text can also be done by selecting the text and applying the Wordart. For that go to Insert- Picture-Wordart- then chose the style you Want and click Ok.

**Step 8:** To change the color of the wordart text, right click on the text and go to Format Wordart.

**Output:**

## October Second, Celebration of Gandhi Jayanti

Gandhi Jayanti is a National Holiday celebrated in India to mark the occasion of the birthday of Mahatma Gandhi, the "Father of the Nation". He was born on October 2, 1869. Hence Gandhi Jayanti is celebrated every year on the 2nd of October. It is one of the three official declared National Holidays of India and is observed in all Indian states and union territories. The United Nations General

Assembly announced on 15 June 2007 that it adopted a resolution which declared that the 2nd of October will be celebrated as the International Day of Non-Violence.[1]

On this day, in India, liquor is neither sold nor consumed in his honour.

Some of the famous quotes by Mahatma Gandhi have been listed below :



Live as if you were to die tomorrow. Learn as if you were to live forever.

Fear is not a disease of the body; fear kills the soul.

## Computers have Become the part of Life



Computers have come a long way spanning all work areas and influencing every one to become computer literate irrespective of the profession they are in. A thorough knowledge of computer has become a prerequisite for any job. Computers are now being used in each and every field of science, engineering

and technology. On an average almost every day an organization or a company is being computerized!

Computers are being used in banks, transport corporations, Finance Institutions, Schools, Colleges, Factories, Grocery shops, Post offices and at many other organization.

### New DTE website

Director of technical Education, Bangalore has launched a new website for its users.

Users are requested to go through the following website for more information.  
<http://dte.karrn.in>.

#### 4. Visiting Card



***Create a Visiting Card of your college using page size as follows***

- ***Page width="3.2"***
- ***Page height="2.2"***

***And use different font styles, sizes, alignments.***

**Procedure:**

**Step 1:** Open MS-Word by click on START button; go to All Programs, then select Microsoft Office Word 2007.

**Step 2:** To open a new document, Click on Office Button then select New - > Blank Document then click on create option.

**Step 3:** Now click on "Page Layout" from the Menu bar. Then click on Margins then click on Custom Margins option. Then the "Page Setup" dialog box appears. In this you find three tabs namely "Margins", "Paper", "Layout". Then in the 'Margins' tab, make all the parameters like Top, Bottom, Left, Right, and Gutter to zero and make Gutter Position to Left. Then in the Page tab, change the width and height options to 3.2 and 2 respectively. Then in the Layout tab, make the Header and Footer to zero. Now this page is set to the visiting card as follows.




**Step 4:** In this step we have to enter the telephone number and Fax number on the top part of the paper. It can be done as follows:


- First go to Insert menu, then select Symbol option.
- Then change Font to "Windings".
- Then select the appropriate to your need i.e., to the telephone option select



, and to the Fax option select




**Step 5:** Now write your institution name and make it to the center alignment button .

**Step 6:** Now write all the details you want to put in your visiting card as your needs. And select the text and make it to center .

**Step 7:** Now change the background color by selecting Page color option from Page Layout menu.

**Step 8:** This is the final step in creating Visiting Card. In this step, we have to save the letter as “Visiting Card.docx” by selecting “Save” option from Office button. Then a prompt window will ask you to write a file name. Now you have to give the file name and press the save button.

**Output:**

 :0878-2223942	 :0878-2223942
<b>B V Raju College</b>	
Website: <a href="http://www.bvricedegree.edu.in">www.bvricedegree.edu.in</a>	
<b>Vishnupur, Bhimavaram</b>	

### 5. To create a memo for the employee in the company

#### Procedure:

- Step 1.** Open MS Office-MS Word – File – New
- Step 2.** Go to View- Header and Footer- Insert the Institution name/code in the Header.
- Step 3.** Go to Insert- Page Number-select the position bottom of the page and Alignment to Center – Click Ok.
- Step 4.** Type the content. Go to File- Page Setup- Margin tab- adjust left, right, top, bottom margins – click ok.
- Step 5.** Use Standard tool bar to align the text to the left, right and center of the page.
- Step 6.** Place the cursor where you want to insert the date then go to Insert- Date and Time Chose in the Available Formats- Click Ok.

CPCP

GOVERNMENT OF KARNATAKA  
DEPARTMENT OF TECHNICAL EDUCATION

No: cpcp/est/2010-2011/156

office of the principal Or II  
Govt Polytechnic,  
Gulbarga Dated: 5-Oct-10

**MEMO**

All the staff members and students of the polytechnic are hereby informed to participate and celebrate the "INDEPENDENCE DAY" at 8:00 am on 15<sup>th</sup> Aug 2010 without fail.

Sd  
PRINCIPAL OR II

To  
All the staff Members  
All the Students

1

**6. *Crate a document on the topic water pollution set the margin, orientation, page color, page border and watermark***

**Step 1. *Open Microsoft Word:*** Launch Microsoft Word on your computer.

**Step 2. *Create a New Document:***

- ✓ Click on "File" in the upper left corner.
- ✓ Select "New" to create a new blank document.

**Step 3. *Set Page Margins:***

- ✓ Click on the "Layout" tab in the top menu.
- ✓ Click on "Margins" and select one of the preset options (e.g., "Narrow" for smaller margins, or "Wide" for larger margins) or choose "Custom Margins" to set your own margins.

**Step 4. *Set Page Orientation:***

- ✓ Still in the "Layout" tab, click on "Orientation."
- ✓ Choose either "Portrait" for vertical orientation or "Landscape" for horizontal orientation.

**Step 5. *Set Page Color:***

- ✓ Click on the "Design" tab in the top menu.
- ✓ Select "Page Color" and choose a color from the palette, or click "More Colors" to pick a custom color.

**Step 6. *Set Page Border:***

- ✓ While still on the "Design" tab, click on "Page Borders."
- ✓ In the "Page Borders" dialog box, you can set various border options, such as line style, color, and width.
- ✓ Once you've configured your border, click "OK" to apply it to the document.

**Step 7. *Add Watermark:***

- ✓ In the "Design" tab, click on "Watermark."

- ✓ You can select a pre-designed watermark from the gallery (e.g., "Confidential" or "Do Not Copy"), or you can create a custom watermark by clicking "Custom Watermark."
- ✓ Configure the watermark settings, such as text, font, size, and color.
- ✓ Click "OK" to insert the watermark into your document.

***Step 8. Start Writing:***

- ✓ Begin typing your content on the document, focusing on the topic of water pollution.

***Step 9. Save Your Document:***

- ✓ Click on "File" and then select "Save" or "Save As" to save your document with a relevant name and location on your computer.

***Step 10. Continuing Editing and Formatting:***

- ✓ You can continue editing and formatting your document as needed, including adding headings, images, bullet points, and more to make it informative and visually appealing

***Output:***

## **7. To create company letter head.**

### **Procedure:**

- Step 1.** Open MS Office-MS Word – File – New.
- Step 2.** Go to View- Header and Footer- Type the complete address of a company.
- Step 3.** Select the Text and click align right on the standard tool bar.
- Step 4.** In order to insert the company logo (create a logo using paint software and save
- Step 5.** it or use the existing one) inside the header go to Insert- Picture- From File-and browse for the required Picture/file/logo where you have saved – click Insert.
- Step 6.** After inserting the logo/image resize the logo to fit the top left corner of the page by right clicking on the logo, go to Format Picture – select Layout tab –select the Wrapping Style to Infront of text- click ok.
- Step 7.** To insert a Line, go to Insert- picture- Auto shapes- Select the line and draw below the Logo and the address inside the header.
- Step 8.** Format the line by Right clicking and selecting Format Auto shape- select the Color and Line tab- chose your style- click ok.
- Step 9.** Then go to the footer- Insert and format a line as did for header.
- Step 10.** Type the text inside the footer and below the line.
- Step 11.** Go to Format-Background-Printed Watermark-Picture Mark-Click Select Picture- Browse for the required background- click Washout Apply.

***Output:***

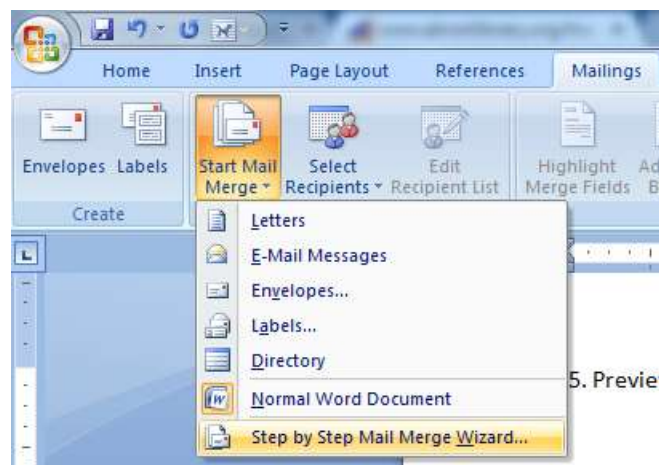
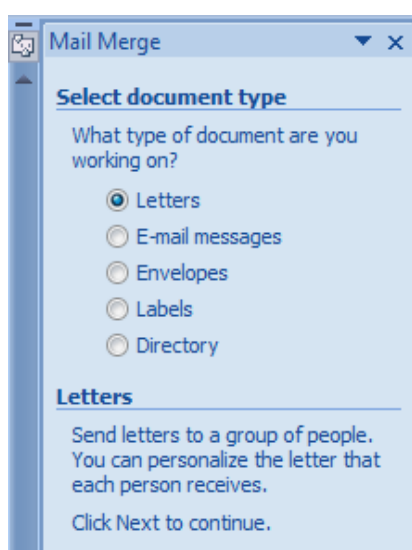


## 8. Mail Merge

### Mail Merge in MS Word

#### To get started:

- Click on the **Mailings** tab, then the **Start Mail Merge** button, and then **Step by Step Mail Merge Wizard**...

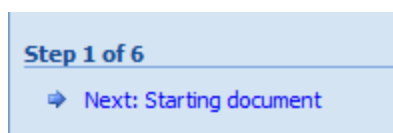


- The wizard will open in the task pane to the right. Select your document type.

#### To create the letters

##### Step 1: Select Document Type

1. Select **Letters** from the Mail Merge task pane and click on next to start the document.
2. Click Next: **Starting document**



##### Step 2: Starting Document

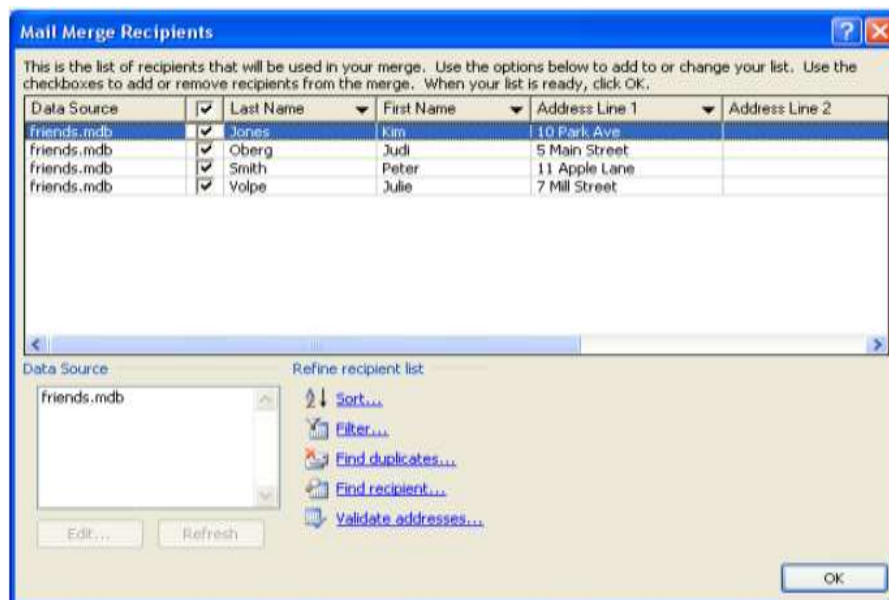
1. Click Use the **Current Document** under Select starting document
2. Click Next: **Select recipients**

**Step 3: Select recipients**

The recipients can come from either an existing Excel file, or Access table or you can create a new list.

***If Using an Existing List:***

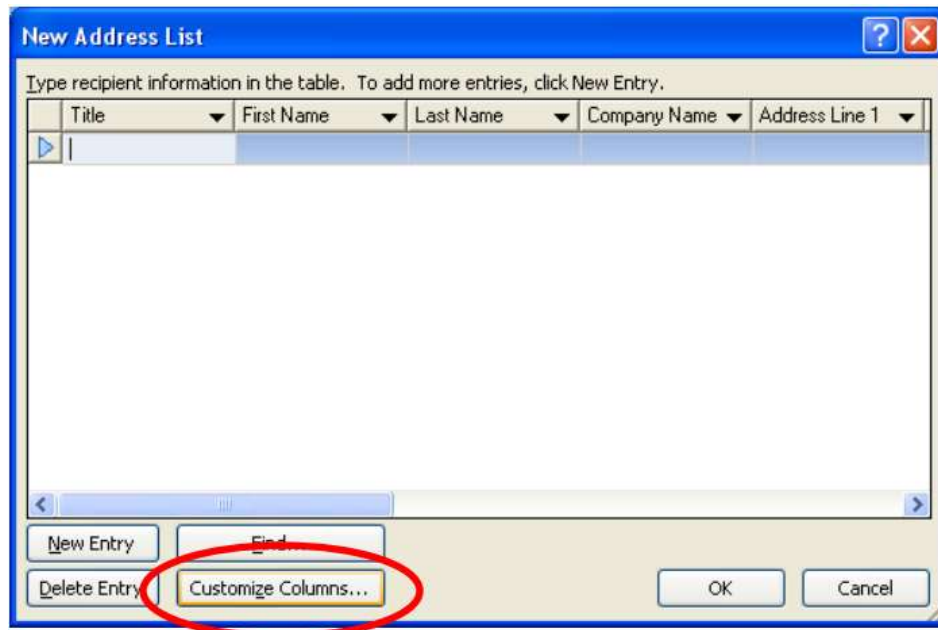
1. Click Use an existing list under Select recipients
2. Click Browse
3. Select the file
4. Click Open Mail Merge Recipients opens showing the names and addresses from your file
5. Click OK



6. Click Next: ***Write your letter***

***To Type a New List:***

- Click **Type a new list** under Select recipients
- 2) Click **Create**
- Click **Customize Columns** to modify the list of fields



- Delete any unnecessary field names and/or add new ones
- Click OK
- Type records here hitting TAB to advance to the next field and to continue adding new records
- Click OK
- Click Save

The recipients list will be saved as a separate file as a Microsoft Access file type. It is saved in the My Data Sources folder. It is recommended to save the file in this folder.

- Click Next: ***Write your letter***

#### ***Step 4: Write your letter***

- 1) Click the location in your document where the data from the mail merge fields need to be inserted
- 2) Insert the fields using ***Insert Merge Field*** from ***Mailings*** tab.



The field name will look like this: <<Name>>

- 3) Repeat this step until all fields have been inserted. Remember to put spaces and punctuation where needed.
- 4) Click Next: Preview your letters

#### ***Step 5: Preview your letters***

- Here is where you can preview the first page with the fields filled in.
- Click Next: Complete the merge

#### ***Step 6: Complete the merge***

##### ***To Complete the Merge:***

- Click Print to send directly to the printer
- Click Edit individual letters to create a new file

**Output:**

D: 20/05/2012,  
Karimnagar.

To,  
Gopinath,  
xxxxxxxxxx,  
James Street,  
Kurnool,  
Kurnool.

Dear Gopinath,

Hai! How are you? Am fine here. How are your studies going on? What about the mid exams. Am happy to say that our college has conducted FRESHERS PARTY for us in the last week at our college premises. Many competitions such as sports, food competition and funny games were conducted before the Fresher's Day. We enjoyed a lot up to the last second of the party and our college has provided food and transportation facility also. Ok bye and "ALL THE BEST" for your exams.

Yours Lovingly,  
A.Ravi kum ar.

---

D:20/05/2012  
Karimnagar.

To,  
Shiva,  
yyyyyyyyyy,  
Geetha Bhavan,  
Karimnagar,  
Karimnagar.

Dear Shiva,

Hai! How are you? Am fine here. How are your studies going on? What about the mid exams. Am happy to say that our college has conducted FRESHERS PARTY for us in the last week at our college premises. Many competitions such as sports, food competition and funny games were conducted before the Fresher's Day. We enjoyed a lot up to the last second of the party and our college has provided food and transportation facility also. Ok bye and "ALL THE BEST" for your exams.

Yours Lovingly,

A.Ravi Kumar.

---

D:20/05/2012,

karimnagar.

To,

Shilpa,

ZZZZZZZZZZZ,

Thimmapur,

Karimnagar,

Karimnagar.

Dear Shilpa,

Hai! How are you? Am fine here. How are your studies going on? What about the mid exams. Am happy to say that our college has conducted FRESHERS PARTY for us in the last week at our college premises. Many competitions such as sports, food competition and funny games were conducted before the Fresher's Day. We enjoyed a lot up to the last second of the party and our college has provided food and transportation facility also. Ok bye and "ALL THE BEST" for your exams.

Yours lovingly,

Ravi Kumar

## ***Module — II Working with MS-Excel***

### ***1. To Create a MS-Excel worksheet to illustrate sorting.***

Procedure :-

To Sort the Data:

1. Type the data in the excel sheet.

2 Select data on list to be sorted.

For example salary in the above figure.

To Sort the Data:

3. Click the *Data Menu* and select the sort option. The sort dialog box appears.

4. Select the ascending and descending option in the *Sort by section*

5. Click the *OK* button

Output:-



The screenshot shows the Microsoft Excel interface with a table containing employee data. The table has columns for Empno, Ename, Design, and Salary. The data is as follows:

	A	B	C	D	E
1	Empno	Ename	Design	Salary	
2	101	Rajkumar	Manager	7800	
3	102	Madhu	Doctor	6500	
4	103	Ravi Kiran	clerk	4500	
5	104	Kulkarni	Manager	9000	
6	105	Satish	Manager	8000	
7	106	Naresh	Doctor	11000	



The screenshot shows the Sort dialog box in Microsoft Excel. The 'Sort by' dropdown is set to 'Salary'. The 'Then by' dropdowns are empty. The 'My data range has' section has 'Header row' selected. The 'Options...' button is visible at the bottom.

Sort by: Salary (Ascending selected)  
Then by: (Ascending selected)  
Then by: (Ascending selected)  
My data range has: Header row (selected)  
Options... OK Cancel



**2. Create a suitable examination data base and find the sum of the marks(total) of each Student and respective class secured by the student**

**Rules**

- Pass if marks in each subject  $\geq 35$ ,
- Distinction if average  $\geq 70$ ,
- First class if average  $\geq 60$  but  $< 70$ ,
- Second class if average  $\geq 50$  but  $< 60$ ,
- Third class if average  $\geq 35$  and but  $< 50$ ,
- Fail if marks in any subject is  $< 35$ .

**Display average marks of the class, subject wise and pass percentage**

**Solution:**

To find the grade of a student we need to follow the following steps

**Step 1: Typing Student database in Excel 2007**

Type the student database with the required fields starts from A1 cell as follows

	A	B	C	D	E	F	G	H	I	J	K
1	Name of the Student	Maths	Physics	Chemistry	English	Sanskrit	Total	Average	P/F	Grade	
2	Ravi	45	75	64	48	98					
3	Vamsi	65	74	85	85	86					
4	Rao	35	95	48	74	82					
5	Satya	32	48	78	76	79					
6	Siva	46	31	86	78	75					
7	Ramesh	89	45	45	82	72					
8	Ramu	75	56	73	74	81					
9											
10											
11											
12											

**Step 2: To find Total Marks of Student**

To find the total marks of a student click on the cell “G2” and type the following formula

=SUM(B2:F2)

To find the total marks for the remaining students select “G2” cell and drag down to the remaining students.



I2		=IF(AND(B2>=35,C2>=35,D2>=35,E2>=35,F2>=35),"Pass","Fail")								
	A	B	C	D	E	F	G	H	I	J
1	Name of the Student	Maths	Physics	Chemistry	English	Sanskrit	Total	Average	P/F	Grade
2	Ravi	45	75	64	48	98	330	66	Pass	
3	Vamsi	65	74	85	85	86	395	79	Pass	
4	Rao	35	95	48	74	82	334	66.8	Pass	
5	Satya	32	48	78	76	79	313	62.6	Fail	
6	Siva	46	31	86	78	75	316	63.2	Fail	
7	Ramesh	89	45	45	82	72	333	66.6	Pass	
8	Ramu	75	56	73	74	81	359	71.8		
9										
10										

### Step 5: To find Grade

To find the grade of a student click on the cell "J2" and type the following formula

**=IF(AND(B2>=35,C2>=35,D2>=35,E2>=35,F2>=35),IF(H2>=75,"Distinction",IF(H2>=65,"First Class",IF(H2>=50,"Second Class",IF(H2>=35,"Third Class")))),IF(H2<35,"Fail"))**

To find the grade for the remaining students select "J2" cell and drag down to the all the students

J2		=IF(AND(B2>=35,C2>=35,D2>=35,E2>=35,F2>=35),IF(H2>=75,"Distinction",IF(H2>=65,"First Class",IF(H2>=50,"Second Class",IF(H2>=35,"Third Class")))),IF(H2<35,"Fail"))								
	A	B	C	D	E	F	G	H	I	J
1	Name of the Student	Maths	Physics	Chemistry	English	Sanskrit	Total	Average	P/F	Grade
2	Ravi	45	75	64	48	98	330	66	Pass	First Class
3	Vamsi	65	74	85	85	86	395	79	Pass	Distinction
4	Rao	35	95	48	74	82	334	66.8	Pass	First Class
5	Satya	32	48	78	76	79	313	62.6	Fail	Fail
6	Siva	46	31	86	78	75	316	63.2	Fail	
7	Ramesh	89	45	45	82	72	333	66.6	Pass	
8	Ramu	75	56	73	74	81	359	71.8	Pass	
9										

Finally we get the following student database with total, average and grade

**Output:**

	A	B	C	D	E	F	G	H	I	J	K
1	Name of the Student	Maths	Physics	Chemistry	English	Sanskrit	Total	Average	P/F	Grade	
2	Ravi	45	75	64	48	98	330	66	Pass	First Class	
3	Vamsi	65	74	85	85	86	395	79	Pass	Distinction	
4	Rao	35	95	48	74	82	334	66.8	Pass	First Class	
5	Satya	32	48	78	76	79	313	62.6	Fail	Fail	
6	Siva	46	31	86	78	75	316	63.2	Fail	Fail	
7	Ramesh	89	45	45	82	72	333	66.6	Pass	First Class	
8	Ramu	75	56	73	74	81	359	71.8	Pass	First Class	
9	Raju	64	78	84	65	87	378	75.6	Pass	Distinction	

**3. Create an electronic spread sheet which shows the sales of different products for 5 years. Create column chart for the following data**

Year	Product_1	Product_2	Product_3	Product_4
2010	1000	800	900	1000
2011	800	80	500	900
2012	1200	190	400	800
2013	400	200	300	1000
2014	1800	400	400	1200

**Creating the spread with different products of 5 year**

Before you can make a chart, you must first enter data into a worksheet. To create the spread sheet with 5 years different products follow the below steps

**Step1:**

Click on the cell B1 and type “Product\_1” and go to the next cell by clicking the Tab key on the keyboard.

**Step 2:**

In the cell C1 type “Product\_2” and click on tab key to activate next cell. Repeat the above to steps to enter “Product\_3”, “Product\_4”

**Step3:**

Click on cell “A2” and type “2010” then press on tab key to activate the next cell.

**Step 4:**

Repeat the above procedure to enter the details of four products for five years. Finally we have the data of products as follows

	A	B	C	D	E	F
1		Product_1	Product_2	Product_3	Product_4	
2	2010	1000	800	900	1000	
3	2011	800	80	500	900	
4	2012	1200	190	400	800	
5	2013	400	200	300	1000	
6	2014	1800	400	400	1200	
7						

### Creating a Column chart for the above data

#### Definition of Chart:

Charts allow you to present data entered into the worksheet in a visual format using a variety of graph types.

Following steps are given to draw a Chart

1. Enter data in the work sheet: Suppose you entered data as given in below

	A	B	C	D	E	F
1		Product_1	Product_2	Product_3	Product_4	
2	2010	1000	800	900	1000	
3	2011	800	80	500	900	
4	2012	1200	190	400	800	
5	2013	400	200	300	1000	
6	2014	1800	400	400	1200	
7						

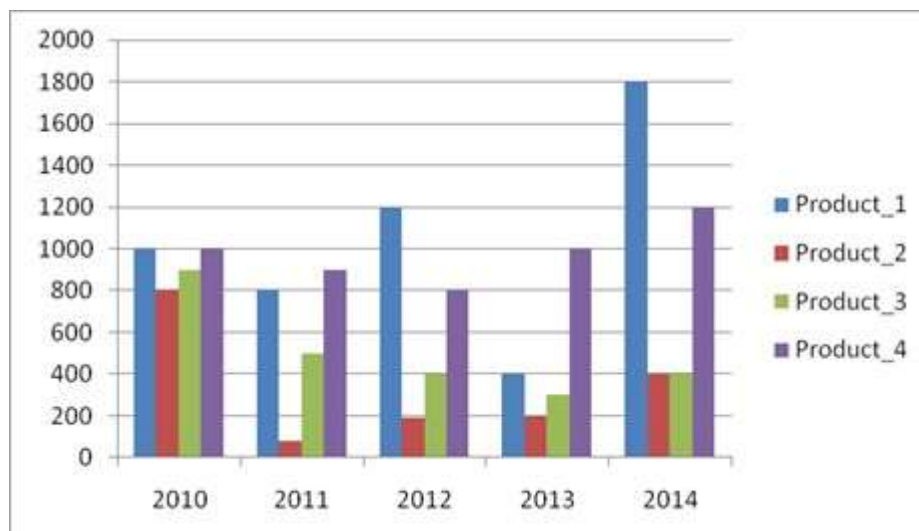
2. Now select data range: By using the mouse high light the range of data you want to take

	A	B	C	D	E	F
1		Product_1	Product_2	Product_3	Product_4	
2	2010	1000	800	900	1000	
3	2011	800	80	500	900	
4	2012	1200	190	400	800	
5	2013	400	200	300	1000	
6	2014	1800	400	400	1200	
7						

3. Click Insert Tab and select a chart type from the chart group and Select the sub type of chart ( In this example selected a 2D Column chart)

Insert --> Chart Group --> Select Column Chart

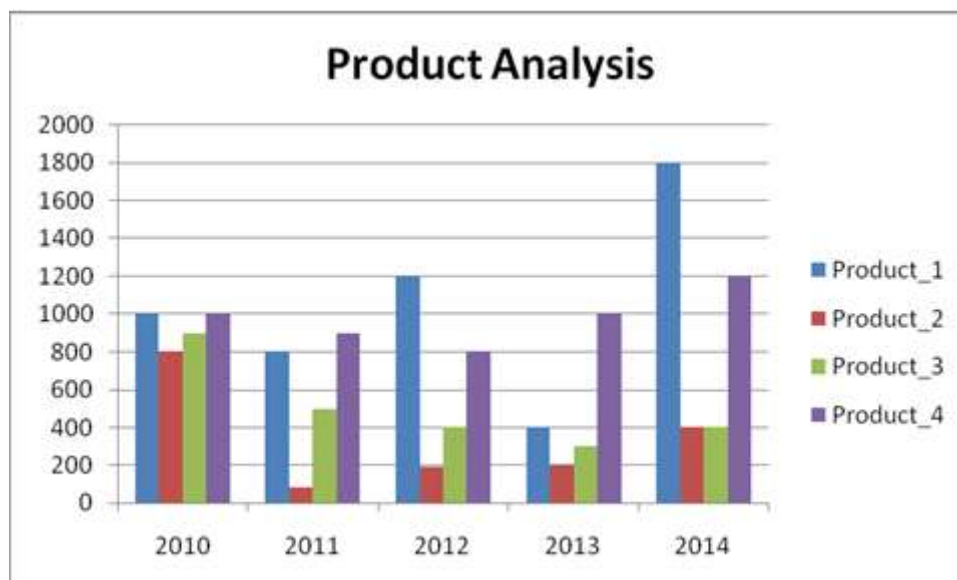
4. The Chart will be displayed as follows



5. Select the Title of the chart

- To give a title to a chart, click on the chart. Now you can see layout tab available. Click on Layout tab.
- Choose(click) on chart title option available in the Label group
- Click on the chart title and write a title “Product Analysis”.

Select Chart--> Layout-->Chart Title

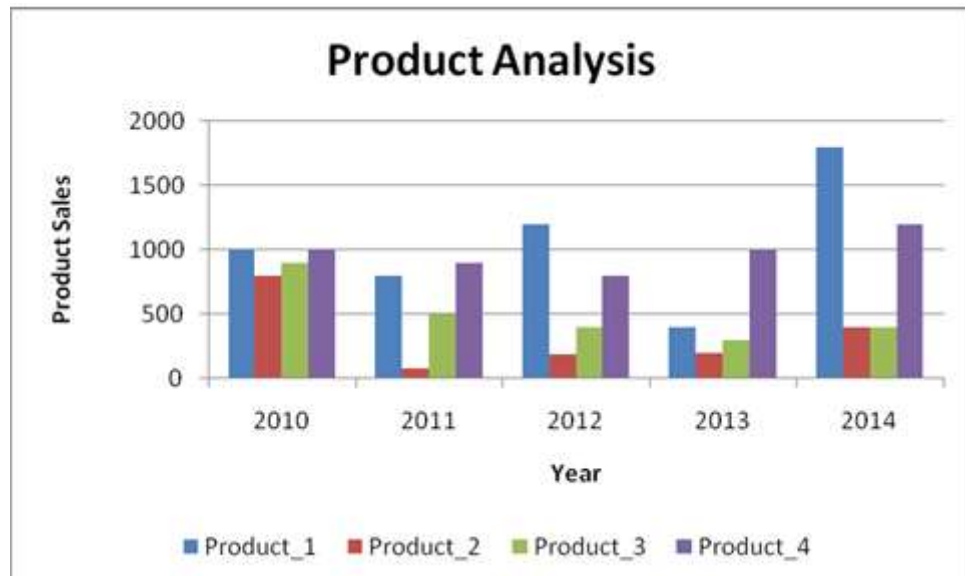


6. Give a name to X-Axis

- Click on Layout tab.
- Then select Axis Titles from Labels Group.
- Select Primary Horizontal Axis Title, as shown below.

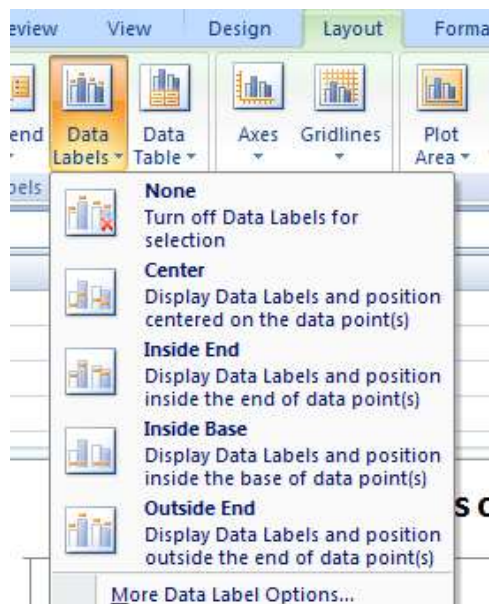
- Now, click on the Axis Title and write an X-axis title “Year”.
- Follow the same steps to give a title to Y-axis “Product Sales”.

**Select Chart-->Layout-->Axis Title-->Primary Horizontal Axis Title**

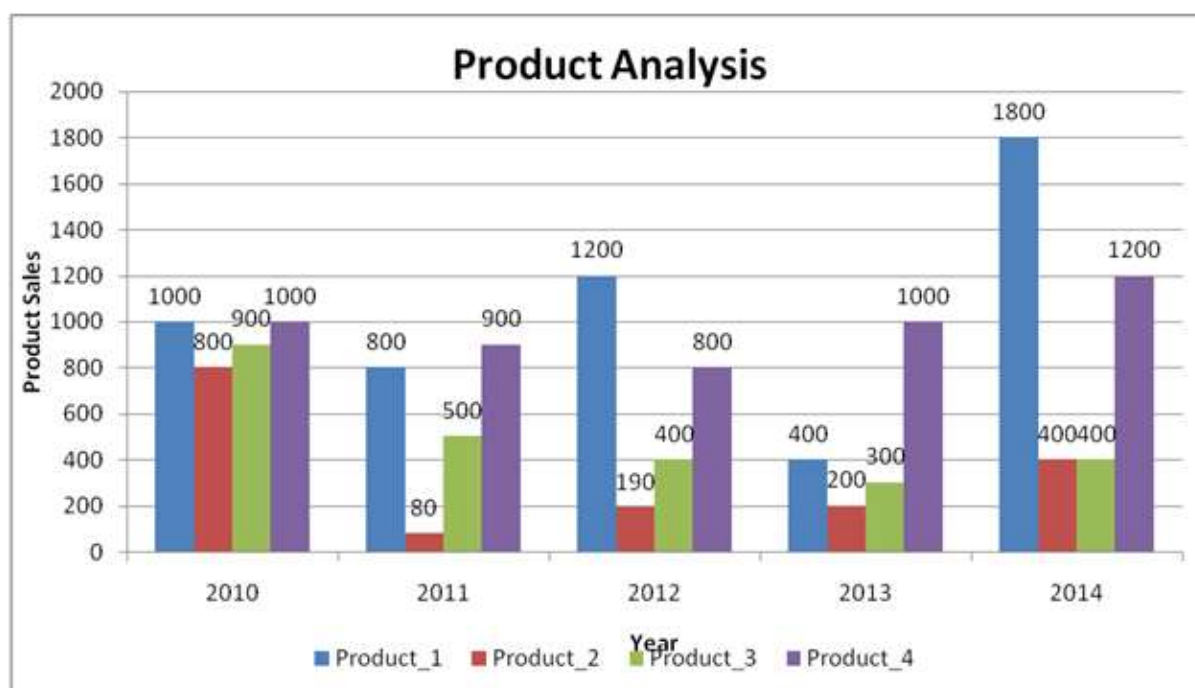


#### 7. Adding Data Labels to the chart

- Click on Layout tab.
- Then click on Data Label option available in Labels Group.
- Now choose a format to display data labels





**Output:**

**4. Create an electronic spread sheet which shows the sales of different products for 5 years. Create pie chart for the following data**

Year	Product_1	Product_2	Product_3	Product_4
2010	1000	800	900	1000
2011	800	80	500	900
2012	1200	190	400	800
2013	400	200	300	1000
2014	1800	400	400	1200

**Creating the spread with different products of 5 year**

Before you can make a chart, you must first enter data into a worksheet. To create the spread sheet with 5 years different products follow the below steps

**Step1:**

Click on the cell B1 and type “Product\_1” and go to the next cell by clicking the Tab key on the keyboard.

**Step 2:**

In the cell C1 type “Product\_2” and click on tab key to activate next cell. Repeat the above to steps to enter “Product\_3”, “Product\_4”



**Step3:**

Click on cell “A2” and type “2010” then press on tab key to activate the next cell.

**Step 4:**

Repeat the above procedure to enter the details of four products for five years. Finally we have the data of products as follows

	A	B	C	D	E	F
1		<b>Product_1</b>	<b>Product_2</b>	<b>Product_3</b>	<b>Product_4</b>	
2	<b>2010</b>	1000	800	900	1000	
3	<b>2011</b>	800	80	500	900	
4	<b>2012</b>	1200	190	400	800	
5	<b>2013</b>	400	200	300	1000	
6	<b>2014</b>	1800	400	400	1200	
7						

**Creating a Pie chart for the above data****Definition of Chart:**

Charts allow you to present data entered into the worksheet in a visual format using a variety of graph types.

Following steps are given to draw a Chart

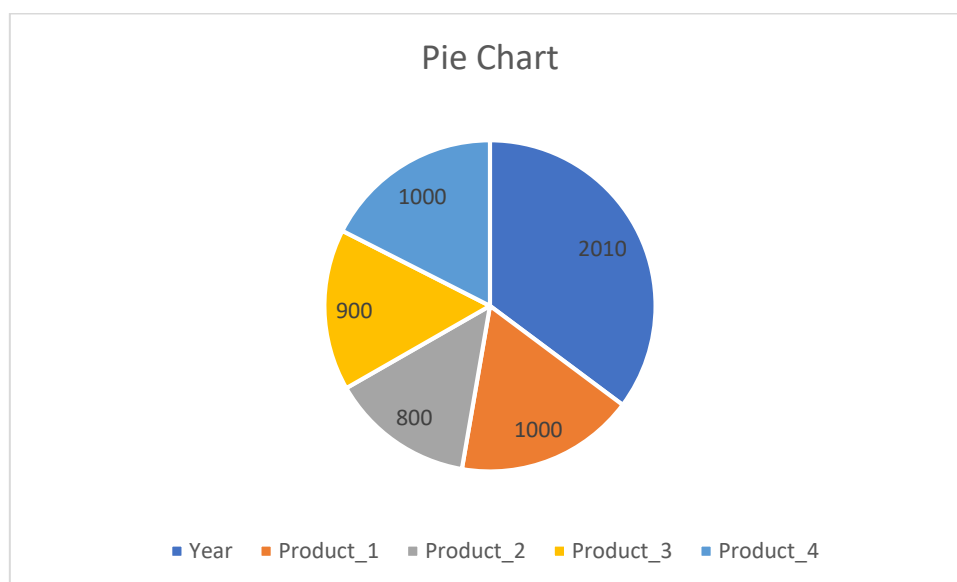
1. Enter data in the work sheet: Suppose you entered data as given in below

	A	B	C	D	E	F
1		<b>Product_1</b>	<b>Product_2</b>	<b>Product_3</b>	<b>Product_4</b>	
2	<b>2010</b>	1000	800	900	1000	
3	<b>2011</b>	800	80	500	900	
4	<b>2012</b>	1200	190	400	800	
5	<b>2013</b>	400	200	300	1000	
6	<b>2014</b>	1800	400	400	1200	
7						

2. Now select data range: By using the mouse high light the range of data you want to take

	A1					
	A	B	C	D	E	F
1		Product_1	Product_2	Product_3	Product_4	
2	2010	1000	800	900	1000	
3	2011	800	80	500	900	
4	2012	1200	190	400	800	
5	2013	400	200	300	1000	
6	2014	1800	400	400	1200	
7						

**Output:**



**5. Create an electronic spread sheet which shows the sales of different products for 5 years. Create bar chart for the following data**

Year	Product_1	Product_2	Product_3	Product_4
2010	1000	800	900	1000
2011	800	80	500	900
2012	1200	190	400	800
2013	400	200	300	1000
2014	1800	400	400	1200

**Creating the spread with different products of 5 year**

Before you can make a chart, you must first enter data into a worksheet. To create the spread sheet with 5 years different products follow the below steps

**Step1:**

Click on the cell B1 and type “Product\_1” and go to the next cell by clicking the Tab key on the keyboard.

**Step 2:**

In the cell C1 type “Product\_2” and click on tab key to activate next cell. Repeat the above to steps to enter “Product\_3”, “Product\_4”

**Step3:**

Click on cell “A2” and type “2010” then press on tab key to activate the next cell.

**Step 4:**

Repeat the above procedure to enter the details of four products for five years. Finally we have the data of products as follows

	A	B	C	D	E	F
1		Product_1	Product_2	Product_3	Product_4	
2	2010	1000	800	900	1000	
3	2011	800	80	500	900	
4	2012	1200	190	400	800	
5	2013	400	200	300	1000	
6	2014	1800	400	400	1200	
7						

**Creating a Pie chart for the above data**

**Definition of Chart:**

Charts allow you to present data entered into the worksheet in a visual format using a variety of graph types.

Following steps are given to draw a Chart

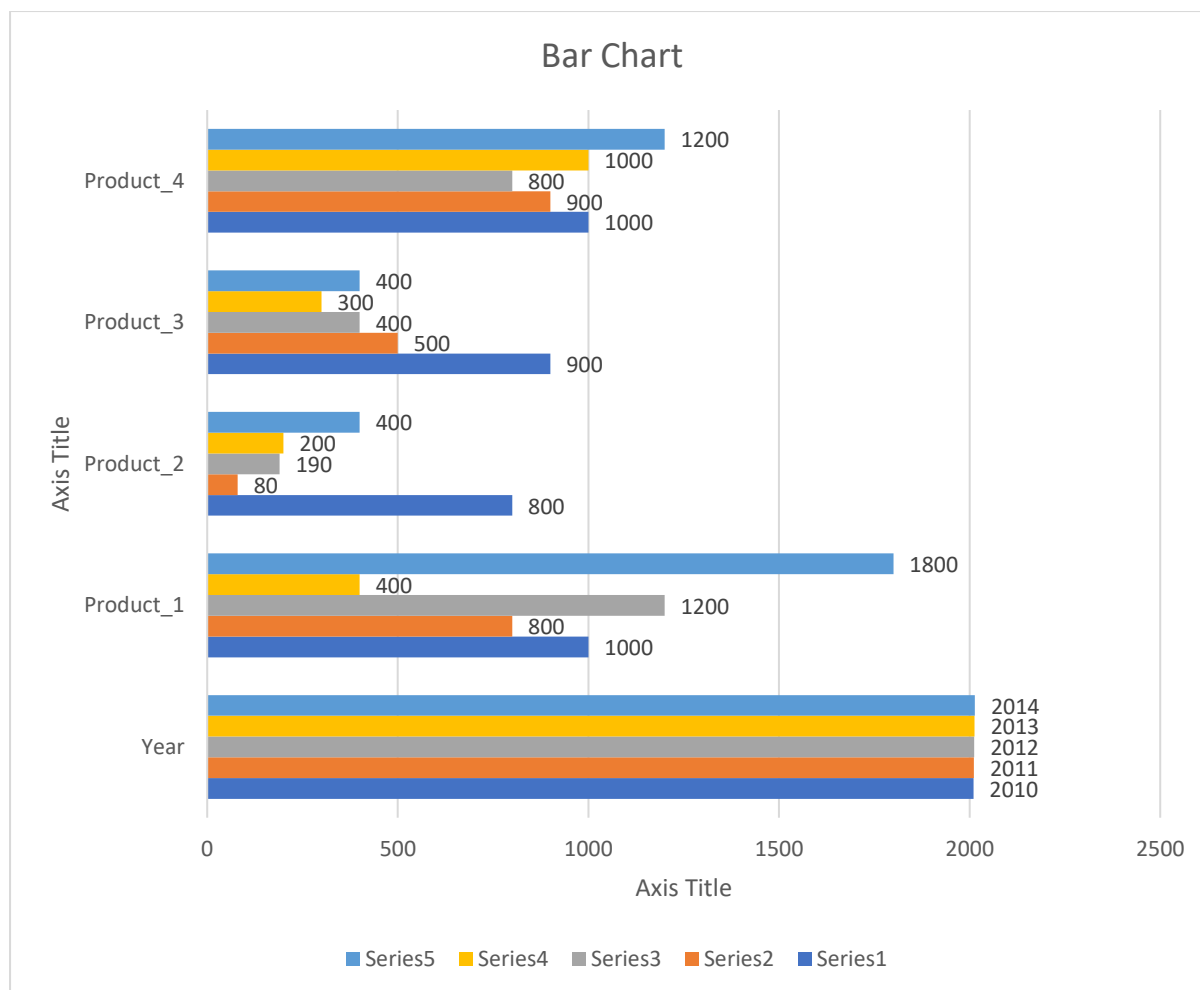
1. Enter data in the work sheet: Suppose you entered data as given in below

D15		fx				
	A	B	C	D	E	
1		<b>Product_1</b>	<b>Product_2</b>	<b>Product_3</b>	<b>Product_4</b>	
2	<b>2010</b>	1000	800	900	1000	
3	<b>2011</b>	800	80	500	900	
4	<b>2012</b>	1200	190	400	800	
5	<b>2013</b>	400	200	300	1000	
6	<b>2014</b>	1800	400	400	1200	
7						

2. Now select data range: By using the mouse high light the range of data you want to take

A1		fx				
	A	B	C	D	E	F
1		<b>Product_1</b>	<b>Product_2</b>	<b>Product_3</b>	<b>Product_4</b>	
2	<b>2010</b>	1000	800	900	1000	
3	<b>2011</b>	800	80	500	900	
4	<b>2012</b>	1200	190	400	800	
5	<b>2013</b>	400	200	300	1000	
6	<b>2014</b>	1800	400	400	1200	
7						

**Output:**



## 6. To Create an employee's salary statement worksheet using MS-Excel

**Aim: -**

To create worksheet with following fields Empno, Ename, Basic pay(BP), Travelling Allowance(TA), Dearness Allowance(DA), House Rent allowance(HRA), Income Tax(IT), Provident Fund(PF), Net Pay(NP)

Given: DA= 30% of BP, HRA=20% of BP, TA=17.5% of BP, IT=15% of BP, PF=12.5% of BP

**Procedure: -**

1. Create an Excel Worksheet for an employee pay roll system.
2. Enter the details of Employee as given and calculate the DA, TA, HRA, IT, PF as a percentage on the basis of Basic Pay.
3. Calculate the Net Pay by using the formulae.

**Gross Pay= DA+TA+HRA+BP**

**Deductions=IT+PF**

**Net Pay= Gross Pay-Deductions**

Empno	Ename	Basic Pay	TA	DA	HRA	Gross Sal	Deductions		Ded. Totals	Net Sal
							I.T	PF		
101	Anil Kumar	5500	962.5	1350	1100	9212.5	825	687.5	1512.5	7700
102	R. Madhu	6000	1050	1300	1200	10050	900	750	1650	8400
103	Ravi Kiran	7500	1312.5	2250	1500	12562.5	1125	937.5	2062.5	10500
104	R. Naresh	4500	787.5	1350	900	7537.5	675	562.5	1237.5	6300
105	Sunil	3500	612.5	1050	700	5862.5	525	437.5	962.5	4900

### ***Module – III Working with MS-PowerPoint***

***SL. No.***

***Experiments***

1. ***Make a Power point presentation of all the details of the books that you had studied in B.Sc. First Year.***

PowerPoint is presentation software that can be used to create slide shows for printing, on-screen projection, or Web-based display.

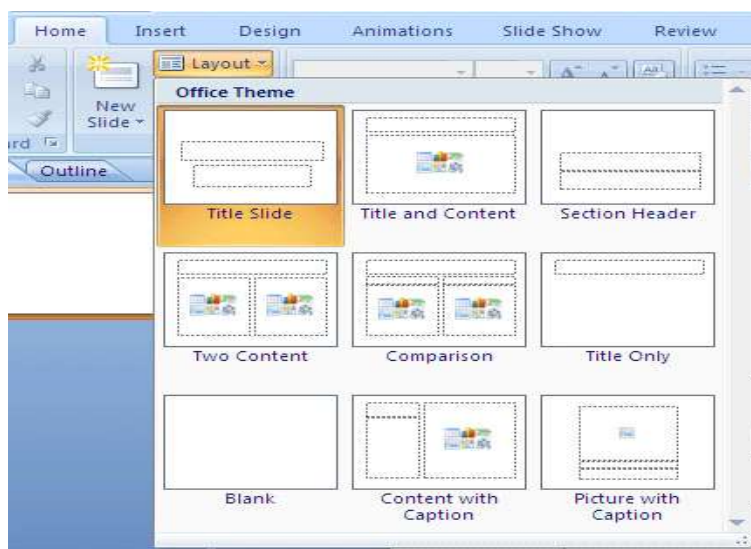
## Opening PowerPoint

**NOTE:** The procedure for opening PowerPoint may vary depending on the setup of your computer.

To open PowerPoint in Windows, click on the Start button --> Programs --> Microsoft PowerPoint

When **PowerPoint 2007** is opened, a blank *Title* slide appears by default as the first slide in your new presentation. You can start a new presentation when you first open PowerPoint or after PowerPoint is already open.

To change the **layout** of an open slide, click on the **Layout** button in the **Home** tab.



## Inserting Slides

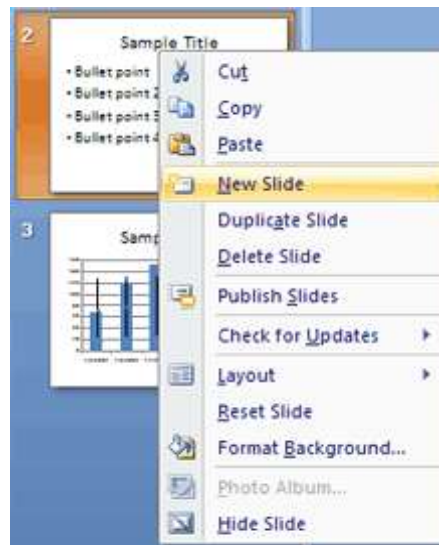
There are two ways to add new slides to your presentation.

- Quick Menu Option
- Ribbon Option

### Quick Menu Options

To insert a new slide using the Quick Menu, in the Slides panel **right click** the slide after which you want a new slide inserted and select **New Slide**.





To change the layout of the slide, **right click** the new slide in the Slides panel, select **Layout** and select the desired theme.

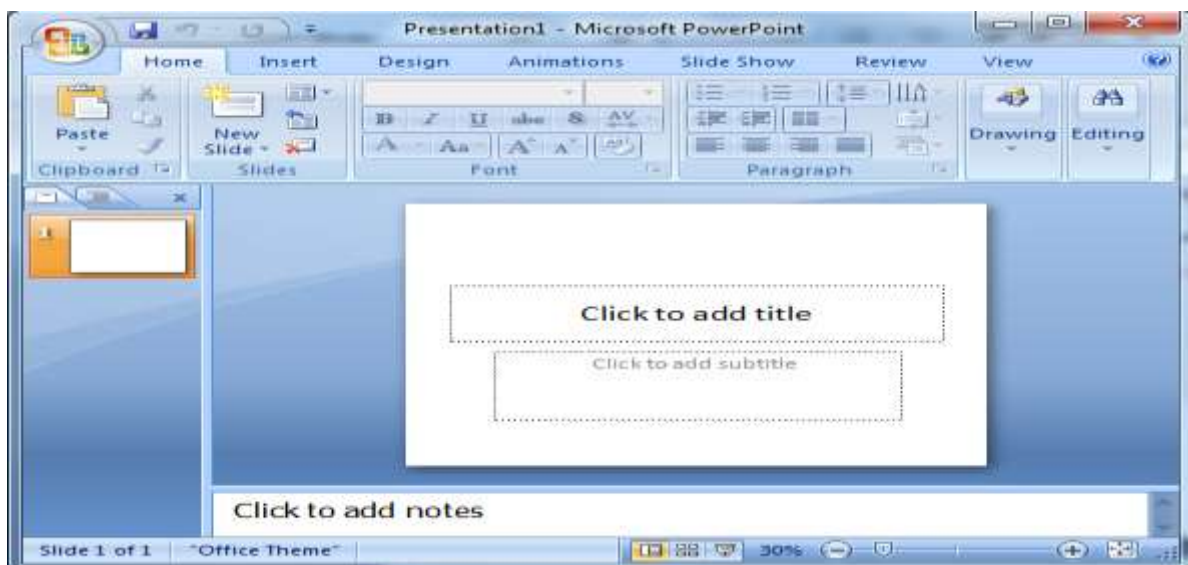
### Ribbon Option

From the **Home tab** in the Slides group, click on **New Slide**. A blank slide will be inserted after your active slide.



If you wish to choose the layout while creating your new slide, click the ▼ on the **New Slide** button and choose a theme.

Select the new slide from Quick menu option or Ribbon Option:





### Apply a new layout to a slide

To change the layout of an existing slide, do the following:

1. On the **Slides** tab, click the slide that you want to apply a new layout to.
2. On the **Home** tab, in the **Slides** group, click **Layout**, and then click the new layout that you want.

### Copy a slide

To copy a slide does the following

1. On the **Slides** tab, right-click the slide that you want to copy, and then click **Copy** on the shortcut menu.
2. Still on the **Slides** tab, right-click where you want to add the new copy of the slide, and then click **Paste** on the shortcut menu.

You can also insert a copy of a slide from one presentation into another presentation.

### Rearrange the order of slides

- On the **Slides** tab, click the slide that you want to move, and then drag it to the location that you want.

To select multiple slides, click a slide that you want to move, and then press and hold CTRL while you click each of the other slides that you want to move.

### Delete a slide

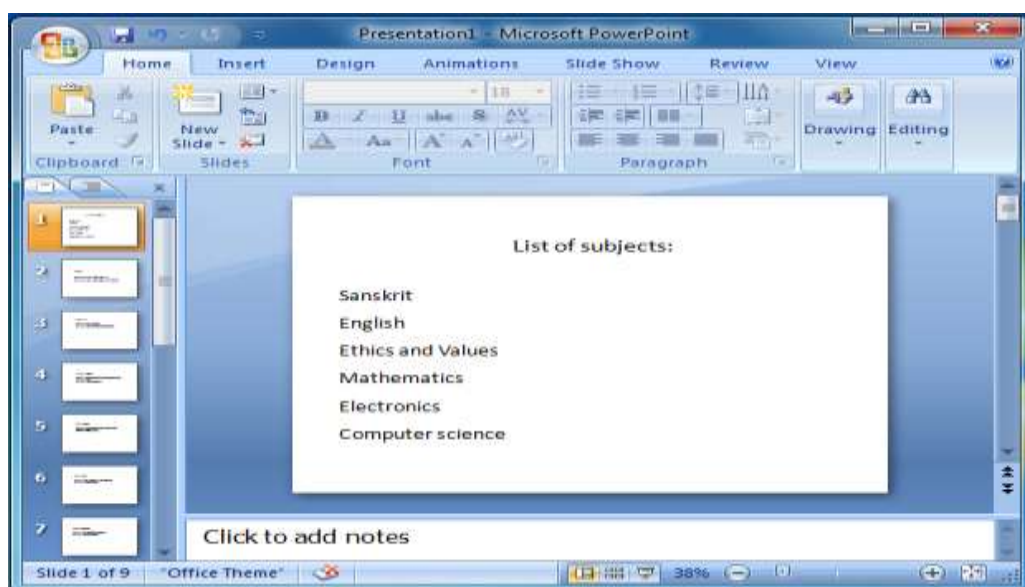
- On the **Slides** tab, right-click the slide that you want to delete, and then click **Delete Slide** on the shortcut menu.

### Adding text to a slide

The Title Slide layout contains text boxes for a title and a subtitle. Try typing text into these boxes.

1. Click in the Title text box. A dashed line border with a circle in each corner and a square box at each midpoint appears around the text box indicating that it is selected.

Adding list of subjects that studying in B.Sc. First Year into the slides:



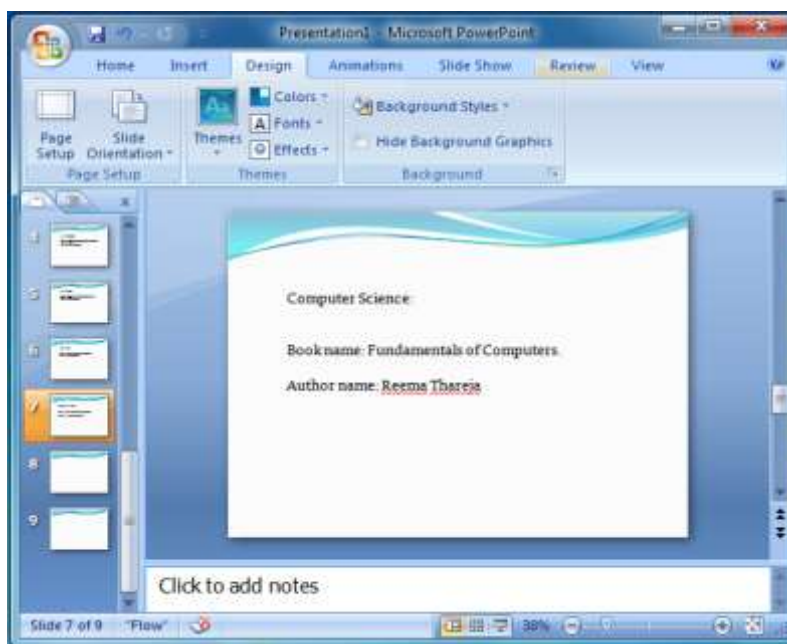
2. Click the Subtitle text box and type a subtitle.

### Selecting a Theme or Designing Slides

Design themes are a convenient way to add a professional flair to your presentation. Themes include preset colors, fonts, backgrounds, and formatting effects. PowerPoint provides you with the option to customize one of their existing themes or to build your own.

### Selecting a Theme

To choose a Theme for an open slide, use the **Theme group** under the **Design tab**. Use the arrows on the right of the Theme group to scroll through the themes, or to see all available themes at once. When you hold your mouse over any of the examples, PowerPoint will show you a preview of the slide.




## Animating Text and Images

### Adding Animation to Slides

1. Click on the object or text box you wish to animate (hold down the Ctrl button while clicking to select more than one).
2. In the **Animations tab** under the **Animations group**, select an option from the **Animate** pull-down list. As you move your mouse over each choice PowerPoint will preview the effect on your slide.
3. Repeat for any other slides or objects you wish to animate.

### Run Your PowerPoint Slide Show

After you create your slides, you can run your slide show:

- Press F5.
- Choose the Slide Show tab. Click the From Beginning button  in the Start Slide Show group.
- Click the Slide Show icon in the bottom-right corner of your screen.

### Saving your presentation

If you are saving a document for the first time, you will need to use the **Save As** command; however, if you have already saved a presentation, you can use the **Save** command.

To use the Save As command:

- Click the **Microsoft Office button**.
- Select **Save As**. A menu will appear.
- Select the type of file you want to save the presentation as
- Enter a **name** for the document.
- Click the **Save** button.

To use the Save command:

- Click the **Microsoft Office button**.
- Select **Save** from the menu.

Using the Save command saves the document in its current location using the same file name.