Unit 1 - Introduction to HTML

Module 1 - Introduction to HTML

HTML tutorial or HTML 5 tutorial provides basic and advanced concepts of HTML. Our HTML tutorial is developed for beginners and professionals. In our tutorial, every topic is given step-by-step so that you can learn it in a very easy way. If you are new in learning HTML, then you can learn HTML from basic to a professional level and after learning HTML with CSS and JavaScript you will be able to create your own interactive and dynamic website. But Now We will focus on HTML only in this tutorial.

The major points of HTML are given below:

- HTML stands for HyperText Markup Language.
- HTML is used to create web pages and web applications.
- HTML is widely used language on the web.
- We can create a static website by HTML only.
- Technically, HTML is a Markup language rather than a programming language.HTML Example with HTML Editor

Let's see a simple example of HTML.

```
<!DOCTYPE>
<html>
<head>
<title>Web page title</title>
</head>
<body>
<h1>Write Your First Heading</h1>
<p>Write Your First Paragraph.</p>
</body>
</html>
```

What is HTML

HTML is an acronym which stands for **Hyper Text Markup Language** which is used for creating web pages and web applications. Let's see what is meant by Hypertext Markup Language, and Web page.

Hyper Text: HyperText 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. HyperText is a way to link two or more web pages (HTML documents) with each other.

Markup language: A markup language is a computer language that is used to apply layout and formatting conventions to a text document. Markup language makes text more interactive and dynamic. It can turn text into images, tables, links, etc.

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**.

Hence, HTML is a markup language which is used for creating attractive web pages with the help of styling, and which looks in a nice format on a web browser. An HTML document is made of many HTML tags and each HTML tag contains different content.

Description of HTML Example

<!DOCTYPE>: It defines the document type or it instruct the browser about the version of HTML.

<html >: This tag informs the browser that it is an HTML document. Text between html tag describes the web document. It is a container for all other elements of HTML except <!DOCTYPE>

<head>: It should be the first element inside the <html> element, which contains the metadata (information about the document). It must be closed before the body tag opens.

<title>: As its name suggested, it is used to add title of that HTML page which appears at the top of the browser window. It must be placed inside the head tag and should close immediately. (Optional)

<body>: Text between body tag describes the body content of the page that is visible to the end user. This tag contains the main content of the HTML document.

<h1>: Text between <h1> tag describes the first level heading of the webpage.

<p>: Text between <p> tag describes the paragraph of the webpage.

Brief History of HTML

In the late 1980's, a physicist, Tim Berners-Lee who was a contractor at CERN, proposed a system for CERN researchers. In 1989, he wrote a memo proposing an internet based hypertext system.

Tim Berners-Lee is known as the father of HTML. The first available description of HTML was a document called "HTML Tags" proposed by Tim in late 1991. The latest version of HTML is HTML5, which we will learn later in this tutorial.

Features of HTML

1) It is a very **easy and simple language**. It can be easily understood and modified.
2) It is very easy to make an **effective presentation** with HTML because it has a lot of formatting tags.
3) It is a **markup language**, so it provides a flexible way to design web pages along with the text.
4) It facilitates programmers to add a **link** on the web pages (by html anchor tag), so it enhances the interest of browsing of the user.
5) It is **platform-independent** because it can be displayed on any platform like Windows, Linux, and Macintosh, etc.
6) It facilitates the programmer to add **Graphics, Videos, and Sound** to the web pages which makes it more attractive and interactive.
7) HTML is a case-insensitive language, which means we can use tags either in lower-case or upper-case.

NOTE: It is recommended to write all tags in lower-case for consistency, readability, etc.

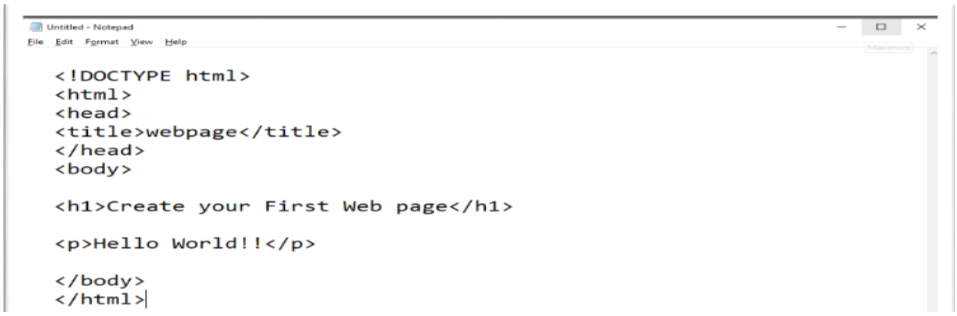
HTML text Editors

- An HTML file is a text file, so to create an HTML file we can use any text editors.
- Text editors are the programs which allow editing in a written text, hence to create a web page we need to write our code in some text editor.
- There are various types of text editors available which you can directly download, but for a beginner, the best text editor is Notepad (Windows) or TextEdit (Mac).
- After learning the basics, you can easily use other professional text editors which are, **Notepad++, Sublime Text, Vim, etc**.

Notepad is a simple text editor and suitable for beginners to learn HTML. It is available in all versions of Windows, from where you easily access it.

Step 1: Open Notepad (Windows)

Step 2: Write code in HTML

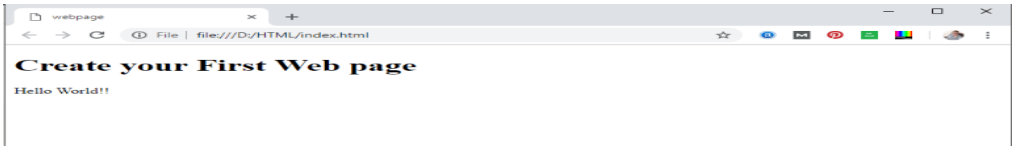


Step 3: Save the HTML file with .htm or .html extension.

Step 4: Open the HTML page in your web browser.

To run the HTML page, you need to open the file location, where you have saved the file and then either double-click on file or click on open with option

Output:



Building blocks of HTML

An HTML document consist of its basic building blocks which are:

- **Tags:** An HTML tag surrounds the content and apply meaning to it. It is written between < and > brackets.
- **Attribute:** An attribute in HTML provides extra information about the element, and it is applied within the start tag. An HTML attribute contains two fields: name & value.

Elements: An HTML element is an individual component of an HTML file. In an HTML file, everything written within tags is termed as HTML elements.

HTML Elements

An HTML element is defined by a start tag, some content, and an end tag.

An HTML element is defined by a start tag, some content, and an end tag:
<tagname>Content goes here...</tagname>
The HTML **element** is everything from the start tag to the end tag:
<h1>My First Heading</h1>
My first paragraph

Start tag	Element content	End tag
<h1>	My First Heading	</h1>
	My first paragraph.	
 	None	*none*

Nested HTML Elements
HTML elements can be nested (this means that elements can contain other elements).
All HTML documents consist of nested HTML elements.
The following example contains four HTML elements (<html>, <body>, <h1> and):
Example: (Try it now)

```
<!DOCTYPE html>
<html>
<body>
<h1>My First Heading</h1>
<p>My first paragraph</p>
</body>
</html>
```

Output:
My First Heading
My first paragraph
Empty HTML Elements
HTML elements with no content are called empty elements.
The
 tag defines a line break, and is an empty element without a closing tag:
**
 Tag**: br stands for break line, it breaks the line of the code.
<hr> Tag: hr stands for Horizontal Rule. This tag is used to put a line across the webpage.
Example:

```
<!DOCTYPE html>
<html>
<body>
<p>This is a <br> paragraph with a line break</p>
</body>
</html>
```

Output:
Thisis a
paragraph with a line break
HTML Tags
HTML tags are like keyword which defines that how web browser will format and display the content. With the help of tags, a web browser can distinguish between an HTML content and a simple content. HTML tags contain three main parts: opening tag, content and closing tag. But some HTML tags are unclosed tags. When a web browser reads an HTML document, browser reads it from top to bottom and left to right. HTML tags are used to create HTML documents and render their properties. Each HTML tags have different properties. An HTML file must have some essential tags so that web browser can differentiate between a simple text and HTML text. You can use as many tags you want as per your code requirement.

- All HTML tags must enclose within <> these brackets.
- Every tag in HTML perform different tasks.
- If you have used an open tag <tag>, then you must use a close tag </tag> (except some tags)

Syntax
<tag> content </tag>

HTML Tag Examples

HTML Code

```
<!DOCTYPE>
<html>
<body>
<p> Paragraph Tag </p>
<h2> Heading Tag </h2>
<b> Bold Tag </b>
<i> Italic Tag </i>
<u> Underline Tag</u>
</body>
</html>
```

Output:

Paragraph Tag

Heading Tag

Bold Tag *Italic Tag* Underline Tag

Unclosed/Empty HTML Tags

Some HTML tags are not closed, for example br and hr.

**
 Tag**: br stands for break line, it breaks the line of the code.

<hr> Tag: hr stands for Horizontal Rule. This tag is used to put a line across the webpage.

HTML Meta Tags

DOCTYPE, title, link, meta and style

HTML Text Tags

<p>, <h1>, <h2>, <h3>, <h4>, <h5>, <h6>, , , <abbr>, <acronym>, <address>, <bdo>, <blockquote>, <cite>, <q>, <code>, <ins>, <del>, <dfn>, <kbd>, <pre>, <samp>, <var> and

HTML Link Tags

<a> and <base>

HTML Image and Object Tags

<img>, <area>, <map>, <param> and <object>

HTML List Tags

, <ol>, <li>, <dl>, <dt> and <dd>

HTML Table Tags

table, tr, td, th, tbody, thead, tfoot, col, colgroup and caption

HTML Form Tags

form, input, textarea, select, option, optgroup, button, label, fieldset and legend

HTML Scripting Tags(script and noscript)

Note: We will see examples using these tags in later charters.

The <!DOCTYPE> Declaration

The <!DOCTYPE> declaration represents the document type, and helps browsers to display web pages correctly. It must only appear once, at the top of the page (before any HTML tags).

The <!DOCTYPE> declaration is not case sensitive.

HTML Text Formatting Elements/Tags

HTML contains several elements for defining text with a special meaning.

Example : (Try it now)

```
<!DOCTYPE html>
<html>
<body>
<p><b>This text is bold</b></p>
<p><i>This text is italic</i></p>
<p>This is<sub> subscript</sub> and <sup>superscript</sup></p>
</body>
</html>
```

HTML Formatting

HTML Formatting is a process of formatting text for better look and feel. HTML provides us ability to format text without using CSS. There are many formatting tags in HTML. These tags are used to make text bold, italicized, or underlined. There are almost 14 options available that how text appears in HTML and XHTML.

In HTML the formatting tags are divided into two categories:

- Physical tag: These tags are used to provide the visual appearance to the text.
- Logical tag: These tags are used to add some logical or semantic value to the text.

NOTE: There are some physical and logical tags which may give same visual appearance, but they will be different in semantics.

Here, we are going to learn 14 HTML formatting tags. Following is the list of HTML formatting text.

Element name	Description
<b>	This is a physical tag, which is used to bold the text written between it.
	This is a logical tag, which tells the browser that the text is important.
<i>	This is a physical tag which is used to make text italic.
	This is a logical tag which is used to display content in italic.
<mark>	This tag is used to highlight text.
<u>	This tag is used to underline text written between it.
<tt>	This tag is used to appear a text in teletype. (not supported in HTML5)
<strike>	This tag is used to draw a strikethrough on a section of text. (Not supported in HTML5)
<sup>	It displays the content slightly above the normal line.
<sub>	It displays the content slightly below the normal line.
<del>	This tag is used to display the deleted content.
<ins>	This tag displays the content which is added
<big>	This tag is used to increase the font size by one conventional unit.
<small>	This tag is used to decrease the font size by one unit from base font size.

The HTML tag is a logical tag, which displays the content in bold font and informs the browser about its logical importance. If you write anything between ? , is shown important text.

See this example:

```
<p><strong>This is an important content</strong>, and this is normal content</p>
```

Output:

This is an important content, and this is normal content

The HTML tag is a logical element, which will display the enclosed content in italic font, with added semantics importance.

See this example:

```
<p><em>This is an important content</em>, which displayed in italic font.</p>
```

Output:

This is an important content, which displayed in italic font.

3) HTML Marked formatting

If you want to mark or highlight a text, you should write the content within <mark>.....</mark>.

See this example:

```
<h2> I want to put a <mark> Mark</mark> on your face</h2>
```

Output:

I want to put a Mark on your face

4) Underlined Text

If you write anything within <u>.....</u> element, is shown in underlined text.

See this example:

```
<p> <u>Write Your First Paragraph in underlined text.</u></p>
```

Output:

Write Your First Paragraph in underlined text.

5) Strike Text

Anything written within <strike>.....</strike> element is displayed with strikethrough. It is a thin line which cross the statement.

See this example:

```
<p> <strike>Write Your First Paragraph with strikethrough</strike>.</p>
```

Output: ~~Write Your First Paragraph with strikethrough.~~

6) Monospaced Font

If you want that each letter has the same width, then you should write the content within <tt>.....</tt> element.

Note: We know that most of the fonts are known as variable-width fonts because different letters have different width. (for example: 'w' is wider than 'i'). Monospaced Font provides similar space among every letter.

See this example:

```
<p>Hello <tt>Write Your First Paragraph in monospaced font.</tt></p>
```

Output:

Hello Write Your First Paragraph in monospaced font.

7) Superscript Text

If you put the content within ^{.....} element, is shown in superscript; means it is displayed half a character's height above the other characters.

See this example:

```
<p>Hello <sup>Write Your First Paragraph in superscript.</sup></p>
```

Output:

Hello Write Your First Paragraph in superscript.

8) Subscript Text

If you put the content within _{.....} element, is shown in subscript ; means it is displayed half a character's height below the other characters.

See this example:

```
<p>Hello <sub>Write Your First Paragraph in subscript.</sub></p>
```

Output:

Hello Write Your First Paragraph in subscript.

9) Deleted Text

Anything that puts within ~~.....~~ is displayed as deleted text.

See this example:

```
<!DOCTYPE>
<html>
<body>
<p>Hello <del>Delete your first paragraph</del></p>
</body>
</html>
```

Output:

Hello ~~Delete your first paragraph.~~

10) Inserted Text

Anything that puts within <ins>.....</ins> is displayed as inserted text.

See this example:

```
<p> <del>Delete your first paragraph.</del><ins>Write another paragraph.</ins></p>
```

Output:

~~Delete your first paragraph.~~Write another paragraph.

11) Larger Text

If you want to put your font size larger than the rest of the text then put the content within <big>.....</big>. It increase one font size larger than the previous one.

See this example:

```
<p>Hello <big>Write the paragraph in larger font.</big></p>
```

Output:

Hello Write the paragraph in larger font.

12) Smaller Text

If you want to put your font size smaller than the rest of the text, then put the content within <small>.....</small>tag. It reduces one font size than the previous one.

See this example:

```
<p>Hello <small>Write the paragraph in smaller font.</small></p>
```

Output:

Hello Write the paragraph in smaller font.

A HTML heading or HTML h tag can be defined as a title or a subtitle which you want to display on the webpage. When you place the text within the heading tags <h1>.....</h1>, it is displayed on the browser in the bold format and size of the text depends on the number of heading.
There are six different HTML headings which are defined with the <h1> to <h6> tags, from highest level h1 (main heading) to the least level h6 (least important heading).
h1 is the largest heading tag and h6 is the smallest one. So h1 is used for most important heading and h6 is used for least important.
Headings in HTML helps the search engine to understand and index the structure of web page.
Note: The main keyword of the whole content of a webpage should be display by h1 heading tag.
Heading elements (h1....h6) should be used for headings only. They should not be used just to make text bold or big.

Module 2 – Attributes, Colors, Hyperlinks

HTML Attribute

- HTML attributes are special words which provide additional information about the elements or attributes are the modifier of the HTML element.
- Each element or tag can have attributes, which defines the behavior of that element.
- Attributes should always be applied with start tag.
- The Attribute should always be applied with its name and value pair.
- The Attributes name and values are case sensitive, and it is recommended by W3C that it should be written in Lowercase only.
- You can add multiple attributes in one HTML element, but need to give space between two attributes.

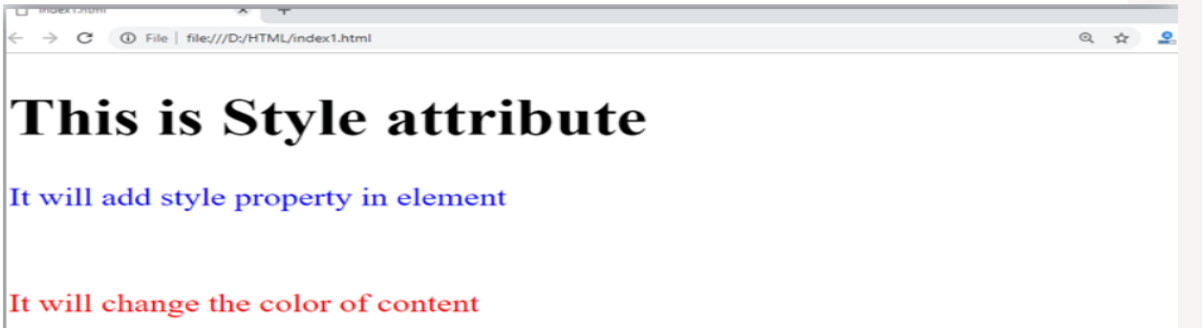
Syntax

```
<element attribute  name="value">content</element>
```

Example

```
<!DOCTYPE html>
<html>
<head>
</head>
<body>
  <h1> This is Style attribute</h1>
  <p style="height: 50px; color: blue">It will add style property in element</p>
  <p style="color: red">It will change the color of content</p>
</body>
</html>
```

Output:



Explanation of above example:

```
<p style="height: 50px; color: blue">It will add style property in element</p>
```

In the above statement, we have used paragraph tags in which we have applied style attribute. This attribute is used for applying CSS property on any HTML element. It provides height to paragraph element of 50px and turns it colour to blue.

```
<p style="color: red">It will change the color of content</p>
```

In the above statement we have again used style attribute in paragraph tag, which turns its colour red.
Note: There are some commonly used attributes are given below, and the complete list and explanation of all attributes are given in HTML attributes List.

The title attribute in HTML

Description: The title attribute is used as text tooltip in most of the browsers. It display its text when user move the cursor over a link or any text. You can use it with any text or link to show the description about that link or text. In our example, we are taking this with paragraph tag and heading tag.

With <h1> tag:

```
<h1 title="This is heading tag">Example of title attribute</h1>
```

The href attribute in HTML

Description: The href attribute is the main attribute of <a> anchor tag. This attribute gives the link address which is specified in that link. The **href** attribute provides the hyperlink, and if it is blank, then it will remain in same page**.**

Example

With link address:

```
<a href="https://www.javatpoint.com/html-anchor">This is a link</a>
```

Without link address:

```
<a href="">This is a link</a>
```

The src Attribute

The **src** attribute is one of the important and required attribute of **<img>** element. It is source for the image

which is required to display on browser. This attribute can contain image in same directory or another directory. The image name or source should be correct else browser will not display the image.

Example

```
<img src="whitepeacock.jpg" height="400" width="600">
```

Note: The above example also have height and width attribute, which define the height and width of image on web page.

Output:

HTML Colors

Colors are very important to give a good look and feel to your website. You can specify colors on page level using <body> tag or you can set colors for individual tags using **bgcolor**attribute.

The <body> tag has following attributes which can be used to set different colors:

- **bgcolor -** sets a color for the background of the page.
- **text -** sets a color for the body text.
- **alink -** sets a color for active links or selected links.
- **link -** sets a color for linked text.
- **vlink -** sets a color for *visited links* - that is, for linked text that you have already clicked on.

HTML Color Coding Methods

- There are following three different methods to set colors in your web page:
- **Color names -** You can specify color names directly like green, blue or red.
- **Hex codes -** A six-digit code representing the amount of red, green, and blue that makes up the color.
- **Color decimal or percentage values -** This value is specified using the rgb() property.

  Now we will see these colouring schemes one by one.

HTML Colors - Color Names

You can specify direct a color name to set text or background color. W3C has listed 16 basic color names that will validate with an HTML validator but there are over 200 different color names supported by major browsers.

Note: Check a complete list of HTML Color Names.

W3C Standard 16 Colors

Here is the list of W3C Standard 16 Colors names and it is recommended to use them.

	Black		Gray		Silver		White
	Yellow		Lime		Aqua		Fuchsia
	Red		Green		Blue		Purple
	Maroon		Olive		Navy		Teal

HTML Colors - Hex Codes

A hexadecimal is a 6-digit representation of a color. The first two digits(RR) represent a red value, the next two are a green value(GG), and the last are the blue value(BB).

A hexadecimal value can be taken from any graphics software like Adobe Photoshop, Paint shop Pro or MS Paint.Each hexadecimal code will be preceded by a pound or hash sign #. Following is a list of few colors using

Color	Color HEX
	#000000
	#FF0000
	#00FF00
	#0000FF
	#FFFF00
	#00FFFF
	#FF00FF
	#C0C0C0
	#FFFFFF

hexadecimal notation.

Example

Here are the examples to set background of an HTML tag by color code in hexadecimal:

```
<!DOCTYPE html>
<html>
<head>
<title>HTML Colors by Hex</title>
</head>
<body text="#0000FF" bgcolor="#00FF00">
<p>Use different color hexa for for body and table and see the result.</p>
<table bgcolor="#000000">
<tr>
<td>
<font color="#FFFFFF">This text will appear white on black background.</font>
</td>
</tr>
</table>
</body>
</html>
```

HTML Colors - RGB Values

This color value is specified using the **rgb( )**property. This property takes three values, one each for red, green, and blue. The value can be an integer between 0 and 255 or a percentage.

Note: All the browsers does not support rgb() property of color so it is recommended not to use it.

Color	Color RGB
	rgb(0,0,0)
	rgb(255,0,0)
	rgb(0,255,0)
	rgb(0,0,255)
	rgb(255,255,0)
	rgb(0,255,255)
	rgb(255,0,255)
	rgb(192,192,192)
	rgb(255,255,255)

Example

Here are the examples to set background of an HTML tag by color code using rgb() values:

```
<!DOCTYPE html>
<html>
<head>
<title>HTML Colors by RGB code</title>
</head>
<body text="rgb(0,0,255)" bgcolor="rgb(0,255,0)">
<p>Use different color code for for body and table and see the result.</p>
```

```
<table bgcolor="rgb(0,0,0)">
<tr>
<td>
<font color="rgb(255,255,255)">This text will appear white on black
background.</font>
</td>
</tr>
</table>
</body>
</html>
```

Background Color Example

Example

```
<!DOCTYPE html>
<html>
<body>

<h1 style="background-color:DodgerBlue;">Hello World</h1>

<p style="background-color:Tomato;">
Lorem ipsum dolor sit amet, consectetuer adipiscing elit, sed diam nonummy nibh euismod
tincidunt ut laoreet dolore magna aliquam erat volutpat.
Ut wisi enim ad minim veniam, quis nostrud exerci tation ullamcorper suscipit lobortis
nisl ut aliquip ex ea commodo consequat.
</p>

</body>
</html>
```

Border ColorExample

```
<!DOCTYPE html>
<html>
<body>

<h1 style="border: 2px solid Tomato;">Hello World</h1>

<h1 style="border: 2px solid DodgerBlue;">Hello World</h1>

<h1 style="border: 2px solid Violet;">Hello World</h1>

</body>
</html>
```

HTML Anchor- Hyper Link

The **HTML anchor tag** defines *a hyperlink that links one page to another page*. It can create hyperlink to other web page as well as files, location, or any URL. The "href" attribute is the most important attribute of the HTML a tag. and which links to destination page or URL.

href attribute of HTML anchor tag

The href attribute is used to define the address of the file to be linked. In other words, it points out the destination page.

The syntax of HTML anchor tag is given below.

```
<a href = "....."> Link Text </a>
```

Let's see an example of HTML anchor tag.

```
<a href="second.html">Click for Second Page</a>
```

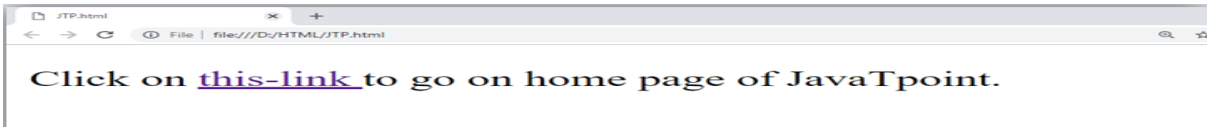
Specify a location for Link using target attribute

If we want to open that link to another page then we can use target attribute of <a> tag. With the help of this link will be open in next page.

Example:

```
<!DOCTYPE html>
<html>
<head>
  <title></title>
</head>
<body>
<p>Click on <a href="https://www.javatpoint.com/" target="_blank"> this-link </a>to go on home page of JavaTpoint.
</p>
</body>
</html>
```

Output:



Note:

- The **target** attribute can only use with href attribute in anchor tag.
- If we will not use target attribute, then link will open in same page.

Appearance of HTML anchor tag

An **unvisited link** is displayed underlined and blue.
A **visited link** displayed underlined and purple.
An **active link** is underlined and red.

Module 3 - HTML Image

HTML img tag is used to display image on the web page. HTML img tag is an empty tag that contains attributes only, closing tags are not used in HTML image element.
Let's see an example of HTML image.

```
<h2>HTML Image Example</h2>
<img src="good_morning.jpg" alt="Good Morning Friends"/>
```

Output:

Attributes of HTML img tag

The src and alt are important attributes of HTML img tag. All attributes of HTML image tag are given below.

1) src: It is a necessary attribute that describes the source or path of the image. It instructs the browser where to look for the image on the server. The location of image may be on the same directory or another server.
2) alt: The alt attribute defines an alternate text for the image, if it can't be displayed. The value of the alt attribute describes the image in words. The alt attribute is considered good for SEO prospective.
3) width: It is an optional attribute which is used to specify the width to display the image. It is not recommended now. You should apply CSS in place of width attribute.
4) height: It h3 the height of the image. The HTML height attribute also supports iframe, image and object elements. It is not recommended now. You should apply CSS in place of height attribute.

Use of height and width attribute with img tag

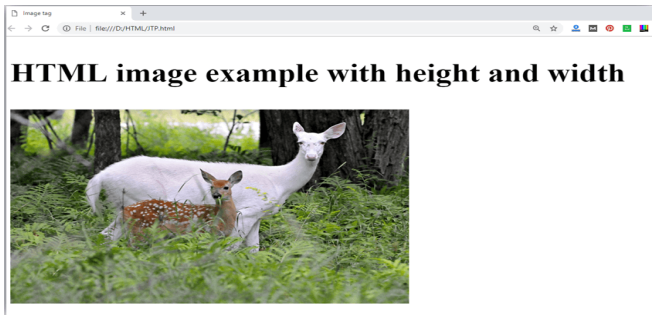
You have learnt about how to insert an image in your web page, now if we want to give some height and width to display image according to our requirement, then we can set it with height and width attributes of image.

```
<img src="animal.jpg" height="180" width="300" alt="animal image">
```

Example:

```
<!DOCTYPE html>
<html>
<head>
        <title>Image tag</title>
</head>
<body>
<h2>HTML image example with height and width</h2>
<img src="C:\Users\RGUKT\Pictures/animal.jpg" height="180"
width="300" alt="animal image">
</body>
</html>
```

Output:



Use of alt attribute

We can use alt attribute with tag. It will display an alternative text in case if image cannot be displayed on browser. Following is the example for alt attribute:

```
<img src="animal.png" height="180" width="300" alt="animal image">
```

Output:

How to get image from another directory/folder?

To insert an image in your web, that image must be present in your same folder where you have put the HTML file. But if in some case image is available in some other directory then you can access the image like this:

```
<img src="E:/images/animal.png" height="180" width="300" alt="animal image">
```

In above statement we have put image in local disk E------>images folder------>animal.png.

Note: If src URL will be incorrect or misspell then it will not display your image on web page, so try to put correct URL.

Use <img> tag as a link

We can also link an image with other page or we can use an image as a link. To do this, put <img> tag inside the <a> tag.

Example:

```
<a href="https://www.javatpoint.com/what-is-robotics"><img src="robot.jpg" height="100" width="100"></a>
```

Output:

Use image as a link

Click on the image to know about robotics

```
<!DOCTYPE html>
<html>
  <head>
        <title>Image tag</title>
  </head>
<body>
        <h2>Use image as a link</h2>
   <p>Click on the image to know about robotics</p>
  <a href="https://www.javatpoint.com/what-is-robotics"><img
src="https://static.javatpoint.com/htmlpages/images/robot.jpg"
height="100" width="100"></a>
  </body>
</html>
```

HTML Background Images

A background image can be specified for almost any HTML element.

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 HTML element:

```
<div style="background-image: url('img_girl.jpg');">
```

```
<!DOCTYPE html>
<html>
<body>

<h2>Background Image</h2>

<p>A background image for a div elememt:</p>

<div style="background-image: url('img_girl.jpg');">
You can specify background images<br>
for any visible HTML element.<br>
In this example, the background image<br>
is specified for a div element.<br>
By default, the background-image<br>
will repeat itself in the direction(s)<br>
where it is smaller than the element<br>
where it is specified. (Try resizing the<br>
browser window to see how the<br>
background image behaves.
</div>
<p>A background image for a p elememt:</p>

<p style="background-image: url('img_girl.jpg');">
You can specify background images<br>
for any visible HTML element.<br>
In this example, the background image<br>
is specified for a p element.<br>
By default, the background-image<br>
will repeat itself in the direction(s)<br>
where it is smaller than the element<br>
where it is specified. (Try resizing the<br>
browser window to see how the<br>
background image behaves.
</p>

</body>
</html>
```

Output:

Background Image

A background image for a div elememt:

A background image for a p elememt:

You can also specify the background image in the <style> element, in the <head> section:

Example

Specify the background image in the `<style>` element:

```
<style>
div {
  background-image: url('img_girl.jpg');
}
</style>
```

```
<!DOCTYPE html>
<html>
<head>
<style>
div {
  background-image: url('img_girl.jpg');
}
</style>
</head>
<body>

<h2>Background Image</h2>

<div>
You can specify background images<br>
for any visible HTML element.<br>
In this example, the background image<br>
is specified for a div element.<br>
By default, the background-image<br>
will repeat itself in the direction(s)<br>
where it is smaller than the element<br>
where it is specified. (Try resizing the<br>
browser window to see how the<br>
background image behaves.
</div>

</body>
</html>
```

Background Image

Background Image on a Page

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

Example

Add a background image for the entire page:

```
<style>
body {
  background-image: url('img_girl.jpg');
}
</style>
```

```
<!DOCTYPE html>
<html>
<head>
<style>
body {
  background-image: url('img_girl.jpg');
}
</style>
</head>
<body>

<h2>Background Image</h2>

<p>By default, the background image will repeat itself if it is smaller than the element where it is specified, in this case the body element.</p>

</body>
</html>
```

Background Cover

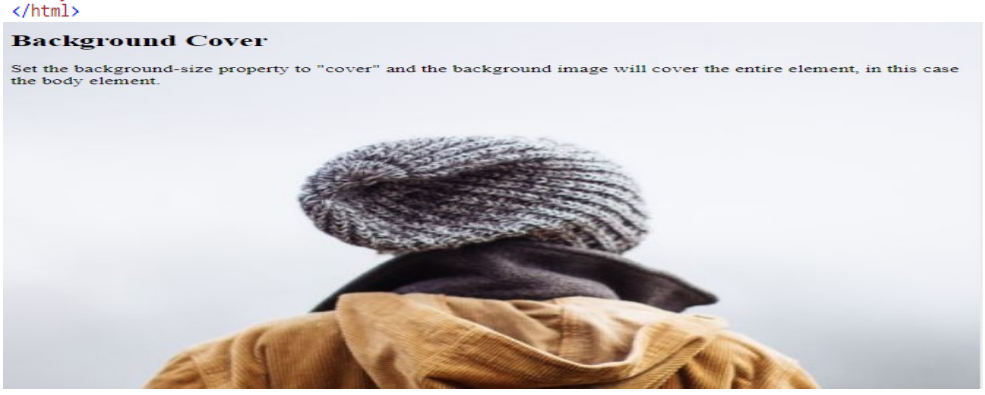
If you want the background image to cover the entire element, you can set the background-size property to cover. Also, to make sure the entire element is always covered, set the background-attachment property to fixed:

This way, the background image will cover the entire element, with no stretching (the image will keep its original proportions):

Example

```
<style>
body {
  background-image: url('img_girl.jpg');
  background-repeat: no-repeat;
  background-attachment: fixed;
  background-size: cover;
}
</style>
```

```
<!DOCTYPE html>
<html>
<head>
<style>
body {
  background-image: url('img_girl.jpg');
  background-repeat: no-repeat;
  background-attachment: fixed;
  background-size: cover;
}
</style>
</head>
<body>

<h2>Background Cover</h2>

<p>Set the background-size property to "cover" and the background image will cover the entire element, in this case the body element.</p>

</body>
</html>
```



HTML Multimedia

Multimedia on the web is sound, music, videos, movies, and animations.

What is Multimedia?

Multimedia comes in many different formats. It can be almost anything you can hear or see, like images, music, sound, videos, records, films, animations, and more.

Web pages often contain multimedia elements of different types and formats.

Browser Support

The first web browsers had support for text only, limited to a single font in a single color.

Later came browsers with support for colors, fonts, images, and multimedia!

Multimedia Formats

Multimedia elements (like audio or video) are stored in media files.

The most common way to discover the type of a file, is to look at the file extension.

Multimedia files have formats and different extensions like: .wav, .mp3, .mp4, .mpg, .wmv, and .avi.

Common Video Formats

There are many video formats out there. The MP4, WebM, and Ogg formats are supported by HTML.The MP4 format is recommended by YouTube.

Format	File	Description
MPEG	.mpg .mpeg	MPEG. Developed by the Moving Pictures Expert Group. The first popular video format on the web. Not supported anymore in HTML.
AVI	.avi	AVI (Audio Video Interleave). Developed by Microsoft. Commonly used in video cameras and TV hardware. Plays well on Windows computers, but not in web browsers.
WMV	.wmv	WMV (Windows Media Video). Developed by Microsoft. Commonly used in video cameras and TV hardware. Plays well on Windows computers, but not in web browsers.

Note: Only MP4, WebM, and Ogg video are supported by the HTML standard.

Common Audio Formats

MP3 is the best format for compressed recorded music. The term MP3 has become synonymous with digital music.

HTML Video

The HTML <video> element is used to show a video on a web page.

Example

Courtesy of Big Buck Bunny:


```
<!DOCTYPE html>
<html>
<body>

<video width="400" controls>
  <source src="mov_bbb.mp4" type="video/mp4">
  <source src="mov_bbb.ogg" type="video/ogg">
  Your browser does not support HTML video.
</video>

<p>
Video courtesy of
<a href="https://www.bigbuckbunny.org/" target="_blank">Big Buck Bunny</a>.
</p>

</body>
</html>
```

The HTML <video> Element

To show a video in HTML, use the <video> element:

Example

```
<video width="320" height="240" controls>
  <source src="movie.mp4" type="video/mp4">
  <source src="movie.ogg" type="video/ogg">
Your browser does not support the video tag.
</video>
```

```
<!DOCTYPE html>
<html>
<body>

<video width="320" height="240" controls>
  <source src="movie.mp4" type="video/mp4">
  <source src="movie.ogg" type="video/ogg">
  Your browser does not support the video tag.
</video>

</body>
</html>
```

How it Works

The controls attribute adds video controls, like play, pause, and volume.
It is a good idea to always include width and height attributes. If height and width are not set, the page might flicker while the video loads.
The <source> element allows you to specify alternative video files which the browser may choose from. The browser will use the first recognized format.
The text between the <video> and </video> tags will only be displayed in browsers that do not support the <video> element.

HTML <video>Autoplay

To start a video automatically use the autoplay attribute:

Example

```
<video width="320" height="240" autoplay>
  <source src="movie.mp4" type="video/mp4">
  <source src="movie.ogg" type="video/ogg">
Your browser does not support the video tag.
</video>
```

```
<!DOCTYPE html>
<html>
<body>

<video width="320" height="240" autoplay>
  <source src="movie.mp4" type="video/mp4">
  <source src="movie.ogg" type="video/ogg">
  Your browser does not support the video tag.
</video>

<p><b>Note:</b> The autoplay attribute does not work on some mobile devices.</p>

</body>
</html>
```

Multiple Choice Questions

1) HTML stands for
   a) Hyper Text Mark-up Language b) Hyper Text Machine Language
   c) Hyper Tech Mark-up Language d)Hyper Teach Mark-up Language
2) HTML is what type of Language?
   a) Programming Language b) Mark-up Language c) Scripting Language d) High level Language

3) Fundamental HTML block is known as……
   a) HTML Body b) HTML Tag c) HTML Element d) HTML Attribute
4) Which of the following tag give similar output of <b> tag?
   a) b) c) <fat> d) None of these
5) Which of the following is the correct hierarchy?
   a) <h6><h5><h3><h1>b) <h5><h6><h3><h1> c) <h6><h5><h1><h3> d) <h1><h3><h5><h6>
6) Which of the following is not an empty tag?
   a)
 b) <hr> c) Both a & b d)
7) Which of the following tag is suitable to create double quote?
   a) <q> b) <mark> c) <u> d)
8) Which of the following is not an attribute?
   a) src b) href c) alt d) <a>
9) Which of the following tag is used to insert image in HTML?
   a) img b) image c) picture d) src
10) Which of the following tag is not used to play the multimedia?
   a) <audio> b) <embed> c) <video> d) <style>
11) Which of the following nested element sequence is correct?
   a) <h1><u>Nested Elements</h1></u>
   b) <h1><u>Nested Elements</h1></u>
   c) <h1><u>Nested Elements</h1></u>
   d) <h1><u>Nested Elements</u></h1>
12) Which of the following tag is used to strikethrough the line?
   a) <del> b) <scratch> c) <sup> d) <sub>
13) Which of the following tag is used to create hyperlink?
   a) b) <a> c) <q> d) <hr>
14) Choose the correct HTML for height attribute and its value?
   a) height=90 b) HEIGHT=”90” c) HEIGHT=90 d) Height=”90”
15) What is the correct HTML for adding background color?
   a) <boy style=” background-color: yellow”> b)<body background = “yellow>
   c) <background>yellow</background> d)All of the above
16) Which of the following mandatory attribute for anchor tag?
   a) Src b) href c) alt d) longdesc
17) How do I add scrolling text to web page?
   a) <Scroll> b) <ciruler> c) <tab> d) <marquee>
18) How can you create an e-mail link?
   a) <a href=”mailto:rgukt@gmail.com”> b) <mail>rgukt@gmail.com</mail>
   c) <mail href=”rgukt@gmail.com> d)<a href= rgukt@gmail.com>
19) HTML is a case sensitive language
   a) True b) False
20) What is the correct HTML for adding a background color?
   a) <background>red<background> b) <body color = “red”>
   c) <body bg color = “red”> d) <body bg = “red”>

Descriptive Questions

1) Describe the HTML?
2) Write the basic structure of the HTML template?
3) What is an anchor tag and how can you open an url into a new tab when clicked?
4) Define attributes in HTML tag with examples?
5) Explain the following tags with examples?
   a) <bdo>b) <sup> c) <sub> d) <del> e) <link>

Unit 2 - HTML Components

Module 1 - HTML Unordered Lists

- Lists are used for displaying data in points by using numbers or bullets.
- There are three types of lists in HTML web pages.
  1. Unordered List
  2. Ordered List
  3. Definition List

Unordered Lists in HTML

- An unordered list is a collection of related items that have no special order or sequence.
- It is nothing but displaying data in points by using bullets or different symbols.
- This list is created by using HTML <ul> tag.
- ul tag consists of multiple list items using <li> tag element.
- Each item in the list is marked with a bullet or different symbols.

Example for Unordered list

```
<html><head>
<title>HTML Unordered List</title>
</head>
<body>
<ul>
<li>Beetroot</li>
<li>Ginger</li>
<li>Potato</li>
<li>Radish</li>
</ul>
</body></html>
```

Output:

- Beetroot
- Ginger
- Potato
- Radish

"Type" Attribute in Unordered List

- You can use type attribute for <ul> tag to specify the type of bullet you like.
- By default, it is a disc.
- Following are the possible options:

```
<ul type="square">
<ul type="disc">
<ul type="circle">
```

Example for Unordered list using "type" attribute

```
<html>
<head>
<title>HTML Unordered List</title>
</head>
<body>
<ul type="square">
<li>Beetroot</li>
<li>Ginger</li>
<li>Potato</li>
<li>Radish</li>
</ul>
</body></html>
```

Output

- Beetroot
- Ginger
- Potato
- Radish

Module 2 - HTML Ordered Lists

Ordered list in HTML

- An ordered list is a collection of related items that have special order or sequence.
- It is nothing but displaying data in points by using numbers or letters.
- This list is created by using HTML <ol> tag.
- tag consists of multiple list items using <li> tag element.
- The numbering starts at '1' and is incremented by 1 for each successive ordered list element tagged with <li>.
- Each item in the list is marked with number or letter.

- If you are required to put your items in a numbered list instead of bulleted then HTML ordered list will be used.

Example for Ordered list in HTML

```
<html><head><title>HTML Ordered List</title></head>
<body>
<ol>
<li>Beetroot</li>
<li>Ginger</li>
<li>Potato</li>
<li>Radish</li>
</ol>
</body>
</html>
```

Output

1. Beetroot
2. Ginger
3. Potato
4. Radish

"Type" Attribute in Ordered List

- You can use type attribute for <ol> tag to specify the type of number or letter you like.
- By default, it is a number.
- Following are the possible options:
  - ✓ <ol type="1"> - Default-CaseNumerals.
  - ✓ <ol type="I"> - Upper-CaseNumerals.
  - ✓ <ol type="i"> - Lower-CaseNumerals.
  - ✓ <ol type="a"> - Lower-CaseLetters.
  - ✓ <ol type="A"> - Upper-CaseLetters.

Example 1 for Unordered list using "type" attribute

Now let's see how <ol type="A"> is used

```
<html><head>
<title>HTML Ordered List</title></head>
<body>
<ol type="A">
<li>Beetroot</li>
<li>Ginger</li>
<li>Potato</li>
<li>Radish</li>
</ol>
</body>
</html>
```

Output:

A. Beetroot
B. Ginger
C. Potato
D. Radish

"start" Attribute list in ordered list

- You can use start attribute for <ol> tag to specify the starting point of numbering you need. Following are the possible options:
  - <ol type="1" start="4"> - Numerals starts with 4.
  - <ol type="I" start="4"> - Numerals starts with IV.
  - <ol type="i" start="4"> - Numerals starts with iv.
  - <ol type="a" start="4"> - Letters starts with d.
  - <ol type="A" start="4"> - Letters starts with D.

Let's see how <ol type="i" start="4" > is used

Example for ordered list using "type" and "start" attribute

```
<html>
<head>
<title>HTML Ordered List</title>
</head>
<body>
<ol type="i" start="4">
<li>Beetroot</li>
<li>Ginger</li>
<li>Potato</li>
<li>Radish</li>
</ol>
```

Output:

iv. Beetroot
v. Ginger
vi. Potato
vii. Radish

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

Definition List in HTML

- HTML and XHTML support a list style which is called definition list where entries are listed like in a dictionary or encyclopedia.
- The definition list is the ideal way to present a glossary, list of terms, or other name/value list.
- Definition List makes use of following three tags.
  - <dl></dl> - Defines the start and end of thelist
  - <dt> - Define term
  - <dd> - Define Term Definition

Example for Definition list

```
<html>
<head>
<title>HTML Definition List</title>
</head>
<body>
<dl>
<dt><b>HTML</b></dt>
<dd>This stands for Hyper Text Markup Language</dd>
<dt><b>HTTP</b></dt>
<dd>This stands for Hyper Text Transfer Protocol</dd>
</dl>
</body>
</html>
```

Multiple Choice Questions

1) Which one of the following is a type of lists that supports in HTML?
a) Ordered List b) Unordered List c)Definition List d) All the above
2) By Which tag an Ordered list is represented?
   a) b) c) d) <dl>
3) What is the default start of item marker in ordered list?
   a) 1 b) I c) id) None of the above
4) Which one from the following is a block level element in HTML
   a) b) c) <class> d) Both A and B
5) If you want to merge two or more rows in table which attribute is used?
   a) rowmerge b) rowsopan c) colmerge d) colspan
6) Which html attribute changes the numbering style of a list
   a) sizeb) value c) typed)number
7) HTML <dl> tag defines the
   a) ordered listb) unordered listc) description listd) descriptive list
8) .... tag is used for
   a) list itemb) item listc) listd) None of the above
9) <dd>....</dd> tag defines the list?
   a) Ordered list b) Unordered listc) Definition listd)Define definition
10) Which tags is used for creating definition list?
a) <dl><dt></dt><dd></dd></dl>b) c) d) None of the above

Descriptive Questions

1) Write short note on Ordered list?
2) Write short note on Unordered list?
3) Write short note on Definition list?
4) Write short note on Nested list?
5) Explain different types of lists in HTML with examples?

Unsolved Problems

1) Write HTML code for creating ordered list by using your university branches?
2) Write HTML code for creating Nested list by using your university?
3) Write HTML code for creating unordered list.
4) Write HTML code for creating definition list.

Module 3 - HTML Tables

The <table> tag defines an HTML table. Each table row is defined with a <tr> tag. Each table header is defined with a <th> tag. Each table data/cell is defined with a <td> tag. By default, the text in <th> elements are bold and centered. By default, the text in <td> elements are regular and left-aligned.

Border color for table in HTML

- The color of the lines inside and outside the table can also be changed using the "bordercolor" attribute.
- It accepts the value as name of the color. If you omit this attribute, the color of the table border is set to its default grey.

Example for "bordercolor" attribute in HTML Table

```
<html><head><title>bordercolor program </title></head>
<body>
<table  border="10"  bordercolor="red">
<tr><td>border line thickness is set to 10</td>
<td>border colour is red</td></tr>
<tr>
<td>red as tomato</td>
<td>red as apple</td>
</tr>
</table></body></html>
```

"bgcolor" attribute in HTML Table

- We can change the background color of a table using the attribute "bgcolor".
- This attribute takes the name of the color or hexadecimal number as value.

Syntax:<table border ="Number" bgcolor= "colorname">

Example program for bgcolor attribute in HTML Table

```
<html><head><title>bgcolor attribute program </title></head>
<body>
<table border="4" bgcolor="yellow" bordercolor="red">
<tr>
<td>border line thickness is set to 4</td>
<td>border color is red</td>
</tr>
<tr>
<td>background color yellow</td>
<td>red as apple</td>
</tr>
</table>
</body></html>
```

Output

border line thickness is set to 4	border colour is red
background color yellow	red as apple

"background" attribute in HTML Table

- If we want to place an image or a picture to the background of the table, then we can use the background attribute.
- This attribute takes the value as the address or the path of the picture.
- The picture may be a bitmap or a graphic image.
- In the following code, the image named "yelloww.jpg" is set as background to the entire table.

Syntax

<table border="number" background="location of the image">

```
<table border ="1" background="c:\yelloww.jpg">
```

The CAPTION tag

- Caption is used for giving additional information to the table.
- The <caption> tag is used to provide a text to the table to explain the contents of the table.
- It is generally in bold, at center with respect to the table. However, the position of the caption can be on either the top or the bottom of the table using the 'align' attribute.

Example for caption to HTML Table

```
<html><head><title>Caption to the Table </title></head>
<body>
<table border="4" bgcolor="yellow" bordercolor="red">
<caption align="bottom">Table with caption</caption>
<tr>
<td>border line thickness is set to 4</td>
<td>border colour is red</td>
</tr>
<tr>
<td>background color yellow</td>
<td>red as apple</td>
</tr>
</table>
</body></html>
```

Output

Table with caption

border line thickness is set to 4	border colour is red
background color yellow	red as apple

Multiple Choice Questions

1) Which HTML tag is used to define a table?
   a) <tb> b) <tl> c) <table> d) <tab>
2) Each cell of the table can be represented by using?
   a) <tr> b) <td> c) <th> d) <thead>
3) For table heading we can use _____
   a) <td> b) <tr> c) <thead> d) <th>
4) Which tag we can use for adding caption to the table?
   a) <caption> b) <thead> c) <th> d) <tr>
5) In table we can't be applied Borders on which tag?
   a) <th> b) <td> c) <tr> d) <thead>
6) Which of the following is not the value for align attribute?
   a) Justify b) Char c) Middle d) Left
7) The gaps between the two cells of same tables are known as…..?
   a) Cell Spacing b) Cell padding c) Cell Difference d) None of these
8) In tables we can combine two or more cells in a table by using …?
   a) Colspan b) Combinecell c) Rowspan d) cell padding
9) If we want to merge two or more rows in a table which attribute we can use?
   a) Rowspan b) Colmerge c) Rowmerge d) Colspan
10) Choose the correct left align the text inside a table cell is?
   a) <td align = "left"> b) <td leftalign> c) <trleftalign> d) <tralignleft>

Descriptive Questions

1) Write about HTML tables
2) Explain different types of tables elements in HTML
3) Explain about caption element in HTML
4) Explain about title element in HTML

Unsolved Problems

1) Write HTML code to create 4X4 (4 rows and 4 columns) table
2) Write HTML code to create 4X4 table with 1st row 2 columns remaining 3 columns
3) Write HTML code to create 4X4 table with last row 3 columns

Unit 3 -HTML Forms
Module 1,2 – Form Elements

- A web form or HTML form on a web page allows a user to enter data that is sent to a server for processing.
- HTML Forms are required when you want to collect some data from the site visitor.
- For example, during user registration you would like to collect information such as name, email address, credit card, etc.
- Forms can resemble paper or database forms because web users fill out the forms using checkboxes, radio buttons, or text fields.
- There are various form elements available like text fields, text area fields, dropdown menus, radio buttons, checkboxes, etc.
- The HTML <form> tag is used to create an HTML form and it has following

  syntax:

  <form action="Script URL" method="GET|POST"> form elements like input, textarea etc. </form>

Form Attributes

- Apart from common attributes, following is a list of the most frequently used form attributes

Attribute	Description
Action	Backend script ready to process your passed data.
Method	Method to be used to upload data. The most frequently used are GET and POST methods.
Target	Specify the target window or frame where the result of the script will be displayed. It takes values like _blank, _self, _parent etc.
Enctype	You can use the enctype attribute to specify how the browser encodes the data before it sends it to the server. Possible values are: ✓ application/x-www-form-urlencoded - This is the standard method most forms use in simple scenarios. ✓ mutlipart/form-data - This is used when you want to upload binary data in the form of files like image, word file etc.

HTML Form Controls

- Form controls are objects that display data or make it easier for users to enter or edit data, perform an action, or make a selection.
- In general, controls make the form easier to use.
- There are different types of form controls that can be used to collect data using HTML form.
  1. Text Input Controls
  2. Checkboxes Controls
  3. Radio Box Controls
  4. Select Box Controls
  5. File Select boxes
  6. Hidden Controls
  7. Clickable Buttons
  8. Submit and Reset Buttons
- All these are defined as attributes of the <input> tag. The <input> tag can have several other attributes depending on the value of the "type" attribute.
- The type attribute specifies the type of input we want from the user.

Text Input Controls

There are 3 types of text input used in forms

- **Single-line text input controls** - This control is used for items that require only one line of user input, such as search boxes or names. They are created using HTML <input> tag.
- **Password inputcontrols** - This is also a single-line text input but it masks the character as soon as a user enters it. They are also created using HTML <input> tag.
- **Multi-line text input controls** - This is used when the user is required to give details that may be longer than a single sentence. Multi-line input controls are created using HTML <textarea> tag.

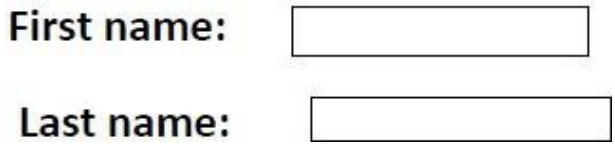
Single-line text input controls

- This control is used for items that require only one line of user input, such as search boxes or names. They are created using HTML <input> tag.

Example of a single-line text input used to take first name and last name

```
<html><head><title>Text Input Control</title></head>
<body>
<form >
                First name: <input type="text" name="first_name" /><br/>
                Last name: <input type="text" name="last_name" />
</form>
</body></html>
```

Output:



Attributes to be used in "input" tag

Following is the list of attributes for <input> tag for creating text field.

Attribute	Description
Type	Indicates the type of input control and for text input control it will be set to text
Name	Used to give a name to the control which is sent to the server to be recognized and get the value.
Value	This can be used to provide an initial value inside the control.
Size	Allows specifying the width of the text-input control in terms of characters.
maxlength	Allows specifying the maximum number of characters a user can enter into the text box.

Password input controls

- This is also a single-line text input control but it masks the character as soon as a user enters it.
- It is also created using HTML <input> tag but "type" attribute used to set password.

Syntax

Password: <input type="password" name="password" />

Following is the list of attributes for <input> tag for creating text field.

Attributes	Description
Type	Indicates the type of input control and for password input control it will be set to password.
Name	Used to give a name to the control which is sent to the server to be recognized and get the value.
Value	This can be used to provide an initial value inside the control.
Size	Allows to specify the width of the text-input control in terms of characters.
maxlength	Allows to specify the maximum number of characters a user can enter into the text box.

Following is the list of attributes for <textarea> tag for creating Multiple-Line Text field control

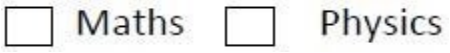
Attribute	Description
Name	Used to give a name to the control which is sent to the server to be recognized and get the value.
Rows	Indicates the number of rows of text area box.
Cols	Indicates the number of columns of text area box

Checkbox Control

- Checkbox controls are used when more than one option is required to select.
- It is also created using HTML <input> tag but "type" attribute is used in checkbox.

Example of a multiple-line text input control

```
<html><head><title>Text Input Control</title></head>
<body>
<form >
<input type="checkbox" name="maths" value="on">Maths
<input type="checkbox" name="physics" value="on"> Physics
</form></body></html>
```



- From the above example value attribute is can be seen in server side not in client side.

Following is the list of attributes for <input> tag for creating Checkbox control

Attribute	Description
Type	Indicates the type of input control and for checkbox input control it will be set to checkbox.
Name	Used to give a name to the control which is sent to the server to be recognized and get the value.
Value	The value that will be used if the checkbox is selected.
checked	Set to checked if you want to select it by default.

Select Box Control

- The <select> tag is used to construct drop-down list boxes and scrolling list boxes.
- Drop-down menus can serve the same purpose as radio buttons or check boxes depending on the options specified.
- The advantage of a drop- down menu, compared to radio buttons or check boxes, is that it takes up 1ess space. But that is a1so a disadvantage, because people can't see all options in the menu right away.
- With the select attribute, the menu can be customized to show more than just one option at a time. Such lists are typically encountered when the user is to select a city or a country.
- The <option>...</option> element is used to specify each item displayed in the list.
- The value of the option last selected is returned when the form data is transmitted.
- Adding the select attribute in the <option> tag indicates which element is displayed initially when the page loads.
- If this is not provided, the first item is se1ected by default.

Example for select box control

```
<html><head><title>select box Control</title>  </head>
<body>
<form >
<select name="dropdown">
<option value="Maths" selected>Maths</option>
<option value="Physics">Physics</option>
</select>
</form></body></html>
```

Output

Following is the list of attributes used for creating Select Box control

Attribute	Description
Name	Used to give a name to the control which is sent to the server to be recognized and get the value.
Size	This can be used to present a scrolling list box.
Multiple	If set to "multiple" then allows a user to select multiple items from the menu.
Value	The value that will be used if an option in the select box is selected.
Selected	Specifies that this option should be the initially selected value when the page loads.
Label	An alternative way of labeling options

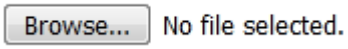
File Upload Box

- The File field allows users to upload files.
- The user can submit any information like picture, a spreadsheet, a scanned document or a word- processor document after verifying that the server on which the form will be processed supports fi1e upload.
- The browser displays a "BROWSE" button next to a text box that lets the user select a field from their computer's storage to use as the value of the file input element.
- Typically, you would have encountered this option when attaching files to your email messages.
- This is also created using the <input> element but type attribute is set to file.

Example for File upload control

```
<html><head><title>file upload box Control</title></head>
<body>
<form >
<input type="file" name="fileupload"  accept="image/*" />
</form></body></html>
```

Output



Following is the list of attributes used for creating File upload control

Attribute	Description
Name	Used to give a name to the control which is sent to the server to be recognized and get the value.
Accept	Specifies the types of files that the server accepts.

Button Controls

- There are various ways in HTML to create clickable buttons.
- You can also create a clickable button using <input> tag by setting its "type" attribute to button.
- The type=" submit" creates a submit button on the form and the type="reset" creates a reset button.
- The Submit button tells the web browser to gather up all the selections, values, and entered text in the form elements and dispatch it off to the destination defined in the action part of the <FORM> tag.
- The Reset button restores the form to its default state.
- The submit or the reset button has a typical shape and size, which may not be in harmony with the rest of the look and feel of the web page.
- Hence you may prefer to use an image instead using the type="image" attribute.
- When a user clicks an image button the form is sent to the address specified in the action attribute of the <form> tag.

Following is the list of attributes used for creating button control

Type	Description
Submit	This creates a button that automatically submits a form.
Reset	This creates a button that automatically resets form controls to their initial values.
Button	This creates a button that is used to trigger a client-side script when the user clicks that button.
Image	This creates a clickable button but we can use an image as background of the button.

Example for button controls

```
<html><head><title>Different button Controls</title></head>
<body><form >
<input type="submit" name="submit" value="Submit" />
<input type="reset" name="reset" value="Reset" />
<input type="button" name="ok" value="OK" />
<input type="image" name="imagebutton" src="/html/images/logo.png" />
</form></body></html>
```

Output

Module 3 - Frames

- A frame divides the screen into separate windows with both vertical and horizontal scroll bars.
- The windows so formed may appear to be sub-windows of the main window (the webpage).
- This allows the user to access different pages of a website from one screen if designed to do so.
- Frames are needed when you want to have menus on one side and the contents or the operations on the other side.
- When the user clicks on one of the menu items, the contents become visible on the other side.
- A frame divides the webpage into different windows. It makes some structural changes to the main window.
- Hence, it is not written inside the body element, but it forms its own element, outside the head section of HTML document called the FRAMESET.
- A FRAMESET element is the parent element that defines the characteristics of individual frame pages.

The basic syntax of FRAMESET element

```
<html><head> … </head>
    <FRAMESET cols= "number%,number%"
        <frame src = "address of HTML document">
        <frame src = "address of HTML document">
    </FRAMESET></html>
```

The attributes used with the FRAMESET element are given in the table below:

FRAMESET element works in conjunction with the FRAME tag whose attributes are discussed below

Attributes	Value accepted	Description
Rows	Number in percentage or star (*) indicating the rest of the window	Divides the main window horizontally in proportion to main window
Cols	Number in percentage or star (*) indicating the rest of the window	Divides the main window vertically in proportion to main window
Border	Number	Increases the width of the outer border.
Frame border	Number	Used with netscape navigator to surround the sub-window with a border. If value is set to zero, no outer lines appear around the frame.
Frame spacing	Number	If set to zero, it removes the ugly grey lines appearing between the two frames, otherwise it increases the width of the grey line.

S No	Attributes	Value	Description
1	src (mandatory attribute)	Address of the HTML document	To load the HTML webpage in the frame defined by FRAMESET element
2	scrolling	Yes/Auto/No	**Yes** will insert both the scrolls irrespective of the size of the sub-window. **Auto** will insert the scrolls only when the contents of the sub-window are not visible **No** will not insert any scrolls even when all the contents are not visible.
3	Noresize	Noresize	when used this attribute does not allow the user to adjust the size of the frame in the webpage.
4.	Name	A noun	Assign a name to the frame
	Target	The noun of name attribute	This attribute specifies the name of a frame where a document is to be opened.

- If we want to make a webpage that uses two frames divided in columns, we should follow these steps

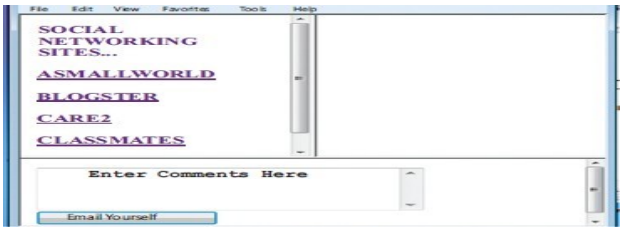
HTML Page with FRAMESET element

```
<html><head><title>frameset program <title></head>
    <FRAMESET cols="120,*">
        <frame src="menu.html" name="menu">
        <frame src="main.html" name="main">
    </FRAMESET></html>
```

- In the above code, the left side of the column is 120 pixels and the Star indicates the rest of the screen. A star has been used because the screen size varies; it may be 640 pixels across, or 800 or 1024 pixels across.
- The 'src' attribute opens the menu.html document in left side frame and main.html gets opened in right side frame. Both the frames are given a name through the 'name' attribute.
- Create a number of HTML documents (containing the body tag) that you would like to load into the frames. (At least as many as the frames that are there in the FRAMESET element from the above example you need to have two documents; menu.html andmain.html)
- When a FRAMESET page is loaded, the browser automatically loads the HTML documents designed.

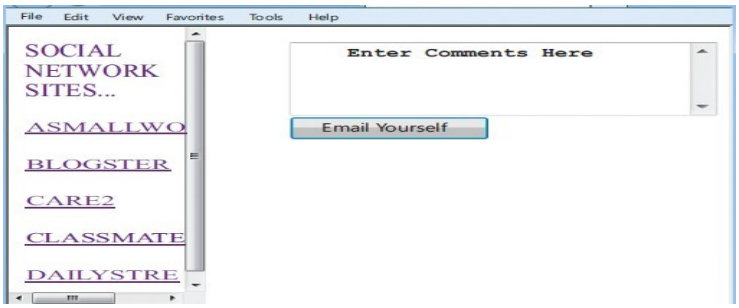
- If you want to divide the webpage in equal sizes horizontally, and want to display a.html in first and b.html in second, you will use the following code.

```
<html><head>…</head>
        <FRAMESET rows= “50%,50%”>
            <frame src = “a.html”>
            <frame src = “b.html”>
        </FRAMESET></html>
```



- The first FRAMESET divides the screen into two rows. Now since the control has come to the first row of the new screen, the second FRAMESET can divide it into two columns.
- Open the two HTML documents, a.html and b.html, give them a name and then move the control to the second row of the new screen. Since now the control is on the second screen, SRC can now open secondrow.html as shown in below figure.
- If we want to insert scrolls in the frames, when the contents of the frame are not visible, use the value “auto” as shown below:

```
<html><head></head>
    <FRAMESET cols=”120,*”
    frameborder=0 framespacing=50>
        <frame src=”a.html”
        scrolling=auto>
        <frame src=”formbutton.html”
        scrolling=no>
    </FRAMESET></html>
```



Multiple Choice Questions

1) Which HTML element is used to define a multi-line input field?
   a) <multi-line> b) <textarea> c) <textfields> d) <textline>
2) In forms the _lank target attribute opens a linked document in …?
   a) same window b) New tab c) New window d) Both A & B
3) Which character tells the end of a tag in HTML?
   a) / b) \ c) >d) <
4) In HTML form the <input type=”text”> is used for …?
   a) One line text b)Block of text c)one paragraph d) All of the these
5) Which tag is used for a submit button?
   a) <button> b) <submit> c) <submit button> d) <input type=”submit”>
6) Which method is used when form data contains sensitive or personal information?
   a) Get method b) Post method c)Both a &b d) Host method
7) The action attribute in HTML forms specifies ….?
   a) Which HTTP method is used b)Which action is going on c) Where to submit the form
   d) The auto completion of form
8) Which of the following is not the form type for adding text?
   a) Text input b) Text area c) Password input d) Submit button
9) Which of the following creates push button?
   a) Reset b)Check Box c)Input d) All of these
10) Define frameset to create 3 Horizontal frames?
   a) <frameset rows=”20%,60%,20%”>-----</frameset> b)<frameset cols=”20%,60%,20%”>-----</frameset> c) <frameset hrlines=”20%,60%,20%”>-----</frameset> d) None of these

Descriptive Questions

1) Explain about Forms in HTML
2) Explain about Frames in HTML
3) Explain about different fields in HTML Forms
4) Explain about forms and frames with example.

Unsolved Problems

1) Write HTML code to create basic form.
2) Write HTML code to create form.
3) Write HTML code to create your bio-data form.

Unit 4 - Introduction to CSS

Module 1 – Introduction to CSS and Font properties

CSS stands for cascading style sheets. CSS describe how HTML elements are to be displayed CSS saves a lot of work. It can control the layout of multiple web pages all at once.

Why use CSS:

CSS is used to define styles for your web pages, including the design, layout and variations in display for different devices and screen sizes.

CSS Syntax:

A CSS rule – set of a selector and a declaration block:

1) Selector points to the HTML element you want to style.
2) The declaration block contains one or more declarations separated by semicolons.
3) Each declaration included a CSS property name and a value. Separated by a colon.
4) A CSS declaration always ends with a semicolon and declaration blocks are surrounded by curly braces.

The Limitations of CSS:

CSS is very limited in browser compatibility. When you design a web page and you want it to display exactly as you like it. The problem with CSS is that it displays web pages very differently in the different browsers. Your webpage looks perfect in Mozilla may look different in Internet Explorer. This is a big problem for your site's success.

Methods of applying CSS to an HTML document:

There are three ways you can apply CSS to an HTML document. The First method is "In-Line, Second method is "Internal" and the Third method i.e. external which is most important.

Method 1: In-line (the attribute style):

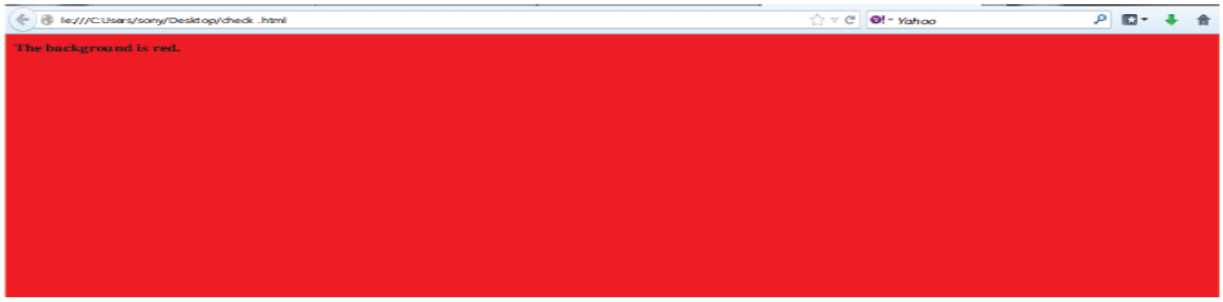
Method 1: In-line (the attribute style)

One way to apply CSS to HTML is by using the HTML attribute style.

Example 1: To apply the red background color in a webpage, it can be applied in the following manner.

```
<html>
    <head>
            <title>Example</title>
    </head>
    <body style="background-color: #FF0000;">
            <p>The background is red.</p>
    </body>
    </html>
```

Output:



Method 2: In-line (the tag style):

Method 2: Internal (the tag style)

Another way is to include the CSS codes using the HTML tag <style>. For example like this:

```
<html>
<head>
        <title>Example</title>
        <style type="text/css">
        body {background-color: #0000FF ;}
        </style>
</head>
<body>
        <p> The background is Blue.</p>
</body>
</html>
```

Output:

Method 3: External (link to a style sheet):

The method to link html with style sheet is called external style sheet. An external style sheet is a text file with the extension **.css**. Like other files, we can place the style sheet on your web server or hard disk. For example,

save the style sheet with the name **style.css** and place it in a folder named **style**.

To create a link from the HTML document (default.htm) to the style sheet (style.css). The following code will be inserted in the header section of the HTML code i.e. between the <head> and </head> tags. HTML file. *<link rel="stylesheet" type="text/css"* ***href="style/style.css"*** */>*

The code will be as follows:

Default.htm

```
<html>
    <head>
            <title>My document</title>
            <link rel="stylesheet" type="text/css" href="style.css" />
    </head>
    <body>
            <h1>My stylesheet Page</h1>
    </body>
    </html>
```

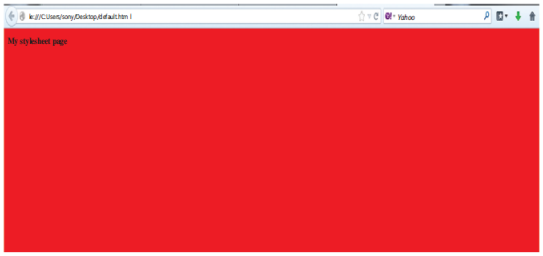
In the other notepad rename as style.css we have to write the following code

style.css

```
body {
   background-color: #FF0000;
}
```

This link will display the layout from the CSS file in the browser when displaying the HTML file.

Output:



One CSS file can be used to control the layout of many HTML documents. Using CSS, the change can be made in a few seconds just by changing one code in the central style sheet.

Font Properties:

Following are the font properties.

1. Font – family
2. Font – style
3. Font – size
4. Font – variant
5. Font – weight

1. Font family:

The property font-family is used to apply prioritized list of fonts in a web page. If the first font of the list is not installed on the computer then the next font of the list will be displayed until a suitable font is found. Font's family is divided into two categories

a. Generic Family: a group of font families with a similar look (like "Serif" or "Monospace")

b. Font Family: a specific font family (like "Times New Roman" or "Arial")

Example: sans-serif, which is a collection of fonts without "feet".

Times New Roman Garamond **Georgia**	These three font-families belong to the genetic family **serif**. They are characterized by all having "feet".
Trebuchet **Arial** **Verdana**	These three font-families belong to the genetic family **sans-serif**. They are all characterized by all having "feet".
Courier Courier New **Andele Mono**	These three font-families belong to the genetic family **monospace**. They are all characterized by all characters having a fixed width.

An example of a inserting list of fonts in a web page:

```
h1 {font-family: arial, comic sans-serif, "Times New Roman";}
h2 {font-family: "Times New Roman", verdana, serif;}
```

Code inserted in font.html:

```
<html >
<head>
<title>Example </title>

<link rel="stylesheet" href="ex1.css" type="text/css"/>
</head>
<body>
    <h1>Heading 1 in Arial</h1>

    <h2>Heading 2 in Times New Roman</h2>

    </body>
</html>
```

Code inserted in ex1.css:

```
h1 {font-family: arial, comic sans-serif, "Times New Roman";}
h2 {font-family: "Times New Roman", verdana, serif;}
```

Output produced by the above code:

Font style:

The property font-style defines the chosen font either in **normal**, **italic** or **oblique**. In the example below, all headlines marked with <h2> will be shown in italics.

h1 {font-family: arial, verdana, sans-serif; **font-style: oblique;**}
h2 {font-family: "Times New Roman", serif; **font-style: italic;**}

Code inserted in font.html:

```
<html >
<head>
<title>Example </title>

<link rel="stylesheet" href="ex1.css" type="text/css"/>
</head>
<body>
    <h1>Heading 1 in Arial</h1>

    <h2>Heading 2 in Times New Roman</h2>

    </body>
</html>
```

Code inserted in ex1.css:

```
h1 {font-family: arial, comic sans-serif, "Times New Roman"; font-style: oblique;}
h2 {font-family: "Times New Roman", verdana, serif; font-style: italic;}
```

Output produced by the above code:

Font variant:

This property is used to select **normal** or **small-caps** variants of a font. A **small-caps** font displays the smaller sized capitalized letters (upper case) instead of lower case letters.

Sans Book SC	Sans Bold SC	Serif Book SC	Serif Bold SC
ABCABC	ABCABC	ABCABC	ABCABC

If font-variant is set to **small-caps** and no small-caps font is available the browser will most likely show the text in uppercase instead.

h1 {font-variant: small-caps;}
h2 {font-variant: normal;}

Code inserted in font.html:

```
<html >
<head>
<title>Example </title>

<link rel="stylesheet" href="ex1.css" type="text/css"/>
</head>
<body>
    <h1>Heading 1 in Arial</h1>

    <h2>Heading 2 in Times New Roman</h2>
    </body>
</html>
```

Code inserted in ex1.css:

h1 {font-family: arial, comic sans-serif, "Times New Roman"; **font-variant: small-caps;**}
h2 {font-family: "Times New Roman", verdana, serif; **font-style: italic; font-variant: normal;**}

Output produced by the above code:

Font weight:

This property describes how bold or "heavy" a font should be presented. A font can either be **normal** or **bold**. Some browsers supports the use of numbers between 100 and 900(in hundreds) to describe the weight of a font.

p {font-family: arial, verdana, sans-serif; font-weight: normal;}
td {font-family: arial, verdana, sans-serif; **font-weight: bold;**}

Code inserted in font.html:

```
<html >
<head>
<title>Example</title>

<link rel="stylesheet" href="ex1.css" type="text/css" media="all" />
</head>
<body>
    <h1>Heading 1 30px</h1>
    <h2>Heading 2 12pt</h2>
    <h3>Heading 3 120%</h3>
    <p> Heading 4 paragraph</p>

    </body>
</html>
```

Code inserted in ex1.css:

```
h1 {font-size: 30px;}
h2 {font-size: 12pt;}
h3 {font-size: 120%;}
p {font-size: 1em;}
```

Output produced by the above code:

The units '**px**' and '**pt**' make the font size absolute, while '**%**' and '**em**' allow the user to adjust the font size as he/she see fit. Some users suffer from poor vision or a monitor of bad quality. To make your website readable for everybody, you should use adjustable units such as '**%**' or '**em**'

Combining [font] styles. All the different font properties can be combined in one single property. For example, to apply different font-properties for <p> tag following code can be used.

p { **font-style: italic;**
font-weight: bold;
font-size: 30px;
font-family: arial, sans-serif; }

The order of values for font properties is:

Example:

Display the content using all the font properties of font family.

Code to be inserted in Font.html

```
<html >
<head>
<title>Example </title>
<link rel="stylesheet" href="ex1.css" type="text/css" media="all" />
</head>
<body>
<p> This an example of combining all the Font properties.</p>
</body>
</html>
```

Code to be inserted in ex1.css

```
p {         font-style: italic;
            font-weight: bold;
            font-size: 30px;
            font-family: arial, sans-serif;    }
```

Output produced by the following above code:

Module 2 - Text Properties

The following are text properties applied to text on the web page

1) Text-indent 2) Text-align 3) Text-decoration 4) Letter-spacing 5) Text-transform

Text – indent:

The text-indent property allows you to add effects to text paragraphs by applying an indent to the first line of the paragraph.

Example:

To apply **40px** indentation to all text paragraphs marked with <p>, the following code will be used:

Code to be inserted in font.html:

```
<html >
<head>
<title>Example</title>
<link rel="stylesheet" href="ex1.css" type="text/css" media="all" />
</head>
<body>
<p> This an example of Text Indentation.</p>
</body>
</html>
```

Code to be inserted in ex1.css

```
    p {
            text-indent: 60px;
      }
```

Text alignment:

The text-align property gives the same effect as attribute align gives in old versions of HTML. The text can either be aligned to the **left**, to the **right** or **center** of the screen. CSS allows you to apply justified alignment on text which is not available in HTML. The value **justify** will stretch each line so that both the right and left margins are straight.

Example:

Display the text in table headings <th> aligned to the right while the table data <td> in the centre of the browser window and normal text in paragraphs to be justified.

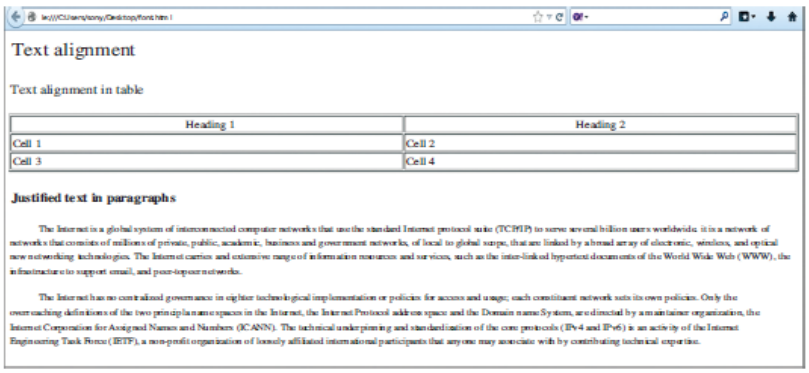
Code to be inserted in font.html

```
<html >
<head>
<title>Example </title>
<link rel="stylesheet" href="ex1.css" type="text/css" media="all" />
</head>
<body>
    <h1>Text alignment</h1>

    <h2>Text alignmen in table</h2>

    <table border="1" width="100%">
    <tr>
    <th>Heading 1</th>
    <th>Heading 2</th>
    </tr>
    <tr>
    <td>Cell 1</td>
    <td>Cell 2</td>
    </tr>
    <tr>
    <td>Cell 3</td>
    <td>Cell 4</td>
    </tr>
    </table>

<h2>Justified text in paragraphs</h2>
<p>The Internet is a global system of interconnected computer networks that use the
standard Internet protocol suite (TCP/IP) to serve several billion users worldwide. It is

    a network of networks that con
    and government networks, of lo
    electronic, wireless, and optical n
    range of information resources ar
    of the World Wide Web (WWW
    networks.

    </p>

        <p>The Internet has no centra
    or policies for access and usage;
    overreaching definitions of the t
    Protocol address space and the
    organization, the Internet Corpo
    technical underpinning and stan
    activity of the Internet Engineerir
    affiliated international participant
    expertise.</p>

    </body>

    </html>
```

Code to be inserted in ex1.css

```
th {
        text-align: right;
    }

td {
        text-align: center;
    }

p  {
        text-align: justify;
    }
```

Output produced by the following above code:



Text alignment

Text alignment in table

Heading 1	Heading 2
Cell 1	Cell 2
Cell 3	Cell 4

Justified text in paragraphs

The Internet is a global system of interconnected computer networks that use the standard Internet protocol suite (TCP/IP) to serve several billion users worldwide. it is a network of networks that consists of millions of private, public, academic, business and government networks, of local to global scope, that are linked by a broad array of electronic, wireless, and optical new networking technologies. The Internet carries and extensive range of information resources and services, such as the inter-linked hypertext documents of the World Wide Web (WWW), the infrastructure to support email, and peer-to-peer networks.

The Internet has no centralized governance in either technological implementation or policies for access and usage; each constituent network sets its own policies. Only the overreaching definitions of the two principle name spaces in the Internet, the Internet Protocol address space and the Domain name System, are directed by a maintainer organization, the Internet Corporation for Assigned Names and Numbers (ICANN). The technical underpinning and standardization of the core protocols (IPv4 and IPv6) is an activity of the Internet Engineering Task Force (IETF), a non-profit organization of loosely affiliated international participants that anyone may associate with by contributing technical expertise.

Text decoration:

The text-decoration property makes it is possible to add different “decorations” or “effects”to text. For example, you can underline the text, have a line through or above the text,etc. In the following example, <h1> are underlined headlines, <h2> are headlines with a line above the text and <h3> are headlines with a line though the text.

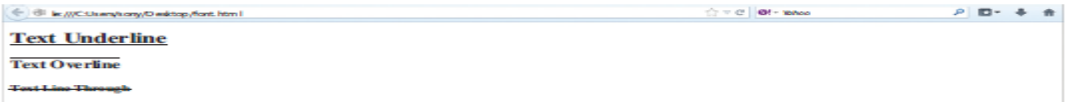
Code to be inserted in font.html

```
<html >
<head>
<title>Example </title>
<link rel=”stylesheet” href=”ex1.css” type=”text/css” media=”all” />
</head>
<body>
    <h1>Text Underline</h1>

    <h2>Text Overline</h2>
    <h3> Text Line Through</h3>

</body>
</html>
```

Code to be inserted in ex1.css

```
h1 {
        text-decoration: underline;
    }

h2 {
        text-decoration: overline;
    }

h3 {
        text-decoration: line-through;
    }
```

Output produced by the above following code:



Letter space:

This property is used to give the specified spacing between the text characters. The value of the property is simply the desired width.

Example:

To give **3px** spacing between the letters in a text paragraph <p> and **6px** between letters in headlines <h1> the following code will be used:

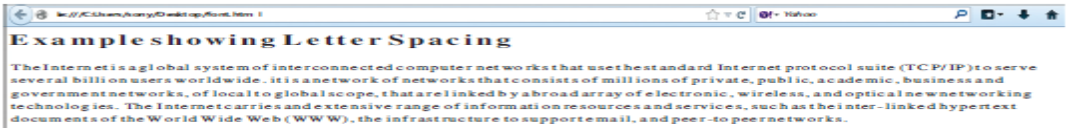
Code to be inserted in font.html

```
<html >
<head>
<title>Example </title>
<link rel=”stylesheet” href=”ex1.css” type=”text/css” media=”all” />
</head>
<body>
<h1>Example showing Letter Spacing</h1>

<p>The Internet is a global system of interconnected computer networks that use the standard Internet protocol suite (TCP/IP) to serve several billion users worldwide. It is a network of networks that consists of millions of private, public, academic, business, and government networks, of local to global scope, that are linked by a broad array of electronic, wireless, and optical networking technologies. The Internet carries an extensive range of information resources and services, such as the inter-linked hypertext documents of the World Wide Web (WWW), the infrastructure to support email, and peer-to-peer networks.
</p>

</body>
</html>
```

Code to be inserted in ex1.css

```
h1 {
        letter-spacing: 6px;
    }
 p  {
        letter-spacing: 3px;
    }
```

Output produced by the following above code:



Text transformation:

The text-transform property controls the capitalization of a text. You can choose **capitalize**, use **uppercase** or **lowercase** regardless of how the original text is looks in the HTML code. An example could be the word“headline” which can be presented to the user as “HEADLINE” or “Headline”. There are four possible values fortext-transform:

Capitalize: Capitalizes the first letter of each word. For example: “information technology” will be “Information Technology”.

Uppercase: Converts all letters to uppercase. For example: “information technology” will be “INFORMATION TECHNOLOGY”.

Lowercase: Converts all letters to lowercase. For example: “INFORMATION TECHNOLOGY” will be “information technology”.

None: No transformations - the text is presented as it appears in the HTML code.

Example:

Display the heading in Capital letters and list items in uppercase.

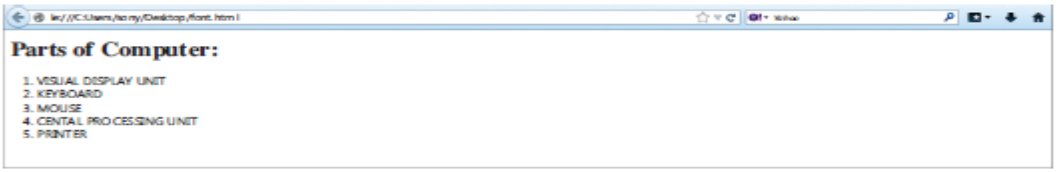
Code to be inserted in font.html

```
<html >
<head>
<title>Example </title>
<link rel=”stylesheet” href=”ex1.css” type=”text/css” media=”all” />
</head>
<body>
<h1>Parts of Computer:</h1>
<ol>
<li> Visual Dispaly Unit</li>
<li> Keyboard </li>
<li> Mouse </li>
<li> Central Processing Unit </li>
<li> Printer </li>
</ol>
</body>
</html>
```

Code to be inserted in ex1.css

```
h1 {
        text-transform: capitalize;
}

li  {
        text-transform: uppercase;
}
```

Output produced by the following above code:



Module 3 - Border Properties and Back ground Properties

The border properties allow you to specify how the border of the box representing an element should look. There are three properties of a border you can change.

a. The **border-color** specifies the color of a border.
b. The **border-style** specifies whether a border should be solid, dashed line, double line, or one of the other possible values.
c. The **border-width** specifies the width of a border.

The border - color property:

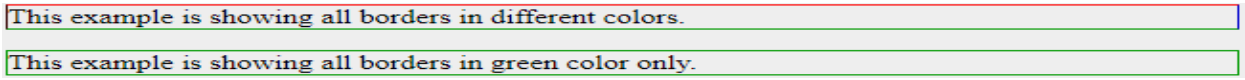
The border-color property allows you to change the color of the border surrounding an element. You can individually change the color of the bottom, left, top and right sides of an element's border using the properties

a. **border-bottom-color** changes the color of bottom border.
b. **border-top-color** changes the color of top border.
c. **border-left-color** changes the color of left border.
d. **border-right-color** changes the color of right border.

Example:

```
<html><head>
<style type = "text/css">
p.example1 {
border:1px solid;
border-bottom-color:#009900; /* Green */
border-top-color:#FF0000; /* Red */
border-left-color:#330000; /* Black */
border-right-color:#0000CC; /* Blue */
}
p.example2 {
border:1px solid;
border-color:#009900; /* Green */
}
</style>
</head>
<body>
<p class = "example1">
This example is showing all borders in different colors.
</p>
<p class = "example2">
This example is showing all borders in green color only.
</p>
</body></html>
```

Output:



The border-style property:

The border-style property allows you to select one of the following styles of border

a. **none** − No border. (Equivalent of border-width:0;)
b. **solid** − Border is a single solid line.
c. **dotted** − Border is a series of dots.
d. **dashed** − Border is a series of short lines.
e. **double** − Border is two solid lines.
f. **groove** − Border looks as though it is carved into the page.
g. **ridge** − Border looks the opposite of groove.
h. **inset** − Border makes the box look like it is embedded in the page.
i. **outset** − Border makes the box look like it is coming out of the canvas.
j. **hidden** − Same as none, except in terms of border-conflict resolution for table elements.

You can individually change the style of the bottom, left, top, and right borders of an element using the following properties:

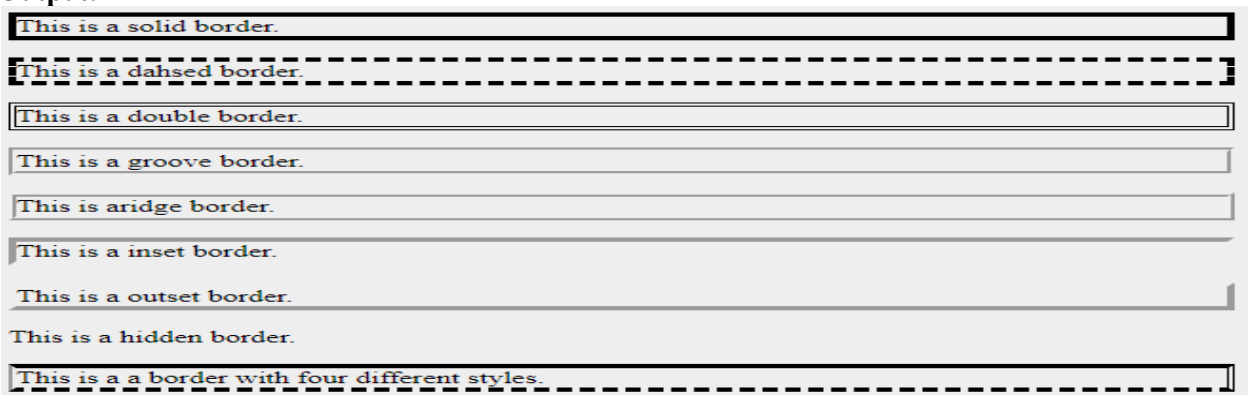
a. **border-bottom-style** changes the style of bottom border.
b. **border-top-style** changes the style of top border.
c. **border-left-style** changes the style of left border.
d. **border-right-style** changes the style of right border.

Example:

```
<html><head>
</head>
<body>
<p style = "border-width:4px; border-style:none;">
This is a border with none width.
</p>
<p style = "border-width:4px; border-style:solid;">
This is a solid border.
</p>
<p style = "border-width:4px; border-style:dashed;">
```

This is a dashed border.
* *
* *
This is a double border.
* *
* *
This is a groove border.
* *
* *
This is a ridge border.
* *
* *
This is a inset border.
* *
* *
This is a outset border.
* *
* *
This is a hidden border.
* *
<p style = "border-width:4px;
border-top-style:solid;
border-bottom-style:dashed;
border-left-style:groove;
border-right-style:double;">
This is a border with four different styles.
* *
</body>
</html>

Output:



The border width property:

The border-width property allows you to set the width of an element borders. The value of this property could be either a length in px, pt or cm or it should be set to thin, medium or thick.

You can individually change the width of the bottom, top, left, and right borders of an element using the following properties.

a. **border-bottom-width** changes the width of bottom border.
b. **border-top-width** changes the width of top border.
c. **border-left-width** changes the width of left border.
d. **border-right-width** changes the width of right border

Example:

<html>
<head>
</head>
<body>
* *

This is a solid border whose width is 4px.
* *
* *
This is a solid border whose width is 4pt.
* *
* *
This is a solid border whose width is thin.
* *
* *
This is a solid border whose width is medium;
* *
* *
This is a solid border whose width is thick.
* *
<p style = "border-bottom-width:4px;border-top-width:10px;
border-left-width: 2px;border-right-width:15px;border-style:solid;">
This is aa border with four different width.
* *
</body>
</html>

Output:

This is a solid border whose width is 4px.

This is a solid border whose width is 4pt.

This is a solid border whose width is thin.

This is a solid border whose width is medium;

This is a solid border whose width is thick.

This is a a border with four different width.

Background Properties:

1. FOREGROUND-COLOR
2. BACKGROUND-COLOR
3. BACKGROUND-IMAGE
4. BACKGROUND-REPEAT

Foreground color: the 'color' property:

The color property describes the foreground color of a text to be displayed in browser.
Example, display all headlines in a document to be in green color.

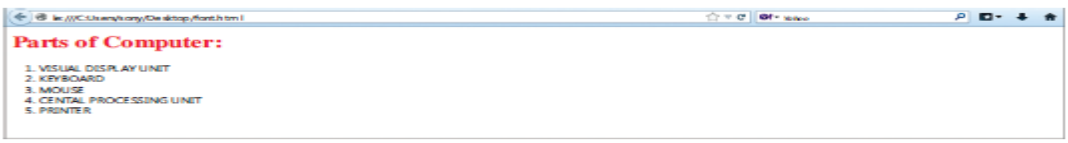
Code to be inserted in font.html:

```
<html >
<head>
<title>Example </title>
<link rel="stylesheet" href="ex1.css" type="text/css" media="all" />
</head>
<body>
    <h1>Parts of Computer:</h1>

    <ol>
    <li> Visual Dispaly Unit</li>
    <li> Keyboard </li>
    <li> Mouse </li>
    <li> Central Processing Unit </li>
    <li> Printer </li>
    </ol>
</body>
</html>
```

Code to be inserted in ex1.css:

```
h1 {
            color: #ff0000;
    }
```

Output produced by the following above code:



Background-color property:

The background-color property describes the background color of browser window. To change the background color of an entire page, the background-color property should be applied to the <body> tag. You can also apply background colors to other elements including headlines and text.

Example, apply different background colors to <body> and <h1> tags.

Code to be inserted in font.html

```
<html >
<head>
<title>Example </title>
<link rel="stylesheet" href="ex1.css" type="text/css" media="all" />
</head>
<body>
    <h1> Example displaying foreground and background colour.</h1>
</body>
</html>
```

Code to be inserted in ex1.css

```
body {
        background-color: #FFCC60;
}

h1 {
        color: #990011;
        background-color: #FC9004;
}
```

Notice that two properties have been applied to <h1> by dividing them by a semicolon.

Code produced by the following above code:

Background images [background-image]:

The background-image property is used to insert a background image in a web page. To insert the image of the earth as a background image for a web page, simply apply the background-image property to <body> and specify the location of the image.

Code to be inserted in font.html

```
<html >
<head>
<title>Example </title>
<link rel="stylesheet" href="ex1.css" type="text/css" media="all" />
</head>
<body>
    <h1> Inserting Image</h1>

</body>
</html>
```

Code to be inserted in ex1.css

```
body {
        background-color: #FFCC66;
        background-image: url("earth.gif");
}

h1 {
        color: #990000;
        background-color: #FC9804;
}
```

Output to be produced by the above following code:

Image will be inserted by giving the specified the location of the image as **url("earth.gif")**. This means that the image is located in the same folder as the style sheet. You can also refer to images in other folders using **url("../images/earth.gif")** or even on the Internet indicating the full address of the file: **url("http://www.example.net/earth.gif")**.

Repeat background image [background-repeat]:

As you have seen in the example above, that by default the image of the earth was repeated both horizontally and vertically to cover the entire screen. The background-repeat property controls this behavior. The four different values for background-repeat are as follows:

Value	Description
background-repeat: repeat-x	The image is repeated horizontally
background-repeat: repeat-y	The image is repeated vertically
background-repeat: repeat	The image is repeated both horizontally and vertically
background-repeat: no-repeat	The image is not repeated

For example, to avoid repetition of a background image the following code can be used:

Code to be inserted in font.html

```
<html >
<head>
<title>Example </title>
<link rel="stylesheet" href="ex1.css" type="text/css" media="all" />
</head>
<body>
    <h1> Inserting Image</h1>
</body>
</html>
```

Code to be inserted in ex1.css

```
body {
        background-color: #FFCC66;
        background-image: url("earth.gif");
        background-repeat: no-repeat;
}

h1 {
        color: #990000;
        background-color: #FC9804;
}
```

Output to be produced by the following above code:

Multiple Choice Questions

1) Can we define the text direction via CSS property?
   a) True b) False
2) Is it possible to declare font-weight, font-face & font-size by using only one CSS property?
   a) True b) False
3) If we want to use a nice looking dotted border around an image, which CSS property will we use?
   a) border-style b) border-color c) border-width d) all the above
4) Which element is used in the <HEAD> section on an HTML page, if we want to use an external style sheet file to decorate the page?
   a) <style> b) <link> c) <src> d) <css>
5) When we write <imgsrc="img.png">, what "img.png" inside double quote implies?
   a) Element b) tag c) attribute d) value
6) How will you make all paragraph elements 'RED' in color ?
   a) p.all {color: red;} b) p {color: red;} c) p.all {color: #990000;} d) all.p {color: #998877;}
7) How do you display a border like this: The top border = 10 pixelsThe bottom border = 5 pixelsThe left border = 20 pixelsThe right border = 1pixel?
   a) border-width:10px 20px 5px 1pxb) border-width:10px 1px 5px 20px
   c) border-width:5px 20px 10px 1px d) border-width:10px 5px 20px 1px
8) Which of the following statements is not correct?
   a) classes are identified a number sign (#)
   b) classes are identified with a dot (.)
   c) ids are identified a number sign (#)
9) In what form are style rules presented?
   a) selector { property: value }
   b) selector { property = value }
   c) selector (property: value)
   d)selector (property= value)
10) How do you display hyperlinks without an underline?
   a) a {text-decoration: no underline}
   b) a {decoration:no underline}
   c) a {text-decoration:none}
   d) a {underline:none}

11) To define the space between the element's border and content, you use the padding property, but are you allowed to use negative values?
    a) True b) False
12) How do you make a list that lists its items with squares?
    a) list-type: square b) type: squarec) type: 2 d) list-style-type: square
13) In CSS what does h1 is called
    a) Selectorb)Tag c)Attribute d)Value
14) Which of the following does CSS not do?
    a) Layoutb)Style c)Design d)Content
15) How do you insert a comment in a CSS file?
    a) /* this is a comment */ b)' this is a comment c) // this is a comment // d) // this is a comment
16) Which HTML tag is used to define an internal style sheet?
    a) <style>b)<css> c)<script>
17) Which CSS property controls the text size?
    a) text-size b) text-style c) font-size d) font-style
18) How do you add a background color for all <h1> elements?
    a) h1.all {background-color:#FFFFFF;}
    b) h1 {background-color:#FFFFFF;}
    c) all.h1 {background-color:#FFFFFF;}
    d) None

Descriptive Questions:

1. Explain CSS with reference to DHTML
2. List some advantages and disadvantages of CSS.
3. What is the extension of a CSS file?
4. Explain how would we embedded Style in your HTML.
5. List down the various font-family property ? Give an example of each.
6. What do you mean by font-variant? Give example of each.
7. Explain font-weight?
8. Mention the properties of CSS used to insert Letter spacing in a line.
9. How many types of text alignments can be included in a CSS page.
10. How the text in a webpage can be capitalized using CSS properties?
11. Explain the CSS properties to set the foreground and background color of the webpages.
12. Which property of CSS controls the repetition of image inserted in a web page as a background?

Unit 5 - Introduction to JS

Module 1 - Introduction to JavaScript

JavaScript is a cross-platform, object-oriented scripting language used to make web pages interactive (e.g., having complex animations, clickable buttons, popup menus, etc.). There are also more advanced server side versions of JavaScript such as Node.js, which allow you to add more functionality to a website than downloading files (such as realtime collaboration between multiple computers). Inside a host environment (for example, a web browser), JavaScript can be connected to the objects of its environment to provide programmatic control over them.

History of JavaScript: It was created in 1995 by Brendan Eich while he was an engineer at Netscape. It was originally going to be named LiveScript but was renamed. Unlike most programming languages, the JavaScript language has no concept of input or output. It is designed to run as a scripting language in a host environment, and it is up to the host environment to provide mechanisms for communicating with the outside world. The most common host environment is the browser.

What is JavaScript? JavaScript is an interpreted computer programming language. • it was originally implemented as a part of web browser so that client-side scripts could interact with the user, control the browser, communicate asynchronously, and alter the contents of the document. • It is tightly integrated with HTML code but executed by the JavaScript Interpreter built into the browser. • JavaScript contains a core set of objects such as array, date, Math, and a core set of language elements like operators, control structures, and statements.

Features of JavaScript: According to a recent survey conducted by **Stack Overflow**, JavaScript is the most popular language on earth.

With advances in browser technology and JavaScript having moved into the server with Node.js and other frameworks, JavaScript is capable of so much more. 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 another object. 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 needed.

Platform for 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. All detailed view is just a click away, and this is possible only because of JavaScript. It uses Application Programming Interfaces(APIs) that provide extra power to the code. The Electron and React is helpful in this department.
- **Server Applications:** With the help of Node.js, JavaScript made its way from client to server and node.js is the most powerful in the server-side.
- **Games:** Not only in websites, JavaScript also helps in creating games for leisure. The combination of JavaScript and HTML 5 makes JavaScript popular in game development as well. It provides the EaseJS library which provides solutions for working with rich graphics.
- **Smartwatches:** JavaScript is being used in all possible devices and applications. It provides a library PebbleJS which is used in smartwatch applications. This framework works for applications that require the internet for its functioning.
- **Art:** Artists and designers can create whatever they want using JavaScript to draw on HTML 5 canvas, make sound more effective also can be used p5.js library.
- **Machine Learning:** This JavaScript ml5.js library can be used in web development by using machine learning.

Limitations of JavaScript:

- **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. But as JavaScript is used to perform simple tasks in a browser, so performance is not considered a big restriction in its use.

- **Complexity:** To master a scripting language, programmers must have a thorough knowledge of all the programing concepts, core language objects, client and server side objects otherwise it would be difficult for them to write advance scripts using JavaScript.
- **Weak error handling and type checking facilities:** It is weakly typed language as there is no need to specify the data type of the variable. So wrong type checking is not performed by compiler

Applications/Uses of JavaScript

1) Developing Multimedia Applications: the users can use JavaScript to add multimedia elements. With JavaScript you can show, hide, change, resize images and create images rollovers. you can create scrolling text across the status bar, thus making multimedia applications more interactive.
2) Create Pages Dynamically: based on the user's choice, JavaScript can generate pages that are customized by the user.
3) Interact with the user: JavaScript can do some processing of forms and can validate user input when the user submits the form.
4) JavaScript objects are similar to dictionaries: in JavaScript, objects are just a collection of name-value pairs. JavaScript objects are considered as a dictionary with string keys. The users can get and set the properties of an object using either the familiar "." (dot) operator, or the "()" operator, which is typically used when dealing with a dictionary.
5) Extension: JavaScript can be extended for different purposes by supplementing it with additional objects.
6) Client-Side JavaScript: It extends the core language by supplying objects to control a browser (navigator or another web browser) and its document object Model (DOM). For example, client-side extensions allow an application to place elements on an HTML form and respond to the user events such as mouse clicks, form input and page navigation.
7) Server-Side JavaScript: extends the core language by supplying objects relevant to running JavaScript on a server. For example, server-side extensions allow an application to communicate with a relational database, which provide continuity of information from one invocation to another of the application, or perform file manipulations on a server.

Basic Structure of JavaScript program Client-side JavaScript code is embedded in HTML code between the either tag or tag surrounded by tag. If code contains the object, functions definitions then it is preferred to add the code between the tag. If code have to placed one or multiple scripts in a document, then tag is used.
Below block shows the basic structure of JavaScript code but it is not limited to this.

```
<!DOCTYPE>
<html>
<head><title></title></head>
<body >
<script type="text/javascript">
Add javascript code here
</script>
</body>
</html>
```

Module 2 - Interactivity in HTML

JavaScript can be implemented using JavaScript statements that are placed within the **<script>... /script>** HTML tags in a web page. You can place the **<script>** tags, containing your JavaScript, anywhere within your web page, but it is normally recommended that you should keep it within the **<head>** tags.
The <script> tag alerts the browser program to start interpreting all the text between these tags as a script. A simple syntax of your JavaScript will appear as follows.

```
<script ...>
   JavaScript code
</script>
```

The script tag takes two important attributes −

- **Language** − This attribute specifies what scripting language you are using. Typically, its value will be javascript. Although recent versions of HTML (and XHTML, its successor) have phased out the use of this attribute.
- **Type** − This attribute is what is now recommended to indicate the scripting language in use and its value should be set to "text/javascript".

So your JavaScript segment will look like −<script language = "javascript" type = "text/javascript">
JavaScript code</script>

Your First JavaScript Code

Let us take a sample example to print out "Hello World". We added an optional HTML comment that surrounds our JavaScript code. This is to save our code from a browser that does not support JavaScript. The comment ends with a "//-->". Here "//" signifies a comment in JavaScript, so we add that to prevent a browser from reading the end of the HTML comment as a piece of JavaScript code. Next, we call a function **document.write** which writes a string into our HTML document.

This function can be used to write text, HTML, or both. Take a look at the following code.

```
<html><body><script language = "javascript" type = "text/javascript">
<!--
document.write("Hello World!")
      //-->
</script></body></html>
```

This code will produce the following result −Hello World!

Whitespace and Line Breaks

JavaScript ignores spaces, tabs, and newlines that appear in JavaScript programs. You can use spaces, tabs, and newlines freely in your program and you are free to format and indent your programs in a neat and consistent way that makes the code easy to read and understand.

Semicolons are Optional

Simple statements in JavaScript are generally followed by a semicolon character, just as they are in C, C++, and Java. JavaScript, however, allows you to omit this semicolon if each of your statements are placed on a separate line. For example, the following code could be written without semicolons.

```
<script language = "javascript" type = "text/javascript">
<!--
   var1 = 10
   var2 = 20
  //-->
</script>
```

But when formatted in a single line as follows, you must use semicolons −

```
<script language = "javascript" type = "text/javascript">
<!--
   var1 = 10; var2 = 20;
  //-->
</script>
```

Note − It is a good programming practice to use semicolons.

Case Sensitivity

JavaScript is a case-sensitive language. This means that the language keywords, variables, function names, and any other identifiers must always be typed with a consistent capitalization of letters.

So the identifiers **Time** and **TIME** will convey different meanings in JavaScript.

NOTE − Care should be taken while writing variable and function names in JavaScript.

Comments in JavaScript

JavaScript supports both C-style and C++-style comments, Thus −

- Any text between a // and the end of a line is treated as a comment and is ignored by JavaScript.
- Any text between the characters /* and */ is treated as a comment. This may span multiple lines.
- JavaScript also recognizes the HTML comment opening sequence <!--. JavaScript treats this as a single-line comment, just as it does the // comment.
- The HTML comment closing sequence --> is not recognized by JavaScript so it should be written as //-->.

Example: The following example shows how to use comments in JavaScript.

```
<script language = "javascript" type = "text/javascript">
<!--
   // This is a comment. It is similar to comments in C++
   /*
   * This is a multi-line comment in JavaScript
   * It is very similar to comments in C Programming
   */
  //-->
</script>
```

There is a flexibility given to include JavaScript code anywhere in an HTML document. However, the most preferred ways to include JavaScript in an HTML file are as follows −

- Script in <head>...</head> section.
- Script in <body>...</body> section.
- Script in <body>...</body> and <head>...</head> sections.
- Script in an external file and then include in <head>...</head> section.

In the following section, we will see how we can place JavaScript in an HTML file in different ways.

JavaScript in <head>...</head> section

If you want to have a script run on some event, such as when a user clicks somewhere, then you will place that script in the head as follows −

```
<html><head>
<script type = "text/javascript">
<!--
        function sayHello() {
alert("Hello World")
        }
      //-->
</script>
</head>
<body>
<input type = "button" onclick = "sayHello()" value = "Say Hello" />
</body></html>
```

This code will produce the following results − JavaScript in <body>...</body> section

If you need a script to run as the page loads so that the script generates content in the page, then the script goes in the <body> portion of the document. In this case, you would not have any function defined using JavaScript.

Take a look at the following code.

```
<html><head></head>
<body>
<script type = "text/javascript">
<!--
document.write("Hello World")
      //-->
</script>
<p>This is web page body </p>
</body></html>
```

This code will produce the following results − JavaScript in <body> and <head> Sections. You can put your JavaScript code in <head> and <body> section altogether as follows −

```
<html><head>
<script type = "text/javascript">
<!--
        function sayHello() {
alert("Hello World")
        }
      //-->
</script>
</head>
<body>
<script type = "text/javascript">
<!--
document.write("Hello World")
      //-->
</script>
<input type = "button" onclick = "sayHello()" value = "Say Hello" />
</body></html>
```

This code will produce the following result −JavaScript in External File

As you begin to work more extensively with JavaScript, you will be likely to find that there are cases where you are reusing identical JavaScript code on multiple pages of a site.
You are not restricted to be maintaining identical code in multiple HTML files. The **script** tag provides a mechanism to allow you to store JavaScript in an external file and then include it into your HTML files.
Here is an example to show how you can include an external JavaScript file in your HTML code using **script** tag and its **src** attribute.

```
<html><head>
<script type = "text/javascript" src = "filename.js" ></script>
</head>
<body>
    .....
</body></html>
```

To use JavaScript from an external file source, you need to write all your JavaScript source code in a simple text file with the extension ".js" and then include that file as shown above.
For example, you can keep the following content in **filename.js** file and then you can use **sayHello** function in your HTML file after including the filename.js file.

```
function sayHello() {
alert("Hello World")
}
```

JavaScript supports the following forms of if..else statement −

- if statement
- if...else statement
- if...else if... statement.

if statement
The **if** statement is the fundamental control statement that allows JavaScript to make decisions and execute statements conditionally.
Syntax
The syntax for a basic if statement is as follows −

```
if (expression) {
  Statement(s) to be executed if expression is true
}
```

Here a JavaScript expression is evaluated. If the resulting value is true, the given statement(s) are executed. If the expression is false, then no statement would be not executed. Most of the times, you will use comparison operators while making decisions.
Example: Try the following example to understand how the **if** statement works.

```
<html><body>
<script type = "text/javascript">
<!--
var age = 20;

if( age> 18 ) {
document.write("<b>Qualifies for driving</b>");
        }
      //-->
</script>
<p>Set the variable to different value and then try...</p>
</body></html>
```

Output: Qualifies for driving
Set the variable to different value and then try...
if...else statement
The **'if...else'** statement is the next form of control statement that allows JavaScript to execute statements in a more controlled way.
Syntax

```
if (expression) {
  Statement(s) to be executed if expression is true
} else {  Statement(s) to be executed if expression is false}
```

Here JavaScript expression is evaluated. If the resulting value is true, the given statement(s) in the 'if' block, are executed. If the expression is false, then the given statement(s) in the else block are executed.
Example: Try the following code to learn how to implement an if-else statement in JavaScript.

```
<html><body>
<script type = "text/javascript">
<!--
var age = 15;
if( age> 18 ) {
document.write("<b>Qualifies for driving</b>");
        } else {
document.write("<b>Does not qualify for driving</b>");
        }
      //-->
</script>
<p>Set the variable to different value and then try...</p>
</body></html>
```

Output: **Does not qualify for driving**
Set the variable to different value and then try...
if...else if... statement
The **if...else if...** statement is an advanced form of **if…else** that allows JavaScript to make a correct decision out of several conditions.
Syntax
The syntax of an if-else-if statement is as follows −

```
if (expression 1) {
  Statement(s) to be executed if expression 1 is true
} else if (expression 2) {
  Statement(s) to be executed if expression 2 is true
} else if (expression 3) {
  Statement(s) to be executed if expression 3 is true
} else {
  Statement(s) to be executed if no expression is true
}
```

There is nothing special about this code. It is just a series of **if** statements, where each **if** is a part of the **else** clause of the previous statement. Statement(s) are executed based on the true condition, if none of the conditions is true, then the **else** block is executed.
Example: Try the following code to learn how to implement an if-else-if statement in JavaScript.

```
<html><body>
<script type = "text/javascript">
<!--
var book = "maths";
if( book == "history" ) {
document.write("<b>History Book</b>");
        } else if( book == "maths" ) {
document.write("<b>Maths Book</b>");
        } else if( book == "economics" ) {
document.write("<b>Economics Book</b>");
        } else {
document.write("<b>Unknown Book</b>");
        }
      //-->
</script>
<p>Set the variable to different value and then try...</p>
</body><html>
```

Output: **Maths Book**
Set the variable to different value and then try...

Module 3 - Introducing basic effects into web page

You can use JavaScript to create a complex animation having, but not limited to, the following elements −

- Fireworks, Fade Effect, Roll-in or Roll-out, Page-in or Page-out and Object movements

You might be interested in existing JavaScript based animation library:

This module provides a basic understanding of how to use JavaScript to create an animation.

JavaScript can be used to move a number of DOM elements (<img />, or any other HTML element) around the page according to some sort of pattern determined by a logical equation or function.

JavaScript provides the following two functions to be frequently used in animation programs.

- **setTimeout(function, duration)** − This function calls **function** after **duration** milliseconds from now.
- **setInterval(function, duration)** − This function calls **function** after every **duration** milliseconds.
- **clearTimeout(setTimeout_variable)** − This function calls clears any timer set by the setTimeout() functions.

JavaScript can also set a number of attributes of a DOM object including its position on the screen. You can set *top* and left attribute of an object to position it anywhere on the screen. Here is its syntax.

// Set distance from left edge of the screen.
object.style.left = distance in pixels or points;
or
// Set distance from top edge of the screen.
object.style.top = distance in pixels or points;

Manual Animation

So let's implement one simple animation using DOM object properties and JavaScript functions as follows. The following list contains different DOM methods.

- We are using the JavaScript function **getElementById()** to get a DOM object and then assigning it to a global variable **imgObj**.
- We have defined an initialization function **init()** to initialize **imgObj** where we have set its **position** and **left** attributes.
- We are calling initialization function at the time of window load.
- Finally, we are calling **moveRight()** function to increase the left distance by 10 pixels. You could also set it to a negative value to move it to the left side.

Example:

```
<html><head>
<title>JavaScript Animation</title>
<script type = "text/javascript">
<!--
varimgObj = null;
                function init() {
imgObj = document.getElementById('myImage');
imgObj.style.position= 'relative';
imgObj.style.left = '0px';
        }
        function moveRight() {
imgObj.style.left = parseInt(imgObj.style.left) + 10 + 'px';
        }
window.onload = init;
      //-->
</script>
</head>
<body>
<form>
<img id = "myImage" src = "/images/html.gif" />
<p>Click button below to move the image to right</p>
<input type = "button" value = "Click Me" onclick = "moveRight();" />
</form>
</body></html>
```

Output: Automated Animation

In the above example, we saw how an image moves to right with every click. We can automate this process by using the JavaScript function **setTimeout()** as follows −

Here we have added more methods. So let's see what is new here −

- The **moveRight()** function is calling **setTimeout()** function to set the position of *imgObj*.
- We have added a new function **stop()** to clear the timer set by **setTimeout()** function and to set the object at its initial position.

Example:

```
<html><head>
<title>JavaScript Animation</title>
<script type = "text/javascript">
<!--
varimgObj = null;
varanimate ;
                function init() {
imgObj = document.getElementById('myImage');
imgObj.style.position= 'relative';
imgObj.style.left = '0px';
        }
        function moveRight() {
imgObj.style.left = parseInt(imgObj.style.left) + 10 + 'px';
          animate = setTimeout(moveRight,20);    // call moveRight in 20msec
        }
        function stop() {
clearTimeout(animate);
imgObj.style.left = '0px';
        }
window.onload = init;
      //-->
</script></head>
<body>
<form>
<img id = "myImage" src = "/images/html.gif" />
<p>Click the buttons below to handle animation</p>
<input type = "button" value = "Start" onclick = "moveRight();" />
<input type = "button" value = "Stop" onclick = "stop();" />
</form>
</body></html>
```

Rollover with a Mouse Event

Here is a simple example showing image rollover with a mouse event.

Let's see what we are using in the following example −

- At the time of loading this page, the 'if' statement checks for the existence of the image object. If the image object is unavailable, this block will not be executed.
- The **Image()** constructor creates and preloads a new image object called **image1**.
- The src property is assigned the name of the external image file called /images/html.gif.
- Similarly, we have created **image2** object and assigned /images/http.gif in this object.
- The # (hash mark) disables the link so that the browser does not try to go to a URL when clicked. This link is an image.
- The **onMouseOver** event handler is triggered when the user's mouse moves onto the link, and the **onMouseOut** event handler is triggered when the user's mouse moves away from the link (image).
- When the mouse moves over the image, the HTTP image changes from the first image to the second one. When the mouse is moved away from the image, the original image is displayed.
- When the mouse is moved away from the link, the initial image html.gif will reappear on the screen.

```
<html>
<head>
<title>Rollover with a Mouse Events</title>
<script type = "text/javascript">
<!--
        if(document.images) {
var image1 = new Image();     // Preload an image
           image1.src = "/images/html.gif";
var image2 = new Image();     // Preload second image
           image2.src = "/images/http.gif";
         }
       //-->
</script></head>
<body>
<p>Move your mouse over the image to see the result</p>
<a href = "#" onMouseOver = "document.myImage.src = image2.src;"
onMouseOut = "document.myImage.src = image1.src;">
<img name = "myImage" src = "/images/html.gif" />
</a>
</body></html>
```

Multiple Choice Questions

1) Which type of JavaScript language is ___
   a) Object-Oriented b)Object-Based c) Assembly-language b) High-level
2) Which of the following is the correct output for the following JavaScript code:

```
varx=5,y=1
var obj ={ x:10}
with(obj)
{
    alert(y)
}
```

   a) 1 b) Error c) 10 d) 5
3) In JavaScript, what is a block of statement?
   a. Conditional block
   b. block that combines a number of statements into a single compound statement
   c. both conditional block and a single statement
   d. block that contains a single statement
4) The "function" and " var" are known as:
   a. Keywords b. Data types c. Declaration statements d. Prototypes

```
var x=3;
var y=2;
var z=0;
If(x==y)
document.write(x);
elseif(x==y)
document.write(x);
else
document.write(z);
```

5) Which of the following is the correct output for the following JavaScript code
   a. 3 b. 0 c. Error d.2

Descriptive Questions

1) What is client side java script and server side java script?
2) Write java script history and features of it?
3) What are the limitations of java script
4) write any simple java script in html
5) Write java script example for if, if..else and if..else if
6) write any 3 basic effects using java script

UNIT 6 -Basics of Algorithms and Flow charts

Module 1 – Introduction to problem Solving

Introduction:

Intelligence is one of the key characteristics which differentiate a human being from other living creatures on the earth. Basic intelligence covers day to day problem solving and making strategies to handle different situations which keep arising in day to day life. One person goes Bank to withdraw money. After knowing the balance in his account, he/she decides to with draw the entire amount from his account but he/she has to leave minimum balance in his account. Here deciding about how much amount he/she may with draw from the account is one of the examples of the basic intelligence. During the process of solving any problem, one tries to find the necessary steps to be taken in a sequence. In this Unit you will develop your understanding about problem solving and approaches.

Problem Solving:

Can you think of a day in your life which goes without problem solving? Answer to this question is of course, No. In our life we are bound to solve problems. In our day to day activity such as purchasing something from a general store and making payments, depositing fee in school, or withdrawing money from bank account. All these activities involve some kind of problem solving. It can be said that whatever activity a human being or machine do for achieving a specified objective comes under problem solving. To make it clearer, let us see some other examples.

Example1: If you are watching a news channel on your TV and you want to change it to a sports channel, you need to do something i.e. move to that channel by pressing that channel number on your remote. This is a kind of problem solving.

Example 2: One Monday morning, a student is ready to go to school but yet he/she has not picked up those books and copies which are required as per timetable. So here picking up books and copies as per timetable is a kind of problem solving.

Example 3: If someone asks to you, what is time now? Seeing time in your watch and telling is also a kind of problem solving.

Example 4: Some students in a class plan to go on picnic and decide to share the expenses among them. So calculating total expenses and the amount an individual have to give for picnic is also a kind of problem solving.

- Now, we can say that problem is a kind of barrier to achieve something and problem solving is a process to get that barrier removed by performing some sequence of activities.
- Here it is necessary to mention that all the problems in the world cannot be solved. There are some problems which have no solution and these problems are called Open Problems.
- If you can solve a given problem, then you can also write an algorithm for it. In next section we will learn what is an algorithm?

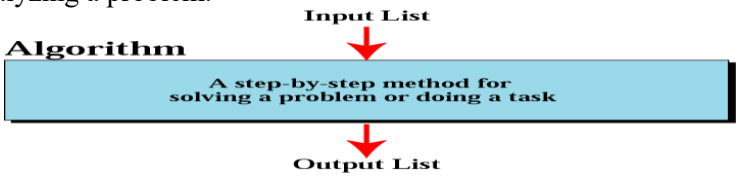
Module 2 –Algorithm

Introduction

Algorithm is a step-by-step process of solving a well-defined computational problem. In practice, in order to solve any complex real-life problems, first we have to define the problem and then, design algorithm to solve it. Writing and executing a simple program may be easy; however, for executing a bigger one, each part of the program must be well organized. In short, algorithms are used to simplify the program implementation. The study of algorithms is one of the key foundations of computer science. Algorithms are essential to the way computers process information, because a computer program is essentially an algorithm that tells the computer what specific steps to perform (in what specific order) in order to carry out a specified task, such as calculating employees' paychecks or printing students' report cards. Thus, an algorithm can be considered to be any sequence of operations that can be performed by a computing device such as a computer.

What is an Algorithm?

- Algorithm is a step by step procedure to solve a particular problem or "A sequence of activities to be processed for getting desired output from a given input.
- It's a easy way of analyzing a problem.



Structure of an algorithm:

START

- STEP 1
- SETP 2
- SETP 3
- SETP 4
- ……so on

STOP

Example 1: A simple algorithm to print „Good Morning".

Step 1: Start

Step 2: Print „Good Morning"

Step 3: Stop

Example 2: Let us take one simple day-to-day example by writing algorithm for making "Maggi Noodles" as a food.

Inputs to the algorithm: Maggie Noodles packet, Pan, Water, Gas

Expected output: Maggie Noodles

Algorithm:

Step 1: Start
Step 2: Take pan with water
Step 3: Put pan on the burner
Step 4: Switch on the gas/burner
Step 5: Put Maggie and masala
Step 6: Give two minutes to boil
Step 7: Take off the pan
Step 8: Take out the magi with the help of fork/spoon
Step 9: Put the Maggie on the plate and serve it
Step 10: Stop

Example3: Find the area of a Circle of radius r.

Inputs to the algorithm: Radius r of the Circle.

Expected output: Area of the Circle

Algorithm:

Step1: Start
Step2: Read\input Radios
Step3: Set 3.14 to PI
Step4: Set Area ←PI* Radios * Radios // calculation of area
Step5: Print Area
Step6: stop

Expressing Algorithms:

An algorithm is a set of steps designed to solve a problem or accomplish a task. Algorithms are usually written in pseudo code, or a combination of your speaking language and one or more programming languages, in advance of writing a program**.** An algorithm may be expressed in a number of ways, including:

- **Natural language:** usually verbose and ambiguous.
- **Flow charts:** avoid most (if not all) issues of ambiguity; difficult to modify w/o specialized tools; largely standardized.
- **Programming language:** Tend to require expressing low – level details that are not necessary for a high – level understanding.

Qualities of a good algorithm

1) Inputs and outputs should be defined precisely.
2) A good algorithm should produce correct and accurate results for any set of legal or correct inputs.
3) Each step in algorithm should be clear and unambiguous.
4) Algorithm should be most effective among many different ways to solve a problem.
5) An algorithm shouldn't have computer code. Instead, the algorithm should be written in such a way that it can be used in similar programming languages Whenever we write an algorithm, we should make sure that all of these parameters given for a good algorithm are incorporated.

Advantage of algorithm

1) It is a step-wise representation of a solution to a given problem, which makes it easy to understand.
2) An algorithm uses a definite procedure.
3) It is not dependent on any programming language, so it is easy to understand for anyone even without programming knowledge.
4) Every step in an algorithm has its own logical sequence so it is easy to debug.
5) By using algorithm, the problem is broken down into smaller pieces or steps hence, it is easier for programmer to convert it into an actual program

Disadvantagesof algorithm

1) Writing algorithm takes a long time.
2) An Algorithm is not a computer program; it is rather a concept of how a program should be
3) Big tasks are difficult to put in Algorithms.
4) Difficult to show Branching and Looping in Algorithms.

Properties of algorithm:

1) **Finiteness:** An algorithm must always terminate after a finite number of steps. It means after every step one reach closer to solution of the problem and after a finite number of steps algorithm reaches to an end point.
2) **Definiteness:** Each step of an algorithm must be precisely defined. It is done by well thought actions to be performed at each step of the algorithm. Also the actions are defined unambiguously for each activity in the algorithm.
3) **Input:** An algorithm has zero or more inputs, taken from a specified set of objects.
4) **Output:** An algorithm has one or more outputs, which have a specified relation to the inputs.
5) **Effectiveness:** All operations to be performed must be sufficiently basic that they can be done exactly and in finite length

Algorithm Efficiency

Algorithmic efficiency is a property of an algorithm which relates to the number of computational resources used by the algorithm. An algorithm must be analyzed to determine its resource usage, and the efficiency of an algorithm can be measured based on usage of different resources.

- Speed - Method that takes the least time
- Space - Method that uses the least memory
- Code — Method that is shortest to describe

Speed is now the most important factor Example

A real world example: Travelling from Vempalli to Hyderabad in Vehicle

Case 1:

1. Take vehicle
2. Check vehicle condition if not get it repair
3. Drive from vempalli to Proddutur
4. Then Proddutur to Hyderabad
5. Travel is successfully completed.

Case 2:

1. Take vehicle
2. Check vehicle condition if not get it repair
3. Drive from Vempalli to kadapa
4. Then Kadapa tokurnool
5. Then Kurnool to Hyderabad
6. Travel is successfully completed

While comparing both cases Algorithm executed successfully with same output but in case 1 travelling time is lesser than the case 2 travelling time.

Tracing an Algorithm

Tracing of an Algorithm means showing how the value of a variable changes during the execution of an algorithm.

Steps in tracing:

1. Identify the variables in the algorithm that need to be traced.
2. Examine the value of the variables at each step in the execution of the algorithm.
3. Determine if the algorithm is giving the correct outputs for a set of legitimate/legal input. Values.
4. Analyze and determine what the purpose of the algorithm

In this module we will see examples of computational algorithms and the techniques used for tracing algorithms. A computational algorithm is a set of precise instructions expected to be expressed as a computer program that can execute on a computer.

Let us now get started with an example of adding two numbers.

Algorithm: Add two numbers

Step1: Start
Step2: Input x
Step3: Input y
Step4: Set z ←x+ y
Step5: Output z
Step6: Stop

If you notice the algorithm it takes two inputs, performs an addition operation and gives the output. Let us see how the algorithm executes step by step.

Input x/* Takes input from the user by using input devices like keyboard. After entering input from the key board, for ex: 5 the value is stored in a memory location. The name of this memory location is x in this particular case. Any time this variable x can store only one value. For ex: if you set x value 6 you will find the new value is referred by x is 6 but not 5 that means the old value which was stored in x values is lost. This is a very good example that shows a variable can store only one value at a time. Input value :5*/ Input y /* Similar to previous statement where new memory location is created for variable y and the user inputs to y is stored.Input value :2*/ Set z to x+ y /* in this statement actual operation that is performed on the input values which is to add the numbers referred to by the variables x and y. In this example you will notice after the addition operation is executed and the values 5 and 2 are added and set to new variable z. */

Output z //the result value i.e. z is displayed to an output device like monitor?

You will notice that algorithm also can be visualized as functions. A function performs a specific operation like addition and multiplication or it could be more complex operations like finding square roots or sorting list of numbers.

Variable:

- Variables are used to handle the data by the algorithm.
- Variables refer to a memory location where the actual value is stored.
- Variable can store one value at a time.
- Old values of the variable are lost when a new value is assigned.

Control structure:

Control Structures are just a way to specify flow of control in programs. Any algorithm or program can be clearer and more understood if they use self-contained modules called as logic or control structures. It basically analyzes and chooses in which direction a program flows based on certain parameters or conditions.

Types of control structures:

There are three types of control structures

1) Sequence
2) Selection or Branching
3) Loop or Repetition

Sequence Control Structure: This refers to the line – by – line execution, in which statements are executed sequentially, in the same order in which they appear in the script.They might, for example, carry out a series of read or write operations, arithmetic operations, or assignments to variables.

Example problem:Write algorithm to find the greater number between two numbers

Step1: Start
Step2: Read\input the Number1
Step3: Read\input the Number2.
Step4: Set Average ← (Number1+ Number2)/2
Step5: Print Average
Step6: End

Selection or Decision Control Structure: The branch refers to a binary decision based on some condition. Depending on whether a condition is true or false. If the condition is true, one of the two branches is explored; if the condition is false, the other alternative is taken. This is usually represented by the 'if-then' construct in pseudo-codes and programs. This structure is also known as the selection structure

Example problem:A person whose age is more than or equals to 18 years is eligible to vote. Now write an algorithm to check whether he is eligible to vote?

Step 1: Start

Step2: Input/Read age //Taking age value from the user

Step 3: Check if age >= 18

a) Print "Eligible to vote" //If Condition is true

Step 4: Otherwise/else

a) Print "Not eligible to vote" //if Condition is false

Step 5: Stop

Repetition or Loop Control Structure: This is a control structure that allows the execution of a block of statements multiple times until a specified condition is met. The loop allows a statement or a sequence of statements to be repeatedly executed based on some loop condition. It is represented by the 'while' and 'for' constructs in most programming languages, for unbounded loops and bounded loops respectively. (Unbounded loops refer to those whose number of iterations depends on the eventuality that the termination condition is satisfied; bounded loops refer to those whose number of iterations is known before-hand.) In the flowcharts, a back arrow hints the presence of a loop. A trip around the loop is known as iteration. You must ensure that the condition for the termination of the looping must be satisfied after some finite number of iterations, otherwise it ends up as an infinite loop, a common mistake made by inexperienced programmers. The loop is also known as the repetition structure.

Example problem:Algorithm to print all-natural numbers upto "n" (Here you need to accept a number from the user, and to print all the natural numbers till 'n' i.e. 1 to n)

Step 1: Start

Step2: Read/Input n

Step 3: Store 1 to "i" //initialization

Step 4: Do the following statements until i<=n//condition

a) print i

b) add 1 to i//increment

Step 5: Stop

Nested control structures: Combining the use of these control structures, for example, a loop within a loop (nested loops), a branch within another branch (nested if), a branch within a loop, a loop within a branch, and so forth, is not uncommon. Complex algorithms may have more complicated logic structure and deep level of nesting, in which case it is best to demarcate parts of the algorithm as separate smaller modules. Beginners must train themselves to be proficient in using and combining control structures appropriately, and go through the trouble of tracing through the algorithm before they convert it into code.

Module 3 - Introduction to Flow chart

Flowchart

The diagrammatic representation of an algorithm is called "Flowchart". Before you start coding a program it is necessary to plan the step by step solution to the task your program will carry out. Such a plan can be symbolically developed using a diagram. This diagram is then called a flowchart. Hence a flowchart is a symbolic representation of a solution to a given task. A flowchart can be developed for practically any job. Flowcharting is a tool that can help us to develop and represent graphically program logic sequence.

For example, suppose you are going for a picnic with your friends then you plan for the activities you will do there. If you have a plan of activities, then you know clearly when you will do what activity. Similarly, when you have a problem to solve using computer or in other word you need to write a computer program for a problem then it will be good to draw a flowchart prior to writing a computer program. Flowchart is drawn according to defined rules.

Features of flowchart:

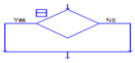
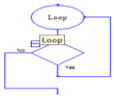
The features of a flowchart are:

1. It is an easy method of communication.
2. It is independent of a programming language.
3. It is the key to correct programming.
4. It clearly indicates the task to be performed at each level.

Flowchart Symbols

There are 6 basic symbols commonly used in flowcharting of assembly language Programs: Terminal, Process, and input/output, Decision, Connector and Predefined Process. This is not a complete list of all the possible flowcharting symbols; it is the ones used most often in the structure of Assembly language programming.

Purpose	Symbol	Name	Description
Start/Stop		Start & Stop	It is Oval symbol
INPUT		Input statement	Allow the user to enter data. Each data value is stored in a ***variable***.
OUTPUT		Output statement	Display (or save to a file) the value of a ***variable***.
PROCESSING		Assignment statement	An assignment statement is used to modify or replace the data stored at the memory location associated with a variable. .

Purpose	Symbol	Name	Description
PROCESSING (Function)		Procedure call	Execute a group of instructions defined in the named procedure. In some cases some of the procedure arguments (i.e., ***variables***) will be changed by the procedure's instructions.
Selection		Decision making statement	Allows you to make 'decision ' in boolean type
Loop		Iteration statements	Which allows you to execute repeatedly until certain condition satisfied. (An iteration control statement controls how many times a block of code is executed)

1) **Process Symbol:**
   - A process symbol is used to represent arithmetic and data movement instructions in the flowchart. All arithmetic processes of addition, subtraction, multiplication and division are indicated in the process symbol.
   - The logical process of data movement from one memory location to another is also represented in the process box. If there are more than one process
2) **Input/ Output Symbol:**
   - This symbol is used to denote any input/output function in the program. Thus, if there is any input to the program via an input device, like a keyboard, tape, card reader etc. it will be indicated in the flowchart with the help of the Input/output symbol.
   - Similarly, all output instructions, for output to devices like printers, plotters, magnetic tapes, disk, monitors etc. are indicated in the Input/ Output symbol.
3) **Decision Symbol:**
   - The decision symbol is used in a flowchart to indicate the point where a decision is to be made and branching done upon the result of the decision to one or more alternative paths. The criteria for decision making are written in the decision box.
   - All the possible paths should be accounted for. During execution, the appropriate path will be followed depending upon the result of the decision.
4) **Loops:** Looping is a problem-solving technique through which a group of statements is executed repeatedly, until certain specified condition is satisfied. Looping is also called a repetitive or an iterative control statement
5) **Connectors:**
   This symbol shows continuation of the flow chart from one page to another. When you reach the bottom of the page or need to jump to another page, draw flow chart connector symbol and connect it to the last item on the chart. Label the inside of the symbol with a letter, typically beginning with an "A".
6) **Predefined process (Function):**
   This symbol indicates a sequence of actions that perform a specific task embedded within a larger process.
7) **Terminal Symbol:**
   - Every flowchart has a unique starting point and an ending point.
   - The flowchart begins at the start terminator and ends at the stop terminator.
   - The Starting Point is indicated with the word START inside the terminator symbol. The Ending Point is indicated with the word STOP inside the terminator symbol. There can be only one START and one STOP terminator in your entire flowchart.

General Rules for flowcharting

1) All boxes of the flowchart are connected with Arrows. (Not lines)
2) Flowchart symbols have an entry point on the top of the symbol with no other entry points. The exit point for all flowchart symbols is on the bottom except for the Decision symbol.
3) The Decision symbol has two exit points; these can be on the sides or the bottom and one side.
4) Generally, a flowchart will flow from top to bottom. However, an upward flow can be shown as long as it does not exceed 3 symbols.
5) Connectors are used to connect breaks in the flowchart. Examples are:
    a. From one page to another page.
    b. From the bottom of the page to the top of the same page.
6) Subroutines (Functions) and Interrupt programs have their own and independent flowcharts.
7) All flow charts start with a Terminal or Predefined Process (for interrupt programs or subroutines) symbol.
8) All flowcharts end with a terminal or a contentious loop.

Flowcharting uses symbols that have been in use for a number of years to represent the type of operations and/or processes being performed. The standardized format provides a common method for people to visualize problems together in the same manner. The use of standardized symbols makes the flow charts easier to interpret, however, standardizing symbols is not as important as the sequence of activities that make up the process.

Advantages of using flowchart

As we discussed flow chart is used for representing algorithm in pictorial form. This pictorial representation of a solution/system is having many advantages. These advantages are as follows:

- **Communication:** A Flowchart can be used as a better way of communication of the logic of a system and steps involve in the solution, to all concerned particularly to the client of system.
- **Effective analysis:** A flowchart of a problem can be used for effective analysis of the problem.
- **Documentation of Program:** Program flowcharts are a vital part of good program documentation. Program document is used for various purposes like knowing the components in the program, complexity of the program etc.
- **Efficient Program Maintenance:** Once a program is developed and becomes operational it needs time to time maintenance. With help of flowchart maintenance become easier.
- **Coding of the Program:** Any design of solution of a problem is finally converted into computer program. Writing code referring the flowchart of the solution become easy.

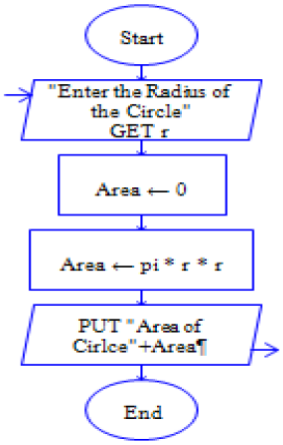
Disadvantagesof using flowchart:

- Drawing flowchart is a time-consuming task.
- Manual tracing is needed to check correctness of flowchart drawn on paper.
- Simple modification in problem logic may lead to complete redraw of flowchart.
- Showing many branches and looping in **flowchart** is difficult.
- In case of complex program/algorithm, **flowchart** becomes **very complex** and **clumpsy.**

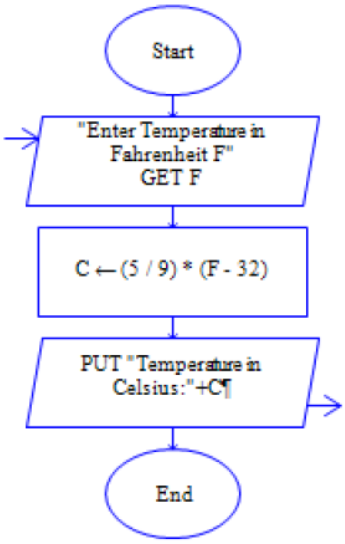
Some examples of Flowcharts

Now, we will discuss some examples on flowcharting. These examples will help in proper understanding of flowcharting technique. This will help you in program development process in next unit of this block.

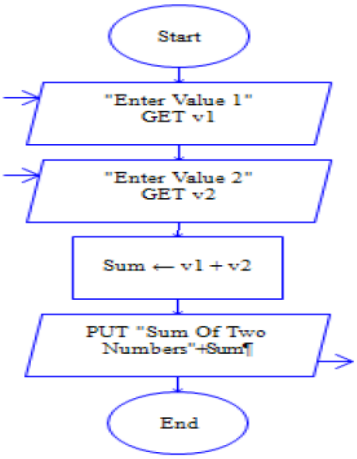
Problem1: Find the area of a circle of radius r



Problem 2: Convert temperature Fahrenheit to Celsius.



Problem3: Flowchart for an algorithm which gets two numbers and prints sum of their value.



Types of control structures:

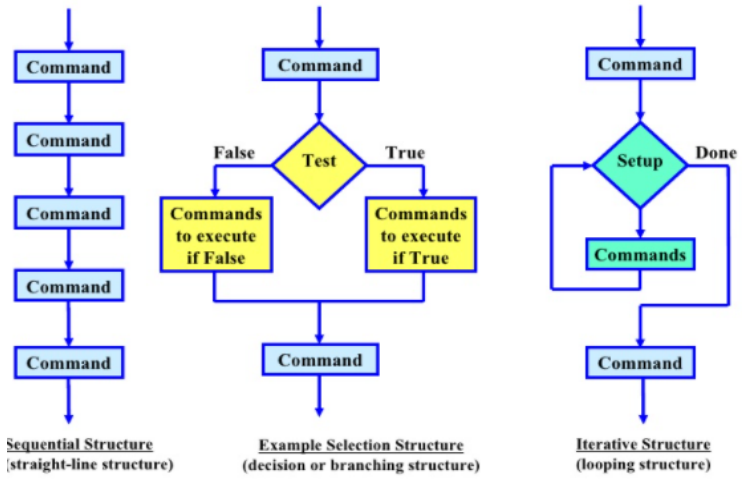
Definition

- The logic of a program may not always be a linear sequence of statements to be executed in that order.
- The logic of the program may require execution of a statement based on a decision.
- Control structures specify the statements to be executed and the order of execution of statements.

Where is the usage of control structure?

Flowchart and Pseudo Code use Control structures for representation

Flowcharts for sequential, selection, and iterative control structures



There are three kind of Control Structure:

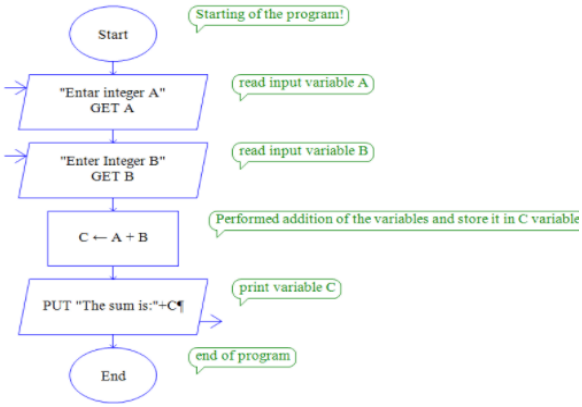
1) Sequential
2) Selection(Branch or conditional)
3) Iterative(loop)

Sequential:

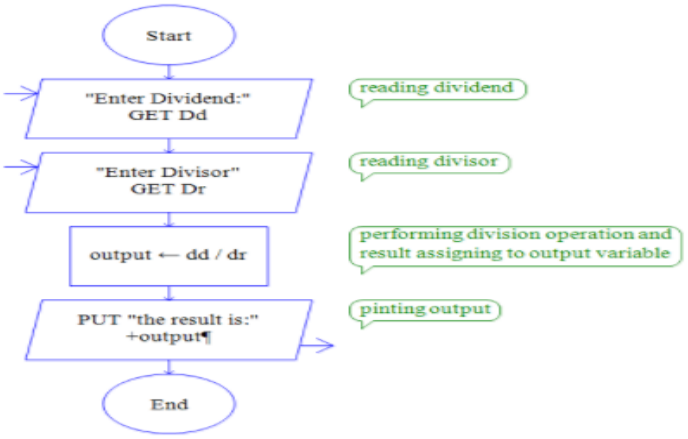
1) Instruction is executed in linear order.
2) Statements are executed in a specified order. No statement is skipped and no statement is executed more than once.

Example problem:

1) Take two numbers from the user print their sum?



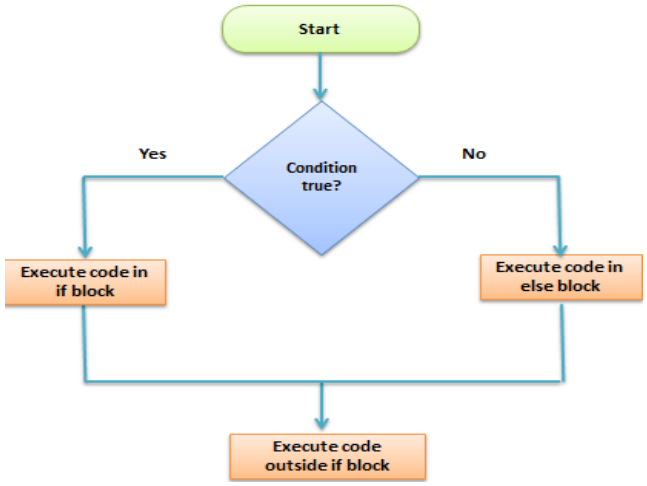
2) Take input of two numbers from the user and do the division of two numbers?



Selection (Branch or conditional):

- It's asked a true/false question THEN selects the next instruction based on the answer.
- It selects a statement to execute on the basis of **condition**. Statement is executed when the condition is true and ignored when it is false

Example: if, if else, switch structures.



Example problem:

3) Draw a flowchart to determine greatest among two numbers?

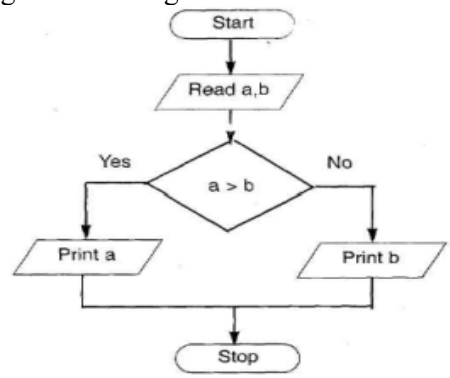
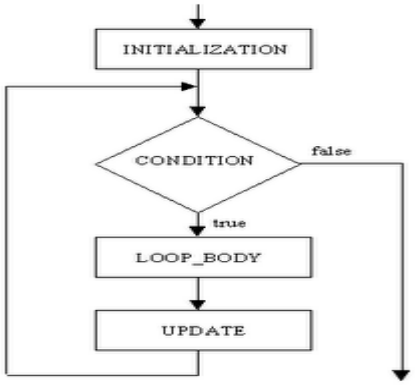


Fig. 2b) Flowchart to determine the greater of two numbers a and b

Iterative (loop):

- Its repeat the execution of a block of instruction.
- In this structure the statements are executed more than one time. It is also known as iteration or loop
- A loop structure is used to execute a certain set of actions for a predefined number of times or until a particular condition is satisfied

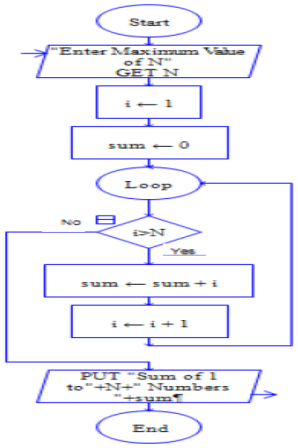
Example: while loop, for loop do-while loops etc.



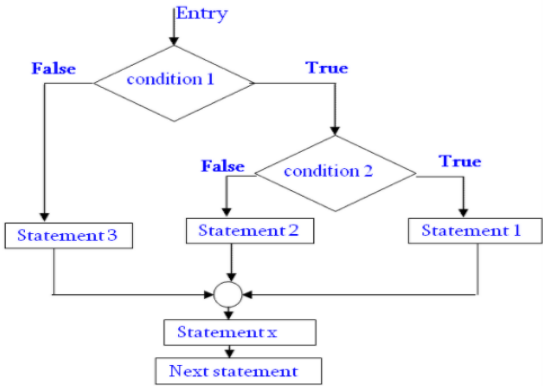
Example problem:Write an algorithm and flowchart to find Total Sum from 1 to 100 natural numbers

Step 1: Start
Step 2: Read value of n
Step 3: Set i←1
Step 4: Set Total_Sum← 0
Step 5: Repeat the steps until i greater th
a) Set Total_Sum←Total_Sum+i
b) increment i // (i.e. i=i+1)
Step 6: Display "1 to 100 Natural Numbers Sum:" Total_Sum
Step 7: End

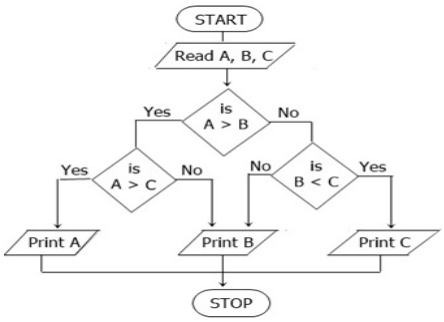
Iterative Structure



Nested Control Structures:A nested control statement is a control statement that is contained within another control statement. You can do this to many levels. You can place control statements inside other control statements, for example an If...Then...Else block within a for. Next loop. A control statement placed inside another control statement is said to be ***nested.***



Example problem:Draw a flowchart to determine largest among three numbers?



Pseudo code

A pseudo code is an informal way of writing a program. However, it is not a computer program. It only represents the algorithm of the program in natural language and mathematical notations. Besides, there is no particular programming language to write a pseudo code. Unlike in regular programming languages, there is no syntax to follow when writing a pseudo code. Furthermore, it is possible to use pseudo codes using simple English language statements.

- Pseudo code is a kind of structured English for describing algorithm.
- It has no syntax like any of the programming language and thus can't be compiled or interpreted by the computer.
- No syntax for Pseudo code
- Not executable program

Advantages of Pseudo code

- Improves the readability of any approach. It's one of the best approaches to start implementation of an algorithm.
- Acts as a bridge between the program and the algorithm or flowchart. Also works as a rough documentation, so the program of one developer can be understood easily when a pseudo code is written out. In industries, the approach of documentation is essential. And that's where a pseudo-code proves vital.
- The main goal of a pseudo code is to explain what exactly each line of a program should do, hence making the code construction phase easier for the programmer

How to write a Pseudo-code?

- To begin the comment double forward slash are used "//".
- Matching braces "{and}" are used to present blocks where a compound statement (set of simple statements) can be illustrated as a block and terminated by a semicolon"**;**". The body of a procedure constructs a block as well.
- All the identifiers start with a letter and the data type of the variables are not declared explicitly.
- An assignment statement is used for the assigning values to the variables.
- To produce the Boolean values (i.e., true and false) the logical operators and, or and not and the relational operators <, ≤, =, =, ≥ and > are provided.
  Input and output are presented by read and write instructions.

Example:A pseudo code to find the total of two numbers is as follows.

```
Sum Of Two Numbers ()
begin
Set sum =0;
Read: number 1, number 2;
Set sum = number1 + number 2;
Print sum;
End
```

2. A pseudo code to find the area of a triangle is as follows.

```
AreaofTrinagle()
Begin
Read: base,height;
Set area =0.5*base*height;
Print area;
End
```

Differences between algorithm and Flowchart:

Algorithm	Flow chart
Step by step instruction representing the process of any solution.	Block by block information diagram representing the data flow.
It is step wise analysis of the work to be done	It is a pictorial representation of a process
Solution is shown in non-computer language like English.	Solution is shown in graphical format.
It is something difficult to understand.	Easy to understand as compared to algorithm.
Difficult to show branching and looping.	Easy to show branching and looping.
Algorithm can be written for any problem.	Flow chart for big problem is impractical.
Easy to debug errors.	Difficult to debug errors.
It is difficult to write algorithm as compared to flowchart.	It is easy to make flowchart.

Differences between algorithm and Pseudocode:

Basis for comparison	Algorithm	Pseudo code
Comprehensibility	Quite hard to understand	Easy to interpret
Uses	Complicated programming language	Combination of programming language and natural language.
Debugging	Moderate	simpler
Ease of construction	Complex	Easier

Difference between flow chart and pseudo code:

Flow chart	Pseudo code
A diagrammatic representation that illustrates a solution model to a given problem.	An informal high – level description of the operating principle of an algorithm.
Written using various symbols.	Written in natural language and mathematical notations help to write pseudo code.

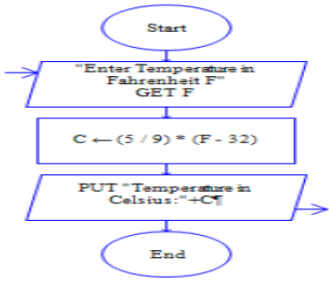
Solved problems:

1) Write an algorithm to go for class picnic.

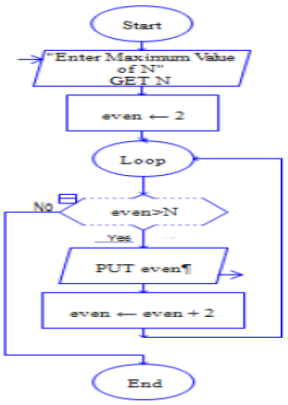
Algorithm:

Step 1: Start
Step 2: Decide the picnic venue, date and time
Step 3: Decide the picnic activities
Step 4: Hire a vehicle to reach to the venue and comeback
Step 5: Goto to the picnic venue on the decided date
Step 6: Do the activities planned for the picnic
Step 7: Come back to school in the hired vehicle
Step 8: Stop

2) To celebrate Teachers' Day
   Algorithm:
   Step 1: Start
   Step 2: Decide the activities for teachers' day like dance performances, plays, etc.
   Step 3: Form groups of students and assign the decided activities from step 2 to each group.
   Step 4: Decide the practice timings for each group.
   Step 5: Each group to practice as per the timings decided in step 4.
   Step 6: Invite the teachers to Teachers' Day celebrations.
   Step 7: Perform the activities planned in step 2 on Teachers' Day
   Step 8: Stop
3) To celebrate New Year
   Algorithm:
   Step 1: Start
   Step 2: Prepare a guest list for New Year party
   Step 3: Decide the venue, food menu, games and fun activities for the party
   Step 4: Invite the guests for the party
   Step 5: On New Year eve, get ready and enjoy the party
   Step 6: Stop
4) Convert temperature Fahrenheit to Celsius with flowchart
   Inputs to the algorithm: Temperature in Fahrenheit
   Expected output: Temperature in Celsius
   Algorithm:
   Step1: Start
   Step2: Read Temperature in Fahrenheit F
   Step3: Set C ←5/9*(F-32)
   Step4: Print "Temperature in Celsius: "+C
   Step5: End

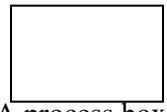


5) Write an algorithm to read two numbers and find their product?
   Inputs to the algorithm: First num1. Second num2.
   Expected output: Sum of the two numbers.
   Algorithm:
   Step1: Start
   Step2: Read\input num1.
   Step3: Read\input num2.
   Step4: set product ← num1 * num2 // calculation of product
   Step5: Print product.
   Step6: End
6) An algorithm to display even numbers between 0 and 99 with flowchart
   Step 1: Start
   Step2: input/read value of n
   Step3: Set even←0
   Step4: Do the following until even < 100
      a) Display even
      b) Set even ←even+2
   Step5: End



7) Design an algorithm which generates even numbers between 1000 and 2000 and then prints them in the standard output. It should also print total sum:
   Step 1: Start
   Step2: input/read value of n
   Step3: Set even 1000
   Step4: Do the following until even < 2000
      a) Display even
      b) Set even even+2
   Step5: End

8) Define the following symbol and use?



Answer: Process Box: A process box is used to represent all types of mathematical tasks like addition, subtraction, multiplication, division, etc.

9) Algorithm to find area of a rectangle?

Algorithm:

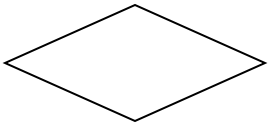
Step 1: Start
Step 2: Take length and breadth and store them as L and B? // Input
Step 4: Multiply L and B and store it in area //Area of Rectangle = L*B
Step 5: Print area //Output
Step 6: Stop.

10) Write an algorithm to find the largest among three different numbers entered by user with flowchart

Step 1: Start
Step 2: Read variables a, b and c.
Step 3: If a greater than b
    a) If a greater than c
        i) Display a is the largest number.
    b) else
        i) Display c is the largest number.
Step 4: else
    a) If b greater than c
        i) Display b is the largest number.
    b) else
        i) Display c is the greatest number.
Step 5: Stop

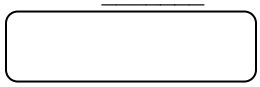
Multiple Choice Questions:

1) An Algorithm is expressed by
    a) Flowchart b) common language c) pseudo code d) all
2) A good algorithm having finite steps?
    a) True b) False
3) Algorithm efficiency is measured by?
    a) Speed b) space c) code d) all of the above
4) How many control structures are there in algorithm?
    a) 3 b) 2 c) 1
5) Which control structure used weather a condition is true or false?
    a) Selection b) iteration c) sequence d) none
6) Which control structure used to check a condition more than one time?
    a) Iteration b) loop c) a & b d) None of the above
7) In computer science, algorithm refers to a pictorial representation of a flowchart.
    a) True b) False
8) The process of drawing a flowchart for an algorithm is called _____
    a) Performance b) Evaluation c) Algorithmic Representation d) Flowcharting
9) Actual instructions in flowcharting are represented in _____
    a) Circles b) Boxes c) Arrows d) Lines
10) A box that can represent two different conditions.
    a) Rectangle b) Diamond c) Circle d) Parallelogram
11) The following box denotes?

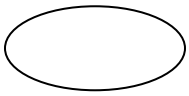


    a) Decision b) Initiation c) Initialization d) I/O
12) There should be certain set standards on the amount of details that should be provided in a flowchart
    a) True b) False

13) Which of the following is not an advantage of a flowchart?
    a) Better communication b) Efficient coding c) Systematic testing d) Improper documentation
14) The symbol denotes _____



    a) I/O n b) Flow c) Terminal d) Decision
15) Pseudo code is an informal high-level description of an algorithm.
    a) True b) false
16) In a flowchart a calculation(process) is represented by
    a) A rectangle b) A rhombus c) A parallelogram d) A circle



17) The above symbol denotes_____
    a) I/O b) Flow c) Loop d) Decision
18) To repeat a task a number times
    a) Loop statement b) Input statement c) Conditional statement d) Output statement
19) In a flow chart how are the symbols connected?
    a) Symbols do not get connected together in a flowchart
    b) With lines and arrow to show the direction of flow
    c) With dashed lines and numbers
    d) With solid lines to link events
20) A flow chart does need to have a start?
    a) True b) False

Unsolved Problems:

1) Write an algorithm and flow chart to add four numbers?
2) Write an algorithm to make tea/coffee
3) Write 6 No's Algorithms for performing day to day activities.
4) Write an algorithm and flow chart to find large number among given two numbers.
5) Draw the flow chart to convert distance entered in Km to Meters.
6) Accept the age of a person and check whether he/she is eligible to vote or not. A person is eligible to vote only when he/she is 18 years or more.
7) Draw the flow chart to find the area of a rectangle whose length and breadth are given
8) Draw the flow chart to find the average of three numbers a, b and c.
9) Write an algorithm and flow chart Print 1 to 100?
10) Write an algorithm to find average of three numbers?