CHAPTER – 6

Advanced Web Publishing (Java Script)

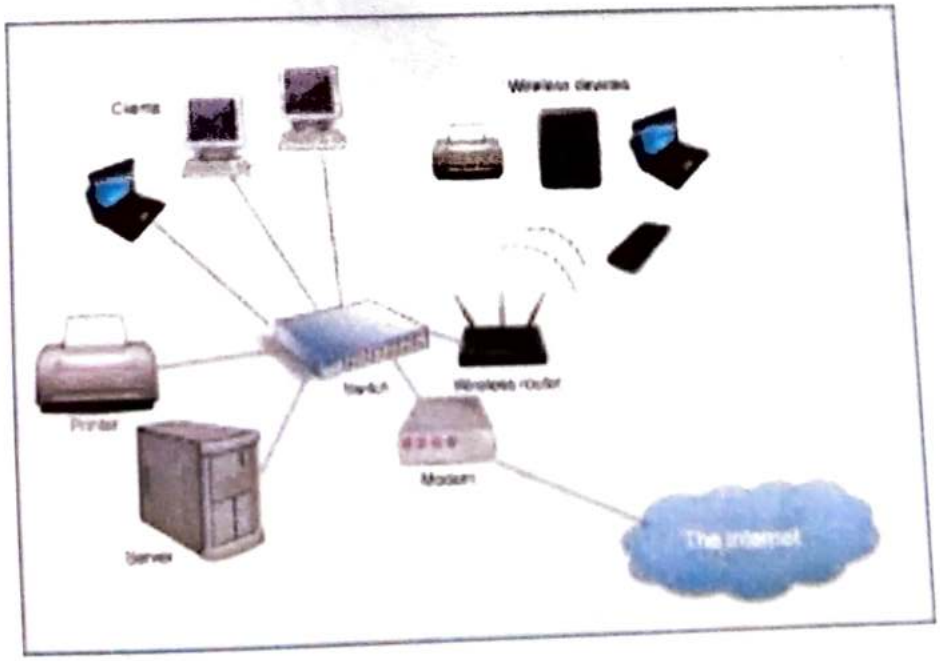
4.1. Networking Fundamentals

INTRODUCTION

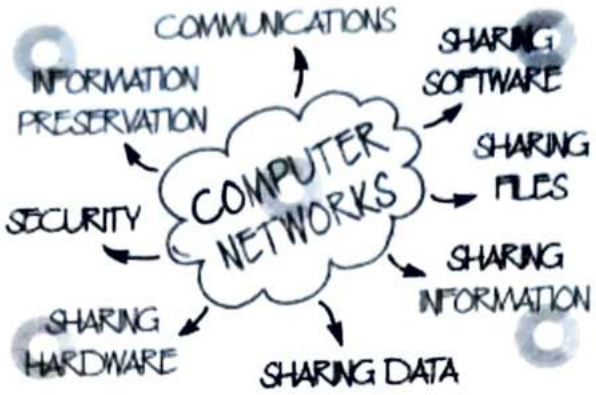
Networking is the practice of linking two or more computing devices together for the purpose of sharing data. Networks are built with computer hardware and computer software. Networking is referred as connecting computers electronically for the purpose of sharing information. Resources such as files, applications, printers and software are common information shared in a networking. The advantage of networking can be seen clearly in terms of security, efficiency, manageability and cost effectiveness as it allows collaboration between users in a wide range. Basically, network consists of hardware component such as computer, hubs, switches, routers and other devices which form the network infrastructure. These are the devices that play an important role in data transfer from one place to another using different technology such as radio waves and wires.

Computer Network :

Def :– A Computer network is an interconnection of geographically distributed multiple computers in such a way that meaningful transmission and exchange of information become possible. Simply, when two or more computers are directly or indirectly connected with each other for the purpose of sharing resources is known as computer network.".



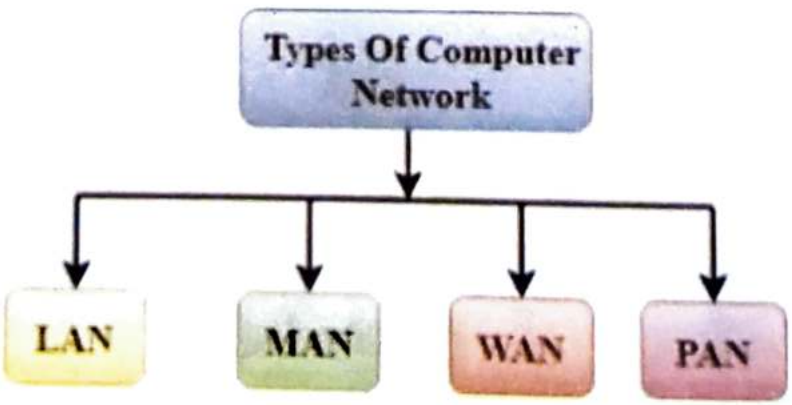
Advantages of Computer Network



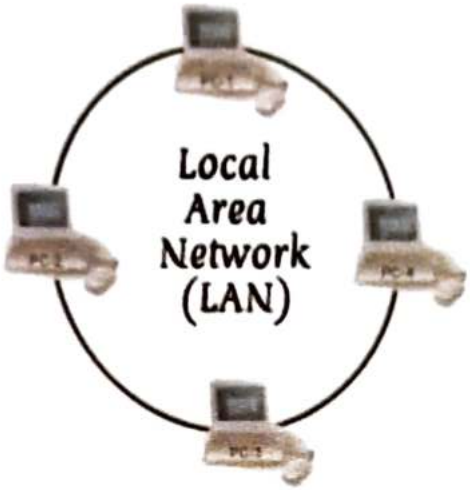
- Hardware such as printers can be shared by all the computers on the network.
- Some software and files such as databases can be shared by different users.
- Users can work together as networked computers can communicate with each other easily and quickly via email or internal messaging systems.
- An Internet connection can be shared.
- File storage facilities can be shared and files therefore accessed from any networked computer.
- Improved security as there is central control over user access, which programs, data and hardware users have access to.
- Files can easily be backed up centrally.

4.2 Computer Network can be classified into different categories:

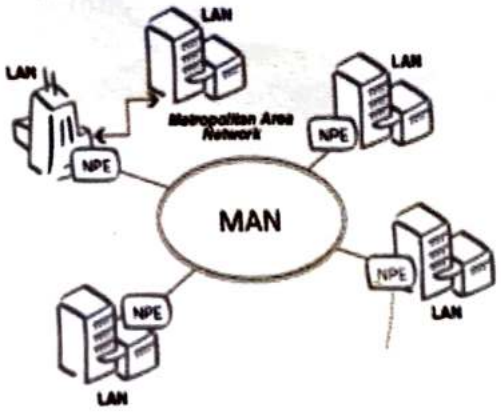
- **LAN (Local Area Network)**
- **MAN(Metropolitan Area Network)**
- **WAN(Wide Area Network)**
- **PAN(Personal Area Network)**



LAN(Local Area Network) : Local Area Network is a group of computers connected to each other in a small area such as building, office.LAN is used for connecting two or more personal computers through a communication medium such as twisted pair, coaxial cable, etc. It is less costly. The data is transferred at an extremely faster rate in Local Area Network. Local Area Network provides higher security.

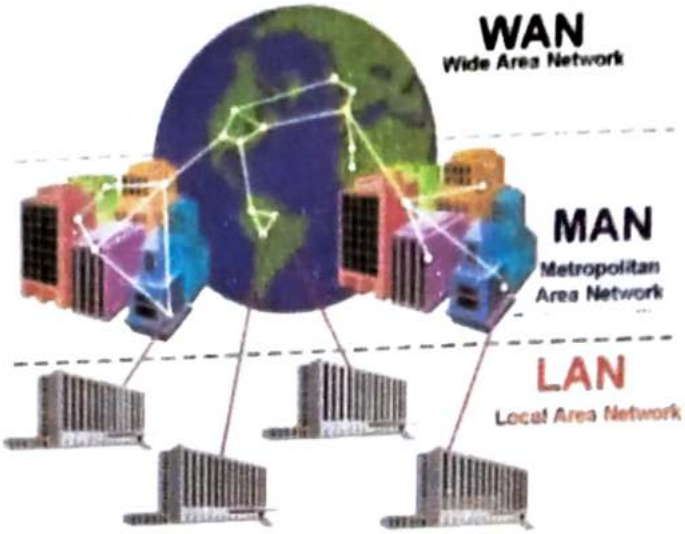


ii. **MAN(Metropolitan Area Network) :** A metropolitan area network is a network that covers a larger geographic area by interconnecting a different LAN to form a larger network. Government agencies use MAN to connect to the citizens and private industries. In MAN, various LANs are connected to each other through a telephone exchange line. It has a higher range than Local Area Network(LAN).e.g. cable TV network



iii. **WAN(Wide Area Network) :** A Wide Area Network is a network that extends over a large geographical area such as states or countries and continent or whole around the world. A Wide Area Network is quite bigger network than the LAN.A Wide Area Network is not limited to a single location. The internet is one of the biggest

WAN in the world. A Wide Area Network is widely used in the field of Business, government, and education.



iv. **PAN(Personal Area Network) :** Personal Area Network is a network arranged within an individual person, typically within a range of 10 meters. Personal computer devices that are used to develop the personal area network are the laptop, mobile phones, media player and play stations. Area Network covers an area of **30 feet**.

4.3 Network Protocols :

A network protocol defines rules and conventions for communication between network devices. Network protocols include mechanisms for devices to identify and make connections with each other, as well as formatting rules that specify how data is packaged into sent and received messages.

Classification of network protocols :

- TCP/IP
- HTTP
- FTP
- TELNET
- SMTP
- PPP

I. **IP addresses :** An IP address is a number identifying of a computer or another device on the Internet. Every machine on the internet has an unique number assigned to it know as IP address. An Internet Protocol address (IP address) is a logical numeric address that is assigned to every single computer, printer, switch, router or any other device that is part of a TCP/IP-based network. An IP address serves two principal functions: host or network interface identification and location addressing. Internet Protocol version 4 (IPv4) defines an IP address as a 32-bit number. An example of an IPv4 address is 192.168.10.35

II. **TCP/IP (Transmission Control Protocol/Internet Protocol) :** TCP/IP is a set of networking protocols that allows two or more computers to communicate. TCP/IP specifies how data is exchanged over the internet by providing end-to-end communications that identify how it should be broken into packets, addressed, transmitted, routed and received at the destination. TCP/IP requires little central management, and it is designed to make networks reliable, with the ability to recover automatically from the failure of any device on the network. TCP/IP uses the client/server model of communication in which a user or machine (a client) is provided a service (like sending a webpage) by another computer (a server) in the network.

There are only four layers in TCP/IP reference model :

Application layer :

Application layer is the top most layer of four layer TCP/IP model. Application layer is present on the top of the Transport layer. Application layer defines TCP/IP application protocols and how host programs interface with Transport layer services to use the network.

Application layer includes the protocols like DNS (Domain Naming System), HTTP (Hypertext Transfer Protocol), FTP (File Transfer Protocol), SMTP (Simple Mail Transfer Protocol) etc.

Internet Layer :

Internet Layer is the second layer of the four layer TCP/IP model. The position of Internet layer is between Network Access Layer and Transport layer. Internet layer pack data into data packets known as IP datagram, which contain source and destination address (logical address or IP address) information that is used to forward the datagram between hosts and across networks. The Internet layer is also responsible for routing of IP datagram. Packet switching network depends upon a connectionless internetwork layer. This layer is known as Internet layer.

Transport Layer :

Transport Layer is the third layer of the four layer TCP/IP model. The position of the Transport layer is between Application layer and Internet layer. The purpose of Transport layer is to permit devices on the source and destination hosts to carry on a conversation. Transport layer defines the level of service and status of the connection used when transporting data.

Network Access Layer

Network Access Layer is the first layer of the four layer TCP/IP model. Network Access Layer defines details of how data is physically sent through the network, This is the lowest level of the TCP/IP protocol

HTTP :

Hypertext Transfer Protocol, used for transmitting and displaying information in the form of web pages on browsers. Hyper Text Transfer Protocol (HTTP) is an application-layer protocol used primarily on the World Wide Web. HTTP uses a client-server model where the web browser is the client and communicates with the web server that hosts the website.

FTP :

File Transfer Protocol, used for file transfer (uploading and downloading) over the Internet.FTP copy files over a network from one computer to another. More generally, it provides for some simple file management on the contents of a remote computer.

TELNET :

Telnet is a protocol that allows you to connect to remote computers (called hosts) over a TCP/IP network (such as the internet). Using telnet client software on your computer, you can make a connection to a telnet server (that is, the remote host). *Telnet is a network protocol used on the Internet or local area networks to provide a bidirectional interactive text-oriented communications.*

SMTP :

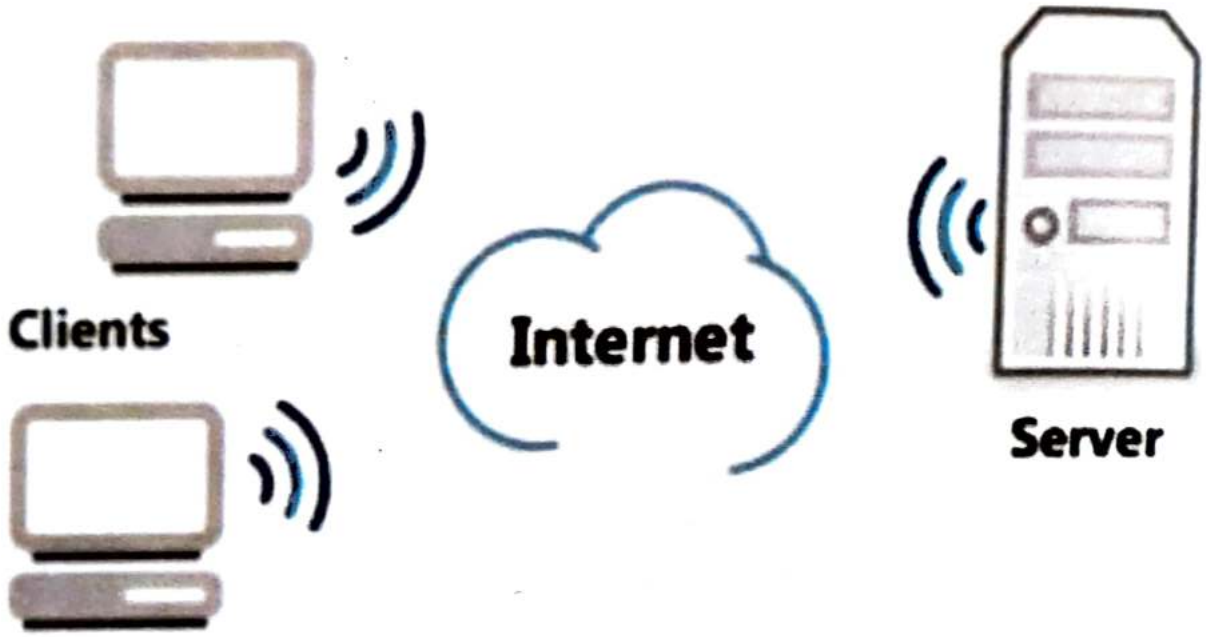
Simple Mail Transfer Protocol used for email. services on a TCP/IP network. SMTP provides the ability to send and receive email messages. SMTP is part of an application-layer protocol that enables the transmission and delivery of email over the Internet.

PPP :

Point - to - Point **Protocol** (**PPP**) is a communication **protocol** of the data link layer that is used to transmit multiprotocol data between two directly connected (point-to-point) computers. This protocol is used for a very basic level of connectivity providing data linkage between the computers. Point-to-point protocol is widely used for the heavier and faster connections.

4.4 Client/Server Architecture

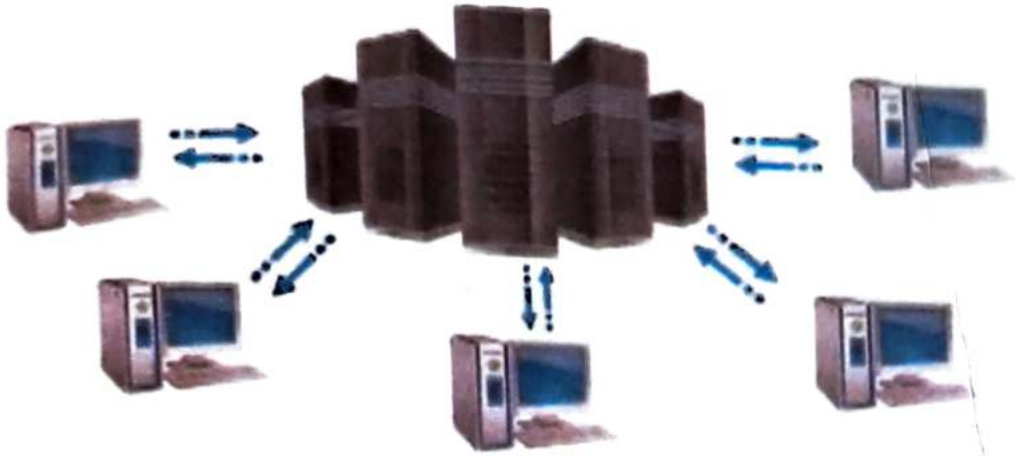
The client-server architecture is also termed as a network-computing structure because every request and their associated services are distributed over a network. So now the question is how the thing works? In the client-server architecture, when the client computer sends a request for data to the server through the internet, the server accepts the requested, process it and deliver the data packets requested back to the client.



i. **Client :** A **client** is a piece of **computer** hardware or software that accesses a service made available by a server. The server is often (but not always) on another **computer** system, in which case the **client** accesses the service by way of a network.

ii. **Server :** A **server** is a **computer** that provides data to other **computers**. It may serve data to systems on a local area network (LAN) or a wide area network (WAN) over the Internet. Many types of **servers** exist, including web **servers**, mail **servers**, and file **servers**. Each type runs software specific to the purpose of the **server**.

A **server** is a physical computer dedicated to run services to serve the needs of other computers. it could be a file server, database server, home media server, print server, or web server.



.5. Need of Web Publishing

ntroduction :

Web publishing, or "online publishing," is the process of publishing content on the nternet. It includes creating and uploading websites, updating web pages, and osting blogs online. The published content may include text, images, videos, and other ypes of media.

Website publishing is the process of uploading content on the internet. It ncludes :

- uploading files
- updating web pages
- posting blogs

Website is published by uploading files on the remote server which is provided by he hosting company.

In order to publish your site, you need the following things :

- Web development software
- Internet Connection
- Web Server

Web development software

It is used for building web pages for your web site. Dreamweaver and Word Press are example of web development software.

Internet Connection

Internet connection is required to connect to a remotely located web server.

Web Server

Web server is the actual location where your website resides on. A web server may host single or multiple sites depending on what hosting service you have paid for.

Web Languages:

Web Language is a scripting language for automating tasks on the World-Wide Web. Web programming refers to the writing, markup and coding involved in Web development, which includes Web content, Web client and server scripting and network security.

The most common languages used for Web programming are XML, HTML, JavaScript and PHP. Web programming is different from just programming, which requires interdisciplinary knowledge on the application area, client and server scripting, and database technology.

4.6 JavaScript Introduction

JavaScript is a client-side scripting language developed by Brendan Eich. **JavaScript** can be run on any operating systems and almost all web browsers. You need a text editor to write **JavaScript** code and a browser to display your web page.

Def :

Java Script is an object oriented language that allows creation of dynamic and interactive web pages. It is lightweight and most commonly used as a part of web pages. JavaScript is the programming language of HTML and the web. Java script allows user entries, which are loaded into HTML form to be processed as required. This empowers a website to return site information according to user's requests. Java script is a client side scripting language.

JavaScript runs on the client-side (browser's side), as opposed to server-side (web server). One benefit of doing this is performance. On the client side, JavaScript is loaded into the browser and can run as soon as it is called. JavaScript enables users to manipulate web pages without sending a request back to the server. Using these capabilities, users can change the contents of a page, change the style of elements on a page, validate user input before a user submits a form.

Advantages of JavaScript

- **Speed** : Client-side JavaScript is very fast because it can be run immediately within the client-side browser.
- **Simplicity:** JavaScript is relatively simple to learn and implement.
- **Popularity:** JavaScript is used everywhere in the web.
- **Server Load**: Being client-side reduces the demand on the website server.
- **No compilation needed**

JavaScript does not require compilation process so no compiler is needed. The owser interprets JavaScript as it HTML tags.

asy to debug and test

The understanding syntax of JavaScript is easy. Any person can learn it very easily d use it to develop dynamic and scalable websites.

latform independent

Any JavaScript-enabled browser can understand and interpret JavaScript code. Any vaScript code can be executed on different types of hardware a JavaScript program ritten for.

vaScript is an Event-Based Programming language

ii. Disadvantages of JavaScript

Security : Due to Client-Side scripting language, it is less Secure.

Browser Support :.

JavaScript is sometimes interpreted differently by different browsers.

4.7 Client-side and Server-side Scripting

Client-side Environment

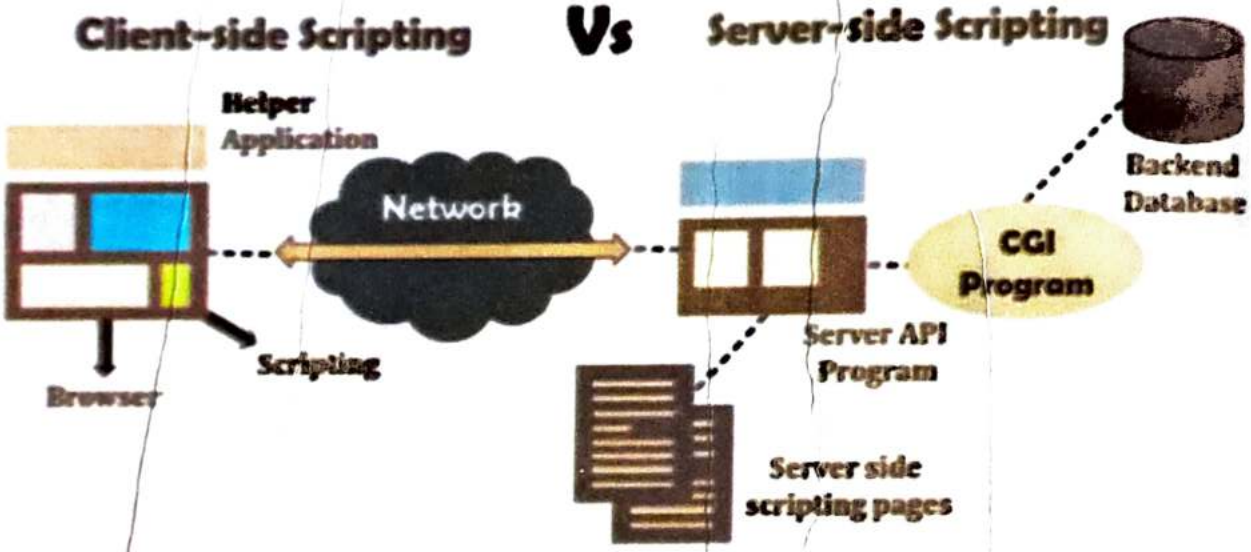
The client-side environment used to run scripts is usually a browser. The processing takes place on the end users computer. The source code is transferred from the web server to the users computer over the internet and run directly in the browser.

The scripting language needs to be **enabled** on the client computer. Sometimes if a user is conscious of **security risks** they may switch the scripting facility off. When this is the case a message usually pops up to alert the user when script is attempting to run.

Server-side Environment

The **server-side environment** that runs a scripting language is a web server. A user's request is fulfilled by running a script directly on the web server to generate dynamic HTML pages. This HTML is then sent to the client browser. It is usually used to provide interactive web sites that interface to databases or other data stores on the server.

This is different from client-side scripting where scripts are run by the viewing web browser, usually in JavaScript. The primary advantage to server-side scripting is the ability to highly customize the response based on the user's requirements, access rights, or queries into data stores.



S.No.	Client Side Scripting	Server Side Scripting
1.	Used when the users browser already have all the code.	Used to create dynamic pages.
2.	The web browser executes the client side scripting.	The web server executes the server side scripting.
3.	Cannot be used to connect to the databases on the web server.	Used to connect to the databases that reside on the web server.
4.	Can't access the file system that resides at the web server.	Can access the file system residing at the web server.
5.	Response from a client-side script is faster as compared to a server-side script.	Response from a server-side script is slower as compared to a client-side script.

4.8 JavaScript Variable

Variable means anything that can vary. JavaScript includes variables which hold the data value and it can be changed anytime.

JavaScript uses reserved keyword **var** to declare a variable. A variable must have a unique name. You can assign a value to a variable using **equal to** (=) operator when you declare it or before using it.

Syntax :

var <*variable-name*>;

var <*variable-name*> = <*value*>;

Example :

```
var x = 25; // variable stores numeric value
var y = 10; // variable stores numeric value
var name = 'HPBOSE';  // variable stores string value
var sum;  // declared a variable without assigning a value
var sum= x + y;
```

4.8.1 Variable Scope in JavaScript

Scope of a variable is the part of the program from where the variable may directly be accessible. The scope of a variable is controlled by the location of the variable declaration, and defines the part of the program where a particular variable is accessible. Scoping rules vary from language to language.

In JavaScript, there are two types of scopes :

1. **Global Scope** – Scope outside the outermost function attached to Window. Any variable declared outside of a function belongs to the global scope and is therefore accessible from anywhere in your code.

```
// global scope
var num = 15;
function one() {
  alert(num); // alerts '15'
}
```

2. **Local Scope** – Inside the function being executed.

```
// local scope
function three() {
  var num = 15;
  alert(num); // alerts 15'
}
```

Program : Write a program in JavaScript to find the sum of two numbers using variable.

Solution :

```
<html>
<body>
<script language="JavaScript">
var x = 50;
var y = 50;
var z=x+y;
document.write("<B>Sum of Values:    "+z+"");
</script>
</body>
</html>
```

Output :

Sum of Values: 100

Program : Write a program in JavaScript to display text message using variable.

Solution :

```
<html>
<body>
<script language="JavaScript">
var text = "Hi, I am JavaScript and You?";
 document.write("<B> "+text +"");
</script>
</body>
</html>
```

Output :

Hi, I am JavaScript and You?

4.9 Data Types in JavaScript

JavaScript provides different **data types** to hold different types of values. There are two types of data types in JavaScript.

1. Primitive data type
2. Non-primitive (reference) data type

JavaScript is a **dynamic type language**, means you don't need to specify type of the variable because it is dynamically used by JavaScript engine. You need to use **var** here to specify the data type. It can hold any type of values such as numbers, strings etc. For example:

```
var a=18;//holding number
var b="Pankaj Dhalouria";//holding string
```

1. Primitive data types

There are five types of primitive data types in JavaScript. They are as follows:

Data Type	Description
String	represents sequence of characters e.g. "hello"
Number	represents numeric values e.g. 100
Boolean	represents boolean value either false or true
Undefined	represents undefined value
Null	represents null i.e. no value at all

2. Non-primitive data types

The non-primitive data types are as follows :

Data Type	Description
Object	represents instance through which we can access members
Array	represents group of similar values
RegExp	represents regular expression

4.10 Operators in JavaScript

An **operator** is capable of manipulating a certain value or operand. **Operators** are used to perform specific mathematical and logical computations on operands. In other words, we can say that an **operator** operates the operands. In **JavaScript operators** are used for compare values, perform arithmetic operations etc.

Let us take a simple expression **7 + 5 is equal to 12**. Here 7 and 5 are called **operands** and '+' is called the **operator**.

JavaScript supports the following types of operators.

- Arithmetic Operators
- Comparison Operators
- Logical (or Relational) Operators
- String Operators

Category	Operator	Name/Description	Example	Result
Arithmetic	+	Addition	3+2	5
	-	Subtraction	3-2	1
	*	Multiplication	3*2	6
	/	Division	10/5	2
	%	Modulus	10%5	0
	++	Increment and then return value	X=3; ++X	4
		Return value and then increment	X=3; X++	3
	--	Decrement and then return value	X=3; --X	2
		Return value and then decrement	X=3; X--	3
Logical	&&	Logical "and" evaluates to true when both operands are true	3>2 && 5>3	False
	||	Logical "or" evaluates to true when either operand is true	3>1 || 2>5	True
	!	Logical "not" evaluates to true if the operand is false	3!=2	True
Comparison	==	Equal	5==9	False
	!=	Not equal	6!=4	True
	<	Less than	3<2	False
	<=	Less than or equal	5<=2	False
	>	Greater than	4>3	True
	>=	Greater than or equal	4>=4	True
String	+	Concatenation(join two strings together)	"A"+"BC"	ABC

4.11 Control Structure

Control statements are designed to allow you to create scripts that can decide which lines of code are evaluated, or how many times to evaluate them. There are two different types of control structures/statements:

1. *Conditional* structures/Statements
2. *Loop* structures/statements.

1. **Conditional Structure/Statements :** *Conditional statements* are used to make decisions. In real life, we make all sorts of decisions based on criteria such as "am I being offered enough money to take this job?" If the answer is "yes," then the result is "take the job." If the answer is "no," then "don't take the job." In JavaScript, you need to make decisions about which sections of code to evaluate.

For example, if you asked a user for input so you could perform a calculation, you will want to carry out the calculation if the input is numeric, but not if it isn't.

The Various Conditional Structures/Statements are :

- **if statement**
- **if else statement**
- **if else if statement**
- **switch statement**
- **if statement :**

The if statement is the most frequently used decision-making statement. It checks a condition and if it evaluates to true, then the statement(s) that it governs are evaluated, but if it evaluates to false then the statement(s) are passed over.

The syntax is as follows :

```
if (condition)
        JavaScript statement here
```

e.g. Write a program in JavaScript to check a person is Eligible for Vote.

Solution :

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

```
        var age = 21;
        if( age >18 ) {
          document.write("<b> Person is eligible  for  Vote</b>");
        }
    </script>
</script>
</body>
</html>
```

Output:

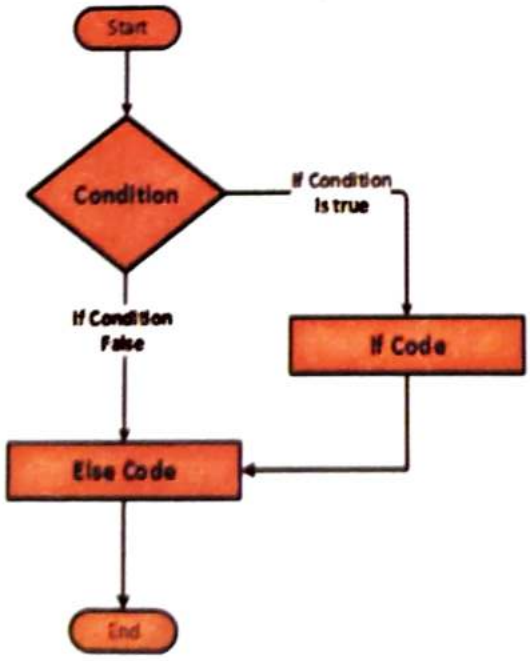
Person is eligible for Vote

- **if else statement:**

The **if/else statement** executes a block of code **if** a specified**condition** is true. **If** the **condition** is false, another block of code can be executed. The **if/else statement** is a part of JavaScript's "Conditional" **Statements**, which are used to perform different actions based on different conditions.

Syntax:

```
if (condition) {
  // block of code to be executed if the condition is true
} else {
  // block of code to be executed if the condition is false
}
```



e.g. Write a program in JavaScript to check whether a person is Eligible for Vote or Not.

Solution :

```
<html>
<body>
<script>
        var age = 17;
        if( age >18 ) {
          document.write("<b> Person is eligible for Vote</b>");
        }
else{
document.write("<b> Person is not eligible for Vote</b>");
}
    </script>
</script>
</body>
</html>
```

Output :

Person is not eligible for Vote

e.g. Write a program in JavaScript to check whether a person is adult or No

Solution :

```
<html>
<body>
<script>
        var age = 19;
        if( age >18 ) {
          document.write("<b> Person is adult</b>");
        }
```

```
else{
document.write("<b> Person is not adult</b>");
}
    </script>
</script>
</body>
</html>
```

Output:

Person is adult.

- **if else if statement:**

An **if** statement can be followed by an optional **else if...else** statement, which is very useful to test various conditions using single if...else if statement.

When using if...else if..else statements, there are few points to keep in mind –

- An if can have zero or one else's and it must come after any else if's.
- An if can have zero to many else if's and they must come before the else.
- Once an else if succeeds, none of the remaining else if's or else's will be tested.

Syntax:

```
if (condition1) {
 // block of code to be executed if condition1 is true
} else if (condition2) {
 // block of code to be executed if the condition1 is false and condition2 is true
} else {
 // block of code to be executed if the condition1 is false and condition2 is false
}
```

e.g. Write a program in JavaScript using with if else if statement to check the entered letter.

Solution :

```
<html>
<body>
<p>Enter letter: a, b, c, d or e?</p>
<input id="myInput" type="text">
<button onclick="myFunction()">Try it</button>
<p id="demo"></p>
<script>
function myFunction() {
  var letter = document.getElementById("myInput").value;
  var text;
  // If the letter is "c"
  if (letter === "c") {
    text = "Spot on! Good job!";
  // If the letter is "b" or "d"
  } else if (letter === "b" || letter === "d") {
    text = "Close, but not close enough.";
  // If the letter is anything else
  } else {
    text = "Your letter is <b>"  +letter +"";
  }
  document.getElementById("demo").innerHTML = text;
}
</script>
</body>
</html>
```

- **Switch Statement :**

The switch statement allows you to choose one of several options The basic syntax of the switch statement is to give an expression to evaluate and several different

statements to execute based on the value of the expression. The interpreter checks each case against the value of the expression until a match is found. If nothing matches, a default condition will be used.

Syntax :

```
switch (expression) {
  case condition 1: statement(s)
                    break;
  case condition 2: statement(s)
                    break;
  ...
  case condition n: statement(s)
                    break;
  default: statement(s)
}
```

e.g. Write a Program in JavaScript using with Switch Statement to check the result according to grade system.

Solution :

```
<html>
<body>
<script>
    var grade='B';
    document.write("<b>Result according to the grade<br/>");
    switch (grade) {
      case 'A': document.write("Excelent<br/>");
        break;
      case 'B': document.write("Very Good<br/>");
        break;
      case 'C': document.write("Good<br/>");
```

```
        break;
      case 'D': document.write("Satisfactory<br/>");
        break;
      case 'F': document.write("Failed<br/>");
        break;
      default:  document.write("Please Enter Grade<br/>")
    }
</script>
</script>
</body>
</html>
```

2. ***Loop*** **or iteration structures/statements :** An **loop** statement, or **iteration**, repeatedly executes a statement, known as the **loop** body, until the **controlling** expression is false (0). The **control** expression must have a scalar type. The while statement evaluates the **control** expression before executing the **loop.**

Looping in programming languages is a feature which facilitates the execution of a set of instructions/functions repeatedly while some condition evaluates to true. For example, suppose we want to print "Hello World" 10 times.

Without Using Loops

```
<script type = "text/javascript">
            document.write("Hello World\n");
            document.write("Hello World\n");
            document.write("Hello World\n");
            document.write("Hello World\n");
            document.write("Hello World\n");
            document.write("Hello World\n");
            document.write("Hello World\n");
            document.write("Hello World\n");
            document.write("Hello World\n");
            document.write("Hello World\n");
        < /script>
```

Using Loops

```
<script type = "text/javascript">
    var i;
    for (i = 0; i < 10; i++)
    {
        document.write("Hello World!\n");
    }
< /script>
```

There are different Types of looping statements :

- For Loop
- While Loop
- Do While Loop
- **For Loop :**

The for loop is used when you know in advance how many times the script should run.

Syntax:

```
for (var=startvalue;var<=endvalue;var=var+increment)
    {
    code to be executed
    }
```

e.g. Write a program in JavaScript to print the first 20 numbers using for loop.

Solution :

```
<html>
<body>
<script type="text/javascript">
var i=0
for (i=0;i<=20;i++)
{
```

```
            document.write("The number is " + i)
            document.write("<br />")
        }
        </script>
        </body>
        </html>
```

Output :

```
The number is 0
The number is 1
The number is 2
The number is 3
The number is 4
The number is 5
The number is 6
The number is 7
The number is 8
The number is 9
The number is 10
The number is 11
The number is 12
The number is 13
The number is 14
The number is 15
The number is 16
The number is 17
The number is 18
The number is 19
The number is 20
```

e.g. Write a program in JavaScript to print your name 5 times using for loop.

Solution :

```
<html>
<body>
<script type="text/javascript">
var i=0
for (i=0;i<=5;i++)
{
        document.write("My Name is Gian Sagar")
        document.write("<br />")
}
</script>
</body>
</html>
```

Output :

My Name is Gian Sagar
My Name is Gian Sagar
My Name is Gian Sagar
My Name is Gian Sagar
My Name is Gian Sagar
My Name is Gian Sagar

- **While Loop :**

The while loop is used when you want the loop to execute and continue executing while the specified condition is true.

Syntax:

```
while (var<=end value)
{
   code to be executed
}
```

Note : The <= could be any comparing statement.

e.g. Write a program in JavaScript to print the first 15 odd numbers using while loop.

Solution :

```
<html>
<body>
<script type="text/javascript">
var i=1;
document.write("The first 15 odd numbers are:<br />")
while (i<=15)
        {
        document.write(i)
        document.write("<br />")
      i=i+2;
        }
</script>
</body>
</html>
```

Output :

The first 15 odd numbers are :

1

3

5

7

9

11

13

15

e.g. Write a program in JavaScript to print the first 20 even numbers using while loop.

Solution :

```
<html>
<body>
<script type="text/javascript">
var i=0;
document.write("The first 20 even numbers are:<br />")
while (i<=20)
        {
        document.write(i)
        document.write("<br />")
      i=i+2;
        }
</script>
</body>
</html>
```

Output :

The first 20 even numbers are:
0
2
4
6
8
10
12
14
16
18
20

- **Do While Loop**

The do...while loop is a variant of the while loop. This loop will always execute a block of code ONCE, and then it will repeat the loop as long as the specified condition is true. This loop will always be executed at least once, even if the condition is false, because the code is executed before the condition is tested.

Syntax :

```
do
{
   code to be executed
}
while (var<=endvalue)
```

e.g. Write a program in JavaScript to print the sum of first 10 numbers using do-while loop.

Solution :

```
<html>
<body>
<script type="text/javascript">
var i=0;
var sum=0;
do
{
        sum=sum+i;
        i=i+1;
}
        while (i<=10)
document.write("The sum of first 10 numbers is:<b> "  +sum)
</script>
</body>
</html>
```

Output :

The sum of first 10 numbers is: **55**

e.g. Write a program in JavaScript to print the sum of first 10 even numbers using do-while loop.

Solution :

```
<html>
<body>
<script type="text/javascript">
var i=0;
var sum=0;
do
{
sum=sum+i;
i=i+2;
}
while (i<=10)
document.write("The sum of first 10 even numbers is:<b> "  +sum)
</script>
</body>
</html>
```

Output :

The sum of first 10 even numbers is: **30**

4.12 Developing Interactive form and Validation

Forms are used in web pages for the user to enter their required details that are further send it to the server for processing. A form is also known as web form or HTML form.

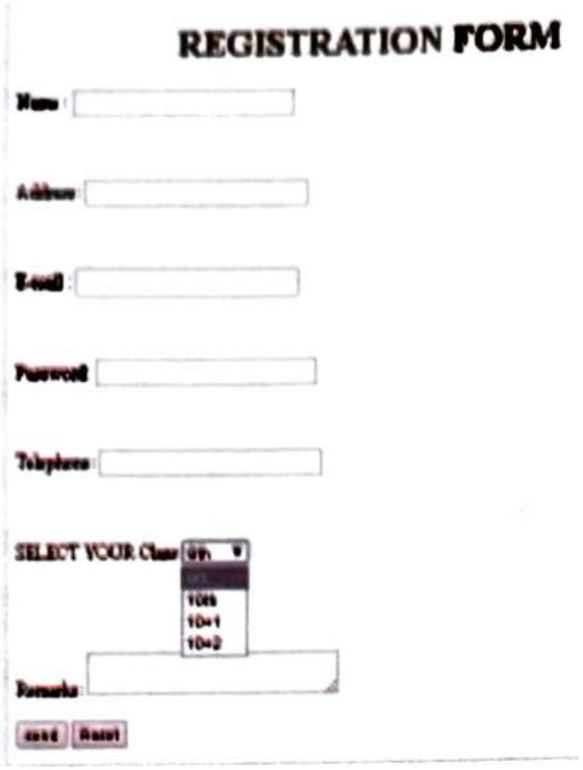
JavaScript provides facility to **validate** the **form** on the client-side so data processing will be faster than server-side **validation**. Most of the web developers prefer **JavaScript form validation**. Through **JavaScript**, we can **validate** name, password, email, date, mobile numbers and more fields.

Creating HTML Form with JavaScript validations :

HTML Form :

```
<html>
<body>
        <h1 style="text-align: center"> REGISTRATION FORM </h1>
        <form     name="RegForm"     action="/submit.php"     onsubmit="return
HPBOSE()" method="post">
            <p>Name    : <input type="text" size=25 name="Name"> </p><br>
           <p> Address: <input type="text" size=25 name="Address">  </p><br>
           <p>E-mail : <input type="text" size=25 name="EMail">  </p><br>
           <p>Password: <input type="text" size=25 name="Password"> </p><br>
          <p>Telephone: <input type="text" size=25 name="Telephone"> </p><br>
           <p>SELECT YOUR Class
             <select type="text" value="" name="Subject">
                <option>9th</option>
                <option>10th</option>
                <option>10+1</option>
                 <option>10+2</option>
        </select></p><br><br>
           <p>Remarks:  <textarea cols="25" name="Remarks">  </textarea></p>
           <p><input type="submit" value="send" name="Submit">
           <input type="reset" value="Reset" name="Reset">
    </p>
</form>
</body>
</html>
```

Output :



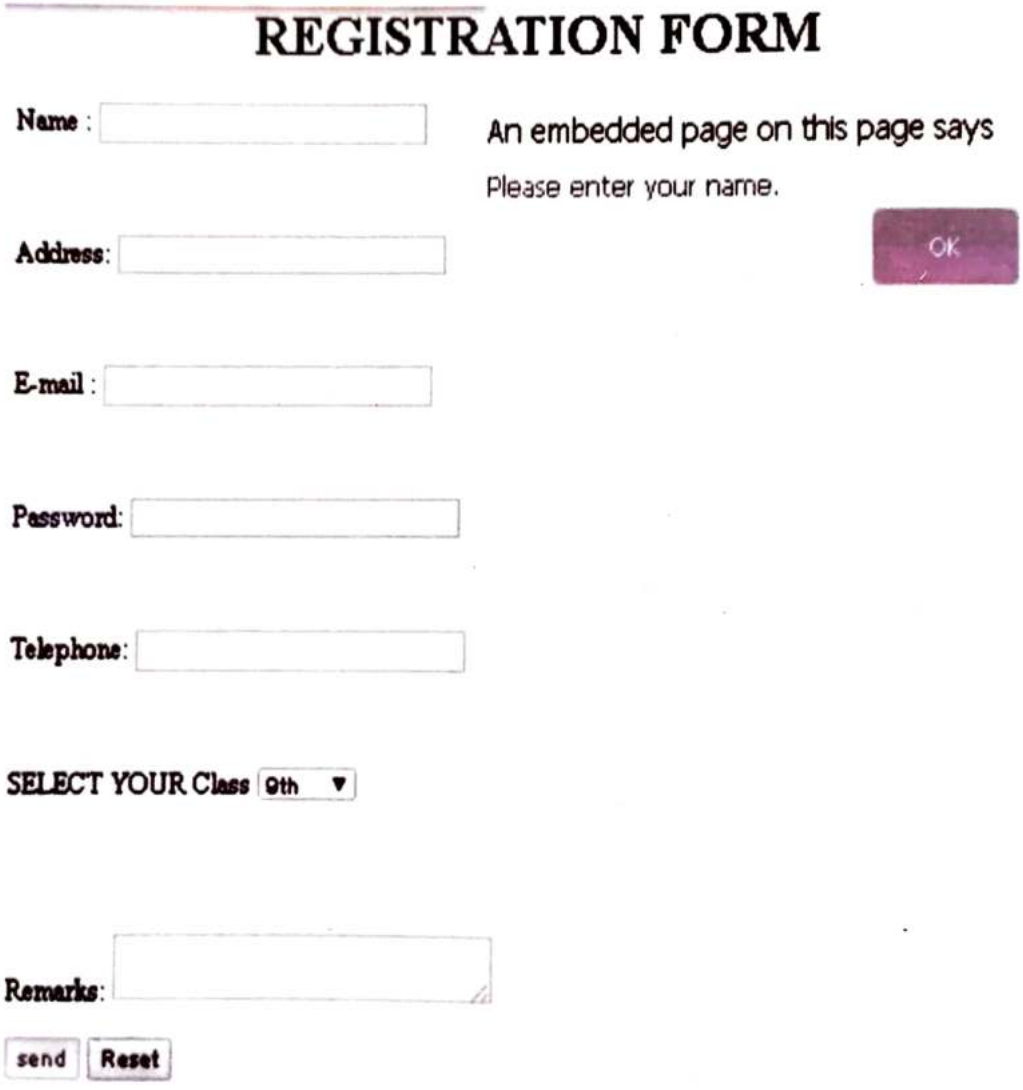
JavaScript Validation for above form :

```
<html>
<head>
<script>
function HPBOSE()
{
  var name = document.forms["RegForm"]["Name"];
  var email = document.forms["RegForm"]["EMail"];
  var phone = document.forms["RegForm"]["Telephone"];
  var what =  document.forms["RegForm"]["Subject"];
  var password = document.forms["RegForm"]["Password"];
  var address = document.forms["RegForm"]["Address"];
  if (name.value == "")
  {
    window.alert("Please enter your name.");
    name.focus();
    return false;
  }
```

```
if (address.value == "")
{
  window.alert("Please enter your address.");
  name.focus();
  return false;
}
if (email.value == "")
{
  window.alert("Please enter a valid e-mail address.");
  email.focus();
  return false;
}
if (email.value.indexOf("@", 0) < 0)
{
  window.alert("Please enter a valid e-mail address.");
  email.focus();
  return false;
}
if (email.value.indexOf(".", 0) < 0)
{
  window.alert("Please enter a valid e-mail address.");
  email.focus();
  return false;
}
if (phone.value == "")
{
  window.alert("Please enter your telephone number.");
  phone.focus();
  return false;
}
if (password.value == "")
{
  window.alert("Please enter your password");
  password.focus();
```

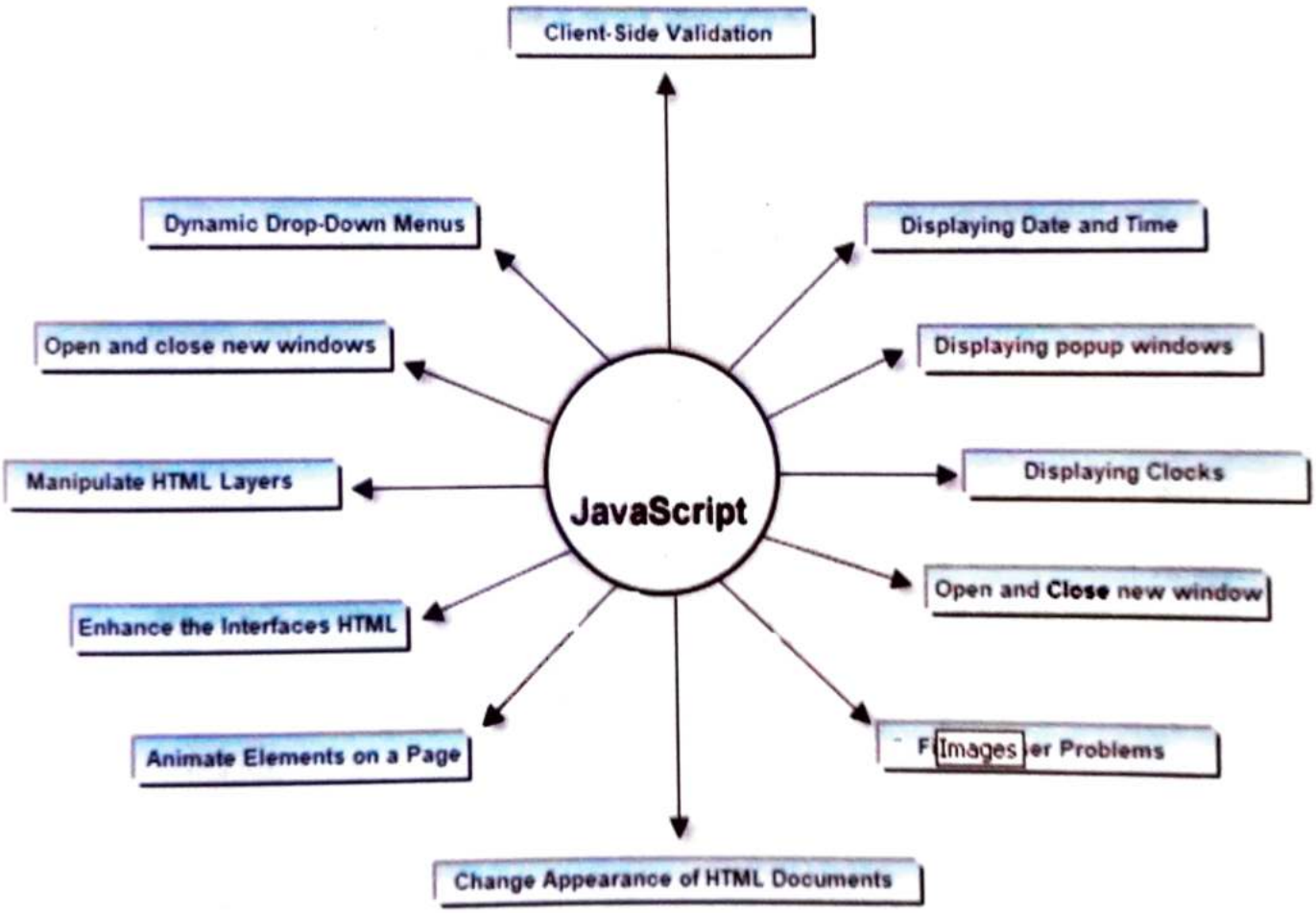
```
    return false;
  }

  if (what.selectedIndex < 1)
  {
    alert("Please enter your course.");
    what.focus();
    return false;
  }
  return true;
      }</script>
```

output : After click on send button, JavaScript validation of look like this



4.13 JavaScript Applications

- Client-side validation
- Dynamic drop-down menus
- Validate user input in an HTML form before sending the data to a server.
- Build forms that respond to user input without accessing a server.
- Change the appearance of HTML documents and dynamically write HTML into separate Windows.
- Open and close new windows or frames.
- Manipulate HTML "layers" including hiding, moving, and allowing the user to drag them around a browser window.
- Build small but complete client side programs .
- Displaying popup windows and dialog boxes (like alert dialog box, confirm dialog box and prompt dialog box)



4.14 Cookies :

A cookie is an amount of information that persists between a server-side and a client-side. A web browser stores this information at the time of browsing. A cookie contains the information as a string generally in the form of a name-value pair separated by semi-colons. It maintains the state of a user and remembers the user's information among all the web pages.

Cookies are data, stored in small text files, on your computer. When a web server has sent a web page to a browser, the connection is shut down, and the server forgets everything about the user. Cookies were invented to solve the problem "how to remember information about the user":

- When a user visits a web page, his/her name can be stored in a cookie.
- Next time the user visits the page, the cookie "remembers" his/her name.

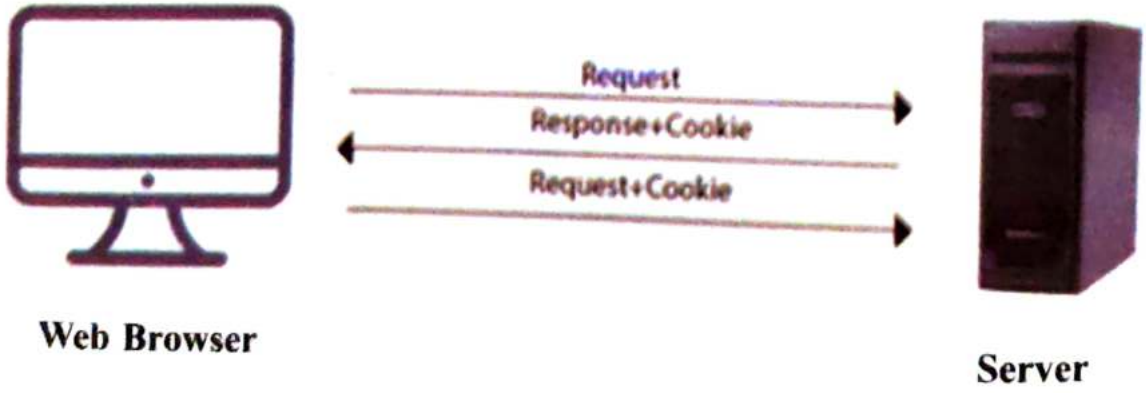
Cookies are saved in name-value pairs like:

username = ABC

When a browser requests a web page from a server, cookies belonging to the page are added to the request. This way the server gets the necessary data to "remember" information about users.

How Cookies Works ?

- When a user sends a request to the server, then each of that request is treated as a new request sent by the different user.
- So, to recognize the old user, we need to add the cookie with the response from the server.
- Browser at the client-side.
- Now, whenever a user sends a request to the server, the cookie is added with that request automatically. Due to the cookie, the server recognizes the users.



Create a Cookie with JavaScript

JavaScript can create, read, and delete cookies with the **document.cookie** property.

Syntax:

document.cookie = "username= ABC";

Read a Cookie with JavaScript

The statement to read cookies is as follow:

var x = document.cookie;

e.g.

```
<html>
  <head>
    <script type = "text/javascript">
      <!—
        function WriteCookie() {
          if( document.myform.customer.value == "" ) {
            alert("Enter some value!");
            return;
          }
          cookievalue = escape(document.myform.customer.value) + ";";
          document.cookie = "name=" + cookievalue;
          document.write ("Setting Cookies : " + "name=" + cookievalue );
        }
      //—>
    </script>
  </head>
  <body>
    <form name = "myform" action = "">
      Enter name: <input type = "text" name = "customer"/>
      <input type = "button" value = "Set Cookie" onclick = "WriteCookie();"/>
    </form>
  </body>
</html>
```

Output

4.15 JavaScript Security

Because of the wide-open nature of the Internet, security is an important issue. JavaScript is not much secure and designed as an open scripting language. JavaScript's first line of defense against malicious code is that the language simply does not support certain capabilities. For example, client-side JavaScript does not provide any way to write or delete files or directories on the client computer. With no File object and no file access functions, a JavaScript program cannot delete a user's data or plant viruses on the user's system. Similarly, client-side JavaScript has no networking primitives of any type.

We cannot treat JavaScript as a full-fledged programming language. It lacks the following important features :

- Client-side JavaScript does not allow the reading or writing of files. This has been kept for security reason.
- JavaScript cannot be used for networking applications because there is no such support available.
- JavaScript doesn't have any multithreading or multiprocessor capabilities.

Once again, JavaScript is a lightweight, interpreted programming language that allows you to build interactivity into otherwise static HTML pages.

SUMMARY

- **Computer Network** is an interconnection of geographically distributed multiple computer.
- **LAN** Local Area Network spread within the room or building.
- **MAN** Metropolitan Area Network spread within the one city to another.
- **WAN** Wide Area Network spread all over the world.
- **PAN** Personal Area Network spread in small area with limited distance.

IP Address is a unique number identify the electronic machine or device which connected to a network.

- **Protocol** is a set of rules or instructions.

- **FTP** stands for File transfer Protocol used for file transfer from one machine to another.
- **HTTP** Hyper text transfer protocol used for transmitting and displaying information.
- **SMTP** Simple mail transfer protocol used for email.
- **PPP** point to point protocol is widely used for the heavier and faster connection.
- **Client** is a machine or device that accesses a service made available by a server.
- **Server** is a powerful machine or device that provide the service to its client.
- **Web Language** used for web programming to develop the web pages.
- **JavaScript** is a client side scripting language that allows creating of dynamic and interactive web pages.
- **variable** holds the value in the program and can change anytime.
- **Operators** are used to perform specific mathematically and logical computations on operands.
- **Loop** is a statement or iteration that repeat and executes a statement.
- **Cookies** are data stored in small text files on your computer.

TRUE/FALSE

1.	Internet is a network of networks.	**True**
2.	PAN stands for public area network.	**False**
3.	An IP address consists of 32 bit information.	**True**
4.	SMTP stands for Simple mail transfer protocol.	**True**
5.	JavaScript is server side scripting language.	**False**
6.	A variable must have unique name.	**True**
7.	HTTP Hyper text transport protocol	**False**
8.	Cookies are data stored in small text files on your computer.	**True**

EXERCISE

Q1. What do you mean by Computer Network ? Explain.

Q2. What is difference between LAN and WAN ?

Q3. What do you mean by TCP/IP Suit ? Explain.

Q4. Explain Application Layer and Transport Layer ?

Q5. **Explain the following terms :**

a.) HTTP b.) FTP c.) SMTP d.) TELNET e.) PPP

Q6. Differentiate between Client Side and Server Side Scripting.

Q7. What is JavaScript ? Write the advantages and disadvantages of JavaScript.

Q8. What is variable ? Write a JavaScript program to illustrate the variable.

Q9. What do you mean by Data Types in JavaScript? Explain.

Q10. What is conditional statement? Explain.

Q11. What is difference between if else and switch statement.

Q12. Explain for loop in JavaScript with an example.

Q13. What is difference between While loop and Do While loop.

Q14. What do you mean by JavaScript Validations? Explain.

Q15. How can you create HTML form using JavaScript validation

Q16. What do you mean by cookies? Explain.

Programs :

1. Write a JavaScript program to check whether a person is eligible for vote or not.
2. Write a JavaScript program to check whether a person is adult or not.
3. Write a program in JavaScript to print the first 20 numbers using **for loop.**
4. Write a program in JavaScript to print the sum of first 10 numbers using **Do while Loop.**
5. Write a program in JavaScript to print the sum of first 10 even numbers using **Do while Loop.**

—End—