year 20...../20.....

No of partials Certified ..... Out of .....

In The Subject of .....N.P.....Lab.....

.....

Teacher In - Charge

.....

Examiner's Signature

.....

Principal

Date : .....

Institution Rubber Stamp

(N.B: The candidate is expected to His/her journal till He/she Passes in the Subject.)

Index

Sr. No.	Name Of the Experiment	Page No	Date of Experiment	Date of Submission	Remarks
01.	To make UTP cable with RJ 45 Connector & build a simple dat network using Crossover & straight cable.			12/3	
02.	To share data from one system to another using folder & drives share.				
03.	To understand Network topology & types of topology.				
04	Study of different type of LAN's & Network Equipments.				
05	To write a java program to calculate the time taken by the program to be executed.	11	8-3-22		
06.	To implement various types of error correction & detection technique.	12	15-3-22		

Sr. No.	Name Of the Experiment	Page No	Date of Experiment	Date of Submission	Remarks
07.	WAP to find XOR operation on two numbers using C/C++/JAVA.	17.	22-3-22		
08	Introduction of Socket programming.	18.	28-3-22		
09	WAP. in C : Hello-Client for TCP.	25	5-6-22		
10.	WAP. in C : Hello-Server for TCP.	27	10-6-22		

Expt. No. 1 Date

Page No. 3

Experiment No - 1

Objective :- To make UTP Cable with RJ 45 Connection & build a simple test networking using crossover cable & straight cable with color code.

Theory :-

UTP Cable stand for Unshielded Twisted Pair cable.

This cable is mostly used for LAN Net network. These cables can be used for voice, low speed data, high speed data, audio & paging system & building automation & control system.

Requirement for making UTP Cable :-

Cable stripper, Wire cutter, RJ 45 connectors, RJ 45 Crimping tool.

Procedure :-

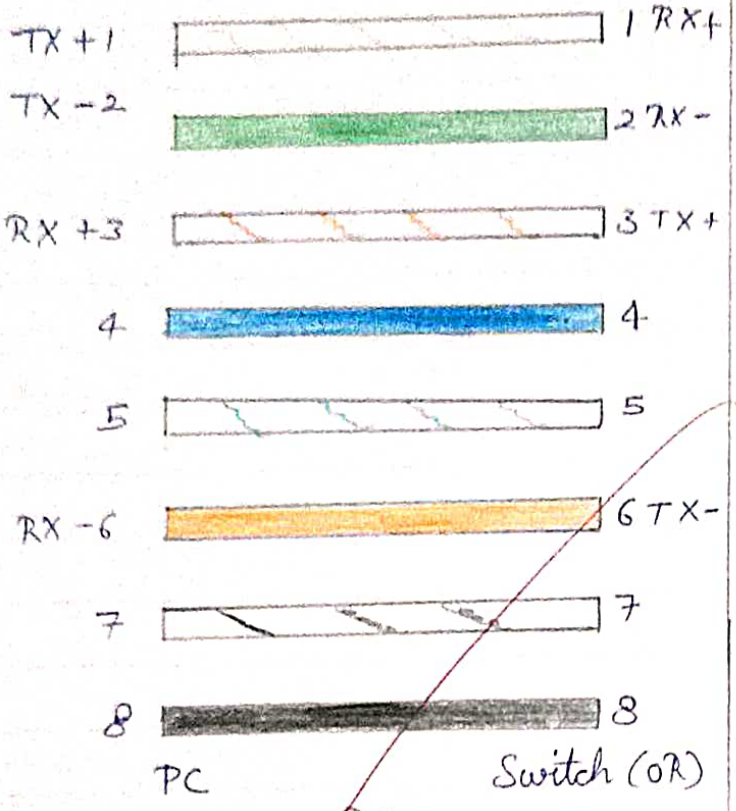
- Use the cable stripper at about 1-2 inches from the end of the code cable to remove the outer jacket (fig 2).
- Untwist the twist pair wires all the way back to the jacket (fig 3).
- Align the untwisted wires all the way back to the jacket order necessary for your need. fig (4)
- Cut the extra wire using wire cutter (fig 5)

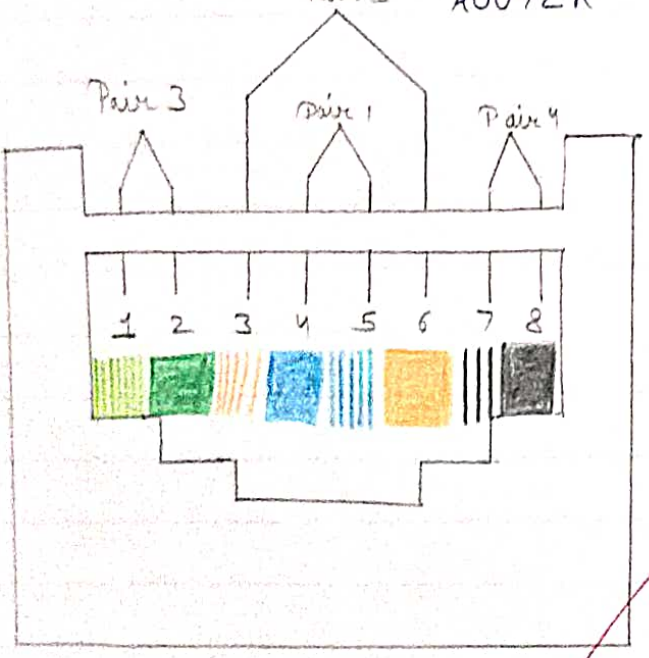
Teacher's Signature

STRAIGHT - THROUGH

Wiring Standards Used

T568A T568A



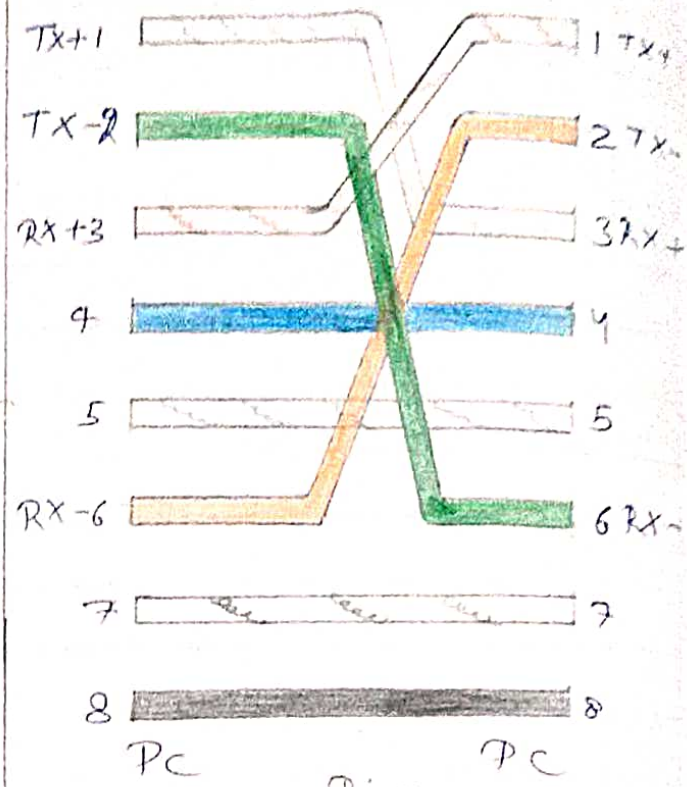


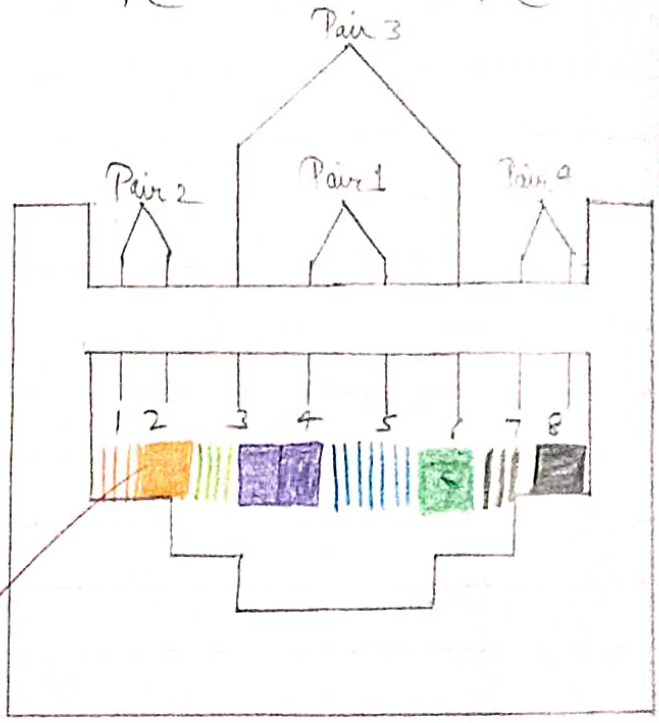
T568 A

CROSS - OVER

Wiring standards Used

T568A T568B





T568 B

- Push the remaining wires into the RJ45 connector head (fig 5).
- Double check that the wire are all the way up into the gold pins of the head & made it up in the proper order.
- Push the head into the open space of the crimping tool & sequence it closed, hard (fig 6)
- Open the crimping tool & remove your newly-crimped ethernet connection.
- Now repeat the crimping process on the otherside of the cable if you are making a completely new cable.
- Now, once the network cable is ready, plug the cable into any two computers, routers, or devices to make a simple test network.

Result :- We made a Network cable using RJ45 connector from UTP cable & using this cable to make simple test network.

Date

Expt. No. 2 Page No. 5

Experiment No - 2

Objective :- To share Data from one system to another using folder share & drive share.

Theory :-

With the help of network cable, connecting two system at a time makes it possible to share data from a folder on drive from system to another.

Procedure :-

1) Sharing file / data from a system :-

- Select the folder which we wants to share & open its properties (fig 1)
- In the properties window, go to the sharing option (fig 2)
- Click on the share button (fig 2)
- select the user with when we wantil to share our data & add it set permission for the user/ system (fig (3)).
- Click on share button and then the data button (fig 4)

The folder is successfully shared.

2) Accessing the file / data from another System :-

Teacher's Signature

Date

Expt. No. _____ Page No. 6

- First find out the I.P. address of first system (fig 5)
- On the second system open "Run" menu by using "Window + R" Key short cut (fig 6).
- Enter the I.P. address of the first system. This will open the network folder (fig 7)
- The network folder also show the number of system/ devices connected to the network.
- The shared folder from the first system will also be available there for use (fig 9).

Result :- Successfully shared the data which are stored in folder / drive completely.

Teacher's Signature _____

Date

Expt. No. 3 Page No. 7

Experiment No - 3

Objective :- To understand network topology and types of network topology.

Theory :-

Topology :- The word topology is derived from two greek words "Topo" and "logy", where topo means "place" and "logy" means "study". In computer network, a topology is used to explain how a network is physically connected & the logical flow of the information in the network.

A topology (flow of the information) mainly describes how devices are connected & interact with each other using communicator links.

⇒ In computer network these are mainly two types of topology :-

1. **Physical topology :-** It describe the way in which the computer or node are connected with each other.
2. **Logical topology :-** It describe the way data flow from one computer to another.

Teacher's Signature

Date

Expt. No. _____ Page No. 8

⇒ In computer network there are mainly six types of topology :-

1. Bus topology :→ Simplest kind of topology in which a common bus or channel is used to communication in the network. The buses connected to various system or devices or droplines.

2. Ring topology :- Topology in which each computer is connected to exactly two other computers to form the rings.

3. Star topology :- Computer network topology in which all the nodes are connected to a centralized hub.

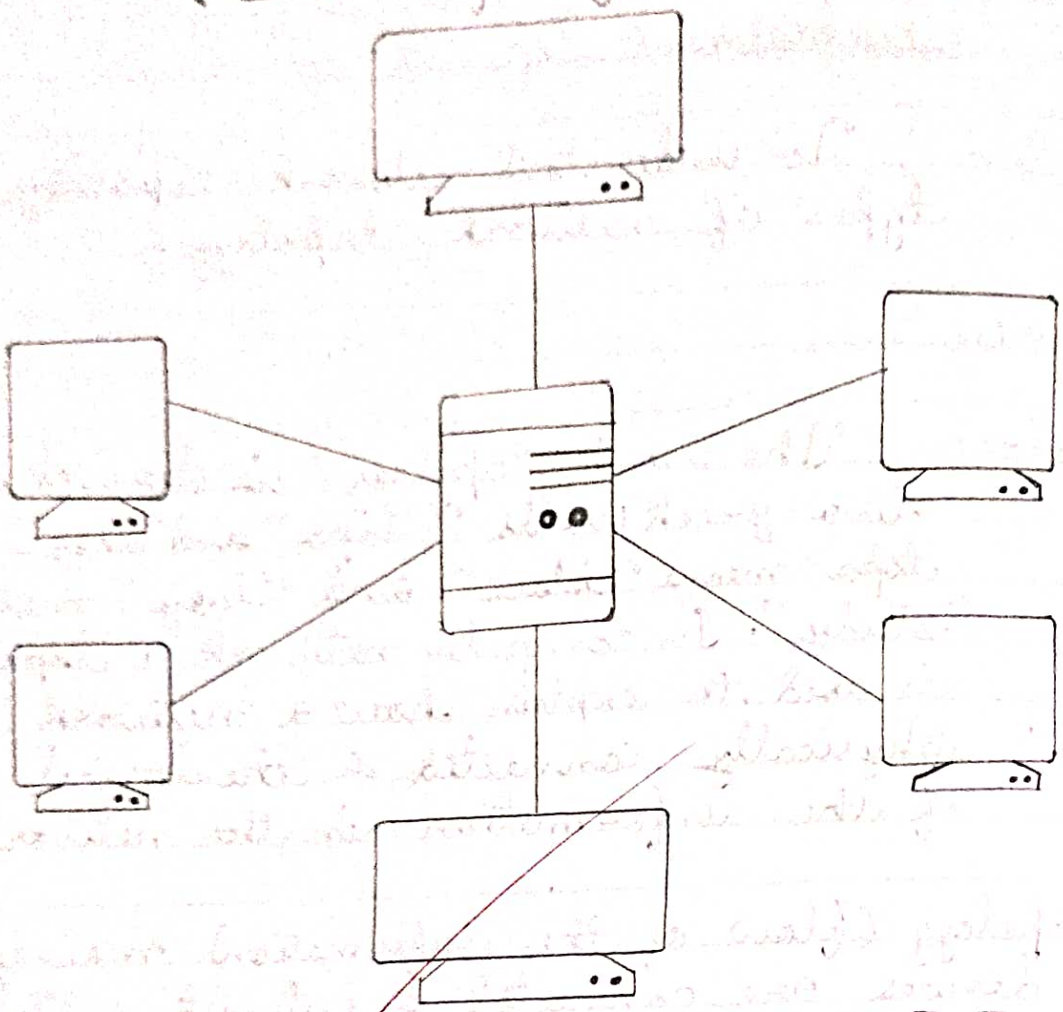
4. Mesh topology :- Computer network topology in which nodes are interconnected with each other. Mesh topology is also known as fully connected topology.

5. Tree topology :- Computer network topology in which all the nodes are directly or indirectly connected to the main bus cable. Tree topology is combination of Bus & star topology.

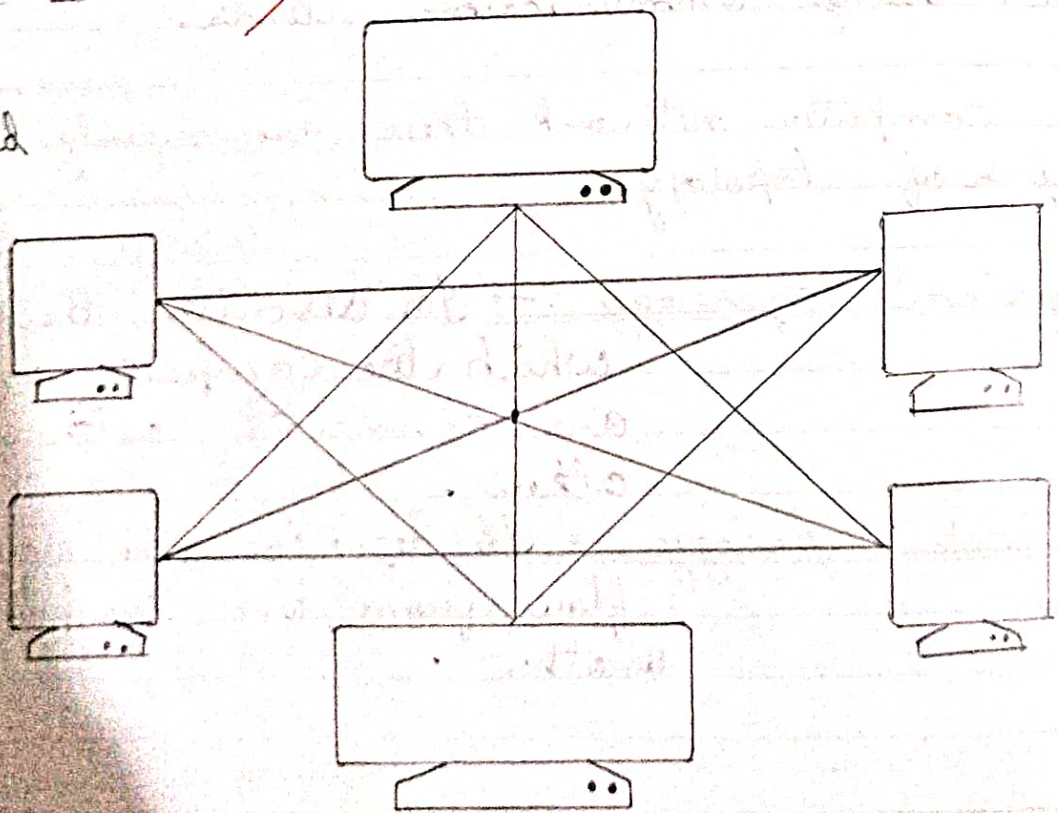
Result : We have study computer network topology & its types.

Teacher's Signature _____

< Star Topology >



Fully Connected



Expt. No. 4 Date

Page No. 3

Experiment No - 4

Objective : Study of different type of LAN's & Network Equipment.

Theory :- A computer network is a group of computer linked to each other that enable the computer to communicate with another computers and share their resources, data & application.

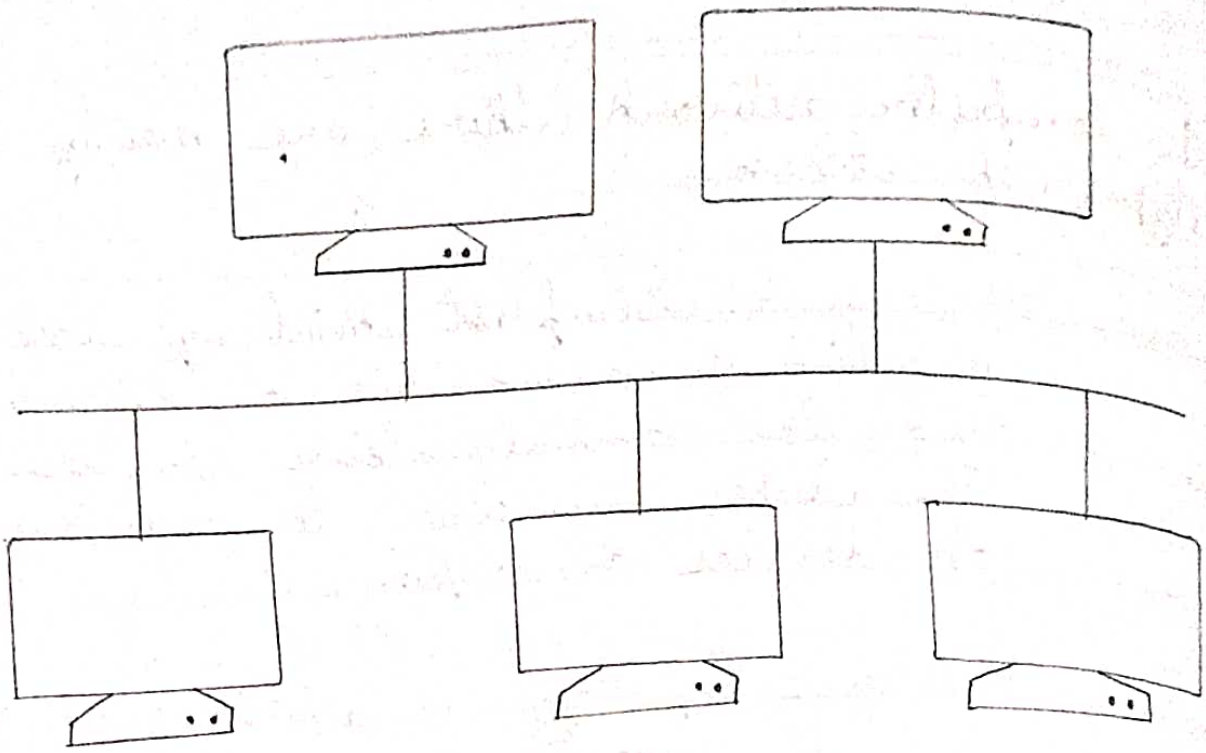
A computer network can be categorised by their size. A computer network is mainly of three types.

(i) LAN (Local Area Network)
(ii) MAN (Metropolitan Area Network)
(iii) WAN (Wide Area Network)

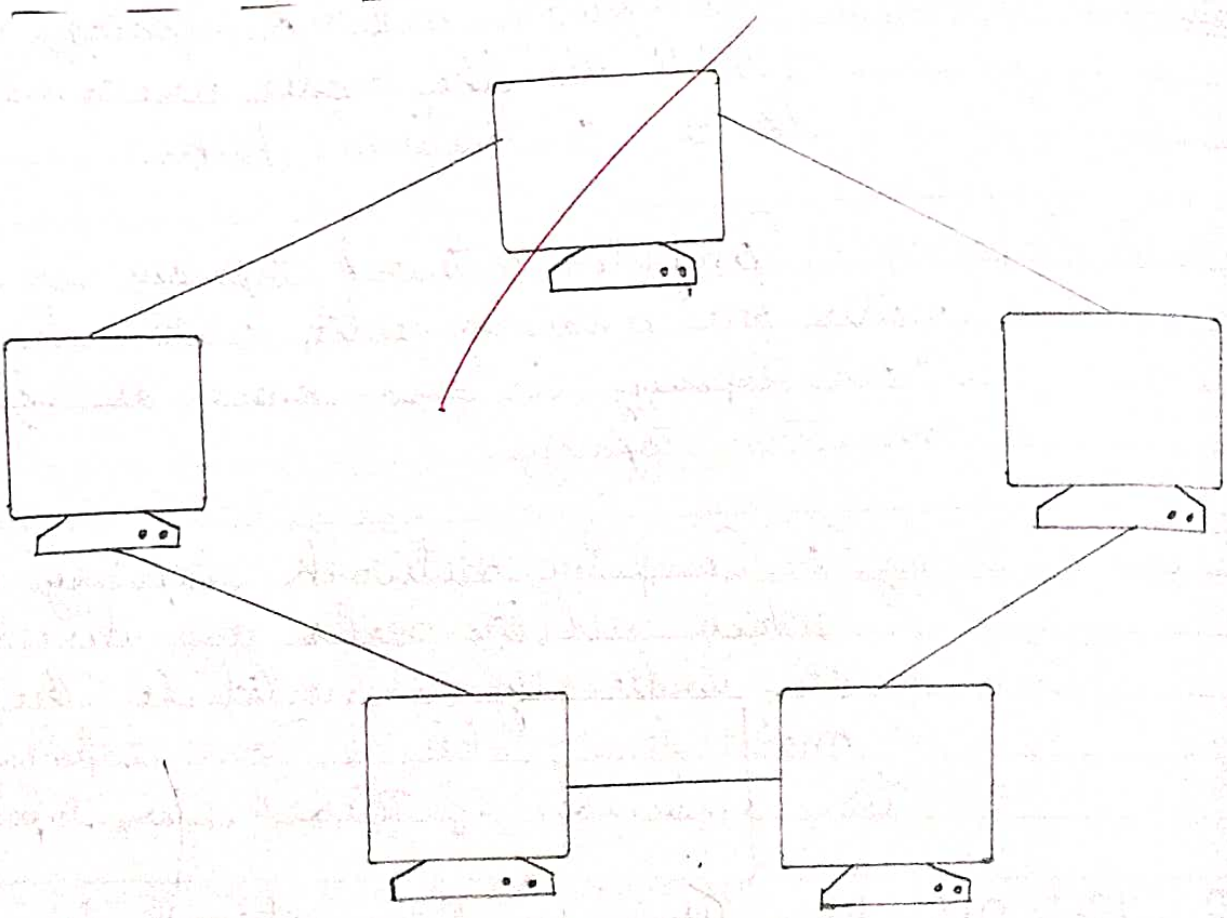
1) **LAN ⇒** Local area network is a group of computers connected to each other in a small area such as building office.

- It is used for connecting 2 or more pc through a communication medium such as twisted pair, coaxial etc.
- It is less costly as it is build with inexpensive hardware such as hubs, network adapter & ethernet cable.
- The data is transfered at an extremely faster rate in LAN.

Teacher's Signature



< BUS Topology >



< RING Topology >

- LAN provide higher security.

2. MAN :- A metropolitan area network is a network that cover a larger geographic area by interconnecting a different LAN to form a larger network.

- In MAN, various LAN are connected to each other through a telephone exchange line.
- The most widely used protocol in MAN are RJ-232 frame relay, ATM, ISDN, OC-3, ADSL etc.
- It has higher range then Local Area Network (LAN).

3. WAN :- A wide area network is a network that extend over a large geographical area such as states or countries.

- It spans over a large geographical area through telephone line, fibre, optical cable or satellitic.
- Internet is one of the biggest WAN in the World.
- A WAN is widely used in field of Business, government and education.

Result :- In this experiment, we have studied computer network & its types.

12/03/22

Teacher's Signature

Expt. No. 5 Date

Page No. 11

Experiment No - 5

Objective :- To write a java program to calculate the time taken by the program to be executed.

Program :-

```
import java.lang.*;
class XYZ
{
  public static void main (String arg[])
  {
    int sum = 0;
    long initial_time = System.CurrentTimeMillis();
    for (int i=0; i<1000000; i++)
    {
      sum = sum + i;
    }
    long last_time = System.CurrentTimeMillis();
    long total Time taken = last time - initial time;
    System.out.println("Total time taken:" + total
                       time taken + "Millisecond");
  }
}
```

Teacher's Signature

Output

Total time taken : 14342 millisecond.

Date

Expt. No. 6 Page No. 12

Experiment No-6

Hamming Code

Objective :-

Write a program to implement various type of error correction & detection technique.

Hamming Code :-

1. Hamming code can be applied to data unit of any length.
2. It is used to detect & correct Signal bit error.
3. All bit position that are power of 2 (2^n) are marked as paritory bits (1, 2, 4, 8, 16, ...) & the other bit are Data bit.

7 bit Structure

D_7	D_6	D_5	D_4	D_3	D_2	D_1

Determine the value of paritory bit :-

- The value of paritory bit is determine by the sequence of bit that is alternativilly check & skip.

For P_1 Paritory

1 bit check, 1 bit skip, 1 bit check, 1 bit skip, - - - - - -

1, 3, 5, 7, - - - - - - - - -

For P_2 Paritory

2 bit check, 2 bit skip, 2 bit Check, - - -

2, 3, 6, 7, - - - - Teacher's Signature

For P_4 Parity

4, 5, 6, 7, 12, 13, 14, 15, ...

Condition :-

- if (even no. of 1's exist in sequence) $P=0$
- if (odd no. of 1's exist in sequence) $P=1$

Error detection process :-

A error is located by forming a 3-bit number out of 3-parity checks.

Condition :-

- if (odd no. of 1's error exist) $P=1$

if (even no. of 1's then no. error exist) $P=0$

After we have found the error word are find if decimal value then we invert the incorrect bit to obtain the correct word.

A 7 bits hamming code is received as 1011011 Assume even parity and state whether the received code is correct or wrong.

D_7	D_6	D_5	D_4	D_3	P_2	P_1
1	0	1	1			

Teacher's Signature

P_1	D_3	D_5	D_7	
1	0	1	1	$= P_1$ = error

P_2	D_3	D_5	D_6	
1	0	0	1	$= P_2$ = No error

P_4	D_5	D_6	D_7	
1	1	0	1	$= P_4$ = error

P_4	P_2	P_1	$\Rightarrow$	D_7	D_6	D_5	P_4	D_3	P_2	P_1
1	0	1		1	0	0	1	0	1	1

Code :- Hamming code

```
#include <stdio.h>
void main()
{
int data[10];
int datatrue[10], c1, c2, c3, i;
Printf("enter 4 bits of data one by one\n");
Scanf("%d", &data[0]);
Scanf("%d", &data[1]);
Scanf("%d", &data[2]);
Scanf("%d", &data[3]);
data[6] = data[0]^data[2]^data[4];
data[5] = data[0]^data[1]^data[4];
data[3] = data[0]^data[1]^data[2];
Printf("\encoded data is\n");
for(i=0; i<7; i++)
{
printf("%d", data[i]);
}
```

Teacher's Signature

```
printf("Enter received data bit one by one\n");
    for(i=0; i<7; i++)
{
scanf("%d", &data tree[i]);
}
    c1 = data tree[6]^ data tree[4]^ data tree[2]^
                                        data tree[0];
c2 = data tree[5]^ data tree[4]^ data tree[1] data tree
                                        [0];
c3 = data tree[3]^ data tree[2] data tree[1] data tree
                                        [0];
c = c3*4+c2*2+c1;
    if (c==0)
{
Printf("No error while transmission of Data\n");
}
else
{
Printf("error on position %d", c);
}
Printf("\n Data sub");
    For (i=0; i<7, i++);
{
  Printf("%d", data tree[i]);
}
  Printf("\n correct msg");
  If (data tree (7-c) == 0)
{
  data tree (7-c) = 1;
}
```

```
else
{
data tree [7-c] = 0;
}
for (i=0; i<7; i++)
{
Printf ("%d", data tree [i]);
}
}
}
```

Date

Expt. No. 07

Page No. 17

Experiment No - 7

Aim: WAP to find XOR operation on two numbers using C/C++/Java.

```
#include <iostream>
using namespace std;
int myXOR (int x, int y)
{
    int res = 0;
    for (int i = 3; i >= 0; i--)
{
    bool b1 = x & (1 << i);
    bool b2 = y & (1 << i);

bool XORed bit = (b1 & b2) ? 0 : (b1 | b2);
    res << = 1
    res | = XORed bit;
{ return res;
    { int main()
        { int x = 3, y = 5;
          cout << "XOR is " << myXOR (x, y);
          return 0;
        }
```

Teacher's Signature

Output

xOR is 6

Date

Expt. No. 08 Page No. 12

Experiments No - 8

Aim :- Introduction of Socket Programming.

Theory :-

Socket :- it allows communication b/w the different process on the same or different machines.

Where is Socket used ?

its was first introduction in 2:1 BSD and Subsequently into their current form with 4.2 BSD.
A unix socket is used in client server application framework.

Socket types :- Two types of Sockets.

a) Stream Sockets :- delivery in a network environment is garanted, if you opened through the stream socket three items "A, B, C, These sockets are TCP (for data transmission).

b) Datagram Sockets :- Delivery in networking environment is not guaranted.

Unix Socket client server model :
most net application use the client-server architecture : one of two process acts as a clients process and another process acts as a server.

Teacher's Signature

Client Process :- This is the process, which typically makes a request for information. After getting the response, this may terminate and may do some other processing.

Server Process :- This is a process which takes a request from the clients, after getting a request from the clients, this process will perform the required processing, gather the requested information and send it the client. Once done, it becomes ready to serve another clients.

Note :- Client needs to know the address of server, but server does not need.

Architecture :- Two types of clients server architecture.

a) 2-tier :- The type of architecture have some security holes & performance.
The clients directly interacts with Server.

b) 3-tier architecture :-

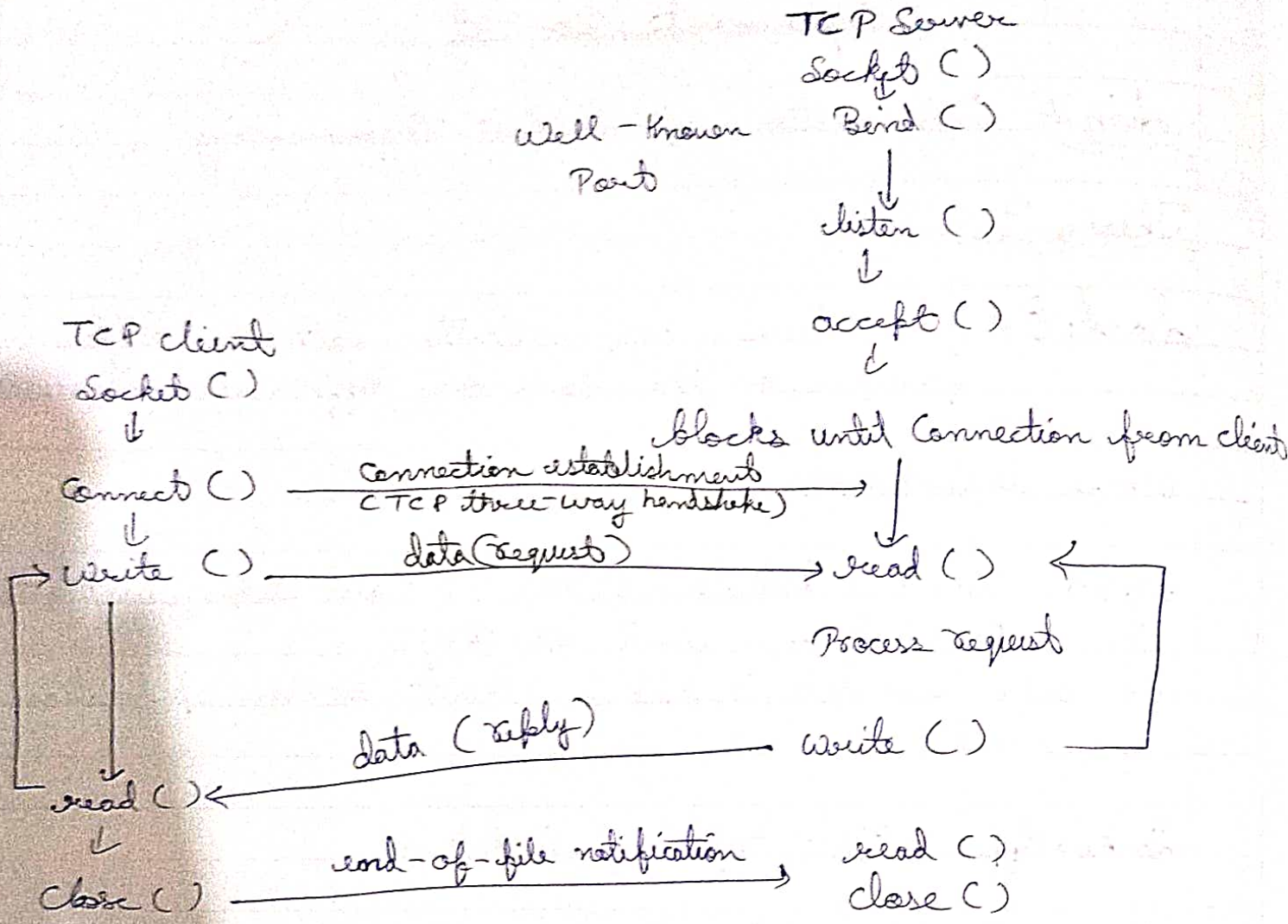
The software called 'middleware, sits b/w the clients and Server.

Types of Server :-

1) Interactive Server :- This is the simplest form of Server where a Server process server one clients and after completing first request, it takes another clients.

Teacher's Signature

Client and Server interaction :



Date

Expt. No. Page No. 20

2) Concurrent server :- The server runs multiple process. The simplest way to write a concurrent server under unix is to fork a client process to handle each client separately.

How to make client?

1) Create a socket with the socket ( ) system call
2) Connect the socket to the address of the server using connect ( ) system call
3) send and received data. Use read and write ( ) System calls.

Unix socket - core functions :-

i) The socket function :-

Syntax :- # include <sys/type.h>
include <sys/socket.h>
int socket (int family, int type, int protocol);

ii) The Connect function :-

The connect function is used by TCP client to establishes connection with a TCP server.

Syntax

include <sys/type.h>
include <sys/socket.h>

unit connect (int sockfd, struct sockaddr *serv-addr, int addrs);

Teacher's Signature

0 if it is successfully connect to server.

1. Socketfd :- Socket descriptor returned by socket function.
2. Serv-addr :- pointer to struct sock addr that contain subscription IP address & Port.
3. Address :- Set it to size of (struct sock addr).

iii) The Bind function :-

Syntax
```
#include <sys/type.h>
#include <sys/socket.h>
int bind (int Socked, struct sockedder * any-addr - int addr);
```

iv) The listen function

Syntax
```
#include <Sys/type.h>
#include <Sys/Socket.h>
int listen (int socked, int backlog);
```

v) The accept function.

Syntax
```
#include <Sys/types.h>
#include <Sys/Socket.h>
int accept (int Sockefd, struct sockaddr *cliaddr, Socklen_t* addr);
```

this call returns non-negative can success, otherwise it return -1.

vi) Send function :-

Syntax
```
int send (int Socket, const void, msg, int less, int glen);
```

vii) The recv function :-

Teacher's Signature

Syntax
int recv (int Socket, void buf, int len, unsigned int flags);

viii) Sendto function ☆ Used to send data over unconnected datagram socket.

ix) The recv from function ☆ receive data from unconnected socket.
int recvfrom (int, Socket, void * buf, int len, unsigned int flags, struct sockaddr *from, int *fromlen);

x) Close Function ☆ Close the communication between client & Server
Syntax
int close (int Socket);

xi) Shut down function ☆ The only difference is that it gives more control in comparision to close function.
int Shutdown (int, socked, int how);

- UNIX - Socket Structure ☆
Various structure are usefull UNIX Socked structure to hold information about address & Parts.

i) Sock adder ☆
struct . Sockadder
{ using nsd short Sa - family ;
char Sa - data [14];
};

ii) Sock adder in 7,

Teacher's Signature

```
struct sockadder_in
{ short int sun_family;
  unsigned short int sun_port;
  struct in adder sin adder;
  unsigned char sin_zero[5];
};
```

iii) in adder *

```
struct in_adder
{unsigned long s. adder; }
```

iv) hostent : struct hostent

```
{ char *h_name;
  char *h_alias;
  int h_adder type;
  int h_length;
  char **h_adder_list;
# define h_adde h_adder_list[0].
```

V) Servent :

```
struct surname { char * s_name };
  char ** s_alias;
  int s_port;
  char * S_proto; };
```

6 Byte ordering function *

function	description
n tons()	has to network short
n tons()	has to network long
n to hl()	network to host long
n to hs()	network to host short.

Teacher's Signature

Date

Expt. No. _____ Page No. 24

Helper Functions

i) The Write Function :-

```
#include <unistd.h>
int write (int fildes, const void *buf, int nbyte);
```

ii) The read function :-

```
#include <Unistd.h>
int read (int fildes, const void *buf, int nbyte);
```

iii) fork function :-

```
#include <Syst/type.h>
#include <unistd.h>
    int fork (void);
```

iv) The b zero function :-

```
Void bzero (Void *s, int n type);
```

v) The bcmp function :-

```
int bcmp (Const + Void *s), Const void *se, int nbyte);
```

vi) The b copy function :-

```
Void cb copy (Const Void *s, void *s2, int n byte);
```

vii) The memset function :-

```
Void * memset (void *s, int c, int n byte)
```

Teacher's Signature _____

Date

Expt. No. Ex-09 Page No. 25

Experiment No - 09

Q: WAP in C: hello - client for TCP [the Server connects to the client, sends the string "Hello world", the closes the connection?

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <unistd.h>
#include <arpa/inet.h>

int main() {
    char *ip = "127.0.0.1";
    int port = 5566;

    int sock;
    struct sockaddr_in addr;
    socklen_t addr_size;
    char buffer[1024];
    int n;
    sock = socket(AF_INET, SOCK_STREAM, 0);
    if (sock < 0) {
        perror("[-] socket error");
        exit(1);
    }
    printf("[+] TCP server socket created.\n");
    memset(&addr, '\0', sizeof(addr));
    addr.sin_family = AF_INET;
    addr.sin_port = port;
```

Teacher's Signature

```
addr.sin_addr.s_addr = inet_addr(IP);

Connect(Sock,(Struct sock addr*) & addr, size of (addr));
    Perror("[-] Socket error");
    exit(1);
}
Printf["[+] TCP server Socket created.\n");
memset(&addr,'10', size of (addr));
addr.sin_family = AF_INET;
addr.sin_Port = Port;
addr.sin_addr.s_addr = inet_addr(IP);

Connect(Sock,(Struct_SockAddr*) &addr, Size of
                                    (addr));
Printf("Connected to the Server.\n");

bzero(buffer, 1024);
strcpy(buffer, "Hello, This is cleint.");
Printf("Cleint: %s\n", buffer);
Send(Sock, buffer, strlen(buffer), 0);
bzero(buffer, 1024);
recv(Sock, buffer, size of (buffer), 0);
Printf("Server: %s\n", buffer);

close(Sock);
Printf("Disconnected from the Server.\n");

return 0;
```

```
addr.sin_addr.s_addr = inet_addr(ip);

Connect(sock,(struct sockaddr*)&addr, sizeof(addr));
    Perror("[-] Socket error");
    exit(1);
}
Printf("[+] TCP server Socket created.\n");
memset(&addr,'\0', sizeof(addr));
addr.sin_family = AF_INET;
addr.sin_port = Port;
addr.sin_addr.s_addr = inet_addr(ip);

Connect(sock,(struct sockaddr*)&addr, sizeof(addr));
Printf("Connected to the Server.\n");

bzero(buffer, 1024);
strcpy(buffer, "Hello, This is client:");
Printf("Client: %s\n", buffer);
Send(sock, buffer, strlen(buffer), 0);
bzero(buffer, 1024);
recv(sock, buffer, sizeof(buffer), 0);
Printf("Server: %s\n", buffer);

close(sock);
Printf("Disconnected from the Server.\n");

return 0;
```

Teacher's Signature

Date

Expt. No. 10

Page No. 27

Experiment No - 10.

Q) WAP to in C: hello - server for TCP S the clients connect to the server) send the string. "Hello world" the closer the connection?

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <onisted.h>
#include <arpa/inet.h>

int main()S
    char *ip = "127.0.0.1";
    int port = 5566;
    int Server_sock, client_sock;
    struct Sockadder_in server_addr, client_addr;
    sockler_t addr_size;
    char buffer[1024];
    int n;

server_sock = Socket - (AF_INET) Sock_STREAM;0);
  if (Server_Sock < 0){
  Perror ("(-) Socket error");
    exit (1);
  }
Printf ("[+] TCP Server socket Created. \n");
memset (& Server_addr, '\0', Size of (Server_addr));
Server addr. sin_family = AF_INET;
Server_addr. sin_port = Port;
```

Teacher's Signature

Date

Expt. No. Page No. 28

```
server_addr.sin_addr.s_addr = inet_addr(IP);
n = bind(server_sock, (struct sockaddr *)&server_addr,
                        sizeof(server_addr));
if(n<0){
    perror("[-] Bind error");
    exit(1);
}
printf("[+] Bind to the port number: %d\n", 7891);

listen(server_sock, 5);
printf("listening...\n");

while(1){
    addr_size = sizeof(client_addr);
    client_sock = accept(server_sock, (struct sockaddr*)
                        &client_addr, &addr_size);

printf("[+] client connected.\n");
bzero(buffer, 1024);
recv(client_sock, buffer, sizeof(buffer), 0);
printf("client: %s\n", buffer);

bzero(buffer, 1024);
strcpy(buffer, "HI, THIS IS SERVER. HAVE A NICE DAY!!!");
printf("Server: %s\n", buffer);
send(client_sock, buffer, strlen(buffer), 0);

close(client_sock);
```

Teacher's Signature

```
    printf("[+] client disconnected.\n\n");
}
    return 0;
}
```