Roll-no:-51
PCE20CS047

POORNIMA
COLLEGE OF ENGINEERING

Affiliated to Rajasthan Technical University, Kota and approved by AICTE

LABORATORY CERTIFICATE

Name....Deepak Dayma.....

Name of the laboratory..Network Programming Lab....**Lab Code**..YCS4-23.....

Section.....A.....**Batch**.....A3.....

Department.....Computer Science.....

This is to certify that Mr./Ms...Deepak Dayma.....with registration no...PCE20CS047.....has satisfactorily completed the course of experiments in the above practical examination conducted by the Department. Total number of experiments performed by him/her are.....10.....

Signature of Faculty Incharge

POORNIMA COLLEGE OF ENGINEERING , JAIPUR

Evaluation Report

Name of the Laboratory....Network Programming Lab.....

Code...YCS4-23.....

S.No.	To be filled by the student: Name of the experiment	Pg. No.	Date of Allotment	Date of Performance	Attendance (2)	To be filled by the faculty: Record* (3)	Performance** (5)	Total (10)	Signature of Faculty With Date
1.	Different Type of LAN (Networks		2/02/22	2/02/22	2	3	5	10	
2.	Various Types of Topologies		18/02/22	18/02/22	2	3	5	10	
3.	LAN installations & configurations		12/05/22	12/05/22	2	3	5	10	
4.	Error correcting (Hamming code)		17/05/22	17/05/22	2	3	5	10	
5.	framing methods (Bit stuffing)		17/05/22	17/05/22	2	3	5	10	
6.	TCP connections (Client & server)		22/05/22	22/05/22	2	3	5	10	
7.	Echo-client & server & Round trip time		23/05/22	23/05/22	2	3	5	10	
8.	UDP with client & server process		31/05/22	31/05/22	2	3	5	10	
9.	multiplexed I/O operations		31/05/22	31/05/22	2	3	5	10	
10.	Bellman Ford Algorithm.		31/05/22	31/05/22	2	3	5	10	
11.									
12.									
13.									
14.									
15.									
16.									
17.									
18.									
19.									
20.									

* Preparation & Lab record

** Overall Quality of Performance, Knowledge about application of experiment, Technical details of equipments, Process & Theory involved in practical & Viva-voce

Max. Marks :.....Marks Obtained.....Average.....

This teach

Web (5)

Only

Blo

POORNIMA

* Experiment - No - (1) - *

*** Aim :-**

Study of Different Type of LAN & Network Equipments.

*** Theory :-**

Types of LAN equipments :-

"Local Area Networking equipments discussed the different types of hardware found in a LAN environment hubs, bridges and switches and how each of hardware functions specifically in a ethernet environment.

a) Repeaters (Layer / Devices) :-

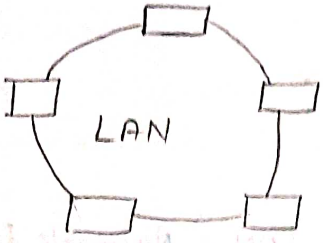
A repeater is a network device used to regenerate or replicate a signal. Repeaters are used in transmission system to generate analog or digital signal distorted by transmission cost. Repeaters are used in both local and wide area networking equipments.

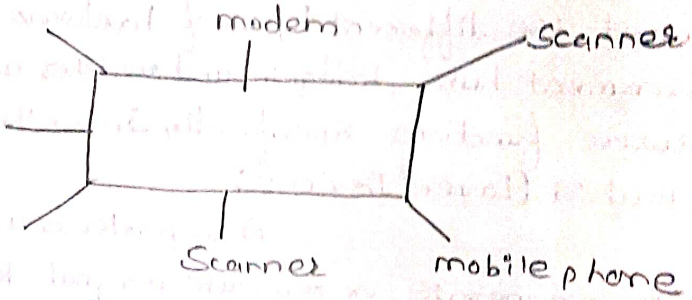
b). Hubs (Layer / Devices) :-

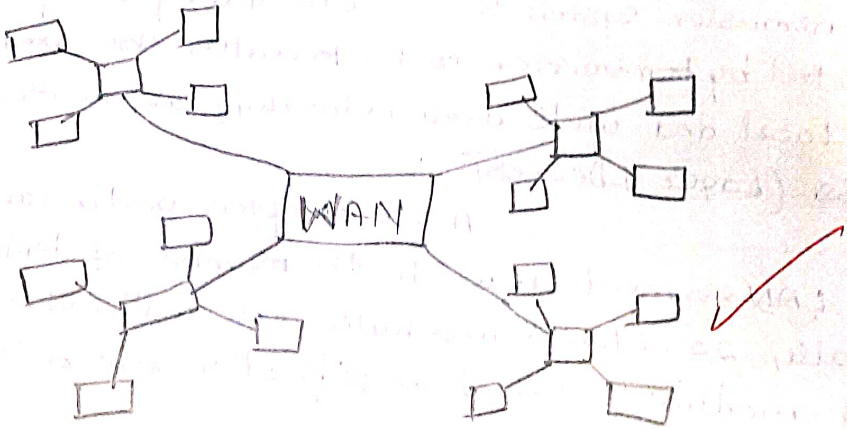
A hub is often used in connected small LAN segment in which the number of device is generally 24 or lower and hubs are multiport repeaters Hubs module the signal amplification required to allow a segment.

(c). Bridges :-

Repeater and hubs no intelligence i they test repeat whatever signal is required or received from one port out all ports without looking at what is being sent or received. Bridges are used of


LAN


modem
Scanner
Scanner
mobile phone


WAN

to networking by using MAC address to build a table of hosts, mapping these hosts to an wide segments.

(d) Switches (Layer 2 Devices):-

Switches sit in the same place in the Network as hubs unlike hubs, however switches examine each frame and process the frame accordingly instead of just repeating the signal to all ports. Switches map the MAC address of the nodes residing on each network segment and then allow only the necessary traffic to pass trans the switch.

(e). Routers (Layer 3 Devices) :-

Routers are devices that forword data packets from one LAN to WAN to another. Based on routing tables and routing protocols, routers read the network address in the packet contained within each transmitted programs.

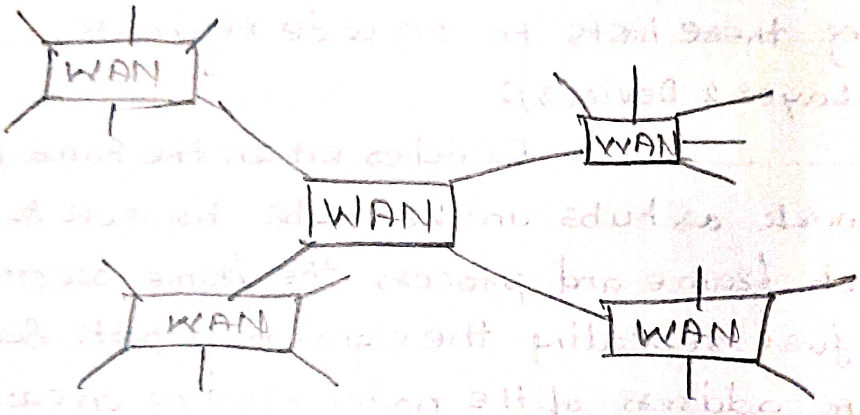
(F). Gateways (Layer 7 Devices):-

Gateways is a device which is used to connect multiple networks and passes from one packets to the other networks. Acting as the 'Gateway' b/w different networking system or computer program on gateway is a device which form a link between the.

* Networking Equipments :-

(1). Network Card :-

Also known as network interfaces cards (NICs) are hardware devices the Connect a Computer with the network. They are installed on the


WAN
WAN
WAN
WAN
WAN

mother board. They are responsible for decomposing a physical connecting between the network and the Computer.

(b) Modems :-

modem is a device which converts the computer generated digital signals of a computer into analog signals to enable their travelling via phone lines. The 'modulator-demodulator' or modem can be used as a dial up for LAN or to connect to an ISD. Modems can be both external as in the devices which concepts to the USB or the serial port of a computer or proprietary devices for household gadgets and other devices, as uses as internal; in the form of fold-add in expansion cards for Computers and PCMIA cards for laptops.

POORNIMA

* Experiment - No - (2) *

***Aim :-**

Study and Verification of Standard Network topologies i.e., Star, Bus, ring etc.

***Theory :-**

Standard Network Topology :-

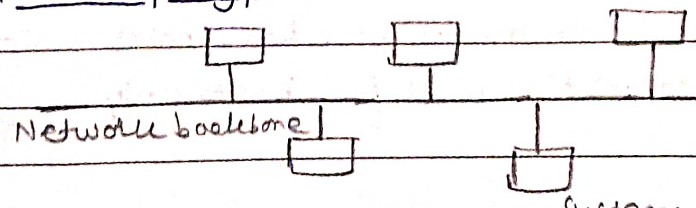
Network topology is the physical interconnections of the elements (Links, nodes, etc) of a computer. A Local area network is one example of a Network that exhibits both a physical topology and a logical topology. Any given node in the LAN has one or more Links to one or more than other nodes in the networks and the mapping of these links and nodes in a graph result in geometrical shape.

The various types of Network topologies are as follows:

(1) **Hierarchial Topology :-**

It is also known as tree topology which is divided into different levels connected with the help of twisted pair, coaxial cable or fiber optics. This types of topology is arranged in the form of a tree structure in which top level contains parent node (root node).

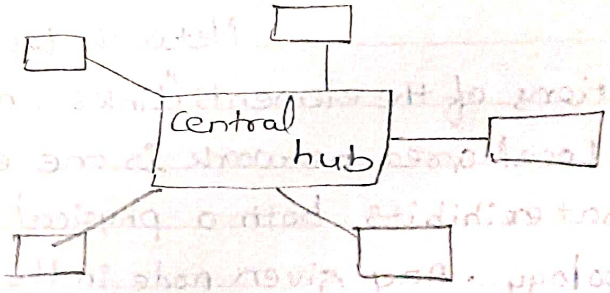
(2) **Bus topology :-**



All devices are single communication line or cable. Bus topology may problem while multiple hosts sending data at same time. Therefore Bus topology uses CSMA/CD technology or recognise one hosts as Bus master to save the issue. is the simplest form of networking.

(3) Star topology:-

All hosts in star topology are connected to a cantool device, known as hub device using a port to point connection There exists a point to point connectings blw hosts and hubs.



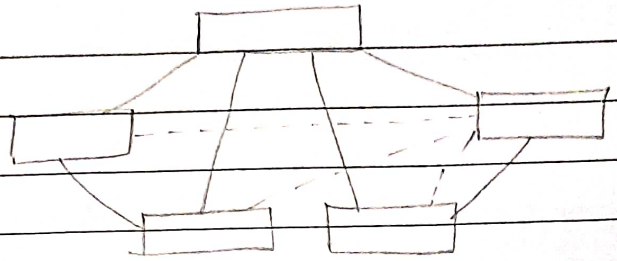
(4). Ring Topology:-

In this each hosts machine connects to exactly two other m/c, creating a circular network structure when one hosts tries to communicate or send messages to a host which is not adjacent to it, the data traverse through all intermediate hosts.

(5). mesh topology:-

A host is connected to one or multiple hosts This topology has hosts in point to point connection with every other hosts on many also have hosts which are in p-top connection with few hosts only.

POORNIMA



(6) Hybrid Topology :-

The hybrid topology is the combination of multiple topologies, used for construction a single cause Topology. The hybrid topology is created with two different network topology are interconnected : if two ring topologies are connected when two different consultant topology is not the hybrid technology

POORNIMA

* Experiment no - (3) *

* **Aim :-**

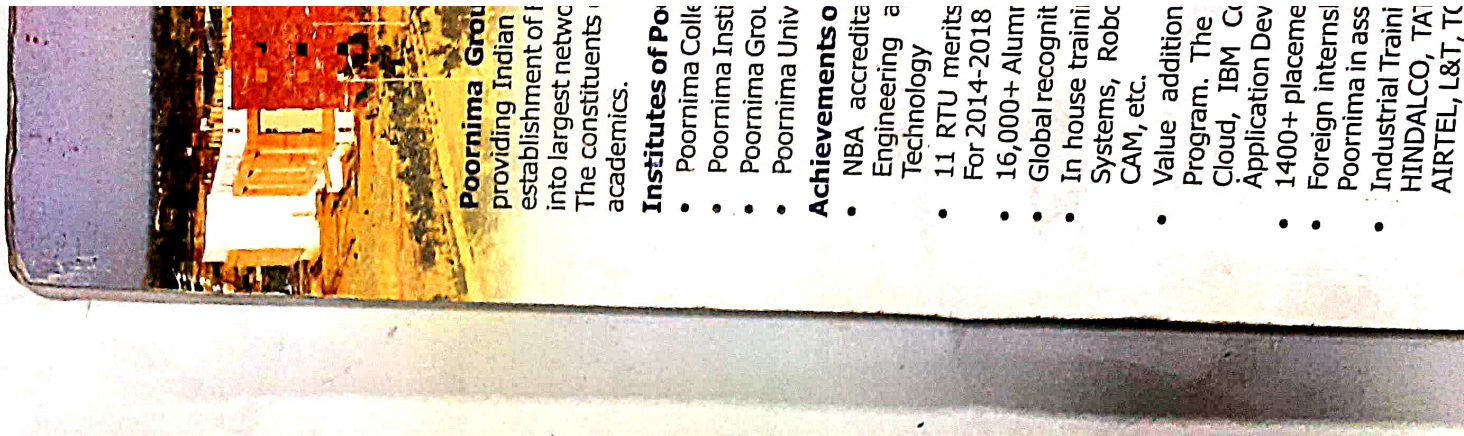
LAN installations and configurations.

* **Theory :-**

Simply put, a LAN is a computer network that connects a small area. most LANS connects work stations and computer each other. Each computer (also known as a "name"), has it's own processing unit and executes it's own programs, however it can also access data and devices anywhere on the LAN. This means that many users can access and share the same information and devices.

STEPS :- Processed with following steps to configure your LAN.

1. From your windows Desktop, click "Start"
2. click on "Control panel".
3. click on "Network and Internet connections".
4. click on "Local area connection".
5. click "properties" on the "Local area connection status" panel.
6. You should find a check mark already in the box next to "Internet protocol (TCP/IP)". Double click "Internet protocol (TCP/IP)"
7. Select "obtain an IP address automatically".
8. Select "obtain DNS secure address automatically".
9. click the "ok" button.
10. click the next "ok" button.



(11). close the "network connections" panel.

How to Set up a Local Area Network(LAN)?

The administrator would need a wireless router linked to a broad-band connection and an ethernet cable to connect the router to the main computer or server to create a wireless network.

This allows other computer devices that already have incorporated or connected wireless network signals and connects to the local area network.

POORNIMA

* Experiment - No - ⑨ *

* **Aim :-**

write a program to implement various type of error correcting techniques.

* **Hamming Code :-**

Haming code is set of error correcting techniques that can be used to detect and correct the error that can occur when the data is moved or stored from the sender to the receiver.

• **Parity Bits :-**

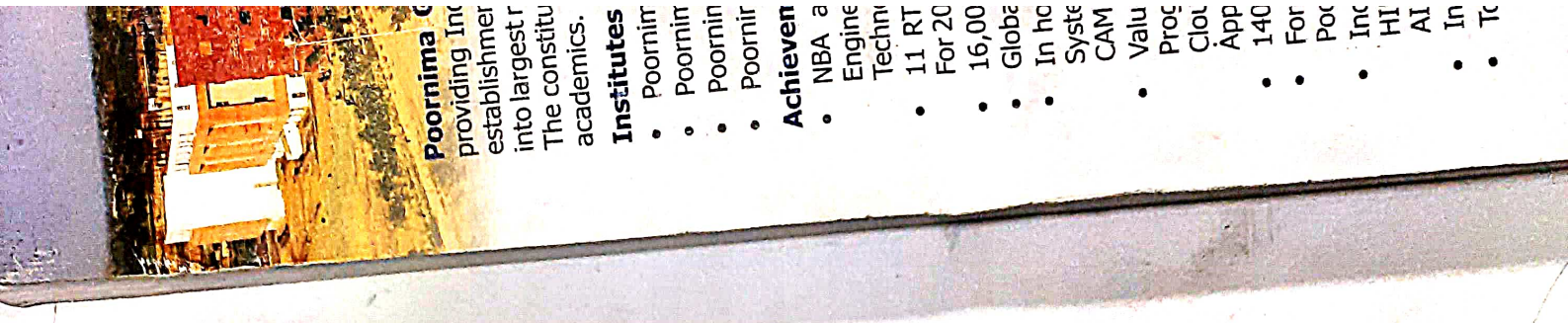
A parity bits is a bit appended to a data of binary bits to ensure that the total number of it's (1's) in the data are even or odd. Parity bits are used for error detection. There are two type of parity bits:

(i) **Even parity Bits :-**

In even parity bits for a given set of the bits. The number of (1's) are counted. If that combination 1's odd, the parity bit values is set of 1. Marking the total count of occurences of 1's an even number.

(ii) **Odd parity Bits :-**

In the case of odd parity, for a given set of bits. The number of (1's) are counted. If that count is even the parity bit value is set to (1), matching the total count of occurences of 1's odd number.



*General Algorithm of Hamming Code :-

The Hamming Code is simply the use of extra parity bits to allow the identification of an error:

(1). Write the bit position starting from 1 in binary form (1, 10, 11, 100) etc.

(2). All the bit positions that are a power of a are marked as parity bits (1, 2, 4, 8, etc)

(3) All the other bits positions are marked as data bits.

(4). Each data bits is included in a unique set of parity bits as determined it's position in binary form.

(5). Since we check for even parity set a parity bit to 1 if, the total number of once in the positions if check is of odd.

(6). Set of parity bit to 0 if the total number of once in the positions if check is even.

POORNIMA

* Program :-

```
#include<iostream>
using namespace std;
int main(){
  int a[10], b[10], c1, c2, c3, c4, *;
  cout<<" Enter a 4 bits";
  cin>>a[3];
  cin>>a[5];
  cin>>a[6];
  cin>>a[7];
  a[1] = a[3]^a[5]^a[7];
  a[2] = a[3]^a[6]^a[7];
  a[4] = a[5]^a[6]^a[7];
 for(int i=1;i<0;i++)
    cout<<a[i];
    cout<<"Enter 7bits";
    for(int i=1;i<8;i++)
      {
      cin>>b[i]
        {
    c1 = b[1]^b[3]^b[5]^b[7];
    c2 = b[2]^b[3]^b[6]^b[7];
    c4 = b[5]^b[5]^b[6]^b[7];
  int p=c1x1+c2x2+c3x4
    if(p==0)
      {
      cout<<"no error";
      }
```

```
else {
    cout << There is error in " << p;
if (b[p] == 0)
        b[p]=1;
    else
        b[p] = 0;
        }
for (int i =1; i<8; i++);
        cout << " b[i];
        return 0;
        }
```

* Lab outcome :-

By this experiment students will able to imple-ments various types of error correcting techniques.

POORNIMA

* Experiment - No - 5 *

***Aim :-**

write a program to implement various types of framing methods.

***Theory :-**

Framing :-

Since the physical layer merely accepts and transmits a stream of bits without any regard to meaning or structure, it is upto the data link layer to create and recognise by attaching special bit patterns to the begining and end of the frame.

The three framing methods that are widely used are :-

- Character Count
- starting and ending character with character stuffing
- Starting and ending flags with bit stuffing.

***Bit stuffing :-**

The third method allows data frames to contain an arbitary number of bits and allows a character codes with an arbitary number of bits per character. At the start and end of each frame is a flag byte consuteetive of the special bit pattern 01111110, whenever the sender's data link layer counters five consecutive 1's in the data it automatically stuffs a zero bits into the

POORNIMA

outgoing bit stream.

* Programs for Bit Stuffing:-

```
#include <stdio.h>
int main(){
    int n, i, count=0;
    printf(" Enter length :");
    scanf("%d", &n);
    int bits[n];
    printf("Enter bits:");
    for(int i=0; i<n; i++){
        scanf("%d", &bits[i]);
        }
    printf("After bit stuffing :");
    for(int i=0; i<n; i++){
        if(bits[i]==1)
            count++;
        else
            count=0;
        printf("%d", bits[i]);
        if(count==5)
            printf("0");
        }
    }
```

POORNIMA

* Experiment - No - (6) *

Aim :- (6.a) →

write a program for successfully connection of TCP Server.

*** Program Code :- (TCP Server) →**

```
#include<stdio.h>
#include<netdb.h>
#include<netinet/in.h>
#include<stdlib.h>
#include<string.h>
#include<sys/socket.h>
#include<sys/types.h>
#define MAX 80
#define PORT 8080
#define SA struct sockaddr.
void function (int connfd)
{
    char buff[MAX];
    int n;
    for(;;){
        bzero(buff, MAX);
    read(connfd, buff, sizeof(buff));
printf("From client : %s\t To client : ", buff);
    bzero(buff, MAX);
    n=0
while((buff[n++] = getchar()) != '\n');
```

*Output :-

```
Socket sucessfully created..
socket sucessfully binded..
Server listening
Server accept the client..
 From client : hi
        To client : hello
 From client : exit
        To client : exit
Server exit..
```

```
write(connfd, buff, sizeof(buff));
if(strncmp("exit", buff, 4)==0){
        printf("Server Exit ....\n");
            break; }
        }
        }
// Driver function
int main() {
        int sockfd, connfd, len;
        struct sockaddr_in servaddr, cli;
        sockfd = socket(AF_INET, SOCK_STREAM, 0);
        if(sockfd==-1) {
                printf("Socket creation failed ....\n");
                exit(0);
                }
        else
                printf("Socket successfully created..\n");
    bzero(&servaddr, sizeof(servaddr));
    servaddr.sin_family = AF_INET;
    servaddr.sin_addr.s_addr = htonl(INADDR_ANY);
        servaddr.sin_port = htons(PORT);
    if ((bind(sockfd, (SA*)&servaddr, sizeof(servaddr)))
                !=0){
    printf("Socket bind failed\n");
        exit(0); }
    else
        printf("Socket Successfully binded\n");
```

```
if((listen(sockfd,5))!=0){
        printf("Listen failed\n");
    exit(0);
    }
else
        printf("Server listening...\n");
         len=sizeof(cli);
connfd=accept(sockfd,(SA*)&cli,&len);
    if(connfd<0){
            printf("Server accept failed\n");
             exit(0);
              }
        else
        printf("Server accept the client\n");
        func(connfd);
  close(sockfd);
        }
```

* (6.b) Aim :-

write a program to implement the TCP client connections.

* Program Code :- (TCP client) →

```
#include<netdb.h>
#include<stdio.h>
#include<stdlib.h>
#include<string.h>
#include<sys/socket.h>
```

Output :-

Socket sucessfully created..
Connected to the server..
Enter the string : hi
from server : hello
Enter the string : exit
from server : exit
client exit...

POORNIMA

```
#define MAX 80
#define PORT 8080
#define SA struct sockaddr
void function(int sockfd) {
    char buff[MAX];
    int n;
    for(;;){
    bzero(buff, sizeof(buff));
    printf("Enter the string: ");
    n=0;
while((buff[n++]=getchar())!='\n');
write(sockfd, buff, sizeof(buff));
    bzero(buff, sizeof(buff));
read(sockfd, buff, sizeof(buff));
printf("From server : %s", buff);
    if((strncmp(buff, "exit", 4))==0){
    printf("Client Exit...\n");
        break;
        }
    }
}
int main(){
    int sockfd, connfd;
struct sockaddr_in servaddr, cli;
sockfd= socket(AF_INET, SOCK_STREAM, 0);
    if(sockfd==-1){
printf("socket creation failed...\n");
    exit(0); }
```

```
else
    printf( "Socket successfully created..\n");
bzero (&servaddr, sizeof (servaddr));
servaddr.sin_family = AF_INET;
servaddr.sin_addr.s_addr = inet_addr("127.0.0.1");
servaddr.sin_port = htons (PORT);
if (connect (sockfd, (SA*) &servaddr, sizeof(servaddr)!
                                        =0) {
    printf("Connection with server failed..\n");
        exit(0);
    }
else
printf ("connected to the server..\n");
    func(sockfd);
close (sockfd);
    }
```

POORNIMA

* Experiment -No-(7) *

Aim :-

Write an Echo server using TCP to estimate the round trip time from client to the server. The server should be such that it can accept multiple connections at any given time, with multiplexed I/O operations.

Program Code for client (for echo server) :-

```
#include <stdlib.h>
#include <netinet/in.h>
#include <sys/types.h>
#include <sys/socket.h>
#include <string.h>
#include <unistd.h>
#define MAXCOUNT 1024
int main(int argc, char* argv[])
{
int sfd;
char msg[MAX COUNT];
char blankmsg[MAX COUNT];
struct sockaddr_in saddr;
memset(&saddr, 0, sizeof(saddr));
sfd = socket(AF_INET, SOCK_STREAM, 0);
saddr.sin_family = AF_INET;
inet_pton(AF_INET, "127.0.0.1", &saddr
,sizeof(saddr));
```

Poornima Group, established
providing Indian Industry with
establishment of First Technical I
into largest network of Professio
The constituents of Poornima G
academics.

Institutes of Poornima Grou

- Poornima College of Enginee
- Poornima Institute of Engine
- Poornima Group of Institutio
- Poornima University

Achievements of Poornima

- NBA accreditation of bra
  Engineering and Poornim
  Technology
- 11 RTU merits (2013-201
  For 2014-2018 Batch merit
- 16,000+ Alumni around th
- Global recognitions & allia
- In house training on PLC,
  Systems, Robotics, JAVA
  CAM, etc.
- Value addition program
  Program. The programs
  Cloud, IBM Cognos for

* Output :-

Sucessfully echo server setupped

POORNIMA

```
for(;;){
    memset(msg, 0, MAX COUNT);
    memset(blankmsg, 0, MAX COUNT);
    fgets(msg, MAXCOUNT, stdin);
    send(sfd, blankmsg, sizeof, strlen(msg), 0);
    recv(sfd, blankmsg, sizeof(blankmsg), 0);
    printf("%s", blankmsg);
    fflush(stdout);
        }
    exit(0);
    }
```

POORNIMA

* Experiment -No- (8) *

* Aim :-

Write a program to implementation of client side and server side for UDP server model.

* Program code for client side UDP server model :-

```
#include<stdio.h>
#include<stdlib.h>
#include<unistd.h>
#include<string.h>
#include<sys/types.h>
#include<sys/socket.h>
#include<arpa/inet.h>
#include<netinet/in.h>
#define PORT 8080
#define MAXSIZE 1024
int main(){
    int sockfd;
    char buffer[MAXLINE];
    char *hello = "Hello from client";
    struct sockaddr_in servaddr;
if((sockfd=socket(AF_INET,SOCK_DGRAM,0))<0)
  { perror("socket creation failed");
    exit(EXIT_FAILURE);
  }
memset(&servaddr,0,sizeof(servaddr));
servaddr.sin_family=AF_INET;
```

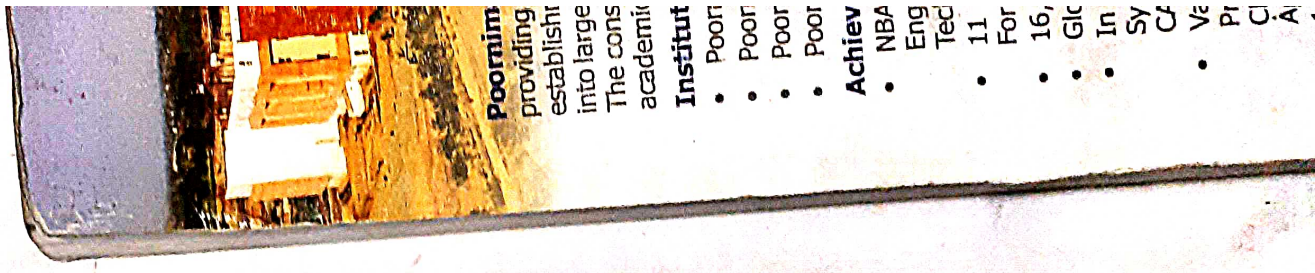
*Output :-

```
$./server
client : Hello from client
Hello message sent.
$ ./client
 Hello message sent
Server : Hello from server.
```

```
servaddr.sin_port = htons(PORT);
servaddr.sin_addr.s_addr = INADDR_ANY;
int n, len;
sendto(sockfd, (const char *)hello, strlen(hello),
    MSG_CONFIRM, (const struct sockaddr *)&
servaddr, sizeof(servaddr));
printf("Hello message.\n");
n = recvfrom(sockfd, (char *)buffer, MAXLINE,
    MSG_WAITALL, (struct sockaddr *)&servaddr,
    &len);
    buffer[n] = '\0';
printf("Server : %s\n", buffer);
    close(sockfd);
    return 0;
    }
```

* Server side implementation of UDP client server model:-

```
#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>
#include <string.h>
#include <sys/types.h>
#include <sys/socket.h>
#include <arpa/inet.h>
#include <netinet/in.h>
#define PORT 8080
#define MAXLINE 1024
```



*Output:-

$./server
client : Hello from client
Hello message sent.
$ ./client
Hello message sent.
Server : Hello from server.

POORNIMA

```
int main(){
    int sockfd;
    char buffer[MAXLINE];
    char* hello = "Hello from server";
    struct sockaddr_in servaddr, cliaddr;
    if ((sockfd = socket(AF_INET, SOCK_DGRAM, 0))<0
    ){
    perror("socket creation failed");
    exit(EXIT_FAILURE);
        }
    memset(&servaddr, 0, sizeof(servaddr));
    memset(&cliaddr, 0, sizeof(cliaddr));
servaddr.sin_family = AF_INET;
servaddr.sin_addr.s_addr = INADDR_ANY;
servaddr.sin_port = htons(PORT);
    if (bind(sockfd, (const struct sockaddr*)&serva
        -ddr, sizeof(servaddr))<0)
        {
        perror("bind failed");
          38
        exit(EXIT_FAILURE);
            }
        int len, n;
    n = recvfrom(sockfd, (char*)buffer, MAXLINE,
    MSG_WAITALL, (struct sockaddr*)&cliaddr,
    &len);
```

```
buffer[n] = '\0';
printf ("client : %s\n", buffer);
sendto(sockfd, (const struct (const char*) hello, strlen
    (hello),
msg_CONFIRM,(const struct sockaddr *) &cliaddr,
len);
printf ("Hello message sent .\n");
    return 0;
    }
```

POORNIMA

* Experiment-No-⑨ *

* **Aim :-**

Write an echo server using TCP for multiplexed I/O operations.

* **Program code for Echo server :-**

```
//Echo Server
#include <sys/types.h>
#include <sys/socket.h>
#include <netdb.h>
#include <stdio.h>
#include <string.h>
int main(){
  char str[100];
   int listen_fd, comm_fd;
  struct sockaddr_in servaddr;
   listen_fd = socket (AF_INET, SOCK_STREAM, 0);
bzero (&servaddr, sizeof (servaddr));
 servaddr.sin_family = AF_INET;
 servaddr.sin_addr.s_addr = htons (INADDR_ANY);
 servaddr.sin_port = htons (22000);
bind (listen_fd, (struct sockaddr *) &servaddr,
  sizeof (servaddr));
listen (listen_fd, 10);
comm_fd = accept (listen_fd, (struct sockaddr *)
  NULL, Null);
  while (1) {
```

Poornima Group,
providing Indian Ind
establishment of Firs
into largest network
The constituents of
academics.

Institutes of Poor
- Poornima Colleg
- Poornima Institu
- Poornima Group
- Poornima Univer

Achievements of
- NBA accreditati
  Engineering an
  Technology
- 11 RTU merits
  For 2014-2018
- 16,000+ Alumn
- Global recognit
- In house trainii
  Systems, Robo
  CAM, etc.
- Value additior
  Program. The
  Cloud, IBM C
  Application De
- 1400+ placem
- Foreign intern
- Poornima in as
- Industrial Trai
  HINDALCO, T

*Prene

*Output:-

0. $./server

(1)

(2). Echoing back Hello

Echoing back How are

(3

(4 Echoing back I m fine

Echoing back Bye.

(5

(6

POORNIMA

```
bzero (str, 100);
read (comm_fd, str, 100);
printf ("Echoing back - %s", str);
write (comm_fd, str, strlen(str)+1);
    }
}
```

* Program code for client process :-

```
#include <sys/types.h>
#include <sys/socket.h>
#include <netdb.h>
#include <stdio.h>
#include <string.h>
int main( int argc, char **argv)
  {
  int sockfd, n;
  char sendline[100];
  char recvline[100];
  struct sockaddr_in servaddr;
  sockfd = socket (AF_INET, SOCK_STREAM
                   ,0);
  bzero (&servaddr, sizeof servaddr);
servaddr.sin_family = AF_INET;
servaddr.sin_port = htons (22000);
inet_pton (AF_INET, "127.0.0.1", &(servaddr
      .sin_addr));
```

*Ex

*Output :-

Hello

Hello

How r u

How r u

I m fine

I m fine

Bye

Bye.

POORNIMA

```
connect(sockfd, (struct sockaddr *)&servaddr, sizeof
    (servaddr));
    while(1)
      {
      bzero(sendline, 100);
      bzero (recvline, 100);
      fgets(sendline, 100, stdin);
write(sockfd, sendline, strlen(sendline)+1);
      read (sockfd, recline, 100);
        printf("%s", recvline);
              {
            }
```

POORNIMA

* Experiment - No - (10) *

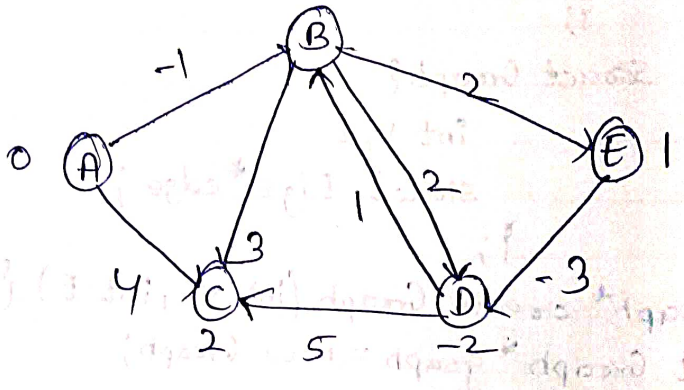
* Aim :-

Stimulate Bellman-Ford Routing Algorithm in NS2.

* Program code Bellman-Ford Routing Algorithm :-

```
#include <bits/stdc++.h>
struct Edge {
        int src, dest, height;
};
struct Graph {
        int V, E
        struct Edge* edge;
};
struct Graph* create Graph (int V, int E) {
    struct Graph* graph = new Graph;
    graph -> V = V;
    graph -> E = E;
    graph -> edge = new Edge[E];
        return graph;
}
void print Arr(int dist[], int n){
printf("vertex Distance from Source\n");
    for(int i=0; i<n; ++i)
    printf("%d \t\t %d \t\n", i, d[i]);
}
void Bellmanford (struct Graph* graph, int src){
        int v = graph -> V;
```

A	B	C	D	E
0	∞	∞	∞	∞
0	-1	∞	∞	∞
0	-1	4	∞	∞
0	-1	2	∞	∞
0	-1	2	∞	1
0	-1	2	1	1
0	-1	2	-2	1



*Output:-

Vertex	Distance from source.
0	0
1	-1
2	2
3	-2
4	1
5	

POORNIMA

```
int V = graph->V;
int E = graph->E;
int dist[V];
for (int i=0; i<V; i++)
    dist[i] = INT_MAX;
dist[src] = 0;
for (int i=0; i<= V-1; i++) {
    for (int j=0; j<E; j++) {
        int u = graph->edge[j].src;
        int v = graph->edge[j].dist;
        int weight = graph->edge[j].weight;
if [dist[u] != INT_MAX && dist[v] + weight <
    dist[v]) dist[v] = dist[v] + weight;
}
}
for (int i=0; i<E; i++){
    int v = graph->edge[i].src;
    int v = graph->edge[i].dist;
    int weight = graph->edge[i].weight
if [dist[u] != INT_MAX && dist[u] +
        weight < dist[v]){
printf ("Graph contains negative weight");
    return 0;
    }
}
Printf Arr[dist.v];
return;
}
```

POORNIMA

```
int main(){
        int V=5;
        int E=8;
struct Graph* graph =create Graph (V,E);
 graph→ edge [0].src =0;
 graph→ edge [0].dist = 1;
 graph→ edge[0]. weight = -1;
  graph→ edge[1].src =0;
  graph→ edge[1] .dist =2
  graph → edge[1].weight = 4;
   graph→ edge [2].src = 1;
   graph→ edge [2]. dist = 2;
    graph→ edge [2]. weight = 3;
    graph→ edge [3]. src = 1;
     graph→ edge[3].dist = 3;
     graph→ edge [3] weight = 2;
    graph→ edge[ 4] dist = 4;
    graph→ edge [4] src = 1;
     graph→ edge [4] weight = 2;
    graph→ edge [5] src = 3;
     graph→ edge [5] weight =5;
     graph→ edge [5] dist =2;
     graph→ edge[6] src=3;
     graph→ edge [6] .dist = 1;
     graph→ edge [6]. weight=1;
     graph→ edge [7] .src = 4;
     graph→ edge [7] .dist = 3;
```

POORNIMA

```
graph -> edge [7].weight = -3;
Bellman_ford (graph, 0);
return 0;
}
```