



VIGNAN'S INSTITUTE OF INFORMATION TECHNOLOGY (AUTONOMOUS)

DUVVADA - VISAKHAPATNAM – 530 049

(An Autonomous Institute, Accredited by NAAC, Affiliated to JNTUK, Kakinada, AP)

COMPUTER NETWORKS LAB MANUAL (VR20)

(III CSE)

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

## INDEX

S.No	Contents	Page No
1	<b>Exercise – 1</b> Implementation a Basic Network using Topologies in Packet Tracer.	3-12
2	<b>Exercise – 2</b> Implement the three CRC polynomials on a data set of characters– CRC 12, CRC 16 and CRC CCIP.	13-15
3	<b>Exercise – 3</b> Switch and Router configuration and Configuring VLAN in packet tracer.	16-23
4	<b>Exercise – 4</b> Implement Dijkstra’s algorithm to compute the Shortest path through a graph.	24-27
5	<b>Exercise – 5</b> Take an example subnet graph with weights indicating delay between nodes. Now obtain Routing table at each node using distance vector routing algorithm.	28-30
6	<b>Exercise – 6</b> Take an example subnet of hosts. Obtain broadcast tree for it.	31-32
7	<b>Exercise – 7</b> Implementation of Connection oriented concurrent service (TCP).	33-34
8	<b>Exercise – 8</b> Implementation of Connectionless Iterative time service (UDP).	35-37
9	<b>Exercise – 9</b> Configuration of DHCP Server and Implementation of DNS.	38-40
10	<b>Exercise – 10</b> Implementation of HTTP (Hyper Text Transfer Protocol).	41-44

## 1. Implementation a Basic Network using Topologies in Packet Tracer.

**Bus Topology:** A bus topology is a network in which nodes are directly linked with a common half-duplex link. A host on a bus topology is called a station. In a bus network, every station will accept all network packets, and these packets generated by each station have equal information priority. A bus network includes a single network segment and collision domain.

### Steps to Configure and Setup Bus Topology

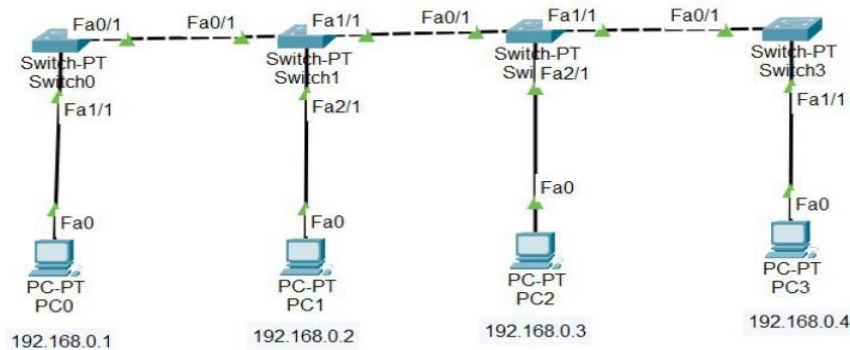
**Step 1:** First, open the cisco packet tracer desktop and select the devices given below:

S.NO	Device	Model-Name
1.	PC	PC
2.	Switch	PT-Switch

### IP Addressing Table

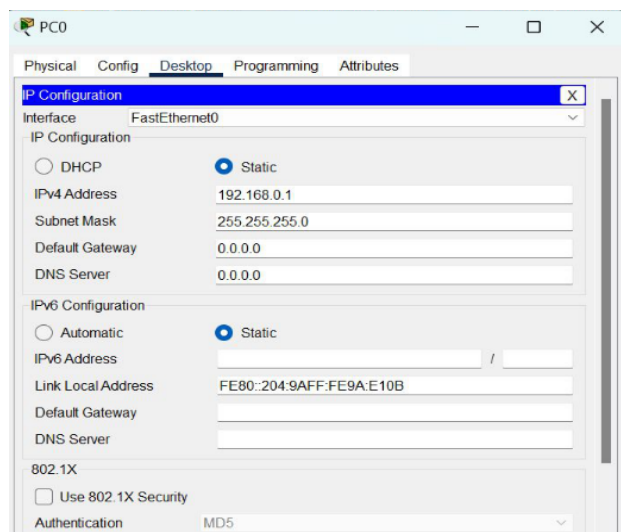
S.NO	Device	IPv4 Address	Subnet Mask
1	pc0	192.168.0.1	255.255.255.0
2	pc1	192.168.0.2	255.255.255.0
3	pc2	192.168.0.3	255.255.255.0
4	pc3	192.168.0.4	255.255.255.0

- Then, create a network topology as shown below image:
- Use an Automatic connecting cable to connect the devices with others.

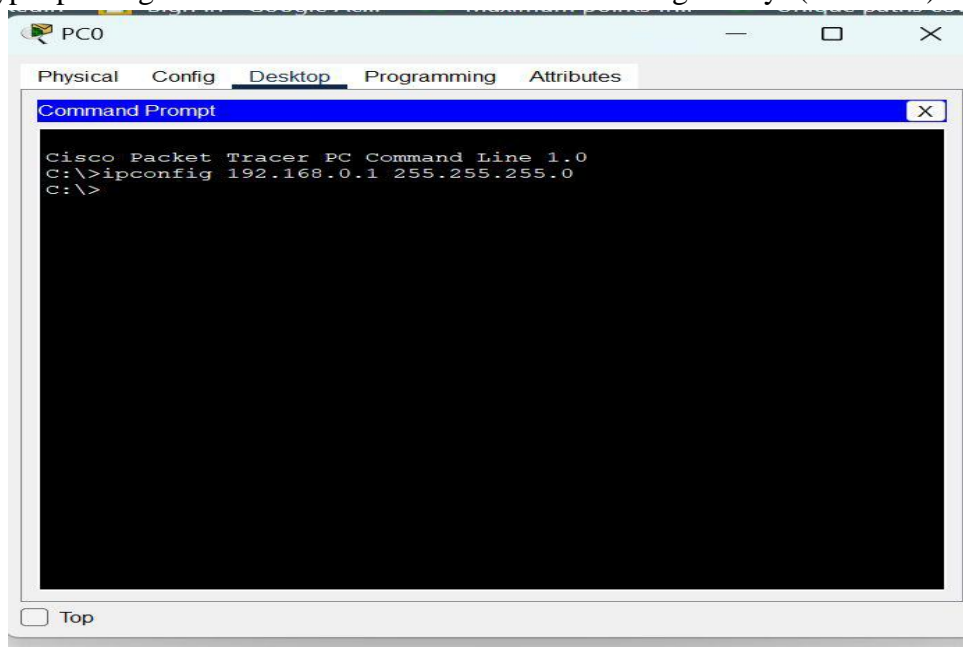


**Step 2:** Configure the PCs (hosts) with IPv4 address and Subnet Mask according to the IP addressing table given above.

- To assign an IP address in PC0, click on PC0.
- Then, go to desktop and then IP configuration and there you will IPv4 configuration.
- Fill IPv4 address and subnet mask.



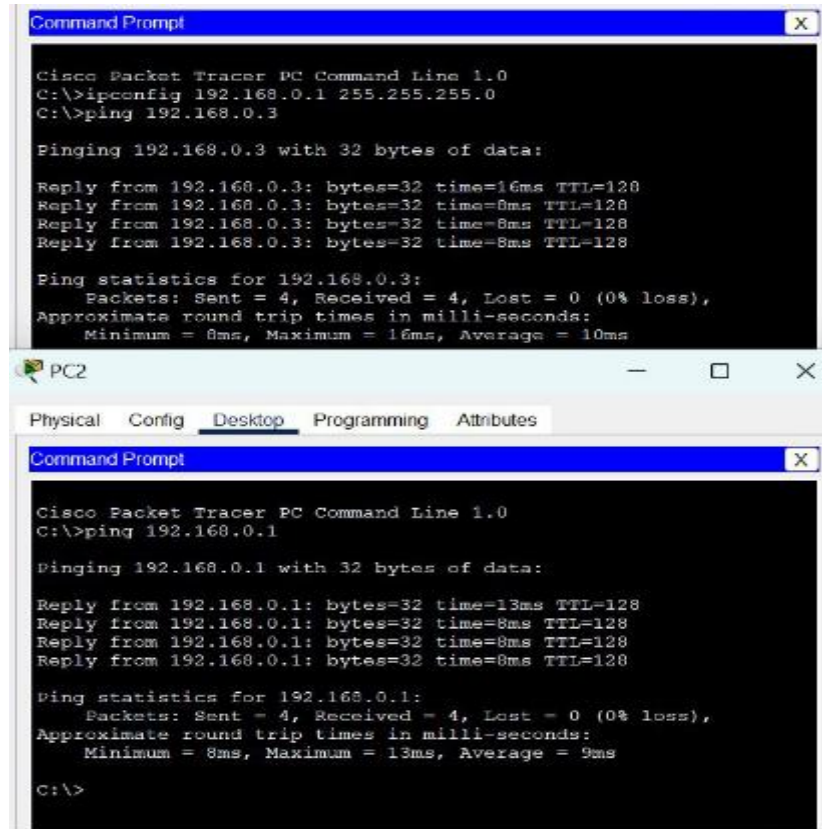
- Assigning an IP address using the ipconfig command, or we can also assign an IP address with the help of a command.
- Go to the command terminal of the PC.
- Then, type ipconfig <IPv4 address><subnet mask><default gateway>(if needed)



- Repeat the same procedure with other PCs to configure them thoroughly.

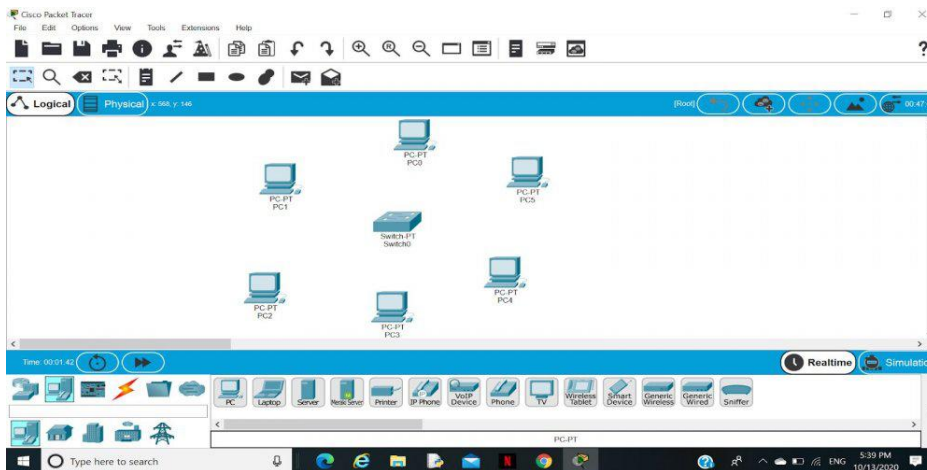
**Step 3:** Verify the connection by pinging the IP address of any host in PC0.

- Use the ping command to verify the connection.
- As we can see we are getting replies from a targeted node on both PCs.
- Hence the connection is verified.

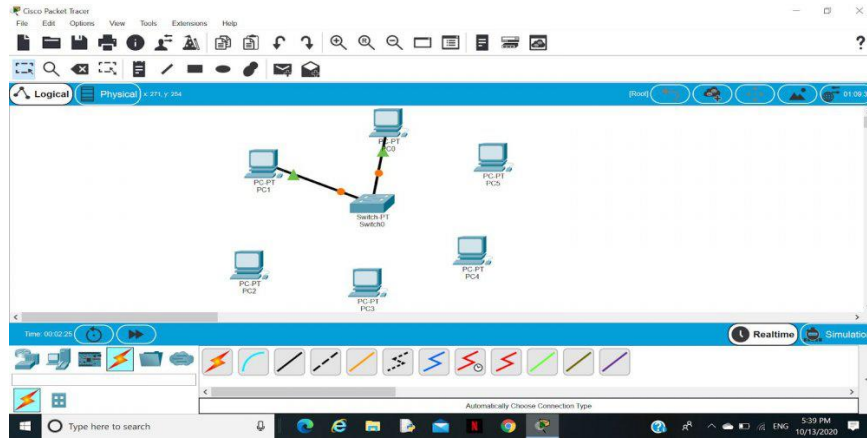


**Star Topology:** A star topology for a Local Area Network (LAN) is one in which each node is connected to a central connection point, such as a hub or switch. Whenever a node tries to connect with another node then the transmission of the message must be happening with the help of the central node. The best part of star topology is the addition and removal of the node in the network but too many nodes can cause suffering to the network.

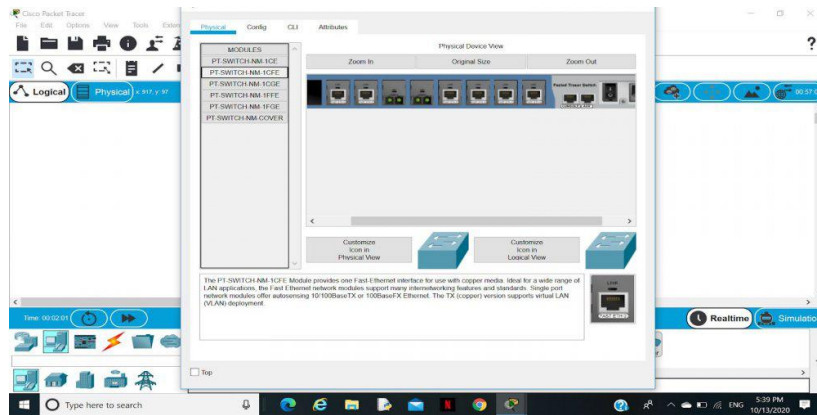
**Step 1:** We have taken a switch and linked it to six end devices.



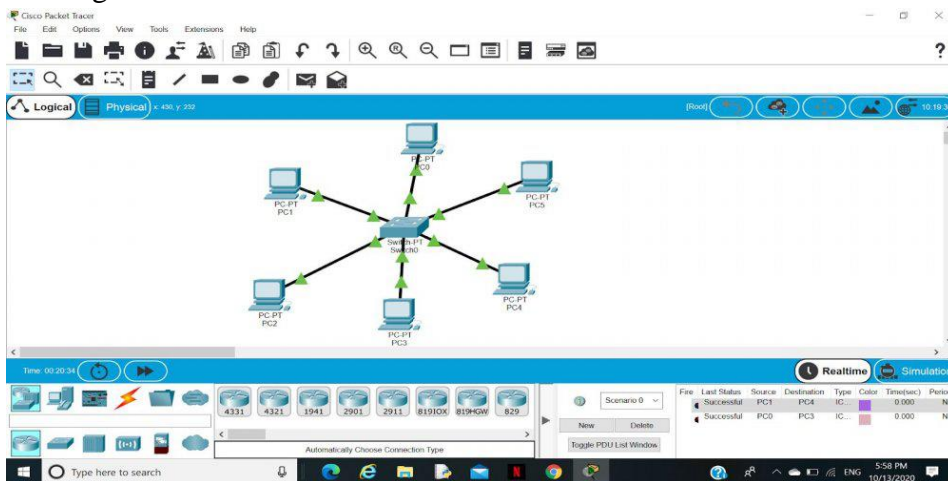
**Step 2:** Link every device with the switch.



**Step 3:** Provide the IP address to each device.



**Step 4:** Transfer message from one device to another and check the Table for Validation.



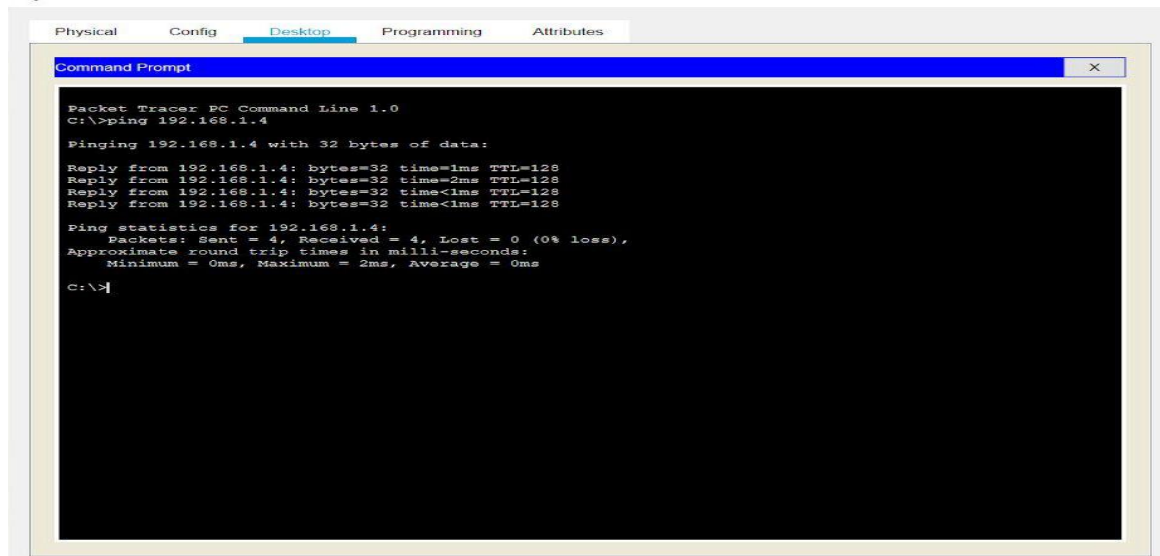
Now to check whether the connections are correct or not

**Command:**

"ping ip\_address\_of \_any\_device"

**Example:**

ping 192.168.1.4



**Ring topology:** Ring topology is a kind of arrangement of the network in which every device is linked with two other devices. This makes a circular ring of interconnected devices which gives it its name. Data is usually transmitted in one direction along the ring, known as a unidirectional ring. The data is delivered from one device to the next until it reaches the decided destination. In a bidirectional ring, data can travel in either direction.

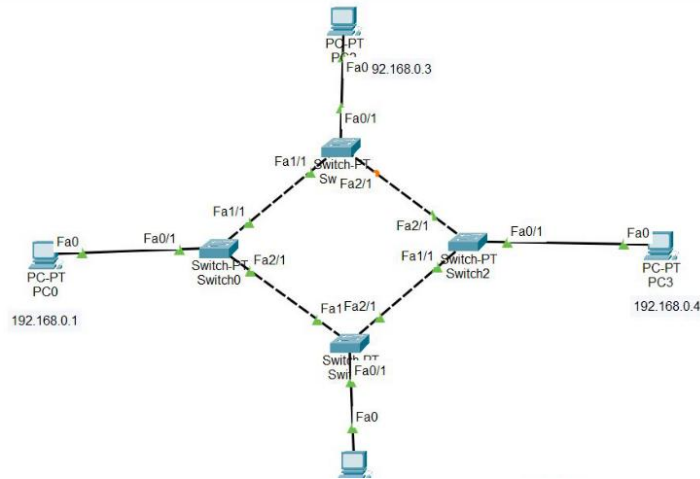
**Step 1:** First, open the cisco packet tracer desktop and select the devices given below:

S.NO	Device	Model Name
1.	PC	PC
2.	Switch	PT-Switch

### IP Addressing Table

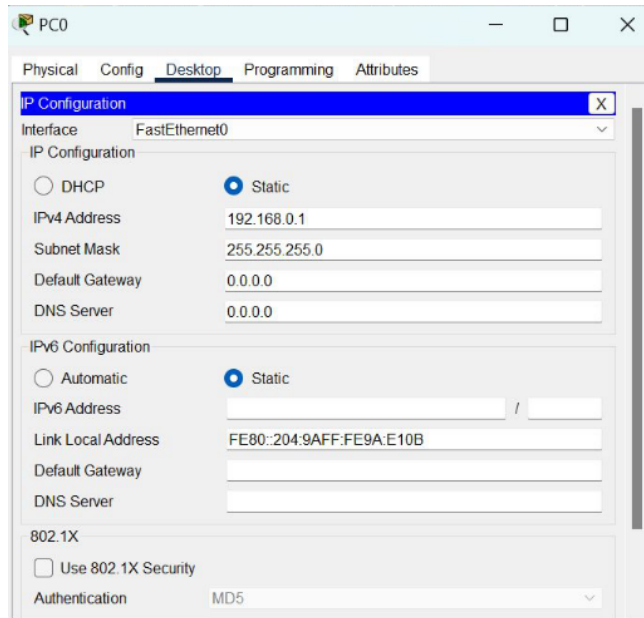
S.NO	Device	IPv4 Address	Subnet Mask
1.	pc0	192.168.0.1	255.255.255.0
2.	pc1	192.168.0.2	255.255.255.0
3.	pc2	192.168.0.3	255.255.255.0
4.	pc3	192.168.0.4	255.255.255.0

- Then, create a network topology as shown below the image.
- Use an Automatic connecting cable to connect the devices with others.



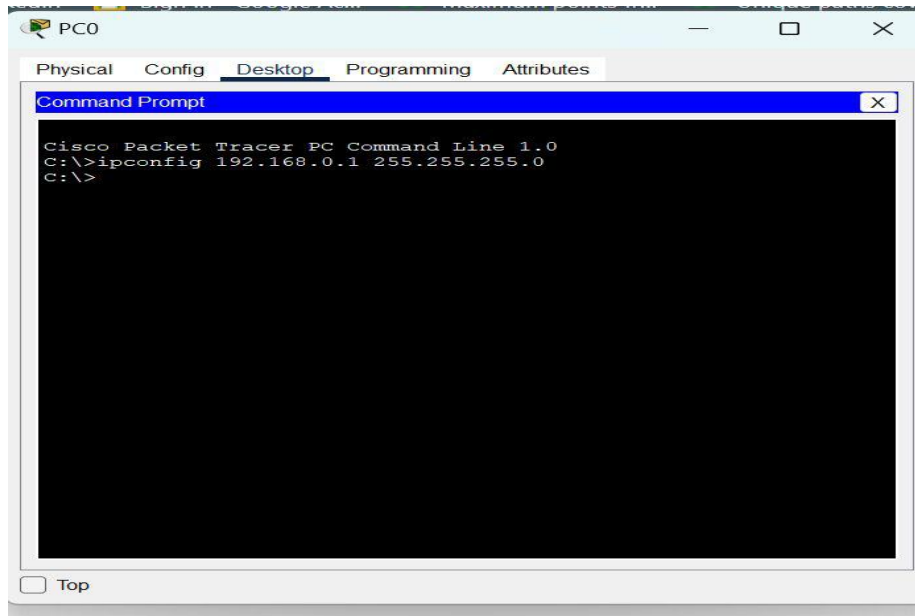
**Step 2:** Configure the PCs (hosts) with IPv4 address and Subnet Mask according to the IP addressing table given above.

- To assign an IP address in PC0, click on PC0.
- Then, go to desktop and then IP configuration and there you will IPv4 configuration.
- Fill IPv4 address and subnet mask.



- Assigning IP address using the ipconfig command, or we can also assign an IP address with the help of a command.
- Go to the command terminal of the PC.
- Then, type ipconfig <IPv4 address><subnet mask><default gateway>(if needed)

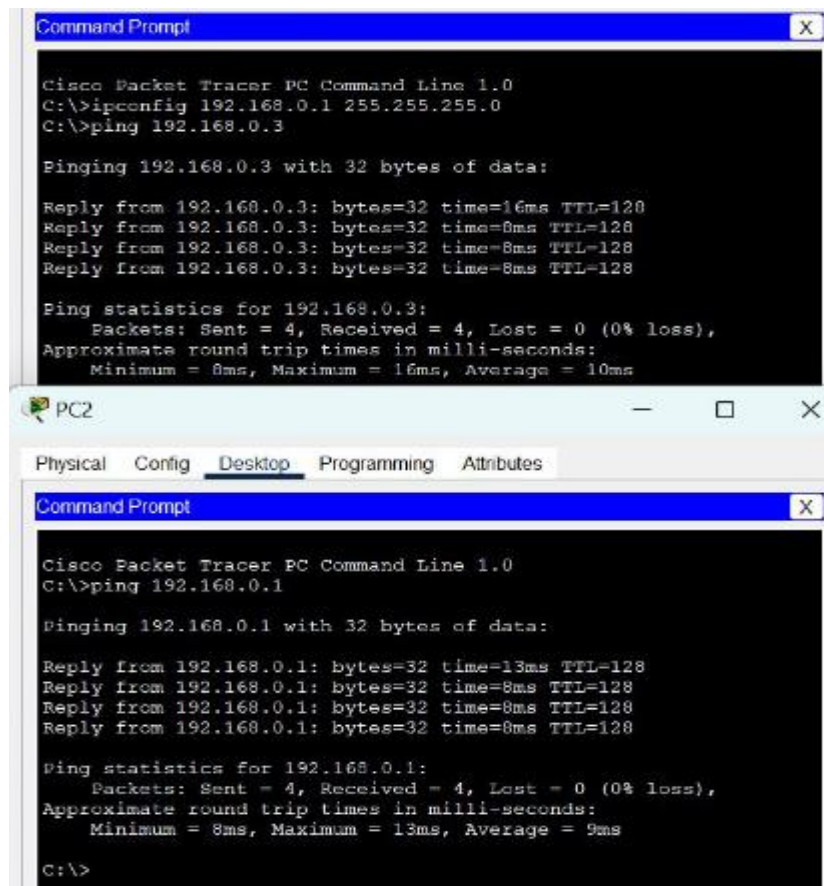




- Repeat the same procedure with other PCs to configure them thoroughly.

**Step 3:** Verify the connection by pinging the IP address of any host in PC0.

- Use the ping command to verify the connection.
- As we can see we are getting replies from a targeted node on both PCs.
- Hence the connection is verified.



**Mesh Topology:** In the mesh topology of networking, each and every device sends its own signal to the other devices that are present in the arrangement of the network.

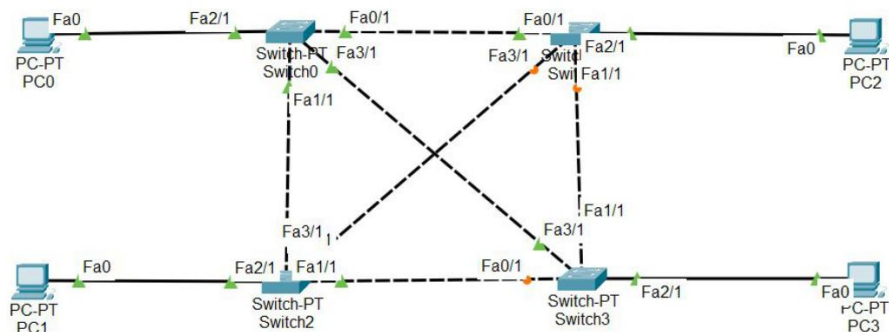
**Step 1:** First, open the Cisco packet tracer desktop and select the devices given below:

S.NO	Device	Model name
1.	PC	PC
2.	Switch	PT-switch

**IP Addressing Table:**

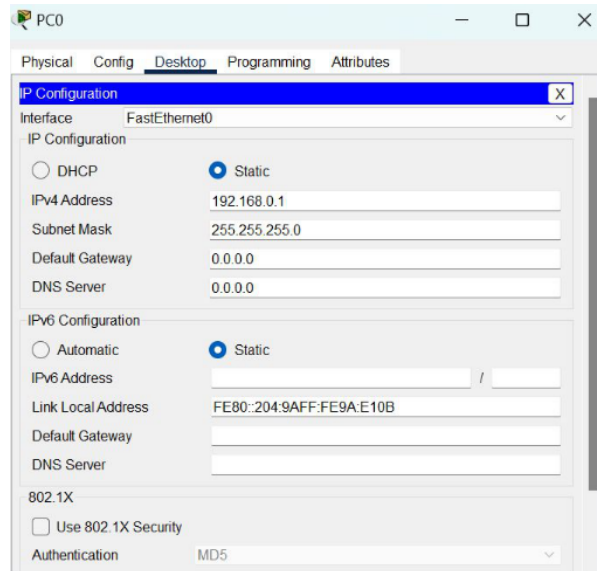
S.NO	Device	IPv4 Address	Subnet Mask
1.	pc0	192.168.0.1	255.255.255.0
2.	pc1	192.168.0.2	255.255.255.0
3.	pc2	192.168.0.3	255.255.255.0
4.	pc3	192.168.0.4	255.255.255.0

- Then, create a network topology as shown below the image.
- Use an Automatic connecting cable to connect the devices with others.

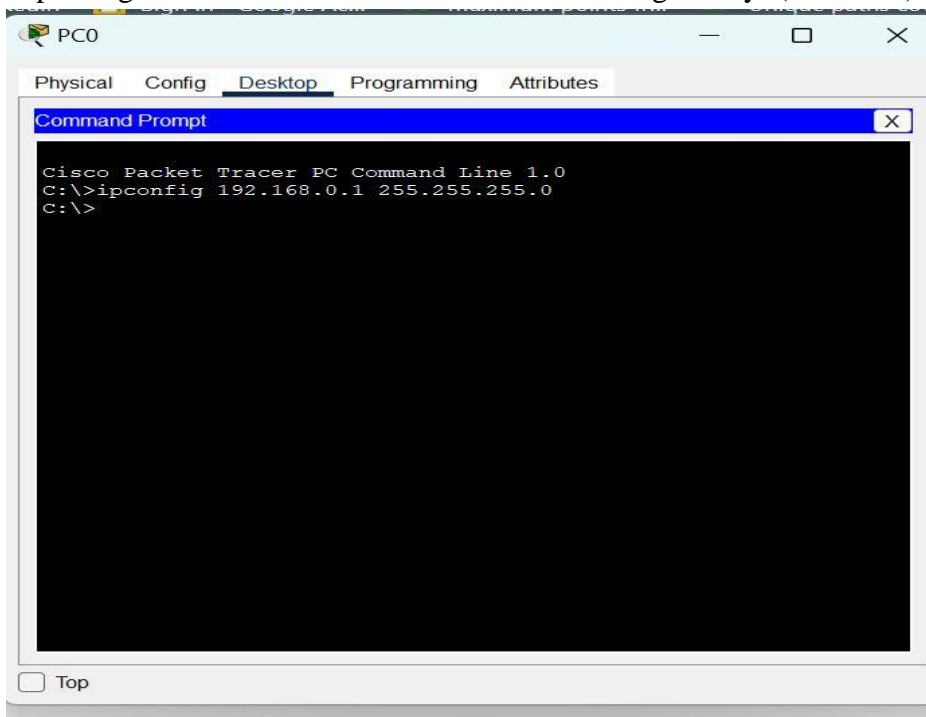


**Step 2:** Configure the PCs (hosts) with IPv4 address and Subnet Mask according to the IP addressing table given above.

- To assign an IP address in PC0, click on PC0.
- Then, go to desktop and then IP configuration and there you will IPv4 configuration.
- Fill IPv4 address and subnet mask.



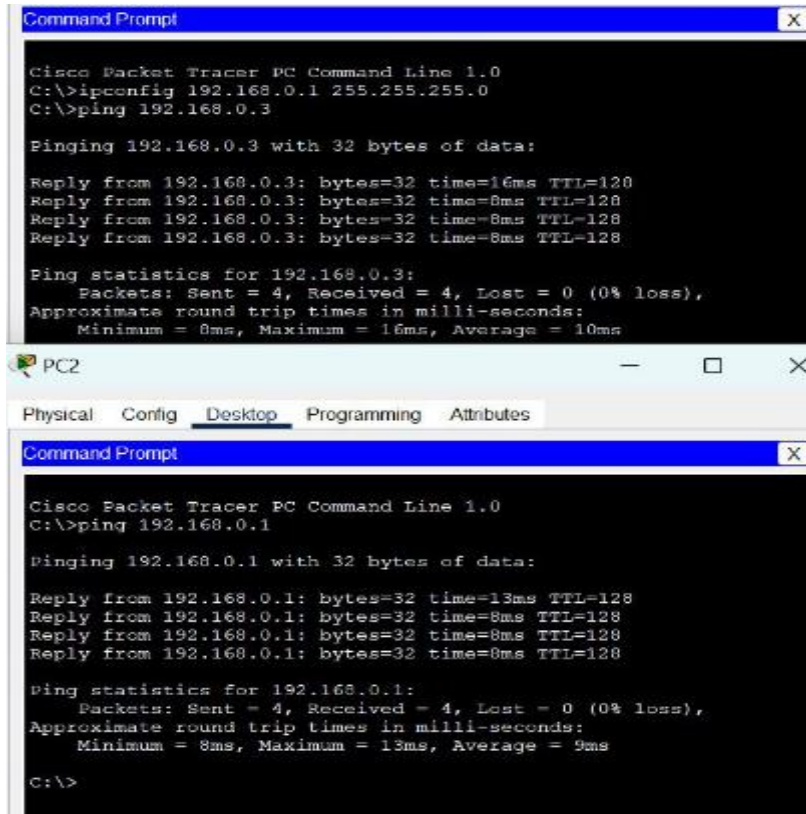
- Assigning IP address using the ipconfig command.
- Also, we can also assign an IP address with the help of a command.
- Go to the command terminal of the PC.
- Then, type ipconfig <IPv4 address><subnet mask><default gateway>(if needed)



- Repeat the same procedure with other PCs to configure them thoroughly.

Step 3: Verify the connection by pinging the IP address of any host in PC0.

- Use the ping command to verify the connection.
- We will check if we are getting any replies or not.
- Here we get replies from a targeted node on both PCs.
- Hence the connection is verified.



## 2. Implement the three CRC polynomials on a data set of characters– CRC 12, CRC 16 and CRC CCIP

CRC method can detect a single burst of length  $n$ , since only one bit per column will be changed, a burst of length  $n+1$  will pass undetected, if the first bit is inverted, the last bit is inverted and all other bits are correct. If the block is badly garbled by a long burst or by multiple shorter burst, the probability that any of the  $n$  columns will have the correct parity that is 0.5. so the probability of a bad block being expected when it should not be  $2^{-n}$ . This scheme sometimes known as Cyclic Redundancy Code

### //PROGRAM FOR CYCLIC REDUNDENCY CHECK

```
int gen[4],genl,frl,rem[4];void
main()
{
int
i,j,fr[8],dupfr[11],recfr[11],tlen
,flag;clrscr();
frl=8; genl=4; printf("enter
frame:");
for(i=0;i<frl;i++){ scanf("%d",&fr[
i]);dupfr[i]=fr[i];}
printf("enter generator:");
for(i=0;i<genl;i++)scanf("%d",&g
en[i]);tlen=frl+genl-1;
for(i=frl;i<tlen;i++){ dupfr[i]=0;
}
remainder(dupfr);
for(i=0;i<frl;i++)
{
recfr[i]=fr[i];
}
for(i=frl,j=1;j<genl;i++,j++)
```

```

{
recfr[i]=rem[j];
}
remainder(recfr);flag=0;

for(i=0;i<4;i++)
{
if(rem[i]!=0)flag++;
}

if(flag==0){
printf("frame received correctly");}
else{
printf("the received frame is wrong");}

getch();}

remainder(int fr[]){ int
k,k1,i,j; for(k=0;k<frl;k++){
if(fr[k]==1){k1=k;
for(i=0,j=k;i<genl;i++,j++)
{
rem[i]=fr[j]^gen[i];
}
for(i=0;i<genl;i++)
{
fr[k1]=rem[i];
k1++;}}}}

```

OUTPUT:

```
enter frame:1
1
1
1
1
1
1
1
1
1
1
enter generator:1
1
0
1
frame received correctly
```

### 3. Switch and Router configuration and Configuring VLAN in packet tracer.

The switch is a network device that is used to segment the networks into different sub networks called subnets or LAN segments. It is responsible for filtering and forwarding the packets between LAN segments based on the MAC address.

Steps to Configure the Switch:

**Step 1.** Open the packet tracer desktop and take a switch (PT-Switch) from the devices.



**Step 2:** Configure the Host name of the switch0.

- Click on switch0 and go to Command Line Interface.
- Then change the hostname to “sh”

**Command:**

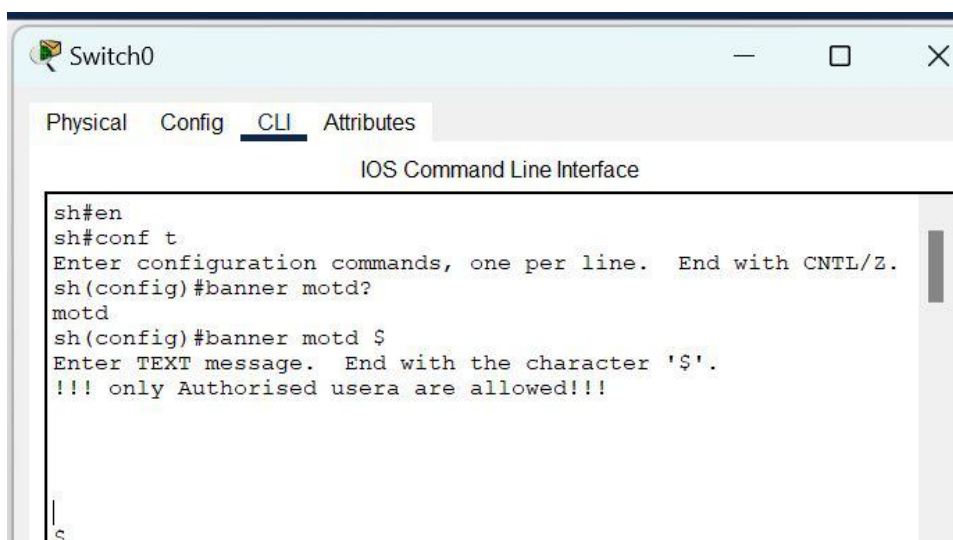
```
switch>
switch>en
switch#conf t
switch(config)#hostname sh
sh(config)#exit
```

**Step 3:** Set a message of the day (MOTD) banner for the users.

**Command:**

```
sh(config)#banner motd $
```

- Then, enter MOTD and end it with ‘\$’ to exit.





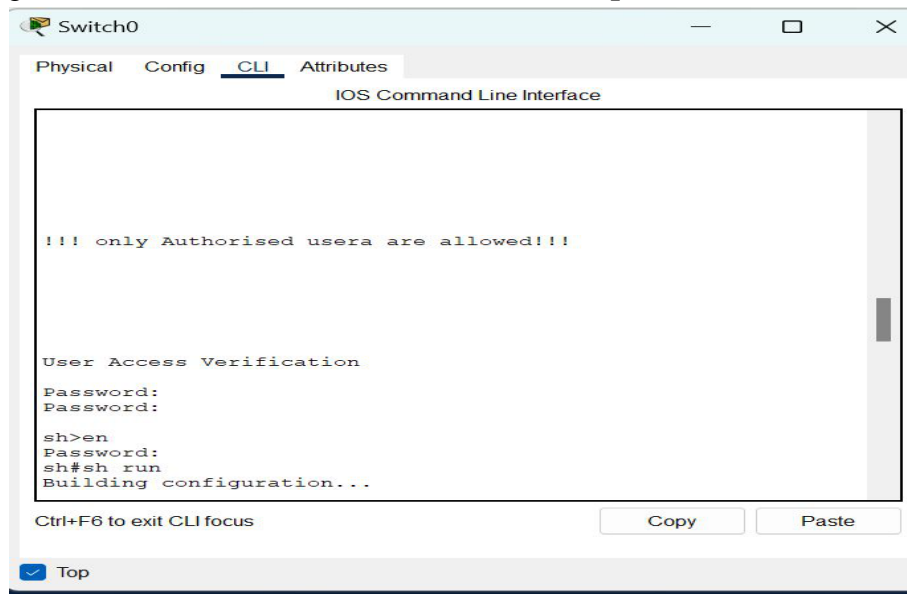
**Step 4:** Set up line control password and enable secret password.

To configure the Line Control password and Enable secret follow the below commands:

```
sh#conf t
sh(config)#
sh(config)#line con 0
sh(config-line)#password 123
sh(config-line)#login
sh(config-line)#exit
```

**Step 5:** Verify the password

- When you try to log in first, it will ask for the **line control password**.
- Then, to configure the terminal it will ask to **enable a secret password**.



To save the configuration use the below command:

**Command:**

```
sh#copy run startup-config
```

Computers are connected with routers using a copper straight-through cable. After forming the network, to check network connectivity a simple PDU is transferred from PC0 to PC1. The network simulation status is successful. From this network, it can be observed that the router handles data transfers between multiple devices.

## Procedure:

### Step-1(Configuring Router1):

1. Select the router and Open CLI.
2. Press ENTER to start configuring Router1.
3. Type enable to activate the privileged mode.

4. Type config t(configure terminal) to access the configuration menu.

5. Configure interfaces of Router1:

6. Type no shutdown to finish.

Router1 Command Line Interface:

Router>enable

Router#config t

Enter configuration commands, one per line. End with CNTL/Z.

Router(config)#interface FastEthernet0/0

Router(config-if)#ip address 192.168.10.1 255.255.255.0

Router(config-if)#no shutdown

Router(config-if)#

%LINK-5-CHANGED: Interface GigabitEthernet0/0, changed state to up

Router(config-if)#interface FastEthernet0/1

Router(config-if)#ip address 192.168.20.1 255.255.255.0

Router(config-if)#no shutdown

Step-2(Configuring PCs):

Assign IP Addresses to every PC in the network.

Select the PC, Go to the desktop and select IP Configuration and assign an IP address, Default gateway, Subnet Mask

Assign the default gateway of PC0 as 192.168.10.1.

Assign the default gateway of PC1 as 192.168.20.1.

Step-3(Connecting PCs with Router):

Connect FastEthernet0 port of PC0 with FastEthernet0/0 port of Router1 using a copper straight-through cable.

Connect FastEthernet0 port of PC1 with FastEthernet0/1 port of Router1 using a copper straight-through cable.

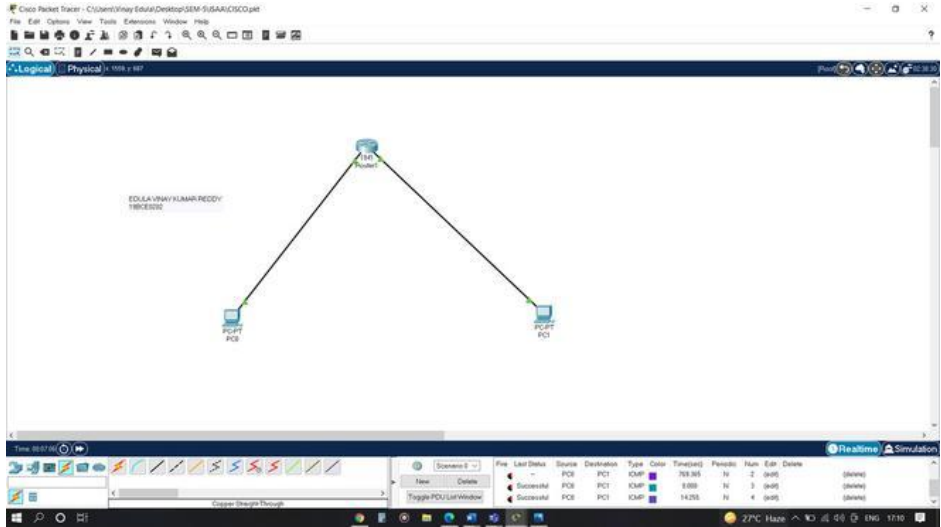
**Router Configuration Table:**

Device Name	IP address FastEthernet0/0	Subnet Mask	IP Address FastEthernet0/1	Subnet Mask
Router1	192.168.10.1	255.255.255.0	192.168.20.1	255.255.255.0

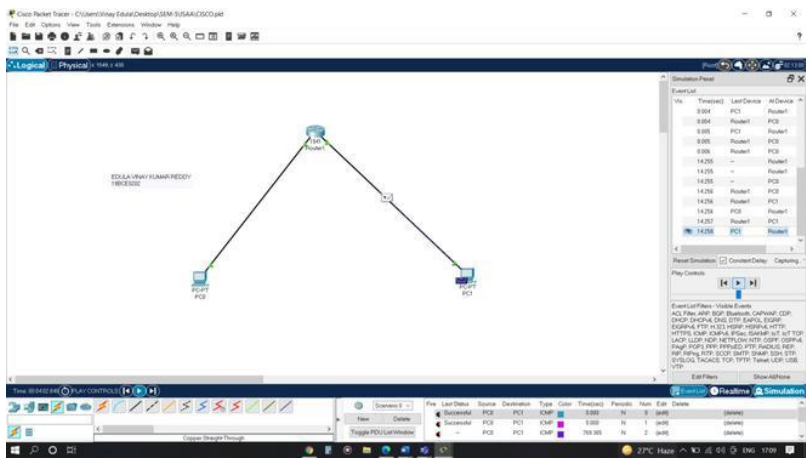
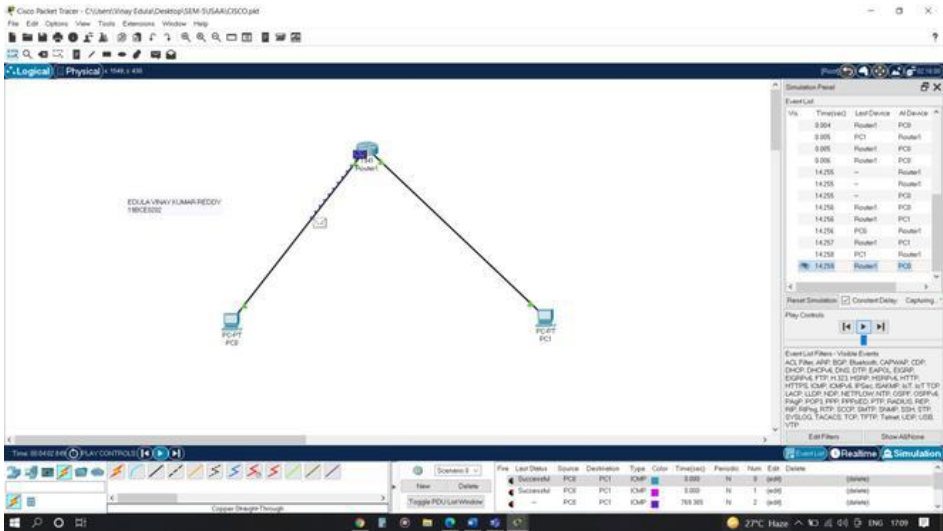
**PC Configuration Table:**

Device Name	IP address	Subnet Mask	Gateway
PC 0	192.168.10.2	255.255.255.0	192.168.10.1
PC 1	192.168.20.2	255.255.255.0	192.168.20.1

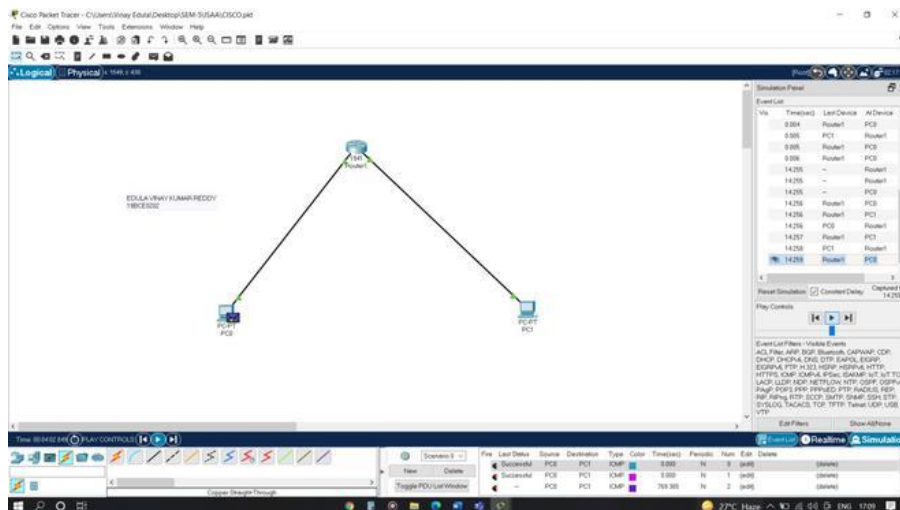
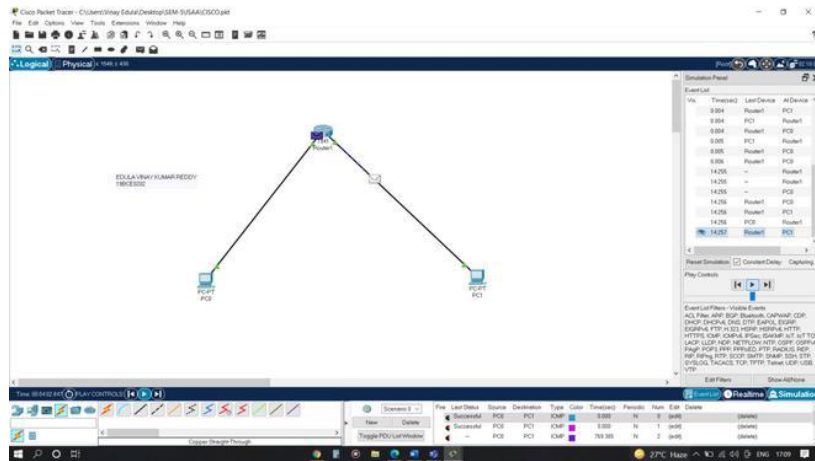
Designed Network topology:



Sending a PDU From PC0 to PC1:

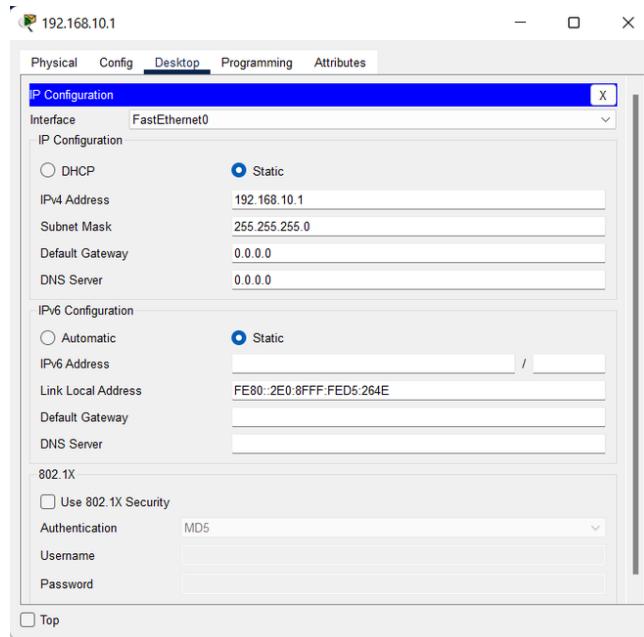


Acknowledgment from PC1 to PC0:

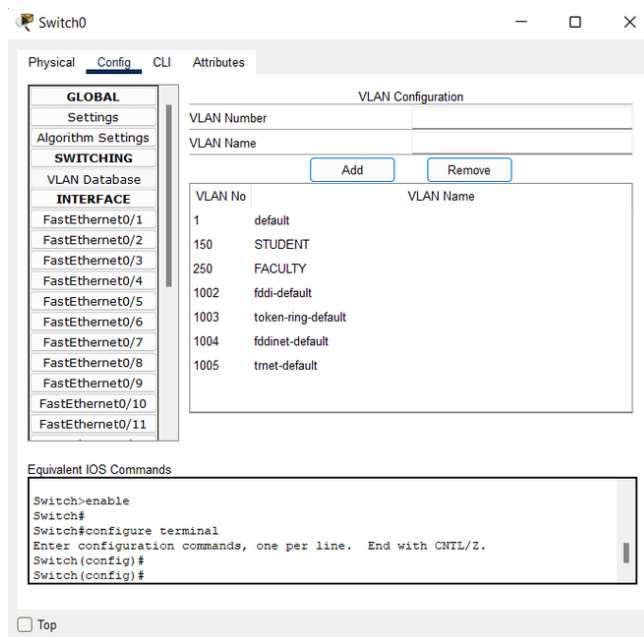


Virtual Local Area Network. This is a custom network we create from one or more existing LANs. It enables a group of devices from multiple networks (both wired and wireless) to be combined into a single Logical network. The result is a VLAN that can be administered like a physical area network. The network equipment like routers or switches must support the VLAN configurations to create a VLAN.

**Step 1:** At first, we create a LAN, LAN-A with 6 hosts. To create a LAN we need one Layer 2 switch Switch0 and 6 end devices. Now we provide IP addresses to the hosts starting from 192.168.10.1 (you can provide any valid IP addresses). To provide an IP address to a host just select that host → Desktop → IP Configuration → IPv4 Addresses and provide an IP address and then ENTER, the Subnet Mask will be provided by default.

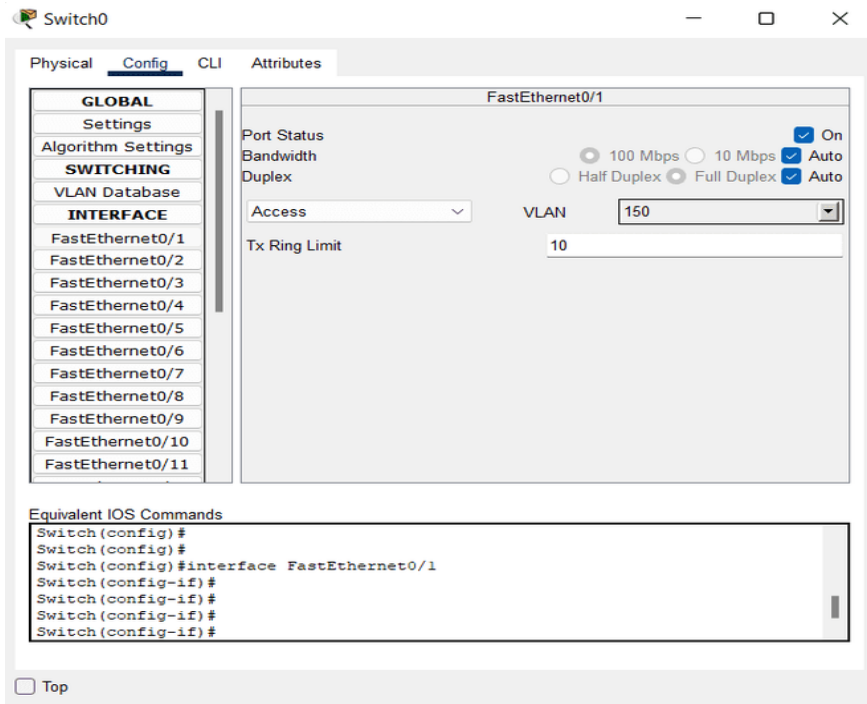


**Step 2:** Let us create 2 VLANs where the name of the first VLAN is VLAN-STUDENT and the second VLAN is VLAN-FACULTY. To configure VLANs we have to go to the switch Switch0 and move to Config → SWITCHING → VLAN Database. Now let us take the VLAN Number for STUDENT is 150 and for FACULTY is 250 and add these numbers to VLAN Database.

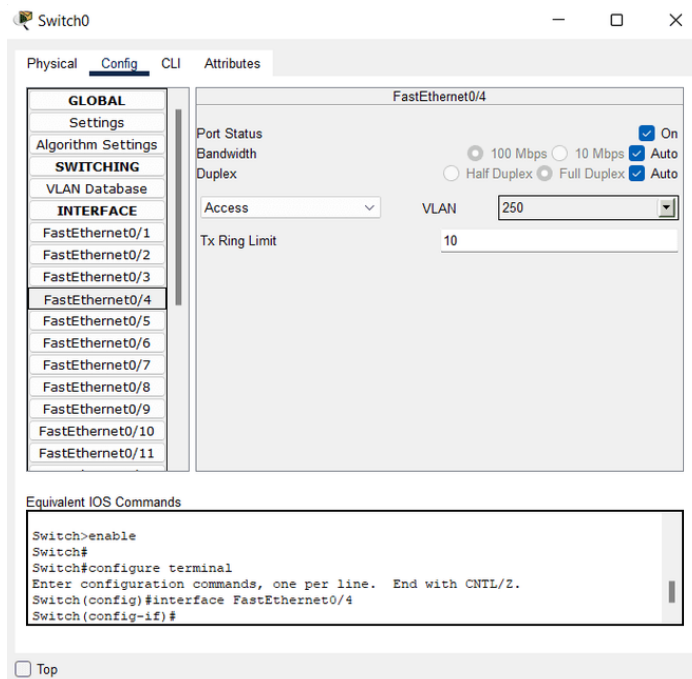


**Step 3:** Next we have to select the hosts under VLAN-STUDENT. Here I have put hosts with IP addresses from 192.168.10.1 to 192.168.10.3 under VLAN-STUDENT. To do so we have to select the switch Switch0 → Config → INTERFACE, here we choose FastEthernet0/1 corresponding to the host 192.168.10.1 which we consider to be in VLAN-STUDENT. Now we select the down arrow beside VLAN and select 150:STUDENT, which is for student VLAN.

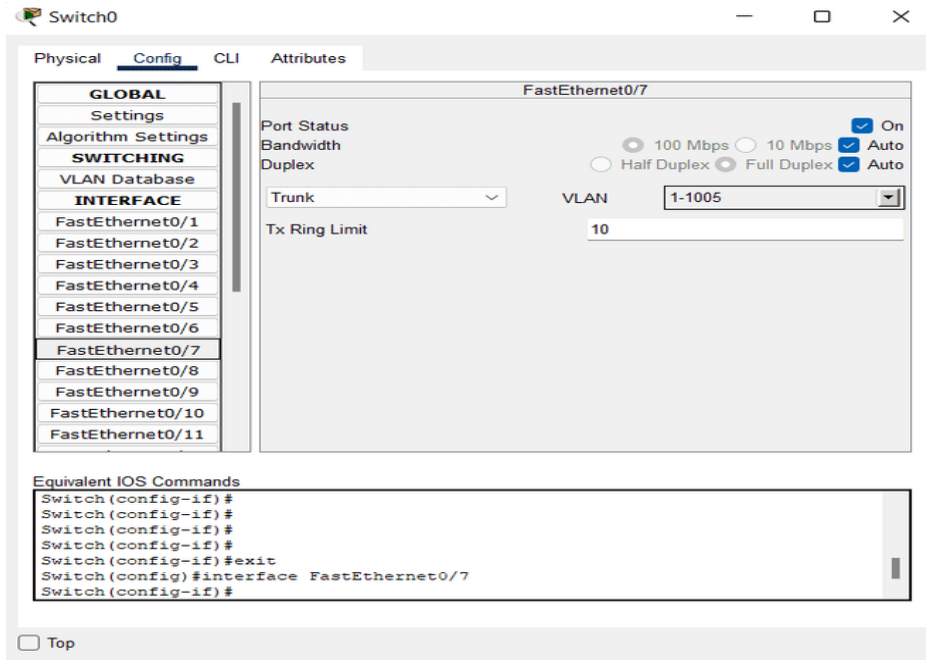
Similarly, we do this same process for FastEthernet0/2 and FastEthernet0/3.



**Step 4:** Now we have to configure the hosts under VLAN-FACULTY. Here I have put hosts with IP addresses 192.168.10.4 to 192.168.10.6 under VLAN-FACULTY. To do so, just follow the process mentioned in Step 3, but instead of selecting the VLAN Number 150:STUDENT, select 250:FACULTY for FastEthernet0/4, FastEthernet0/5, and FastEthernet0/6.

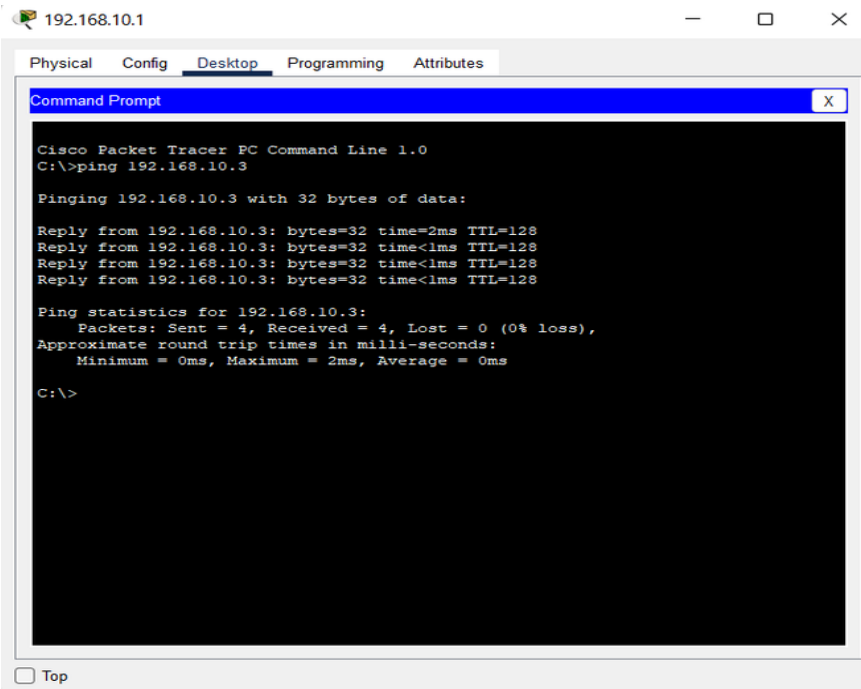


**Step 5:** Lastly, just change the switch port mode from Access to Trunk for FastEthernet0/7.



Now our VLAN configuration is ready, and we can check this by sending data packets from one host to another under LAN-A. Let us ping from 192.168.10.1 to 192.168.10.3. To do so, we have to select the host with IP 192.168.10.1 and then select Desktop → Command Prompt. Now run the following command to ping 192.168.10.3.

ping 192.168.10.3



#### 4. Implement Dijkstra's algorithm to compute the Shortest path through a graph.

Dijkstra's algorithm is an algorithm for finding the shortest paths between nodes in a graph, which may represent, for example, road networks. The algorithm exists in many variants; Dijkstra's original variant found the shortest path between two nodes,[3] but a more common variant fixes a single node as the "source" node and finds shortest paths from the source to all other nodes in the graph, producing a shortest-path tree.

```
#include<stdio.h>

#include<conio.h>

#define INFINITY 9999

#define MAX 10

void dijkstra(int G[MAX][MAX],int n,int startnode);

int main()

{

int G[MAX][MAX],i,j,n,u;

printf("Enter no. of vertices:");

scanf("%d",&n);

printf("\nEnter the adjacency matrix:\n");

for(i=0;i<n;i++)

for(j=0;j<n;j++)

scanf("%d",&G[i][j]);

printf("\nEnter the starting node:");

scanf("%d",&u);

dijkstra(G,n,u);

return 0;

}

void dijkstra(int G[MAX][MAX],int n,int startnode)

{

int cost[MAX][MAX],distance[MAX],pred[MAX];

int visited[MAX],count,mindistance,nextnode,i,j;
```



```

//pred[] stores the predecessor of each node

//count gives the number of nodes seen so far

//create the cost matrix

for(i=0;i<n;i++)

for(j=0;j<n;j++)

if(G[i][j]==0)

cost[i][j]=INFINITY;

else

cost[i][j]=G[i][j];

//initialize pred[],distance[] and visited[]

for(i=0;i<n;i++)

{

distance[i]=cost[startnode][i];

pred[i]=startnode;

visited[i]=0;

}

distance[startnode]=0;

visited[startnode]=1;

count=1;

while(count<n-1)

{

mindistance=INFINITY;

//nextnode gives the node at minimum distance

for(i=0;i<n;i++)

if((distance[i]<mindistance&&!visited[i])

{

```

```

mindistance=distance[i];
nextnode=i;
}
//check if a better path exists through nextnode
visited[nextnode]=1;
for(i=0;i<n;i++)
if(!visited[i])
if(mindistance+cost[nextnode][i]<distance[i])
{
distance[i]=mindistance+cost[nextnode][i];
pred[i]=nextnode;
}
count++;
}

//print the path and distance of each node
for(i=0;i<n;i++)
if(i!=startnode)
{
printf("\nDistance of node%d=%d",i,distance[i]);
printf("\nPath=%d",i);
j=i;
do
{
j=pred[j];
printf("<-%d",j);
}while(j!=startnode);

```

}

}

OUTPUT:

```
Enter no. of vertices:5
Enter the adjacency matrix:
0 10 0 30 100
10 0 50 0 0
0 50 0 20 10
30 0 20 0 60
100 0 10 60 0
Enter the starting node:0
Distance of node 1=10
Path=1<-0
Distance of node 2=50
Path=2<-3<-0
Distance of node 3=30
Path=3<-0
Distance of node 4=60
Path=4<-2<-3<-0
Process returned 5 (0x5)    execution time : 47.471 s
Press any key to continue.
```

**5. Take an example subnet graph with weights indicating delay between nodes. Now obtain Routing table at each node using distance vector routing algorithm.**

Dijkstra algorithm is also called the single source shortest path algorithm. It is based on the greedy technique. The algorithm maintains a list visited[ ] of vertices, whose shortest distance from the source is already known.

If visited[i], equals 1, then the shortest distance of vertex i is already known. Initially, visited[i] is marked as, for source vertex. At each step, we mark visited[v] as 1. Vertex v is a vertex at the shortest distance from the source vertex. At each step of the algorithm, the shortest distance of each vertex is stored in an array distance[ ].

```
#include<stdio.h>
#include<conio.h>
#define INFINITY 9999
#define MAX 10

void dijkstra(int G[MAX][MAX],int n,int startnode);

int main()
{
    int G[MAX][MAX],i,j,n,u;
    printf("Enter no. of vertices:");
    scanf("%d",&n);
    printf("\nEnter the adjacency matrix:\n");
    for(i=0;i<n;i++)
        for(j=0;j<n;j++)
            scanf("%d",&G[i][j]);
    printf("\nEnter the starting node:");
    scanf("%d",&u);
    dijkstra(G,n,u);
    return 0;
}

void dijkstra(int G[MAX][MAX],int n,int startnode)
{
    int cost[MAX][MAX],distance[MAX],pred[MAX];
    int visited[MAX],count,mindistance,nextnode,i,j;
    //pred[] stores the predecessor of each node
    //count gives the number of nodes seen so far
    //create the cost matrix
    for(i=0;i<n;i++)
        for(j=0;j<n;j++)
            if(G[i][j]==0)
                cost[i][j]=INFINITY;
            else
                cost[i][j]=G[i][j];
```

```

//initialize pred[],distance[] and visited[]
for(i=0;i<n;i++)
{
distance[i]=cost[startnode][i];
pred[i]=startnode;
visited[i]=0;
}
distance[startnode]=0;
visited[startnode]=1;
count=1;
while(count<n-1)
{
mindistance=INFINITY;
//nextnode gives the node at minimum distance
for(i=0;i<n;i++)
if(distance[i]<mindistance&&!visited[i])
{
mindistance=distance[i];
nextnode=i;
}
//check if a better path exists through nextnode
visited[nextnode]=1;
for(i=0;i<n;i++)
if(!visited[i])
if(mindistance+cost[nextnode][i]<distance[i])
{
distance[i]=mindistance+cost[nextnode][i];
pred[i]=nextnode;
}
count++;
}

//print the path and distance of each node
for(i=0;i<n;i++)
if(i!=startnode)
{
printf("\nDistance of node%d=%d",i,distance[i]);
printf("\nPath=%d",i);
j=i;
do
{
j=pred[j];
printf("<-%d",j);
}while(j!=startnode);
}
}

```

OUTPUT:

```
Enter no. of vertices:5
Enter the adjacency matrix:
0 10 0 30 100
10 0 50 0 0
0 50 0 20 10
30 0 20 0 60
100 0 10 60 0
Enter the starting node:0
Distance of node 1=10
Path=1<-0
Distance of node 2=50
Path=2<-3<-0
Distance of node 3=30
Path=3<-0
Distance of node 4=60
Path=4<-2<-3<-0
Process returned 5 (0x5)    execution time : 47.471 s
Press any key to continue.
```

## 6. Take an example subnet of hosts. Obtain broadcast tree for it.

This technique is widely used because it is simple and easy to understand. The idea of this algorithm is to build a graph of the subnet with each node of the graph representing a router and each arc of the graph representing a communication line. To choose a route between a given pair of routers the algorithm just finds the broadcast between them on the graph.

```
#include<stdio.h>
int a[10][10],n;
void main()
{
int i,j,root;
clrscr();
printf("Enter no.of nodes:");
scanf("%d",&n);
printf("Enter adjacent matrix\n");
for(i=1;i<=n;i++)
for(j=1;j<=n;j++)
{
printf("Enter connecting of %d-->%d: ",i,j);
scanf("%d",&a[i][j]);
}
printf("Enter root node:");
scanf("%d",&root);
adj(root);
}
adj(int k)
{
int i,j;
printf("Adjacent node of root node:\n");
printf("%d\n",k);
for(j=1;j<=n;j++)
{
if(a[k][j]==1 || a[j][k]==1)
printf("%d\t",j);
}
printf("\n");
for(i=1;i<=n;i++)
{
if((a[k][j]==0) && (a[i][k]==0) && (i!=k))
printf("%d",i);
}
}
```

OUTPUT:

```
Enter no.of nodes:5
Enter adjacent matrix Enter connecting of 1-->1::0
Enter connecting of 1-->2::1
Enter connecting of 1-->3::1
```

Enter connecting of 1-->4::0  
Enter connecting of 1-->5::0  
Enter connecting of 2-->1::1  
Enter connecting of 2-->2::0  
Enter connecting of 2-->3::1  
Enter connecting of 2-->4::1  
Enter connecting of 2-->5::0  
Enter connecting of 3-->1::1  
Enter connecting of 3-->2::1  
Enter connecting of 3-->3::0  
Enter connecting of 3-->4::0  
Enter connecting of 3-->5::0  
Enter connecting of 4-->1::0  
Enter connecting of 4-->2::1  
Enter connecting of 4-->3::0  
Enter connecting of 4-->4::0  
Enter connecting of 4-->5::1  
Enter connecting of 5-->1::0  
Enter connecting of 5-->2::0  
Enter connecting of 5-->3::0  
Enter connecting of 5-->4::1  
Enter connecting of 5-->5::0  
Enter root node:2  
Adjacent node of root node:: 2  
1 3 4  
5



## 7. Implementation of Connection oriented concurrent service (TCP).

If we are creating a connection between client and server using TCP then it has a few functionalities like, TCP is suited for applications that require high reliability, and transmission time is relatively less critical. It is used by other protocols like HTTP, HTTPs, FTP, SMTP, Telnet. TCP rearranges data packets in the order specified. There is absolute guarantee that the data transferred remains intact and arrives in the same order in which it was sent. TCP does Flow Control and requires three packets to set up a socket connection before any user data can be sent. TCP handles reliability and congestion control. It also does error checking and error recovery. Erroneous packets are retransmitted from the source to the destination.

### SOURCE CODE:

#### SERVER:

```
#include<stdio.h>
#include<netinet/in.h>
#include<netdb.h>
#define SERV_TCP_PORT 5035
int main(int argc,char**argv)
{
    int sockfd,newsockfd,clength;
    struct sockaddr_in serv_addr,cli_addr;
    char buffer[4096];
    sockfd=socket(AF_INET,SOCK_STREAM,0);
    serv_addr.sin_family=AF_INET;
    serv_addr.sin_addr.s_addr=INADDR_ANY;
    serv_addr.sin_port=htons(SERV_TCP_PORT);
    printf("\nStart");
    bind(sockfd,(struct sockaddr*)&serv_addr,sizeof(serv_addr));
    printf("\nListening...");
    printf("\n");
    listen(sockfd,5);
    clength=sizeof(cli_addr);
    newsockfd=accept(sockfd,(struct sockaddr*)&cli_addr,&clength);
    printf("\nAccepted");
    printf("\n");
    read(newsockfd,buffer,4096);
    printf("\nClient message:%s",buffer);
    write(newsockfd,buffer,4096);
    printf("\n");
    close(sockfd);
    return 0;
}
```

#### CLIENT:

```
#include<stdio.h>
#include<sys/types.h>
#include<sys/socket.h>
#include<netinet/in.h>
#include<netdb.h>
#define SERV_TCP_PORT 5035
int main(int argc,char*argv[])
{
    int sockfd;
```


```

struct sockaddr_in serv_addr;
struct hostent *server;
char buffer[4096];
sockfd=socket(AF_INET,SOCK_STREAM,0);
serv_addr.sin_family=AF_INET;
serv_addr.sin_addr.s_addr=inet_addr("127.0.0.1");
serv_addr.sin_port=htons(SERV_TCP_PORT);
printf("\nReady for sending...");
connect(sockfd,(struct sockaddr*)&serv_addr,sizeof(serv_addr));
printf("\nEnter the message to send\n");
printf("\nClient: ");
fgets(buffer,4096,stdin);
write(sockfd,buffer,4096);
printf("Serverecho:%s",buffer);
printf("\n");
close(sockfd);
return 0;
}

```

## OUTPUT:

### SERVER:




```

Telnet 172.16.5.90
[student@localhost Jebastin]$ cc echoserver.c
[student@localhost Jebastin]$ ./a.out

Start
Listening...
Accepted
Client Message: Praise the Lord
[student@localhost Jebastin]$

```

### CLIENT:



```

Telnet 172.16.5.90
[student@localhost Jebastin]$ cc echoclient.c
[student@localhost Jebastin]$ ./a.out

Ready for Send
Enter the message to send
Client: Praise the Lord
Server Echo: Praise the Lord
[student@localhost Jebastin]$

```

## 8. Implementation of Connectionless Iterative time service (UDP).

UDP is widespread on the Internet for time-sensitive transmissions like DNS lookups and video playback. By not explicitly establishing a connection before data is sent, it speeds up communications. This enables the delivery of data incredibly quickly, but it can also result in packets being lost in transit, opening up the potential for DDoS assaults and other forms of exploitation. **UDP** is a defined procedure for transferring data between two computers connected by a network, like all networking protocols. In contrast to other protocols, UDP completes this task in a straightforward manner: it delivers packets (units of data transmission) straight to the destination computer without first establishing a connection, specifying the order of such packets, or verifying that they arrived as intended. The term "datagram" is used to describe UDP packets.

*/ 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

// Driver code
int main() {
    int sockfd;
    char buffer[MAXLINE];
    char *hello = "Hello from server";
    struct sockaddr_in servaddr, cliaddr;

    // Creating socket file descriptor
    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));

    // Filling server information
    servaddr.sin_family = AF_INET; // IPv4
    servaddr.sin_addr.s_addr = INADDR_ANY;
    servaddr.sin_port = htons(PORT);

    // Bind the socket with the server address
    if ( bind(sockfd, (const struct sockaddr *)&servaddr,
              sizeof(servaddr)) < 0 )
    {
        perror("bind failed");
```

```

        exit(EXIT_FAILURE);
    }

    int len, n;

    len = sizeof(cliaddr); //len is value/result

    n = recvfrom(sockfd, (char *)buffer, MAXLINE,
                  MSG_WAITALL, ( struct sockaddr *) &cliaddr,
                  &len);

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

    return 0;
}
// Client 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

// Driver code
int main() {
    int sockfd;
    char buffer[MAXLINE];
    char *hello = "Hello from client";
    struct sockaddr_in servaddr;

    // Creating socket file descriptor
    if ( (sockfd = socket(AF_INET, SOCK_DGRAM, 0)) < 0 ) {
        perror("socket creation failed");
        exit(EXIT_FAILURE);
    }

    memset(&servaddr, 0, sizeof(servaddr));

    // Filling server information
    servaddr.sin_family = AF_INET;

```

```

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 sent.\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;
}

```

**Output:**

\$ ./server

Client: Hello from the client

Hello, message sent.

\$ ./client

Hello, message sent.

Server : Hello from the server

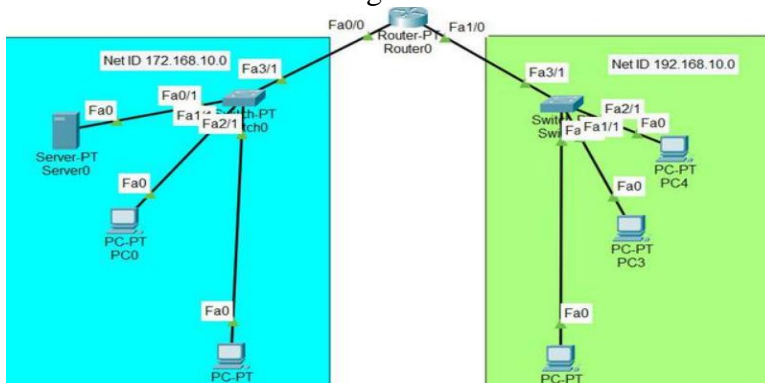
## 9. Configuration of DHCP Server and Implementation of DNS.

DHCP is a network management protocol used in networks to dynamically assign IP addresses and other network configuration information like default gateway, mask, DNS server address, etc. It is an application layer protocol.

**Step 1:** First, open the packet tracer desktop and select the devices given below:

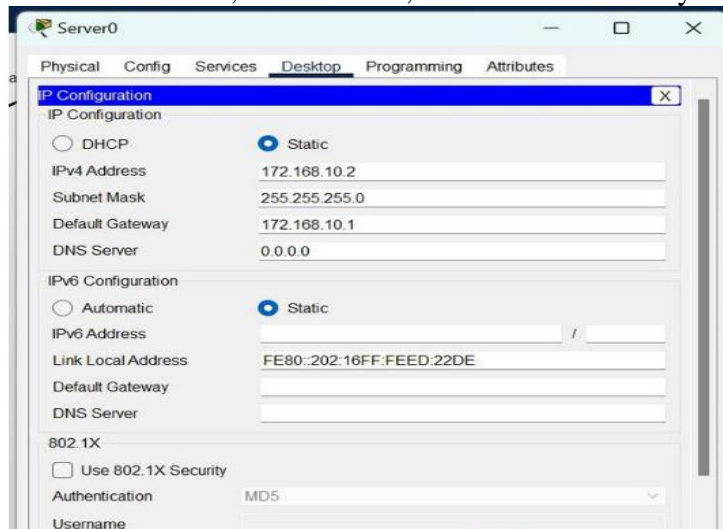
S.NO	Device	Model-Name	Unit
1.	PC	PC	5
2.	Switch	PT-Switch	2
3.	Router	PT-Router	1
4.	Server	Server-PT	1

- Now create a network topology as shown below the image.
- Use an Automatic connecting cable to connect the devices with others.



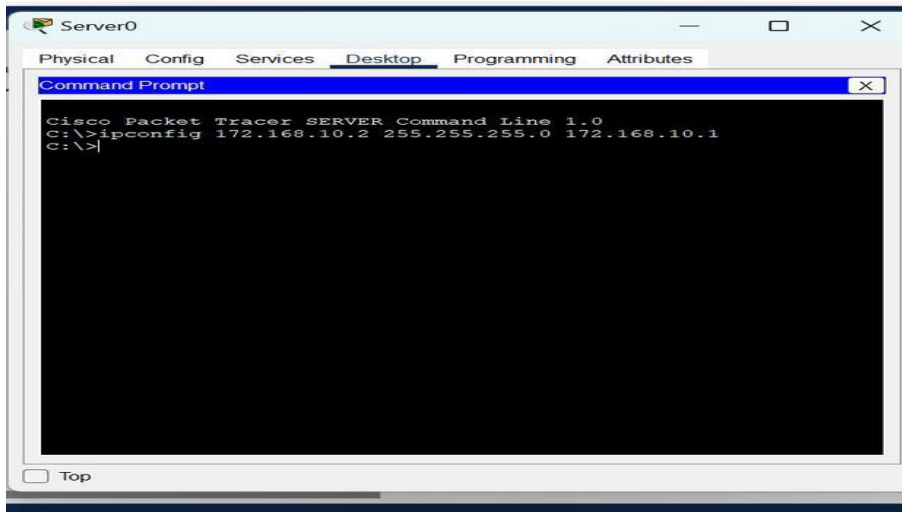
**Step 2:** Configure the Server with IPv4 address and Subnet Mask according to the Data given above.

- To assign an IP address in Server, click on Server-PT.
- Then, go to desktop and IP configuration and there you will find IPv4 configuration.
- Add IPv4 address, subnet mask, and Default Gateway.



2. Assigning IP address using the ipconfig command.

- We can also assign an IP address with the help of a command.
- Go to the command prompt of the server
- Then, type ipconfig <IPv4 address><subnet mask><default gateway>(if needed)

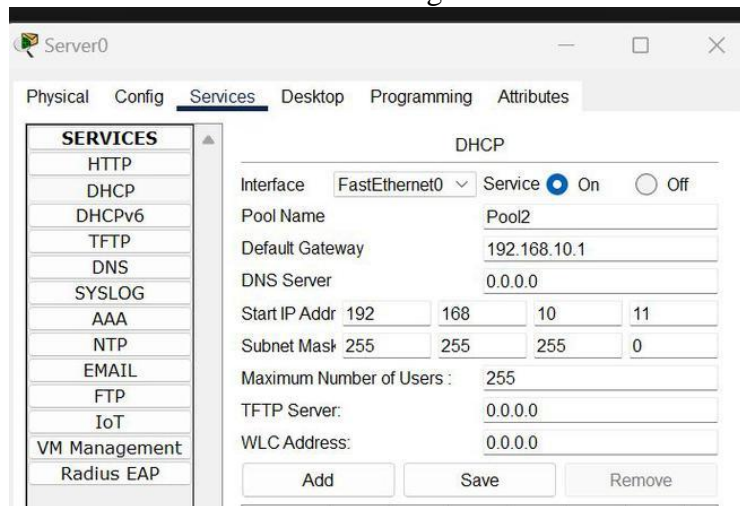


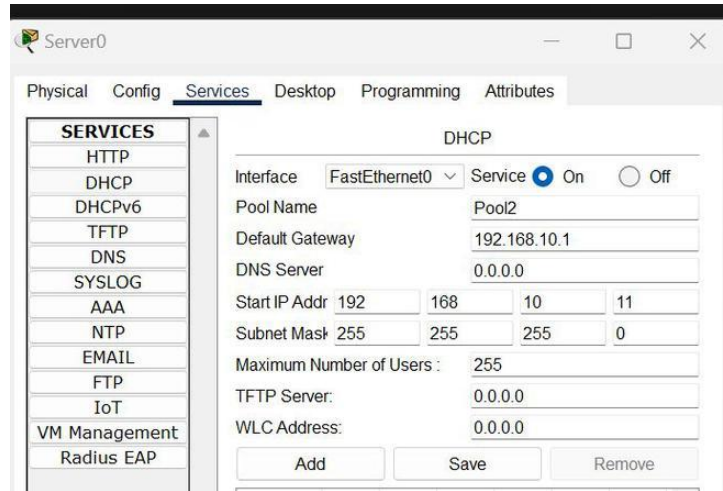
### Step 3: Configuring the DHCP server.

To configure the DHCP server first,

- Click on Server then, Go to services.
- Click on DHCP and turn on the services and, configure the DHCP server with the help of the data given below.
  - Delete the default values of Start IP Address and subnet Mask then save the info.
  - Create two new pools.

POOL1 and POOL2 and fill the data as shown in the images below.





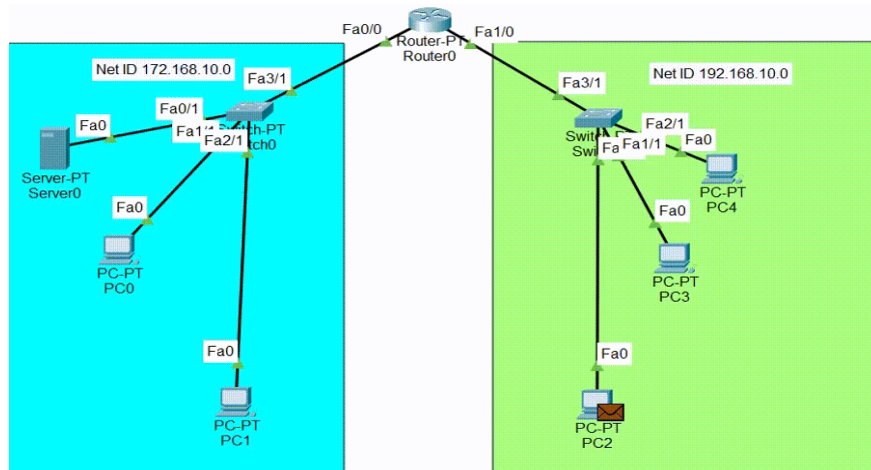
**Step 4:** Configuring Router with IPv4 Address and Subnet Mask.

- To assign an IP address in router0, click on router0.
- Then, go to config and then Interfaces, and make sure to turn on the ports.
- Then, configure the IP address in FastEthernet according to IP addressing Table.
- Fill IPv4 address and subnet mask.

**Step 5:** Configuring the PCs and changing the IP configuration.

- To assign an IP address in PC0, click on PC0.
- Then, go to desktop and IP configuration and there you will find IPv4 configuration.
- Change its state from static to DHCP.
- It will automatically fetch the data and configure itself.
- Repeat the same procedure with other PCs to configure them thoroughly.

OUTPUT:

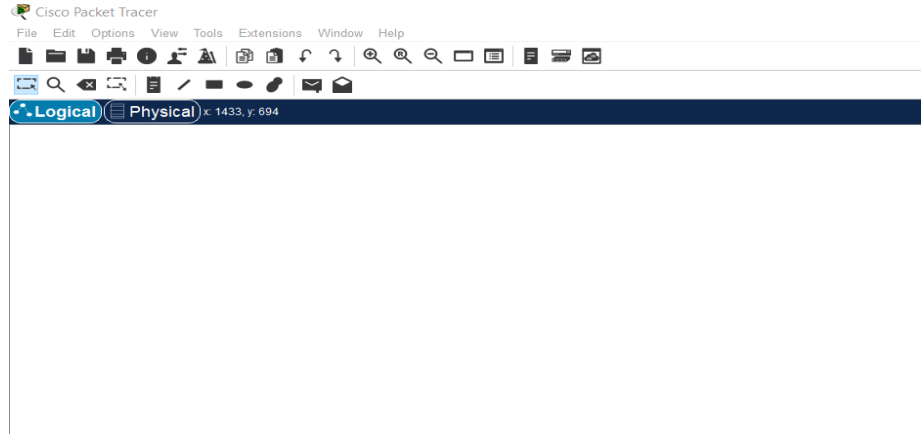




## 10. Implementation of HTTP (Hyper Text Transfer Protocol).

A web server is like a computer that uses an HTTP (Hyper Text Transfer Protocol) and many other protocols. it responds when a client makes a request over the World Wide Web. The main work of the web server is to show website content that is processed, and stored, in the web server to deliver the web pages to the user. The Web server also uses SMTP (Simple Mail Transfer Protocol) for sending mail and FTP (File Transfer Protocol) for file transfer and storage.

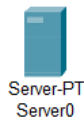
**Step 1:** After Installation Open Packet Tracer.



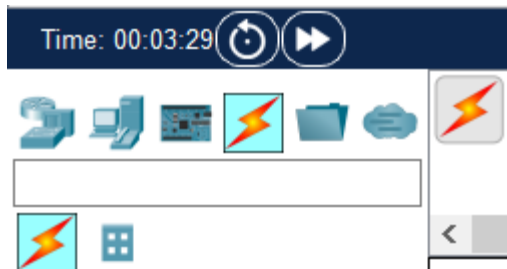
**Step 2: Implementation–** Click End Devices and Then Select PC The Drag Into them In the Screen.



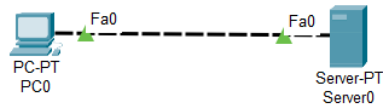
**Step 3:** Again Click End Devices and after that click on a server. Drag Into Screen



**Step 4:** Connect each other with a wire for this click-on connection.

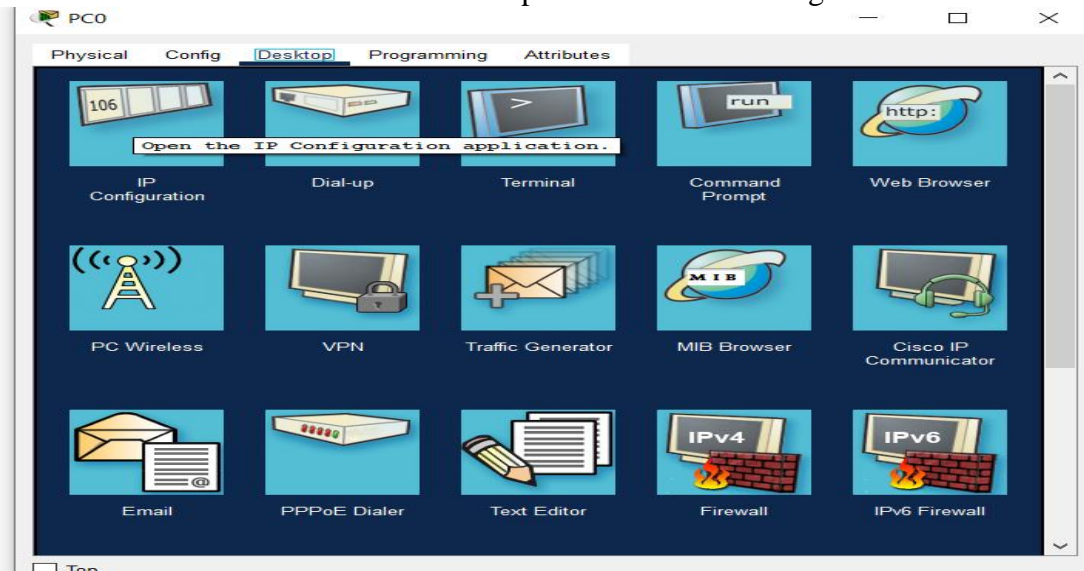


**Step 5:** After Click On this Click on the PC and other hand click on a server. Like This

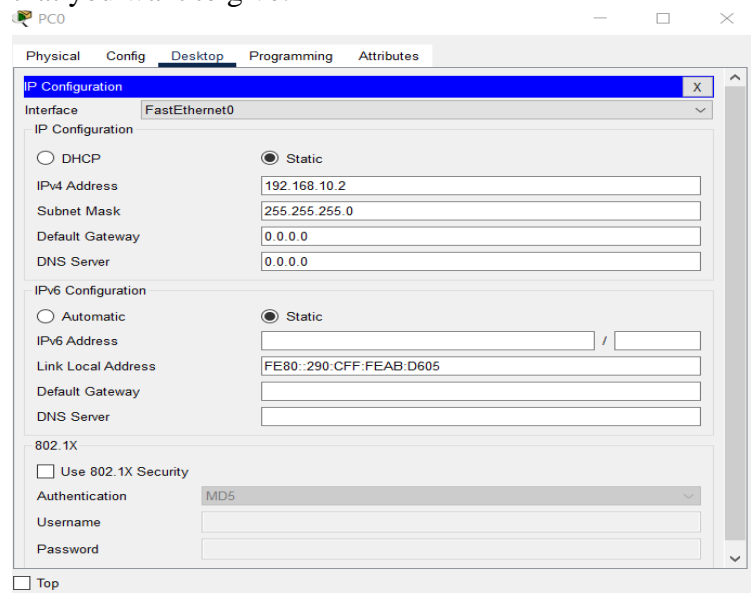


## Configuration:

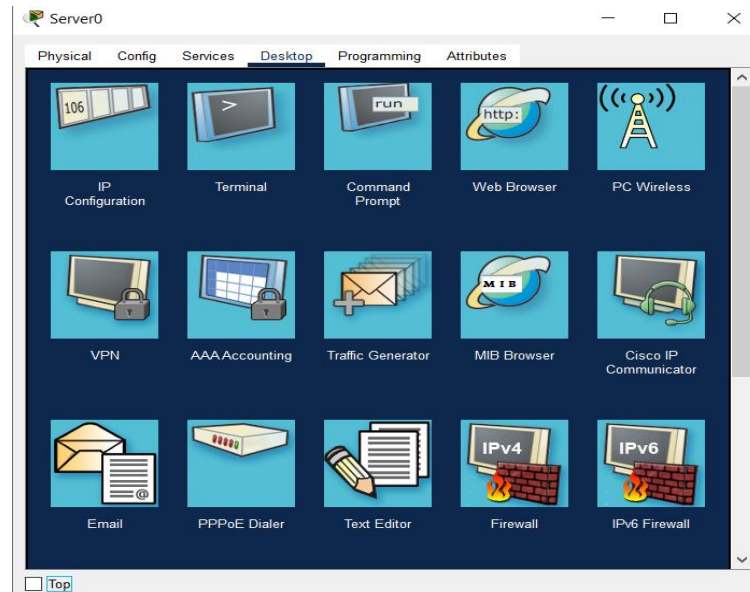
**Step 1:** Double-Click On **PC0** and Click on Desktop Then Go to IP Configuration



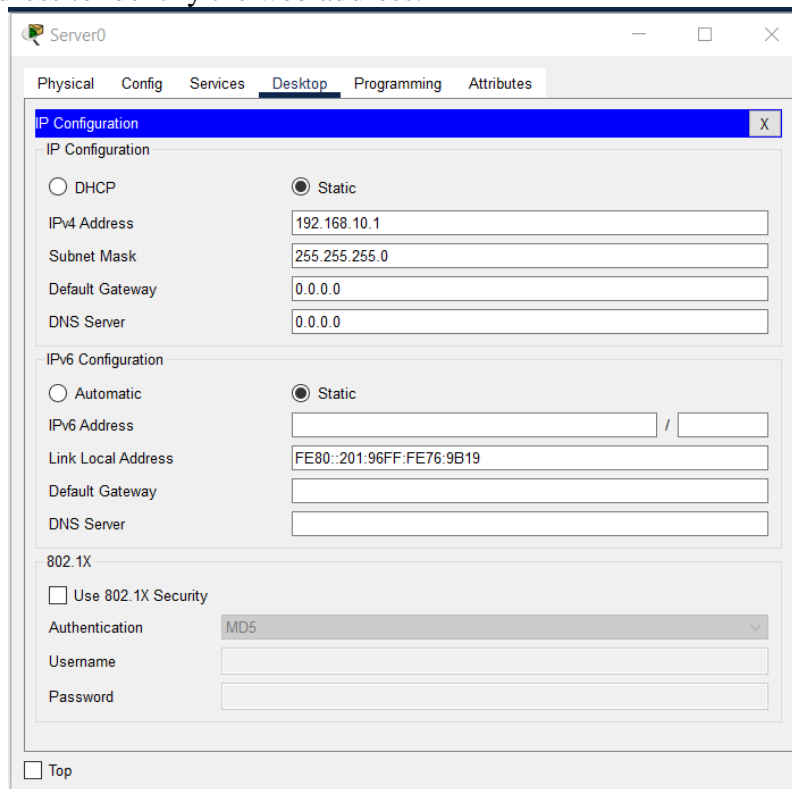
**Step 2:** Then Set the IP that you want to give.



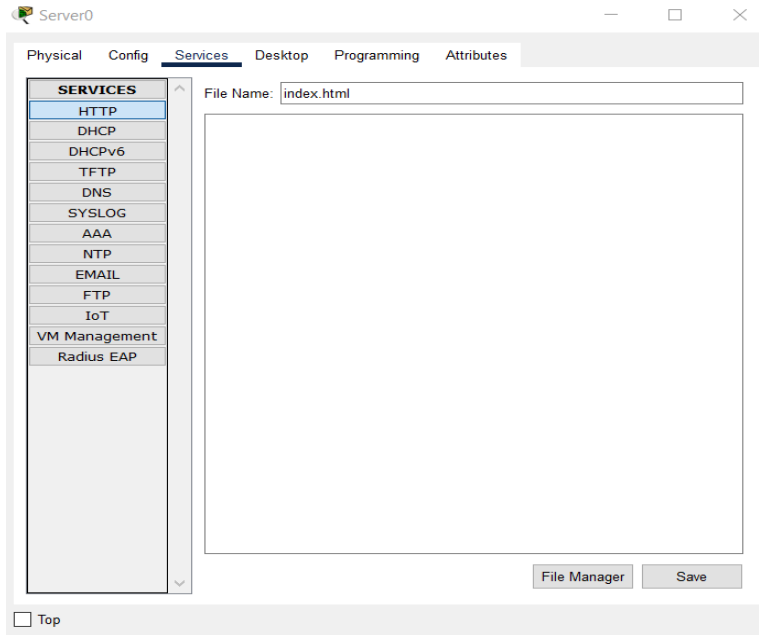
**Step 3:** Double-Click on Server0. Then Click on IP Configuration and Set the IP for the web server.



**Step 4:** Set the IP Address to identify the web address.



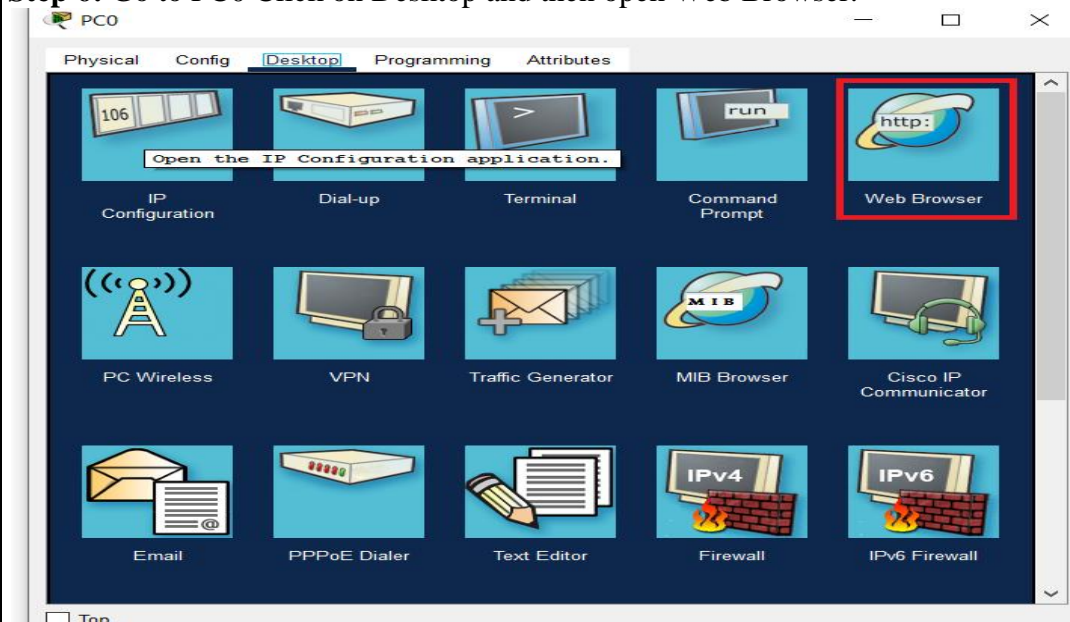
**Step 5:** Now go to Services and add some HTML code to check whether the server is working or not



```
<!DOCTYPE html>
<html>
<head>
<title>Page Title</title>
</head>
<body>
<h2>Welcome To GFG</h2>
<p>Default code has been loaded into the Editor.</p>
</body>
</html>
```

Add this code and save it

**Step 6:** Go to PC0 Click on Desktop and then open Web Browser.



**Step 7:** Remember the IP that we are given to the web server enter the same IP to the address bar. Click on go