Date 3/11/22
Page 1

Lab: week 0

Interface Overview:

Interface has 10 components

1. Menu Bar, main tool bar, Common tol bar, Logical/ Physical Workspace and Navigation Bar, Workspace, Realtime/Simulation Bar, Network Component Bar Device-Type Selection Box, User Created Packet Windows

It has 2 workspaces and 2 modes, In logical you can build your network and in simulation you can run controlled networking.

You can change the settings according to your preference; you can toggle between animation, sounds, show link lights etc,
In admin panel you can disable access to a particular interface such as interface tabs, interface locking etc.

Under Hide panel you can chose to show or hide Phy, config, CLI, Desktop, GUI & HTML. In fact you can change the format options. You can set User profile from the menu bar.

Incomplete
10/18/22

You can have Multiple Algorithms and you can save the packages as PKZ file to share

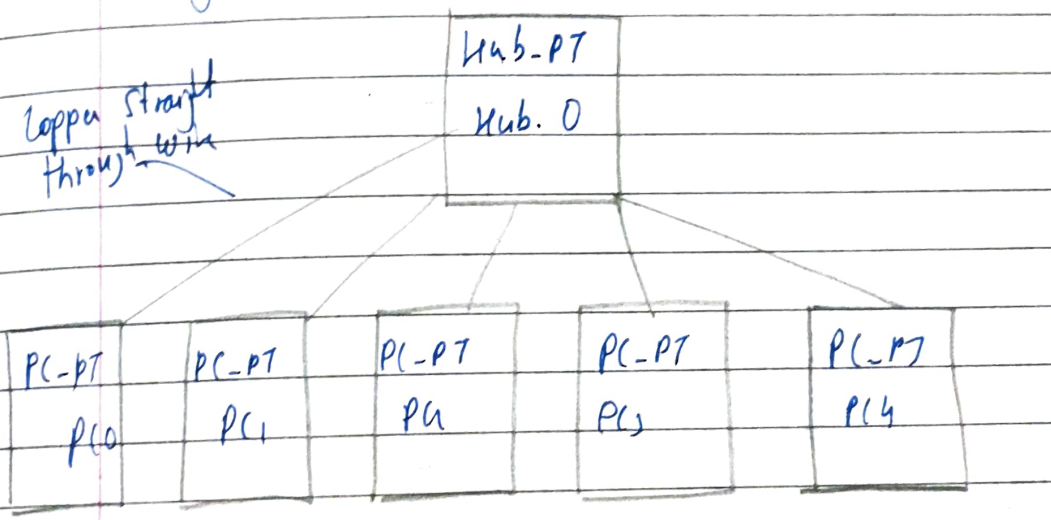
10/11/22

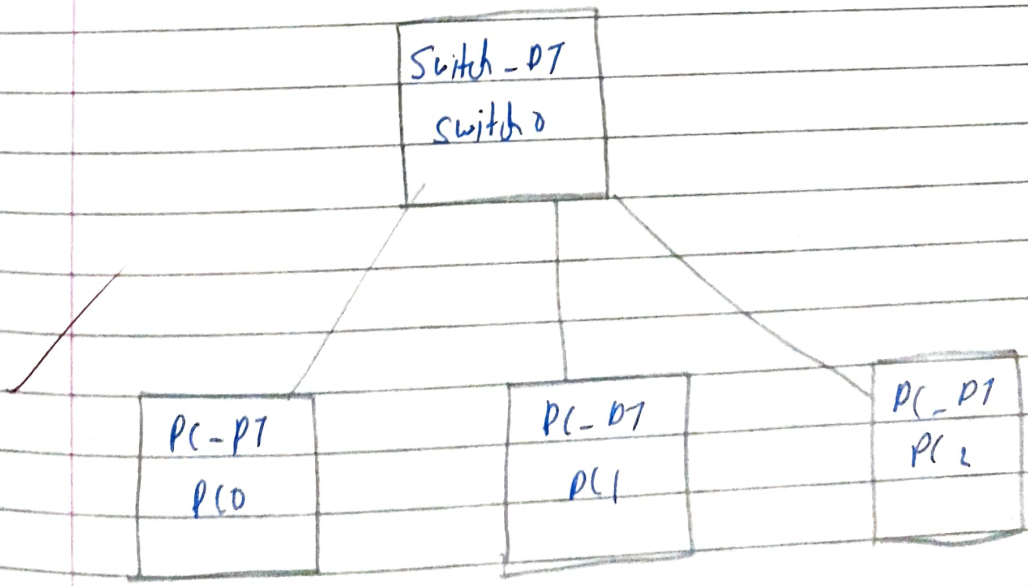
Lab week 1

Aim: Creating a Topology and simulate sending a simple PDU from source to destination using hub and switch as connecting devices.

Topology:

Using hub:





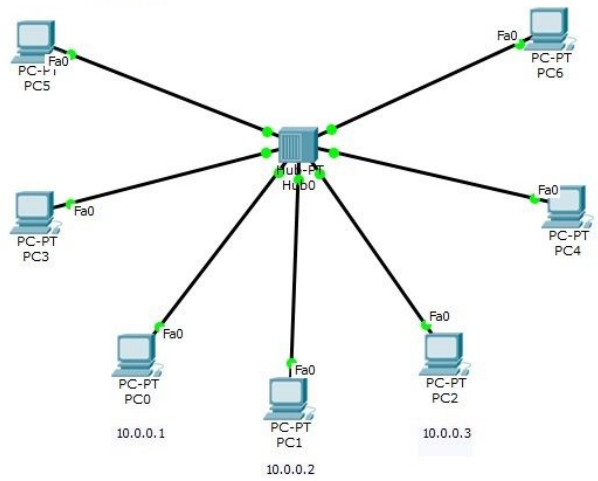
Cisco Packet Tracer Student

File Edit Options View Tools Extensions Help

Logical | [Root] | New Cluster | Move Object | Set Tiled Background | Viewport

1BM20CS068

HUB



Time: 01:40:55 | Power Cycle Devices | Fast Forward Time | Realtime

Connections

Copper Straight-Through

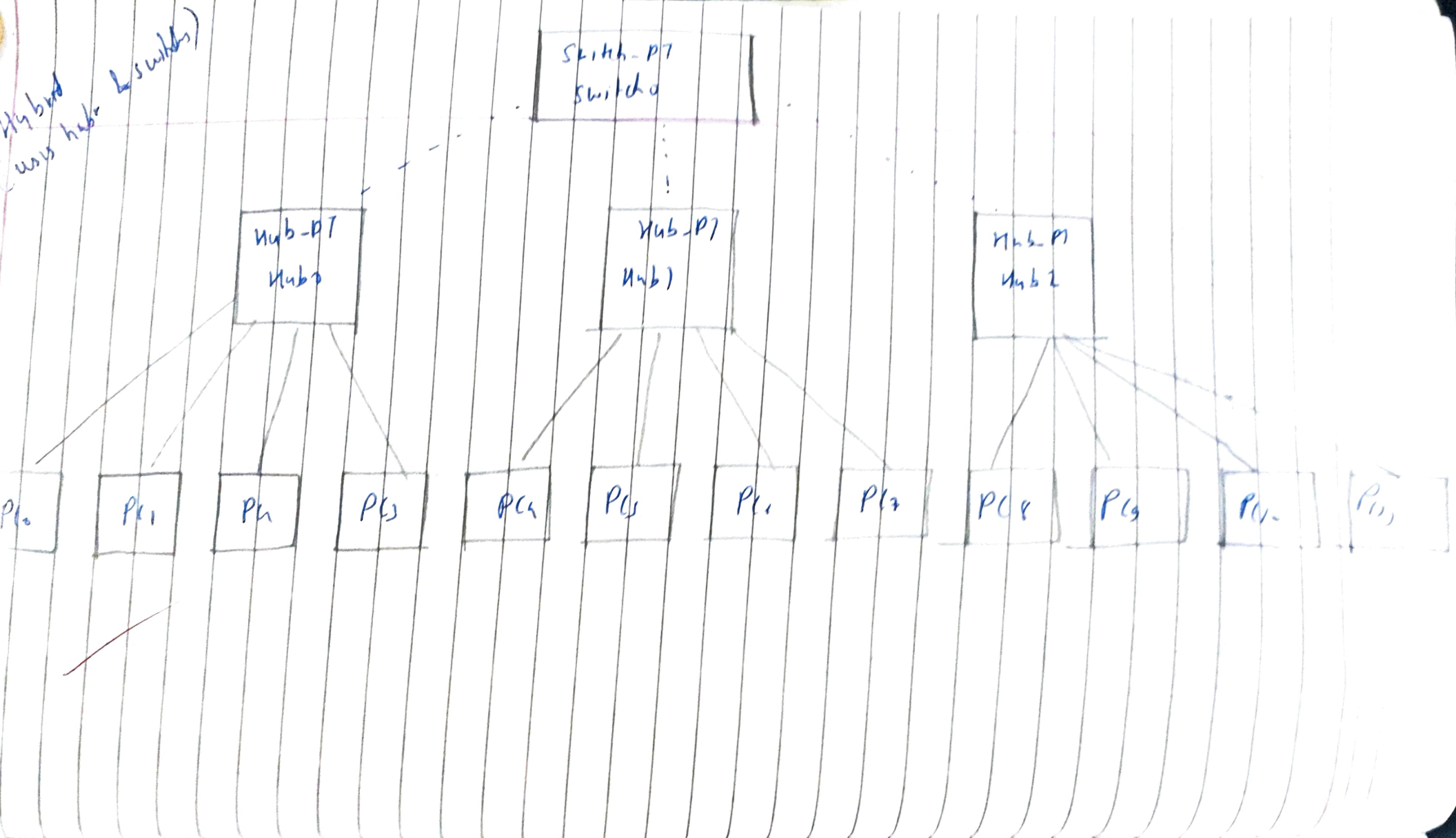
Scenario 0 | New | Delete | Toggle PDU List Window

Fire	Last Status	Source	Destination	Type	Color	Time(sec)	Periodic	Num	Edit	Delete
	Successful	PC0	PC2	ICMP		0.000	N	0	(edit)	(delete)
	Failed	PC0	PC2	ICMP		0.000	N	1	(edit)	(delete)

26°C Partly sunny

Search

06:36 10-11-2022


Switch-P7
Switch 0
Hub-P7
Hub 2
Hub-P7
Hub 1
Hub-P7
Hub 0

Procedure

Using Hub: 1) Add generic hub and seven PCs to workspace.
2) Give the IP address of each PC and make sure that IP is different for each device.
3) With the help of copper straight wire connect all PCs to hub.
4) Hub & PC ~~each~~ are connected to each other ~~&~~ fast ethernet connection.
5) If no of port are insufficient then add extra port, by clicking on device, turn off device & add necessary ports
6) With down IPs of all the end devices below the device.

Real time: select source PC in desktop tab, then select command prompt option in command prompt type.

ping 10.0.0.3: This is going to ping PC2 & response is generated in PC0

Simulation time:
Select simple PDU & select src and destination computer allows us to see how ~~ports~~ packets transformed to and from device

Hybrid Mode:

i) Add a switch, 3 hubs and 12 PCs to workspace

ii) Connect three hubs to switch & 4 PCs to each of the hubs using copper cross over and copper straight through wires respectively.

iii) Configure the IP of each of the PC in configure & add a note below each PC containg IP address.

Real time Mode: Select PC you want to send packet from & open its cmd prmpt. Specify destination PC by specifying its IP address. A response is not sent by destination PC to source PC.

Simulation mode:
Add a simple PDU by selecting the ports of PC and click on auto capture. from right panel.

Observation:

→ Hub:

Learning Outcomes:

* When source sends a packet in network the hub source the packet and endsbroadcast over the network, i.e, it sends

Date __/__/__
Page

data to all the end devices in network and node where it matches with the specified address accepts the packet and acknowledge it. Remaining nodes ignore the message

ii) Comm^n b/n hub & end devices is established through copper straight through wire as they belong to diffent layers.

iii) Number of ports can be added if needed by clicking on the device & adding the necessary ports

Result:

PC > ping 10.0.0.3
pinging 10.0.0.3 with 32 bytes of data
Reply from 10.0.0.3 : byte = 32 time = 0ms
Reply from 10.0.0.3 : byte = 32 time = 0ms

ping statistics for 10.0.0.3
packets: sent = 4, Received = 4, lost = 4

17/11/22

Switches

Learning Options:

When source device sends a message to the switches once a connection is established which takes some time called learning time, the switch receives the packet it initially broadcasts the packet to all connected devices to locate the destination, once the destination is located the message is sent only to that device.

Results:

ping 10.0.0.2

pinging 10.0.0.2 with 32 bytes of data

Reply from 10.0.0.2 : bytes = 32 time = 0ms

Reply from 10.0.0.2 : bytes = 32 time = 0ms

Reply from 10.0.0.2 bytes = 32 time = 0ms

Reply from 10.0.0.2 bytes = 32 time = 0ms

ping statistics for 10.0.0.2

Packets: Sent = 4, Received = 4, Lost = 0

Hybrid Mode:

Learning Options

→ switch and hub are connected through copper cable,

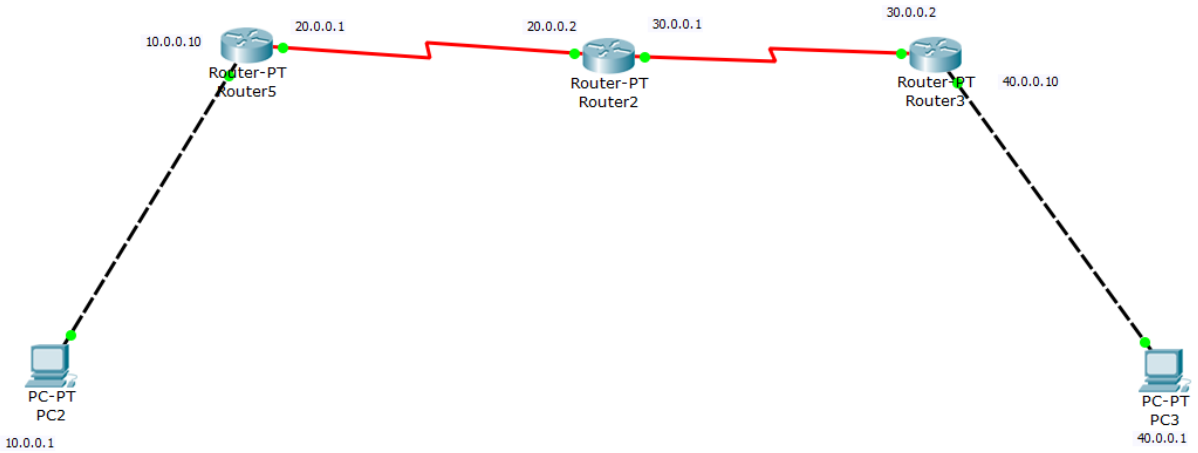
Date __/__/__
Page _____

as they belong to same network layer but PC and hubs are connected through by straight wire.

2) message from source PC to the destination is sent through the hub which then sends to all its connected PCs and the switches. The switches then sends the message to all its connected PC. The destination PC acknowledges that it has received the message by sending a acknowledgement to the source PC.

Present by
17/11/22

1BM20CS068

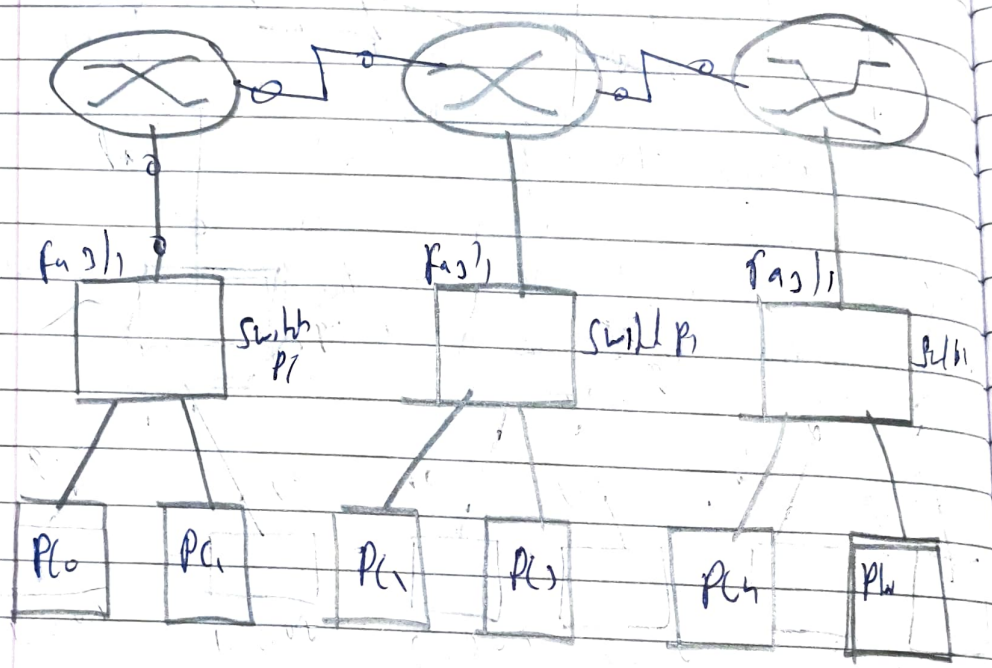

10.0.0.10
20.0.0.1
20.0.0.2
30.0.0.1
30.0.0.2
40.0.0.10
Router-PT
Router5
Router-PT
Router2
Router-PT
Router3
PC-PT
PC2
10.0.0.1
PC-PT
PC3
40.0.0.1

24/11/22

Lab 3

Aim: ~~To configure router to using minimum commands~~ Configure default route to the routers

Topology:



Procedure:

1) Place 3 generic routers switches & 6 generic PCs in the workspace.

2) Connect the PCs to the switches using copper straight through wire

3) ~~Connect~~ Connect the switches to routers also using cu straight ~~wire~~

4) ~~Connect~~ Connect the routers ~~to~~ on another using serial ~~as~~ DCE

5) Set the IP add of each PC and subnet mask in the fast ethernet 0.

6) Set the default gateway for each PC using settings

7) Click on the router & enter the following commands to establish connection b/w the switches

→ enable
→ config t
→ interface fast ethernet 0/0.
→ ip address 10.0.0.1 255.0.0.0
→ no shut.

After some time the light which was amber for the switch will turn green indicating the switches and routers are ready for communication.

Repeat the same for the other router
Click on the router to now establish connection b/w the neighbouring

→ enable
→ config t
→ interface serial 2/0
→ ip address 20.0.0.10 255.0.0.0
→ no shut

click on router1

- enable
- config t
- interface serial 2/0
- ip address 20.0.0.20 255.0.0.0
- no shut

The red light b/n the 2 routers will turn green indicate they are ready for communication.

Teaching Router 0 about network 30, 40, 50

click on router 0, open CLI:

- enable
- config t
- interface serial 2/0
- ip route 0.0.0.0 0.0.0.0 20.0.0.20
- exit
- show ip route

It will show that network 30, 40 50 are connected via gateway 20.0.0.20.

Teaching Router 1 of network to 4 40

Date __/__/__
Page

Tasks router 2 of intn 10.20 & 30

- enable
- config t
- int fa 0/0
- ip add 0.0.0.0 0.0.0.0 30.0.0.1
- exit
- show ip route

- Simulation mode:
  Add a single PDU by selecting the PC, & click on the 'add' option from right panel.

- Real time mode:
  select the PC PC0 and go to the command prompt & ping a PC in network so. At first it will show request timed out & packet will be lost during transmission. But on executing the command once more. The PC now would have learnt the network & the message will be sent successfully to PC.

- Observations

Learning outcome: On this network

router R1 does not have a default route since R0 & R2 can't become a default. similarly R1 if any one of R0 & R2 is default then the packets that are supposed to exit R1 can go to R0/R2 as they are default.

Result

PC> Ping 50.0.0.1
Pinging 50.0.0.1 with 32 bytes of data

Request timed out
Reply from 50.0.0.1: bytes = 32 time = 19ms TTL=128
Reply from 50.0.0.1: bytes = 32 time = 11ms TTL=128
Reply from 50.0.0.1: bytes = 32 time = 3ms TTL=128

Ping statistics for 50.0.0.1:
Packets: Sent = 4, received = 3 lost = 1 (25% loss)

PC> Ping 50.0.0.1
Pinging 50.0.0.1 with 32 bytes of data
Reply from 50.0.0.1: bytes=32 time=31ms TTL=128
Reply from 50.0.0.1 bytes=32 time=2ms TTL=128
Reply from 50.0.0.1: bytes=32 time=11ms TTL=128
Reply from 50.0.0.1 bytes=32 time=2ms TTL=128

Ping statistics for 50.0.0.1:
packets: Sent=4, received=4 lost=0 (0% loss)

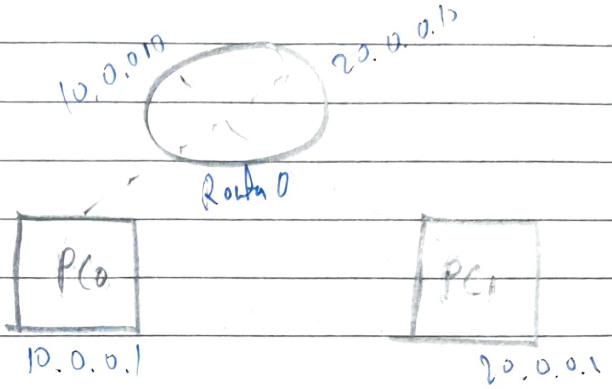
6/12/22

Lab 2

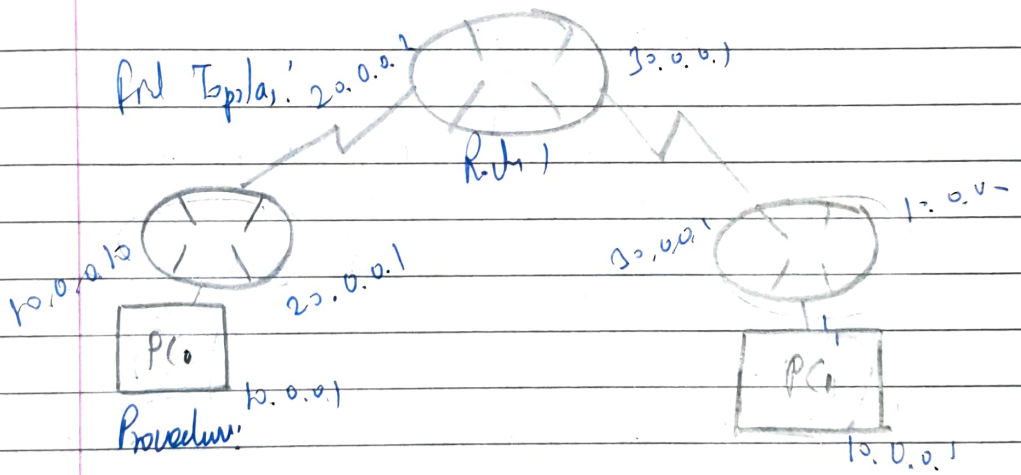
Date / /
Page
17/01/24

Aim: Configuring IP address to Router in packet tracer. Explore the following messages ping Response, Destination unreachable, Request timed out, Reply.

Topology:-



Pvt Topology:



Procedure:

1) Use a copper cross over wire to connect the 2 PCs to the router
2) Configure each PC & a specific IP address. Also enter gateways for both PCs.
3) Open Router CLI & do this:

enable
config t
interface fast ethernet 0/0
ip address 10.0.0.1 255.0.0.0

no shut
exit
interface fastethernet 1/0
ip address 20.0.0.1 255.0.0.0
no shut
exit
exit
exit

4) After entering these commands the lights b/w PC & Router will become green

⇒ For 3 Router & 2 PC's

i) Add 3 routers and 2 PC's to workspace and connect PC's & routers with by using the Crossover and connect Routers by using Serial DTE.

ii) Each PC is configured by a specific IP address and IP address is given by clicking specific PC after that ethernet gateway for both PC's

iii) Follow the following commands in CLI of a router after setting IP address as 2000.2 for

iv) Now for router follow the same commands & ip address as 3.0.0.1 in router and 4.0.0.10 for fast ethernet 0/

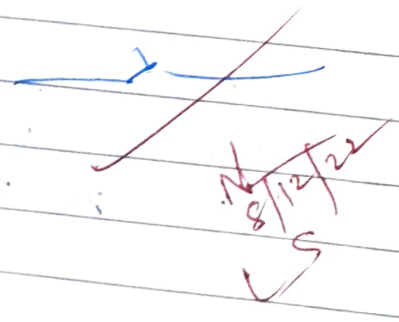
i) Initially we get Req time out then I ...
we get the results

Ping 4.0.0.1
Pinging 4.0.0.1 [32 bytes of data:

Reply from 4.0.0.1 bytes = 32 Time = ...
Reply from 4.0.0.1 bytes = 32 Time = ...
Reply from 4.0.0.1 bytes = 32 Time = ...
Reply from 4.0.0.1 : bytes = 32 Time = ...

Ping Statistics for 4.0.0.1:

Packets sent = 4 Received = 4 lost = 0 (0...
Approximate round trip time in milli seconds
Minimum = 3ms, Maximum = 4ms Average = 7ms

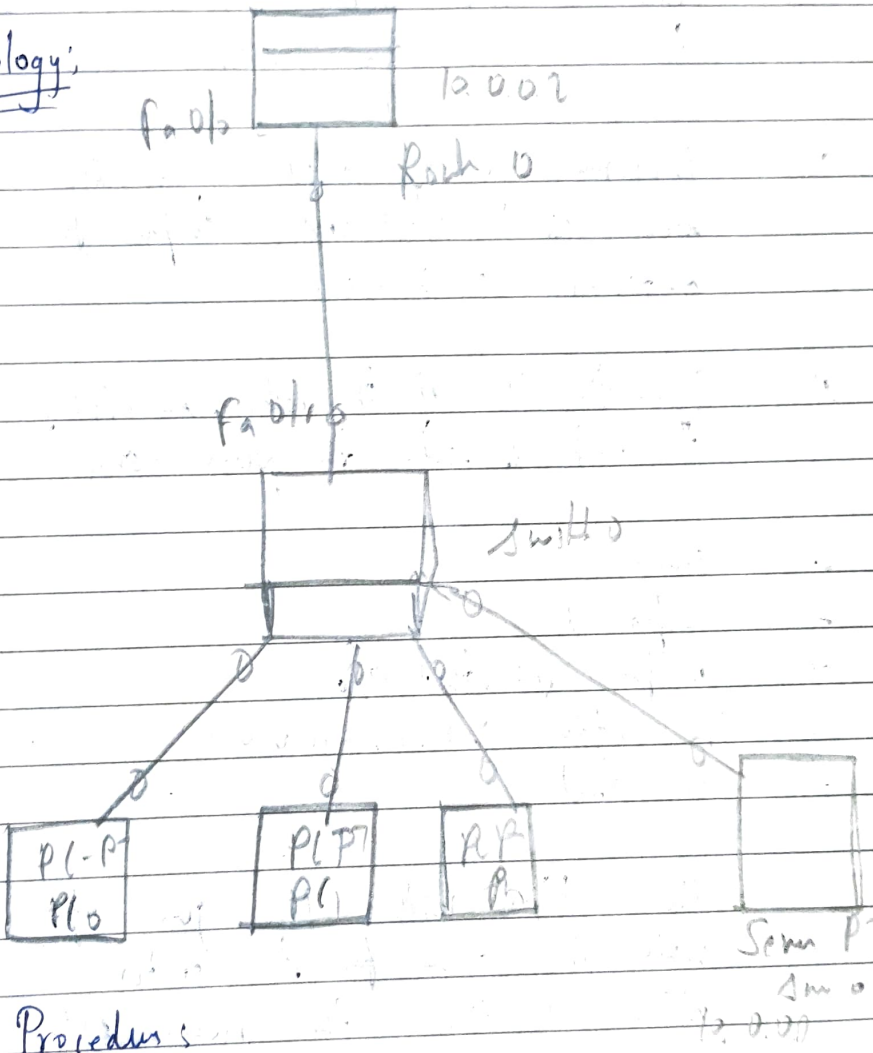


Lab 4

Date 13/12/22
Page

Aim: Configure DHCP within a LAN in a packet tracer.

Topology:



4) Procedure:

1) Place a generic router, a generic switch, 2 generic PC's and server in the workspace as shown in the given topology.

2) Connect the PC's to the switch through the straight wire.

3) Connect the server to the switch & switch to the router using the straight through

4) Place a note below the server & write the ip address as 10.0.0.1,

5) Configure the ip address of the server as 10.0.0.1 and configure the gateway as 10.0.0.1

6) Open the CLI of the router or click the router & enter the following commands

- enable
- config t
- interface fast ethernet 0/0
- ip address 10.0.0.2 255.0.0.0
- no shut

The light will turn green of router 4 ports for the switch After some time, the amber color also change the green.

Now click on the server

Open the services tab
click on DHCP.
Turn the switch ON

Result

1) PC > Ping 10.0.0.5

Pinging 10.0.0.5 with 32 bytes of data:

Reply from 10.0.0.5: bytes = 32 time = 0ms TTL=128
Reply from 10.0.0.5: bytes = 32 time = 0ms TTL=128
Reply from 10.0.0.5: bytes = 32 time = 0ms TTL=128
Reply from 10.0.0.5: bytes = 32 time = 0ms TTL=128

Ping Statistics for 10.0.0.5:

Packets: Sent = 4, Received = 4, Lost = 0 (0% loss)
Approximate round trip times in milli-seconds:
Minimum = 0ms, Maximum = 0ms, Average

2) PC > Ping 10.0.0.4

Pinging 10.0.0.4 with 32 bytes of data:

Reply from 10.0.0.4: bytes = 32 time = 0ms TTL=128
Reply from 10.0.0.4: bytes = 32 time = 0ms TTL=128
Reply from 10.0.0.4: bytes = 32 time = 0ms TTL=128
Reply from 10.0.0.4: bytes = 32 time = 0ms TTL=128

Ping statistics for 10.0.0.4:
Packets Sent = 4, Received = 4, Lost = 0 (0% loss)
App round trip time in milli-seconds
Min = 0ms, Max = 0ms, Avg = 0ms

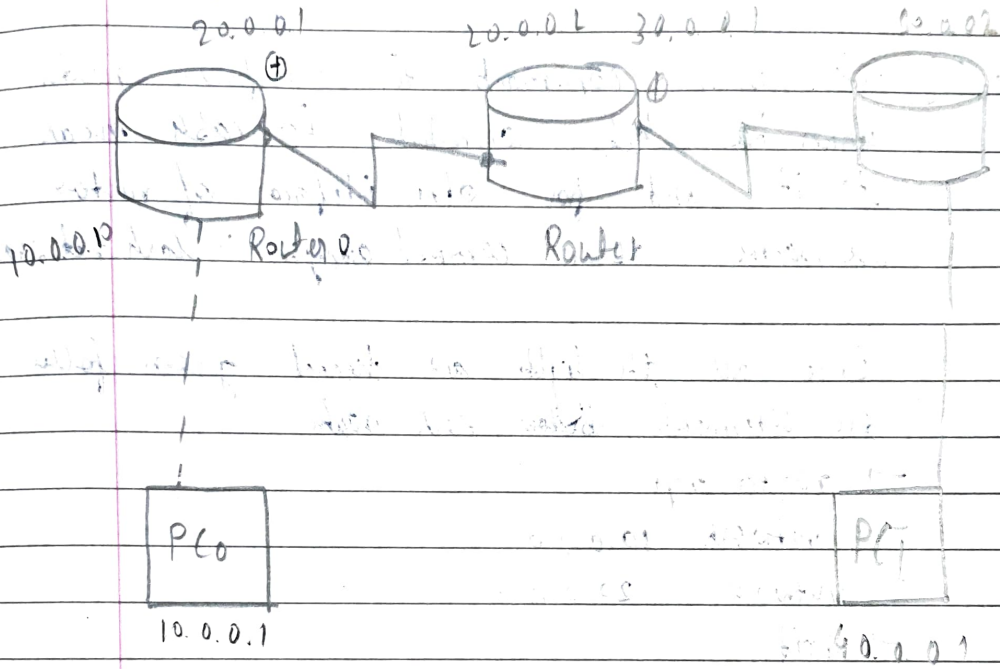
☆ Lab 5 ☆

Date / /
Page 08/12/2(

~~set default gateway ip~~

Aim: Configuring RIP Routing Protocol in Routers

Topology:



Procedure:

- Use 3 generic routers, 2 generic PC and place notes to indicate respective IP address. Use serial DCE cable to connect router and use copper cross cable to connect PC & router 1 and router 3.
- Set IP address, gateway & subnet mask as 10.0.0.1 10.0.0.10, 255.0.0.0 for PC0 set 40.0.0.1, 40.0.0.10, 255.0.0.0 for PC1
- interface PC0 and router 1
  - → interface fastethernet 0/0
  - → IP address 20.0.0.10 255.0.0.0
  - → no shut

- for interfacing serial 2/0 of router 1
  - → interface serial 2/0
  - → IP address 20.0.0.1 255.0.0.0
  - → encapsulation PPP
  - → clock rate 64000
  - → no shut

- Use above commands for interfacing router which has clock symbol in cable near to it and for other interfaces of routers use same above command except "clock rate 64000"

- Once all the lights are turned green follow the commands below each router
  - → router rip
  - → network 10.0.0.0
  - → network 20.0.0.0
  - → exit

- ~~Repeat~~ Repeat the same command for router 2 and 3

- Observation:
  Use RIP routing becomes easy when lots number of routers are present

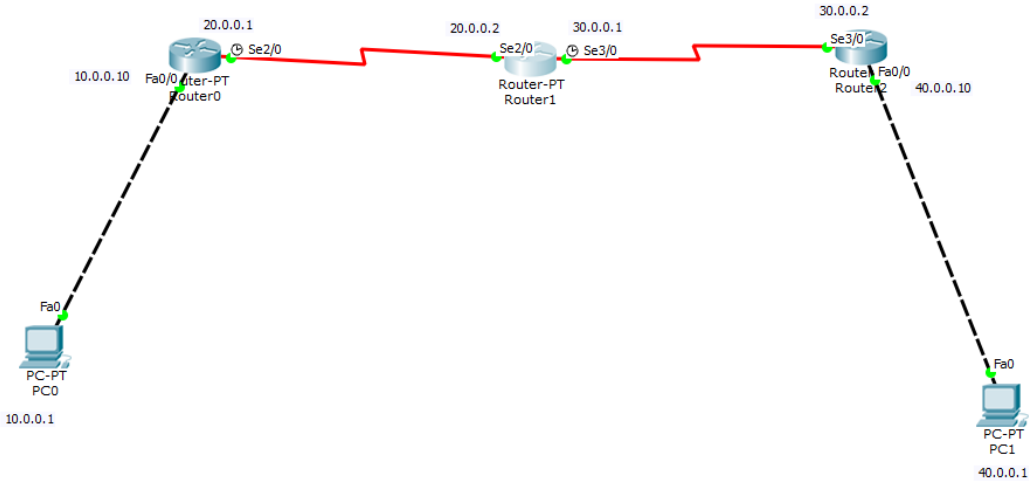
- Result:
  Pinging 10.0.0.1 with 32 byte of data

Cisco Packet Tracer Student

File Edit Options View Tools Extensions Help

Logical | [Root] | New Cluster | Move Object | Set Tiled Background | Viewport

1BM20CS068



Time: 00:09:56 | Power Cycle Devices | Fast Forward Time | Realtime

Connections

Automatically Choose Connection Type

Scenario 0 | New | Delete | Toggle PDU List Window

Fire	Last Status	Source	Destination	Type	Color	Time(sec)	Periodic	Num	Edit	Delete

Type here to search

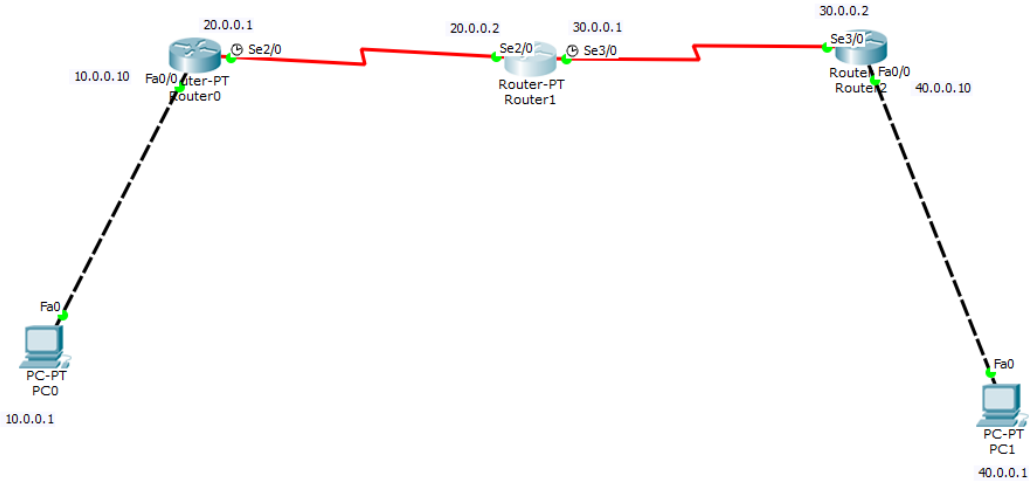
26°C Partly sunny ENG 11:47:21 08-12-2022

Cisco Packet Tracer Student

File Edit Options View Tools Extensions Help

Logical | [Root] | New Cluster | Move Object | Set Tiled Background | Viewport

1BM20CS068



Time: 00:09:56 | Power Cycle Devices | Fast Forward Time | Realtime

Connections

Automatically Choose Connection Type

Scenario 0 | New | Delete | Toggle PDU List Window

Fire	Last Status	Source	Destination	Type	Color	Time(sec)	Periodic	Num	Edit	Delete

Type here to search

26°C Partly sunny ENG 11:47:21 08-12-2022

Date / /
Page

reply from 10.0.0.1 byte = 32
reply from 10.0.0.1 byte = 32
reply from 10.0.0.1 byte = 32
reply from 10.0.0.1 byte = 32

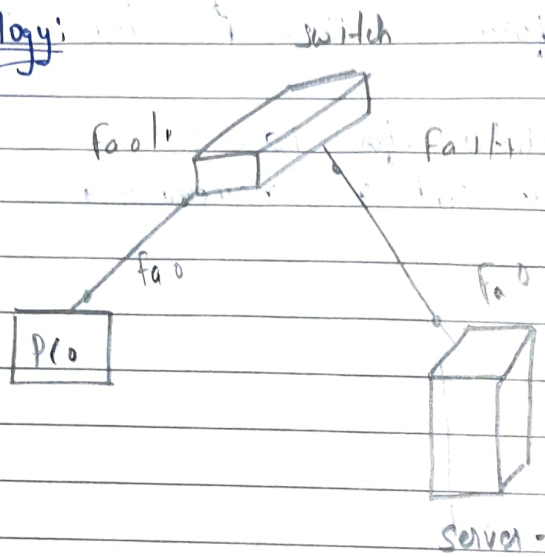
ping statistics for 10.0.0.1
packets: sent = 4, recieved = 4, lost = 0

Wabi
29-12-2022

15/12/22 Lab 6

Aim: Demonstration of WEB server & DNS using Packet Tracer

Topology:



Procedure:

- Set the IP address of the pc and the server respectively as 10.0.0.2 and 10.0.0.1
- Open web-browser in desktop tab of PC and type IP of server "http://10.0.0.2"
- Default home page will be displayed
- Now go to the Services tab of server enable HTTP and change contents of index.html by clicking edit option.
- check again in browser of PC0 to see the updated changes.

Activate D

- enable D
- enter the
need to b
I "
- click
- now g
it displ

Custom

- create a
in http
- chang
created f
- Chek
PC0 by

We ca
"cm" in
it map
name
Mappi
for U
with a

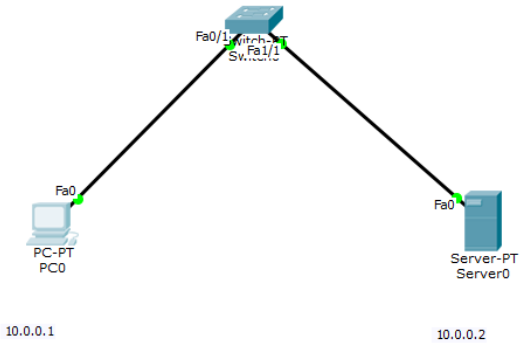
Well
29-12-2022

Cisco Packet Tracer Student

File Edit Options View Tools Extensions Help

Logical | [Root] | New Cluster | Move Object | Set Tiled Background | Viewport

1bm20cs068



Time: 00:47:02 | Power Cycle Devices | Fast Forward Time | Realtime

Connections

Automatically Choose Connection Type

Scenario 0 | New | Delete | Toggle PDU List Window

Fire	Last Status	Source	Destination	Type	Color	Time(sec)	Periodic	Num	Edit	Delete

26°C Mostly sunny

Search

06:48 15-12-2022

Activate DNS :-

- enable DNS on DNS service to activate it
- enter the domain name and IP address need to be mapped.
  ["cn" → 10.0.0.2]
- click add to add the new mapping
- now give name in the web browser to check if it displaying index.html

Custom page:

- create a new page resume.html and save it in http services.
- change hyperlink in index.html to link the above created file
- check the output in the webbrowser of PC0 by click on hyperlink.

Observation

- We can view the webpage when we type "cn" in browser because 10.0.0.2 address is mapped to the name "cn" by domain name system concept.
- Mapping is required because its difficult for users to remember IPs hence IP is mapped with a name.

29-12-2022

Lab 7 (crc)

27/12/22

Aim: Write a program for error detection for CRC-16

```
#include <stdio.h>
#include <string.h>
#define N strlen(gen_poly)
char data[28];
char check_value[28];
char gen_poly[10];
int data_length, i, j;
void XOR(){
for(j=1; j<N; j++)
check_value[j] = ((check_value[j] == gen_poly[j])?'0':'1');
}

void receive(){

printf("Enter the received data: ");
scanf("%s", data);
printf("\n-----\n");
printf("Data received: %s", data);
crc();
for(i=0; (i<N-1) && (check_value[i]!='1'); i++);
if(i<N-1)
printf("\nError detected\n\n");
else
printf("\nNo Error detected\n\n");

void crc(){
```

```
for (i=0; i<N; i++)
    check_value[i] = data[i];

do {
    if (check value[0] == '1')
    error();

    for (j=0; j<N; j++)
    check_value[j] = check_value[j+1];
    check_value[j] = data[i+j];
}
    while (i <= data_length + N - 1);
}

int main() {

printf("\n Enter data to be transmitted: ");
scanf("%s", data);
printf("\n Enter the Gen polynomial: ");
scanf("%s", gendata);
data_length = strlen(data);
for (i=data_length; i<data_length+N-1; i++)
    data[i]='0';

printf("\n Data padded with n-1 zeros : %s", data);
crc();
printf("\n CRC or check value is %s", data);

data[n] = check_value[i-data_length];
```

printf("\n Enter data to be sent: "); scanf

receive();

return 0;

}

Op

Enter the data: 1000100000010000

1000100000010000 $x^{16}+x^{12}+x^5+1$

Enter poly: 10001001 :: GP

Data padded with n-1: 1000100000010000000

CRC or check value :: 01001

Final data sent: 1000100000010001

Enter the received data: 1000100000010001

Data received: :: 1000100000010001

No error detected

5/1/23

☆ Leaky Bucket ☆

Aim: ~~Write a program for distance vector algorithm to find~~ suitable path for ~~transmission~~

```
#include <bits/stdc++.h>
#include <unistd.h>
using namespace std;
# define bucketSize 500

void bucketInput (int a, int b)
 {

  if ( a > bucketSize)
     cout << "\n\t\t Bucket overflow";
  else
  {
    sleep(5);
    while (a>b)
    {
      cout << "\n\t\t" << b << " bytes outputted.";
      a -= b;
      sleep(5);
    }

    if ( a>0)
    cout << "Last" << a << "bytes sent \t";

  cout << " Bucket Output successful";
 }
}
```

```
int main()
{
    int op, pktsize;
    cout << "Enter output rate: ";
    cin >> op;
    for (int i=1; i<=5; i++)
    {
        sleep(rand()%10);
        pktsize = rand()%701;
        cout << "Packet no" << i << " Packet size = " << pktsize;
        bucketInput(pktSize, op);
    }
    cout << endl;
    return 0;
}
```

Enter output rate : 50

Packet no 1 Packet size = 186
50 bytes output.
50 bytes outputted
50 bytes outputted
Last 36 bytes sent.
Packet output successfully.

2
200 50
→ 300 50
50
50
50

Packet no 2 Packet size = 21B
So bytes outputted.
So byte outputted.
So bytes outputted
Last 15 byte sent
Bucket output successful.

Pack no 3 Packet size = 53B
Bucket Overflow.

Packet no 4 Packet size = 49B
So bytes outputted.
So bytes outputted.
So byte outputted
So bytes outputted
So bytes outputted
So byte outputted
So bytes outputted
So bytes outputted
So bytes outputted
Last 42 bytes sent
Bucket output successful.

Pack no 5 Packet size = 52B
Bucket overflow

12/1/23
Repeat again

12/01/23
Aim: With a
#inclu
#inclu
int Ba
C
fo

12/01/23

Aim: With a program for distance vector Algo to fid suitable path.

~~Write a program to implement Dijkstra's algorithm~~

Write a program to implent Bellman ford Algorithm

```
#include <stdio.h>
#include <stdlib.h>

int Bellman_Ford( int G[20][20], int V, int E,
                  int edge[20][3])
{
   int i,u, v, k, distance[20], parent[20], S, flag=1;

   for( i=0; i<V ; i++)
      distance[i]=1000, parent[i]= -1;
   printf(" Enter source: ");
   scanf ( "%d", &S);
   distance[S-1]=0;

   for( i=0; i<V-1; i++)
   {
      for( k=0; k<E; k++)
      {
         u = edge[k][0], v= edge[k][1];

         if (distance[u] + G[u][v] < distance[v])
            distance[v]= distance[u] + G[u][v],
                        parent[v]=u;

      }
   }
```

```
for(k=0; k<E; k++)
  {  u = edge[k][0]   v = edge[k][1];
  if( distance[u]+G[u][v]< distance[v])
      flag=0;

  if (flag)
    for (i=0; i<V; i++)
    printf( "Vertex %d → cost= %d and
          = %d\n", i+1, distance[i], parent[i]+1);

    return flag;
}

int main()
 {

 int V, edge[20][3], G[20][20], i, j, k=0;

 printf(" Bellaman Ford \n");
 printf("Enter no of vertex : ");
  scanf("%d", &V);
 printf("Enter graph in matrix form:\n");

   for( i=0; i<V; i++)
      for(j=0; j<V; j++)
      {
         scanf( "%d", &G[i][j]);
         if( G[i][j]!=0)
           edge[k][0]=i, edge[k++][1]=j;
      }
```

12/08/23

9 Implement Dijkstra's Algo to compute the shortest path from a given topography.

Dijkstra's Algorithm

```
#include<bits/stdc++.h>
#include <limits.h>
#include<stdio.h>
using namespace std;

#define V 9

int minDistance( int dist[], bool sptSet[])
{
    int min = INT_MAX, min_index;

  for( int v=0; v<V, v++)
     if (sptSet[v]==false && dist[v]<=min)
       min = dist[v], min_index = v;

   return min_index;
}

void printSolution (int dist[])
{
  printf( "Vertex \t Distance from Source\n")
     for( int i=0; i<V ; i++)
     printf("%d \t %d \n", i, dist[i]);
}
```

void
{

```
void dijkstra ( int graph[V][V], int src)
{
    int dist[V],
    bool sptSet[V]

    for ( int i=0; i<V; i++)
        dist[i]= INT_MAX, sptSet[i]= false;

    dist[src]=0;

    for (int count=0; count<V-1; count++){

      int u= minDistance( dist, sptSet);

      sptSet[u]= true;

    for ( int v=0; v<V; v++)
        if ( !sptSet[v] && graph[u][v] && dist[u]!= INT_MAX
            && dist[u]+graph[u][v] < dist[v])
          dist[v]= dist[u]+ graph[u][v];
    }

    printSolution(dist);
}

int main()
{
    int graph[V][V]
    Load a 64 from graph.txt
```

```
PS C:\Users\mdsur\Documents\COLLEGE-LAB> cd "c:\Users\mdsur\Documents\COLLEGE-LAB\" ; if ($?) { g++ djikstras.cpp -o djikstras } ; if ($?) { .\djikstras }
Enter the graph
0 9 2 5
9 0 6 8
2 6 0 0
5 8 0 0
Vertex          Distance from Source
0               0
1               8
2               2
3               5
PS C:\Users\mdsur\Documents\COLLEGE-LAB>
```

28/01/23

```
for (int i=0; i<V; i++)
{
    for (int j=0; j<V; j++)
        cin >> graph[i][j];
}

dijkstra(graph, 0);
return 0;
}
```

o/p

Enter graph

0 9 2 5
9 0 6 8
2 6 0 0
5 8 0 0

Vertex	Distance from Source
0	0
1	8
2	2
3	5

28/01/23

Socket (TCP/IP)

Using TCP/IP socket with a client server program to make client sending the file name and the server to send back the contents of the requested file if present.

Client TCP.py

```
from socket import *
serverName = '127.0.0.1'
serverPort = 12000

2
clientSocket = socket(AF_INET, SOCK_STREAM)
clientSocket.connect((serverName, serverPort))
sentence = input("\n Enter file name:")
clientSocket.send(sentence.encode())
fileContents = clientSocket.recv(1024).decode()
print('\n From Server:\n')
print(fileContents)
clientSocket.close()
```

Server.py

```
from socket import *
serverName = "127.0.0.1"
serverPort = 12000
serverSocket = socket(AF_INET, SOCK_STREAM)
serverSocket.bind((serverName, serverPort))
serverSocket.listen(1)
```

20/01/23

Sockets (UDP)

Using UDP sockets, write a client server program to make client sending the filename & the server to send back the contents of the requested file if present

clientudp.py

```
from socket import *
serverName = "127.0.0.1"
serverPort = 12000
clientSocket = socket(AF_INET, SOCK_DGRAM)
sentence = input("\n Enter the name")

clientSocket.sendto(bytes(sentence, "utf-8"),
                    (serverName, serverPort))

fileContents, serverAddress = clientSocket.recvfrom(2048)

print("\n Reply from Server:\n")
print(fileContents.decode("utf-8"))
# for i in fileContents:
    # print(str(i), end="")
clientSocket.close()
clientSocket.close()
```

serverUDP.py

from socket import *
serverPort = 12000
serverSocket = socket(AF_INET, SOCK_DGRAM)

serverSocket.bind(("127.0.0.1", serverPort))
print("The server is ready to receive")

while 1:
    sentence, clientAddress = serverSocket.recvfrom(2048)

    sentence = sentence.decode("utf-8")
    file = open(sentence, "r")
    l = file.read(2048)

    serverSocket.sendto(bytes(l, "utf-8"), clientAddress)
    print('\nSent contents of ' + end = '')
    print(sentence)
    # for i in sentence:
    #   print(str(i), end='')
    file.close()

Op: Server.py

The server is ready to receive
Sent contents of serverop.py
The server is ready to receive

Client.py

Enter the filename: serverop.py

Right from server.

from socket
serverPort = 12000
serverSocket = socket(

serverSocket.bind(
while 1:
    print("The
    sentence, clientA
    sentence = sente
    file = open(sen
    l = file.read(
    serverSocket.sendto

    print("\n Sen
    print(sentence)
    file.close()

>>)

Right from server.

```
from socket import *
serverPort = 12000
serverSocket = socket(AF_INET, SOCK_DGRAM)

serverSocket.bind(("127.0.0.1", serverPort))
while 1:
    print("The server is ready to receive")
    sentence, clientAddress = serverSocket.recvfrom(2048)
    sentence = sentence.decode("utf-8")
    file = open(sentence, "r")
    l = file.read(2048)
    serverSocket.sendto(bytes(l, "utf-8"),
                        clientAddress)
    print("\n Sent back to client")
    print(sentence)
    file.close()
```

>>)

```
The server is ready to receive

Sent contents of  serverUDP.py
```

```
Enter file name: serverUDP.py

Reply from Server:

from socket import *
serverPort = 12000
serverSocket = socket(AF_INET, SOCK_DGRAM)
serverSocket.bind(("127.0.0.1", serverPort))
print ("The server is ready to receive")
while 1:
    sentence, clientAddress = serverSocket.recvfrom(2048)
    sentence = sentence.decode("utf-8")
    file=open(sentence,"r")
    l=file.read(2048)
    serverSocket.sendto(bytes(l,"utf-8"),clientAddress)
    print ('\nSent contents of ', end = ' ')
    print (sentence)
# for i in sentence:
# print (str(i), end = '')
    file.close()
```

```python
from socket import *
serverName="127.0.0.1"
serverPort = 12000
serverSocket = socket(AF_INET,SOCK_STREAM)
serverSocket.bind((serverName,serverPort))
serverSocket.listen(1)
while 1:
    print ("The server is ready to receive")
    connectionSocket, addr = serverSocket.accept()
    sentence = connectionSocket.recv(1024).decode()
    file=open(sentence,"r")
    l=file.read(1024)
    connectionSocket.send(l.encode())
    print ('\nSent contents of ' + sentence)
    file.close()
    connectionSocket.close()
```

```
Python 3.10.9 (tags/v3.10.9:1dd9be6, Dec  6 2022, 20:01:21) [MSC v.1934 64 bit (AMD64)]
on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
========== RESTART: C:/Users/mdsur/Desktop/server.py ==========
The server is ready to receive

Sent contents of server.py
The server is ready to receive
```