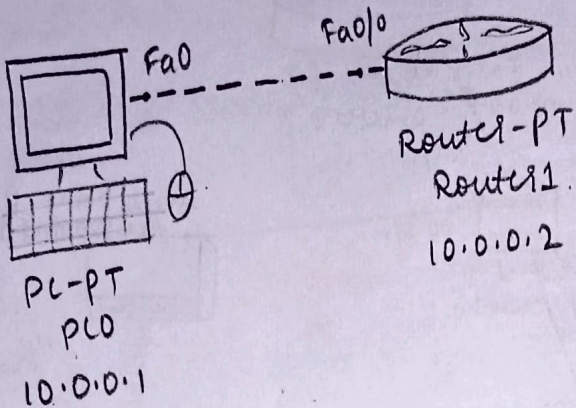
18/8/23

Lab-09

Aim: To understand the operation of TELNET by accessing the router in server room from a PC in IT office.

Topology:



procedure:

Step 1: Create a topology using 1 pc and 1 router as shown above.

Step 2: set the ip address and gateway as 10.0.0.1 and 10.0.0.2 for the pc

Step 3: In the router, go to CLI

```
Router > enable
Router # config t
Router (config) # hostname r1
r1 (config) # enable secret p1
r1 (config) # interface fastethernet 0/0
r1 (config-if) # ip address 10.0.0.2 255.0.0.0
r1 (config-if) # no shut.
r1 (config-if) # line vty 0 5
r1 (config-line) # login
```

% Login disabled on line 132, until 'password' is set

r1(config-line)# password p0

r1(config-line)# exit

r1(config)# exit

r1# wr

Step 4: In command prompt of PC,

PC> 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 TTL = 255

Reply from 10.0.0.2 : bytes = 32 time = 0ms TTL = 255

Reply from 10.0.0.2 : bytes = 32 time = 0ms TTL = 255

Reply from 10.0.0.2 : bytes = 32 time = 0ms TTL = 255

ping statistics for 10.0.0.2 :

packets : Sent = 4, Received = 4, Lost = 0 (0% loss)

Approximate round trip times in milli-seconds:

minimum = 0ms, maximum = 0ms, Average = 0ms

PC> telnet 10.0.0.2

Trying 10.0.0.2 .... open

User Access verification

Password: (p0)

r1> enable

Password: (p1)

r1# show ip route

Codes : C - connected

C 10.0.0.0/8 is directly connected, FastEthernet 0/0

Obs: Using telnet protocol, we can access the router from the PC (which is connected to it)

10/10

28/8/22

1BM21CS154

Command Prompt

```
Packet Tracer PC Command Line 1.0
PC>ping 10.0.0.2

Pinging 10.0.0.2 with 32 bytes of data:

Reply from 10.0.0.2: bytes=32 time=55ms TTL=255
Reply from 10.0.0.2: bytes=32 time=0ms TTL=255
Reply from 10.0.0.2: bytes=32 time=0ms TTL=255
Reply from 10.0.0.2: bytes=32 time=0ms TTL=255

Ping statistics for 10.0.0.2:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 0ms, Maximum = 55ms, Average = 13ms

PC>telnet 10.0.0.2
Trying 10.0.0.2 ...Open


User Access Verification

Password:
r1>enable
Password:
r1#show ip route
Codes: C - connected, S - static, I - IGRP, R - RIP, M - mobile, B - BGP
       D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area
       N1 - OSPF NSSA external type 1, N2 - OSPF NSSA external type 2
       E1 - OSPF external type 1, E2 - OSPF external type 2, E - EGP
       i - IS-IS, L1 - IS-IS level-1, L2 - IS-IS level-2, ia - IS-IS inter area
       * - candidate default, U - per-user static route, o - ODR
       P - periodic downloaded static route

Gateway of last resort is not set

C    10.0.0.0/8 is directly connected, FastEthernet0/0
r1#
```

CYCLE-2

= 1) Write a program for error detecting code using CRC-CCITT (16-bits)

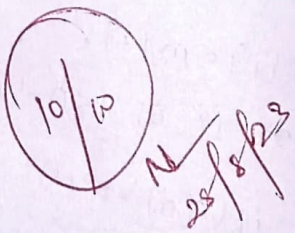
program:

```
#include<stdio.h>
char m[50], g[50], r[50], q[50], temp[50];
void caltrans(int);
void crc(int);
void calram();
void shift();
void main()
{
  int n,i=0;
  char ch, flag=0;
  printf("Enter the frame bits:");
  while((ch=getc(stdin))!='\n')
  m[i++]=ch;
  n=i;
  for(i=0; i<16; i++)
  m[n++]='0';
  m[n]='\0';
  printf("Message after appending 16 zeros: %s",m);
  for(i=0; i<=16; i++)
  g[i]='0';
  g[0]=g[4]=g[11]=g[16]='1'; g[17]='\0';
  printf("\nGenerator: %s\n", g);
  crc(n);
  printf("\nquotient: %s", q);
  caltrans(n);
  printf("\nTransmitted frame: %s", m);
  printf("\nEnter Transmitted frame:");
```

```
scanf("%s",m);
printf("CRC checking \n");
crc(n);
printf("\n\n last remainder: %s",r);
for(i=0;i<16;i++)
 if(r[i]!='0')
 flag='1';
 else
  continue;
 if(flag=='1')
  printf("error during transmission");
 else
  printf("\n Recieved frame is correct");
}

void crc(int n)
{
 int i,j;
 for(i=0;i<n;i++)
  temp[i]=m[i];
 for(i=0;i<16;i++)
  r[i]=m[i];
  for(i=0;i<n-16;i++)
  {
   if(r[0]=='1')
   {
    q[i]='1';
    caldam();
   }
  else
   q[i]='0';
   shiftl();
  }
  r[16]=m[17+i];
  r[17]='\0';
```

```
    for (j=0; j<=17; j++)
      temp[j]= r[j];
  }
  q[n-16]= '\0';
}
void calram()
{
  int i,j;
  for (i=1; i<=16; i++)
  r[i-1] = ((int)temp[i]-48) ^ ((int)q[i]-48)+48;
}
void shiftl()
{
  int i;
  for (i=1; i<=16; i++)
  r[i-1]=r[i];
}
void caltrans(int n)
{
  int i,k=0;
  for (i=n-16; i<n; i++)
  m[i] = ((int)m[i]-48)^((int) r[k++]-48)+48;
  m[i]='\0';
}
```



output:

Enter frame bits: 1011

message after appending 16 zeros: 1011 0000 0000 0000 0000

generator: 10001000000100001

quotient: 1011

Transmitted frame: 1011 1011 0001 0110 1011

Enter Transmitted frame: 1011 1011 0001 0110 1011

Last remainder: 0000 0000 0000 0000

Recieved frame is correct.

OUTPUT:

```
Enter the frame bits:1011
Message after appending 16 zeros:1011000000000000000
generator:10001000000100001


quotient:1011
transmitted frame:10111011000101101011
Enter transmitted frame:10111011000101101011
CRC checking


last remainder:0000000000000000

Received frame is correct
Process returned 0 (0x0)   execution time : 14.468 s
Press any key to continue.
```

```
Enter the frame bits:1001
Message after appending 16 zeros:1001000000000000000
generator:10001000000100001


quotient:1001
transmitted frame:10011001000100101001
Enter transmitted frame:10011001000000101001
CRC checking


last remainder:0000000100000000Error during transmission
Process returned 0 (0x0)   execution time : 19.597 s
Press any key to continue.
```

2) Write a program for congestion control using leaky bucket algorithm.

program:

```
#include<stdio.h>
int main(){
int incoming, outgoing, bucket_size, n, store=0;
printf("Enter bucket size, outgoing rate and no.
  of inputs: ");
scanf("%d %d %d", &bucket_size, &outgoing, &n);
while(n!=0){
printf("Enter the incoming packet size:");
scanf("%d", &incoming);
printf("Incoming packet size %d\n", incoming);
if (incoming <= (bucket_size - store)){
    store += incoming;
printf("Bucket buffer size %d out of %d\n",
  store, bucket_size);
}
else{
printf("Dropped %d no. of packets\n", incoming-
  (bucket_size - store));
printf("Bucket buffer size %d out of %d\n",
        store, bucket_size);
  store = bucket_size;
}
store = store - outgoing;
printf("After outgoing %d packets left out %d
        in buffer\n", store, bucket_size);
  n--;
}
return 0;
}
```

output:

Enter bucket size, outgoing rate and no. of inputs:

20 10 2

Enter the incoming packet size : 30

Incoming packet size 30

Dropped 10 no. of packets

Bucket buffer size 0 out of 20

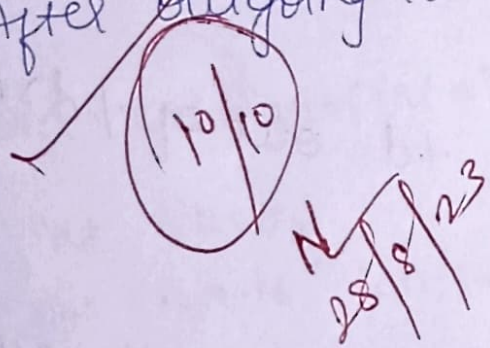
After outgoing 10 packets left out 20 in buffer.

Enter the incoming packet size : 10

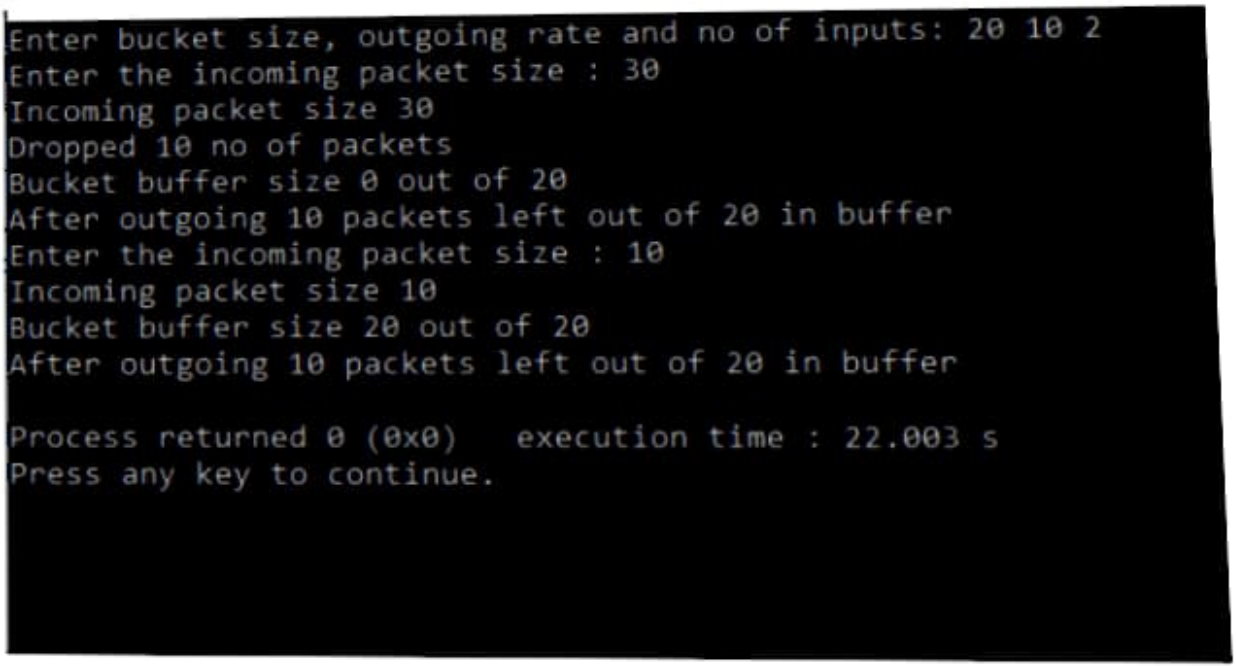
Incoming packet size 10

Bucket buffer size 20 out of 20

After outgoing 10 packets left out of 20 in buffer



OUTPUT:


Enter bucket size, outgoing rate and no of inputs: 20 10 2
Enter the incoming packet size : 30
Incoming packet size 30
Dropped 10 no of packets
Bucket buffer size 0 out of 20
After outgoing 10 packets left out of 20 in buffer
Enter the incoming packet size : 10
Incoming packet size 10
Bucket buffer size 20 out of 20
After outgoing 10 packets left out of 20 in buffer
Process returned 0 (0x0) execution time : 22.003 s
Press any key to continue.