DCN Assignment 1

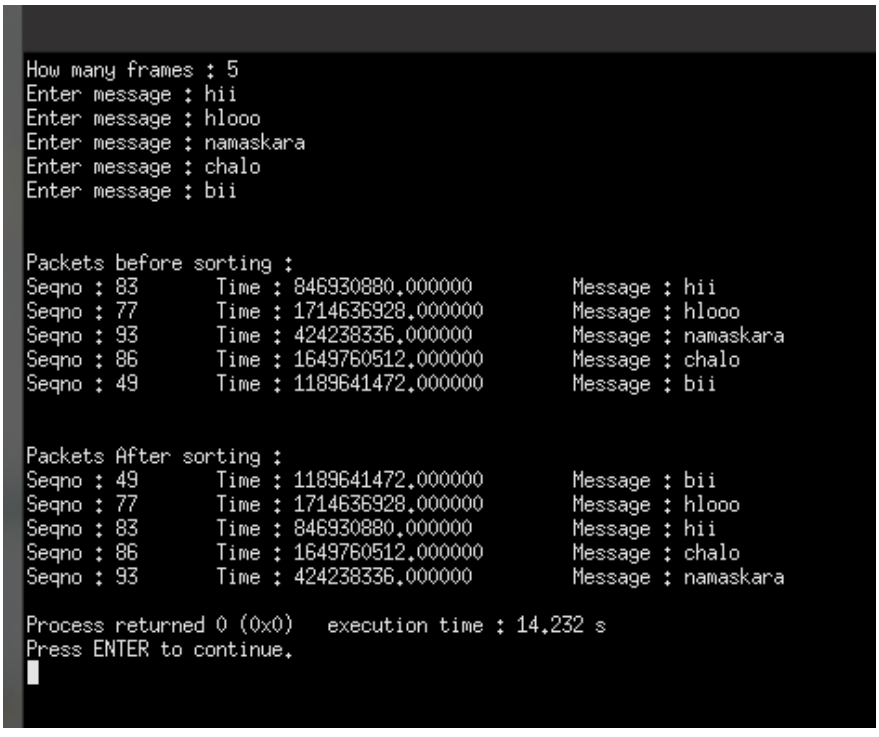
1.Write a C program to implement Frame Sorting

- Each frame contains sequence number, time,message
- Sequence number must be random number of 2 digits Regenerate the repeated sequence no
- Sort the frames based on sequence number using bubble sort techq
- Display all info of packet before sort and after sort

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

struct packet
{   int seqno;
    char msg[50];
    float time;
};

void bubblesort(struct packet * p,int n)
{   struct packet temp;
    for(int i=0;i<n;i++)
    {   for(int j=0;j<n-i-1;j++)
        {   if(p[j].seqno > p[j+1].seqno)
            {   temp = p[j];
                p[j] = p[j+1];
                p[j+1] = temp;
            }
        }
    }
}

int main()
{   int n,num;
    printf("How many frames : ");
    scanf("%d",&n);
    struct packet p[n];

    for(int i=0;i<n;i++)
    {   again:
        num = rand()%100;
        for(int j=0;j<=i;j++)
        {   if(p[i].seqno == num)
```

```
            goto again;
        }
        p[i].seqno = num;
        printf("Enter message : ");
        scanf("%s",p[i].msg);
        p[i].time = rand();
    }

    printf("\n\nPackets before sorting : \n");
    for(int i=0;i<n;i++)
        printf("Seqno : %d\t Time : %f\t Message : %s\n",p[i].seqno, p[i].time, p[i].msg);

    bubblesort(&p,n);

    printf("\n\nPackets After sorting : \n");
    for(int i=0;i<n;i++)
        printf("Seqno : %d\t Time : %f\t Message : %s\n",p[i].seqno, p[i].time, p[i].msg);

    return 0;
}
```

OUTPUT :



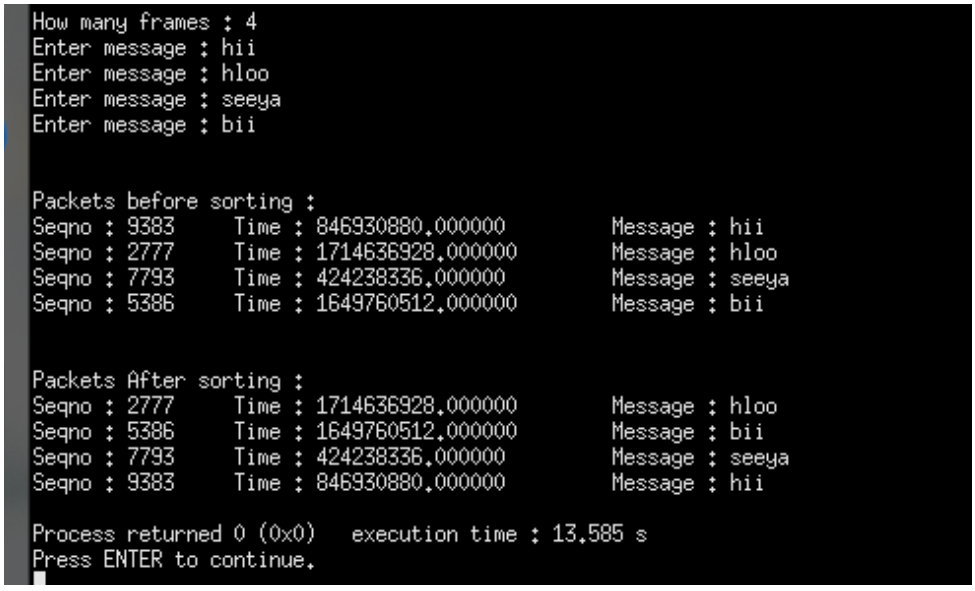
2.Write a C program to implement Frame Sorting

- Each frame contains sequence number, time,message
- Sequence number must be random number of 4 digits Regenerate the repeated sequence no
- Sort the frames based on sequence number using quick sort techq
- Display all info of packet before sort and after sort

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

struct packet
{   int seqno;
   char msg[50];
   float time;
};

void quicksort(struct packet * p,int left,int right)
{   struct packet temp,pivot;
   int i,j;
   if(left<right)
   {   i = left;
      j = right+1;
      pivot = p[left];
      do
      {   do
         {   i++;
         }while(p[i].seqno <pivot.seqno && i<=right);
         do
         {   j--;
         }while(p[j].seqno >pivot.seqno && j>=left);
         if(i<j)
         {   temp = p[i];
            p[i] = p[j];
            p[j] = temp;
         }
      }while(i<j);

      temp = p[left];
      p[left] =p[j];
      p[j] = temp;

      quicksort(p,left,j-1);
      quicksort(p,j+1,right);
   }
}
```

```
int main()
{   int n,num;
    printf("How many frames : ");
    scanf("%d",&n);
    struct packet p[n];

    for(int i=0;i<n;i++)
    {   again:
        num = rand()%10000;
        for(int j=0;j<=i;j++)
        {   if(p[i].seqno == num)
                goto again;
        }
        p[i].seqno = num;
        printf("Enter message : ");
        scanf("%s",p[i].msg);
        p[i].time = rand();
    }

    printf("\n\nPackets before sorting : \n");
    for(int i=0;i<n;i++)
        printf("Seqno : %d\t Time : %f\t Message : %s\n",p[i].seqno, p[i].time, p[i].msg);

    quicksort(&p,0,n-1);
    printf("\n\nPackets After sorting : \n");
    for(int i=0;i<n;i++)
        printf("Seqno : %d\t Time : %f\t Message : %s\n",p[i].seqno, p[i].time, p[i].msg);

    return 0;
}
```

OUTPUT :



3.Write a C program to implement Frame Sorting

- Each frame contains sequence number, packet id, port number,message, source ip address, destination ip address,
- Sequence number must be random number of 3 digits Regenerate the repeated sequence no
- Sort the frames based on sequence number using insertion sort techq
- Display all info of packet before sort and after sort

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

struct packet
{   int seqno,packid,portno;
   char msg[50],source[20],destin[20];
};

void insertionsort(struct packet * p,int n)
{   struct packet temp;
   int i,j;
   for(i=1;i<n;i++)
   {   temp = p[i];
      for(j=i-1;j>=0 && temp.seqno < p[j].seqno;j--)
         p[j+1] = p[j];
      p[j+1] = temp;
   }
}



int main()
{   int n,num;
   printf("How many frames : ");
   scanf("%d",&n);
   struct packet p[n];

   for(int i=0;i<n;i++)
   {   again:
      num = rand()%1000;
      for(int j=0;j<=i;j++)
      {   if(p[i].seqno == num)
            goto again;
      }
      p[i].seqno = num;
      printf("Enter message, Packet id, Port no, Source ip, Destination ip : ");
      scanf("%s%d%d%s%s",p[i].msg, &p[i].packid, &p[i].portno, p[i].source,p[i].destin);
   }
```

```
    printf("\n\nPackets before sorting : \n");
    for(int i=0;i<n;i++)
        printf("Seqno : %d\t Packet Id : %d\t Port No : %d\t Source : %s\t Destination : %s\t Message :
%s\n",p[i].seqno, p[i].packid, p[i].portno, p[i].source, p[i].destin, p[i].msg);

    insertionsort(&p,n);

    printf("\n\nPackets After sorting : \n");
    for(int i=0;i<n;i++)
        printf("Seqno : %d\t Packet Id : %d\t Port No : %d\t Source : %s\t Destination : %s\t Message :
            %s\n",p[i].seqno, p[i].packid, p[i].portno, p[i].source, p[i].destin, p[i].msg);

    return 0;
}
```

OUTPUT :

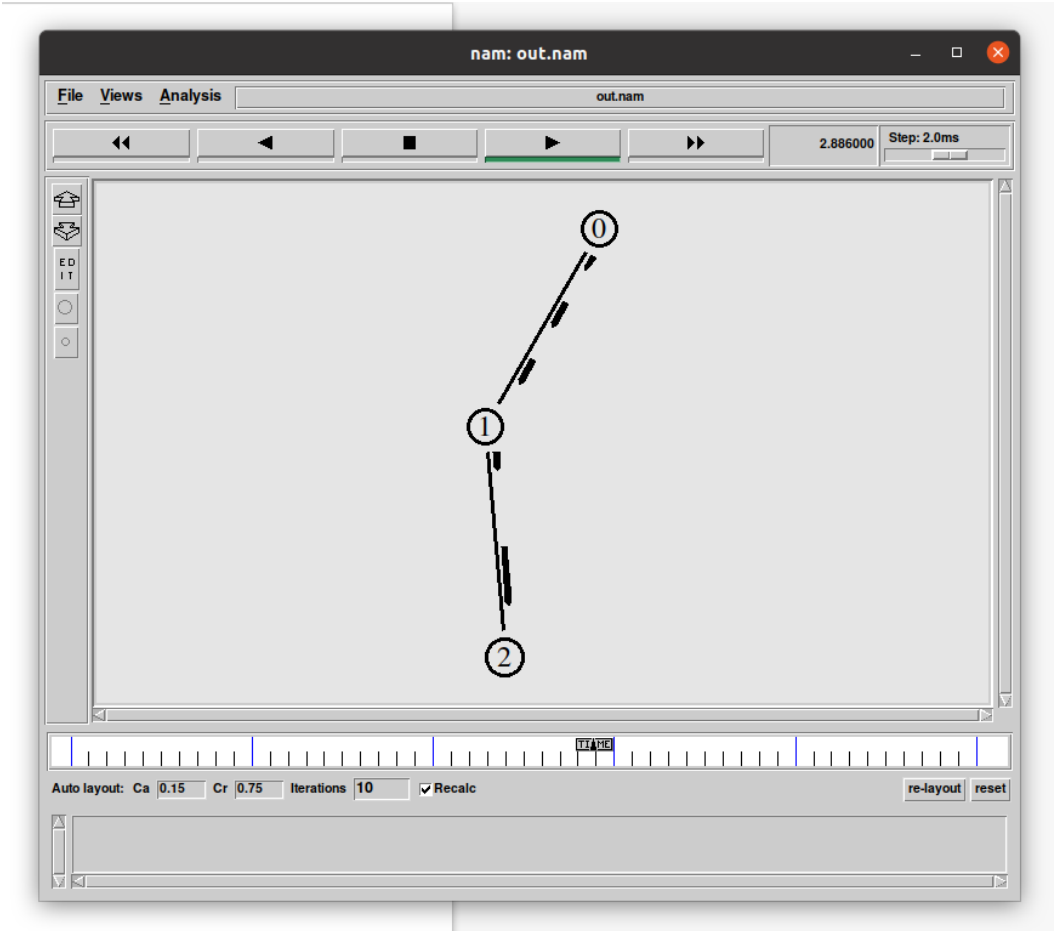
```
How many frames : 5
Enter message, Packet id, Port no, Source ip, Destination ip : hii 2 5 666 333
Enter message, Packet id, Port no, Source ip, Destination ip : hloo 7 4 222 333
Enter message, Packet id, Port no, Source ip, Destination ip : namaskara 9 7 666 555
Enter message, Packet id, Port no, Source ip, Destination ip : chalo 1 4 222 333
Enter message, Packet id, Port no, Source ip, Destination ip : bii 8 6 333 777


Packets before sorting :
Seqno : 383     Packet Id : 2   Port No : 5    Source : 666    Destination : 333      Message : hii
Seqno : 886     Packet Id : 7   Port No : 4    Source : 222    Destination : 333      Message : hloo
Seqno : 777     Packet Id : 9   Port No : 7    Source : 666    Destination : 555      Message : namaskara
Seqno : 915     Packet Id : 1   Port No : 4    Source : 222    Destination : 333      Message : chalo
Seqno : 793     Packet Id : 8   Port No : 6    Source : 333    Destination : 777      Message : bii


Packets After sorting :
Seqno : 383     Packet Id : 2   Port No : 5    Source : 666    Destination : 333      Message : hii
Seqno : 777     Packet Id : 9   Port No : 7    Source : 666    Destination : 555      Message : namaskara
Seqno : 793     Packet Id : 8   Port No : 6    Source : 333    Destination : 777      Message : bii
Seqno : 886     Packet Id : 7   Port No : 4    Source : 222    Destination : 333      Message : hloo
Seqno : 915     Packet Id : 1   Port No : 4    Source : 222    Destination : 333      Message : chalo

Process returned 0 (0x0)   execution time : 52.638 s
Press ENTER to continue.
```

DCN Ass2

Practice Program :

```
set ns [new Simulator]
set nf [open out.nam w]
$ns namtrace-all $nf
set tf [open out.tr w]
$ns trace-all $tf

proc finish {} {
global ns nf tf
$ns flush-trace
close $nf
close $tf
exec nam out.nam &
exit(0)
}

set n0 [$ns node]
set n1 [$ns node]
set n2 [$ns node]
$ns duplex-link $n0 $n1 1Mb 10ms DropTail
$ns duplex-link $n1 $n2 1Mb 5ms DropTail
$ns queue-limit $n0 $n1 50
$ns queue-limit $n1 $n2 50

set udp1 [new Agent/UDP]
$ns attach-agent $n0 $udp1
set null [new Agent/Null]
$ns attach-agent $n2 $null

$ns connect $udp1 $null
set cbr1 [new Application/Traffic/CBR]
$cbr1 set packet-size 500
$cbr1 set interval 0.005
$cbr1 attach-agent $udp1

$ns at 0.5 "$cbr1 start"
$ns at 5.0 "$cbr1 stop"
$ns at 5.5 "finish"
$ns run
```

OUTPUT :


nam: out.nam
File Views Analysis
out.nam
2.886000
Step: 2.0ms
0
1
2
TIME
Auto layout: Ca 0.15 Cr 0.75 Iterations 10 Recalc
re-layout
reset

1. Simulate four node point-to-point network and connect the links as follows :
n0-n1 , n1-n2, n2-n3, n1-n4

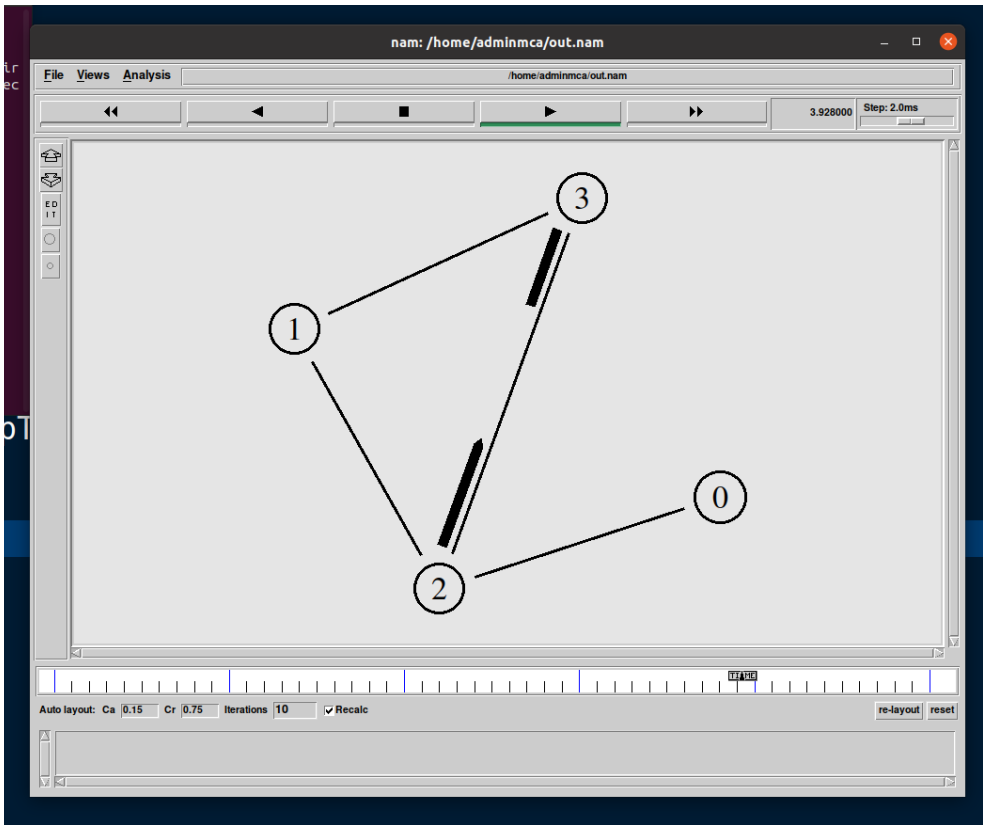
- **Apply UDP Traffic between n0-n3**
- **Apply relevant application over UDP agents chnaging the parameter**
- **Set the queue size vary the bandwidth and find no of packets dropped by UDP**

```
set ns [new Simulator]
set nf [open out.nam w]
$ns namtrace-all $nf
set tf [open out.tr w]
$ns trace-all $tf

proc finish {} {
global ns nf tf
$ns flush-trace
close $nf
close $tf
exec nam out.nam &
exit(0)
}

set n0 [$ns node]
set n1 [$ns node]
set n2 [$ns node]
set n3 [$ns node]

$ns duplex-link $n0 $n2 100Mb 10ms DropTail
$ns duplex-link $n1 $n2 1Mb 5ms DropTail
$ns duplex-link $n2 $n3 1Mb 5ms DropTail
$ns duplex-link $n1 $n3 1Mb 5ms DropTail
$ns queue-limit $n0 $n2 50
$ns queue-limit $n1 $n2 50
$ns queue-limit $n2 $n3 50
$ns queue-limit $n1 $n3 50

set udp1 [new Agent/UDP]
$ns attach-agent $n0 $udp1
set null [new Agent/Null]
$ns attach-agent $n3 $null

$ns connect $udp1 $null
set cbr1 [new Application/Traffic/CBR]
```

```
$cbr1 set packet-size 200
$cbr1 set interval 0.005
$cbr1 attach-agent $udp1

$ns at 0.5 "$cbr1 start"
$ns at 5.0 "$cbr1 stop"
$ns at 5.5 "finish"
$ns run
```

```
#include<stdio.h>
void main()
{       FILE *fp = fopen("/home/student/Documents/out.tr","r");
        char x='d';
        char c;
        int count=0;

        if(!fp)
                printf("Can't open file");

        while( (c=getc(fp)) != EOF )
        {       if( c== 'd')
                        count++;
        }
        printf("packet dropped %d",count);
        fclose(fp);
}
```

OUTPUT :


nam: /home/adminmca/out.nam
File Views Analysis
/home/adminmca/out.nam
3.928000
Step: 2.0ms
3
1
0
2
Auto layout: Ca 0.15 Cr 0.75 Iterations 10 Recalc
re-layout
reset

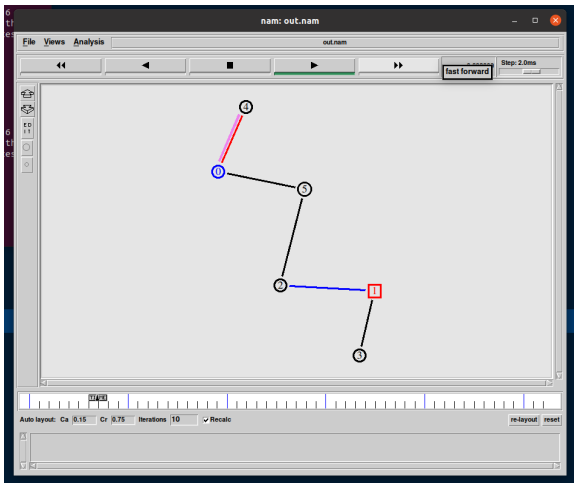
3. Simulate four node point-to-point network and connect the links as follows :
n1-n5, n2-n3, n2-n4, n1-n6, n3-n6

- **Apply FTP agent between n3-n5 and TELNET between n4-n6**
- **Apply relevant application over TCP agents chnaging the parameter**
- **Set the queue size vary the bandwidth and find no of packets dropped by TCP**
- **Change node color, Link color, Packet Color, Chnage node position and shape**

```
set ns [new Simulator]
$ns color 1 violet
set nf [open out.nam w]
$ns namtrace-all $nf
set tf [open out.tr w]
$ns trace-all $tf

proc finish {} {
global ns nf tf
$ns flush-trace
close $nf
close $tf
exec nam out.nam &
exit(0)
}

set n1 [$ns node]
set n2 [$ns node]
set n3 [$ns node]
set n4 [$ns node]
set n5 [$ns node]
set n6 [$ns node]

$ns duplex-link $n1 $n5 0.011Mb 10ms DropTail
$ns duplex-link $n2 $n3 1Mb 5ms DropTail
$ns duplex-link $n2 $n4 1Mb 5ms DropTail
$ns duplex-link $n1 $n6 1Mb 5ms DropTail
$ns duplex-link $n3 $n6 1Mb 5ms DropTail

$ns duplex-link-op $n1 $n5 color red
$ns duplex-link-op $n2 $n3 color blue

$ns duplex-link-op $n1 $n5 orient left
$ns duplex-link-op $n2 $n3 orient right
```

```
$ns queue-limit $n1 $n5 50
$ns queue-limit $n2 $n3 0.011
$ns queue-limit $n2 $n4 50
$ns queue-limit $n1 $n6 50
$ns queue-limit $n3 $n6 50

$n1 color blue
$n2 color red

$n1 shape circle
$n2 shape box

set tcp1 [new Agent/TCP]
$ns attach-agent $n3 $tcp1
$tcp1 set fid_ 1
set tcpsink0 [new Agent/TCPSink]
$ns attach-agent $n5 $tcpsink0
$ns connect $tcp1 $tcpsink0

set tcp2 [new Agent/TCP]
$ns attach-agent $n4 $tcp2
set tcpsink1 [new Agent/TCPSink]
$ns attach-agent $n6 $tcpsink1
$ns connect $tcp2 $tcpsink1

set ftp0 [new Application/FTP]
set telnet0 [new Application/Telnet]

$ftp0 set packet-size 200
$ftp0 set interval 0.005
$ftp0 attach-agent $tcp1
$ns at 0.5 "$ftp0 start"
$ns at 4.0 "$ftp0 stop"

$telnet0 set packet-size 200
$telnet0 set interval 0.0005
$telnet0 attach-agent $tcp2
$ns at 0.5 "$telnet0 start"
$ns at 4.0 "$telnet0 stop"

$ns at 5.5 "finish"
$ns run
```

OUTPUT :


nam: out.nam
File Views Analysis
out.nam
fast forward
Step: 2.0ms
Auto layout: Ca 0.15 Cr 0.75 Iterations 10 Recalc
re-layout reset

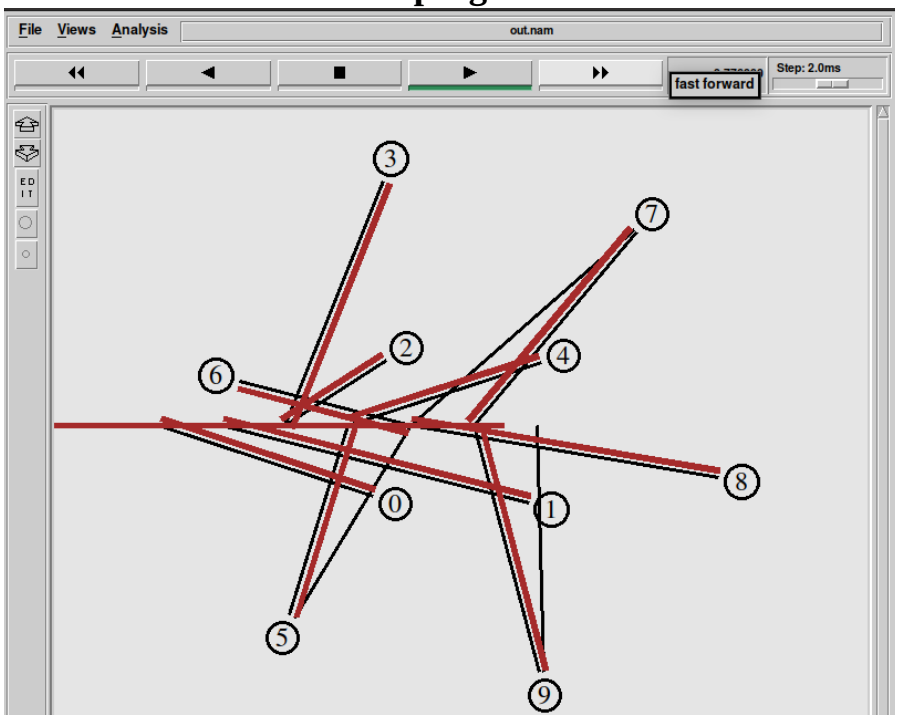
DCN Ass3

1. Simulate an Ethernet LAN with 10 nodes consisting of TCP traffic, Telnet traffic generator.

```
set ns [new Simulator]
set nf [open out.nam w]
$ns namtrace-all $nf
set tf [open out.tr w]
$ns trace-all $tf

$ns color 1 red
$ns color 2 brown

proc finish {} {
global ns nf tf
$ns flush-trace
close $nf
close $tf
exec nam out.nam &
exit(0)
}

set n0 [$ns node]
set n1 [$ns node]
set n2 [$ns node]
set n3 [$ns node]
set n4 [$ns node]
set n5 [$ns node]
set n6 [$ns node]
set n7 [$ns node]
set n8 [$ns node]
set n9 [$ns node]

$ns make-lan "$n0 $n1 $n2 $n3 $n4 $n5 $n6 $n7 $n8 $n9"  0.1Mb 0.1ms LL
Queue/DropTail Mac/802_3

set tcp0 [new Agent/TCP]
$ns attach-agent $n3 $tcp0
$tcp0 set fid_ 1
set tcpsink0 [new Agent/TCPSink]
$ns attach-agent $n5 $tcpsink0
$ns connect $tcp0 $tcpsink0
```

```
set telnet0 [new Application/Telnet]
$telnet0 set packet-size 500
$telnet0 set interval 0.005
$telnet0 attach-agent $tcp0
$ns at 0.5 "$telnet0 start"
$ns at 2.0 "$telnet0 stop"

set tcp1 [new Agent/TCP]
$ns attach-agent $n3 $tcp1
$tcp1 set fid_ 2
set tcpsink0 [new Agent/TCPSink]
$ns attach-agent $n5 $tcpsink0
$ns connect $tcp1 $tcpsink0

set telnet1 [new Application/Telnet]
$telnet1 set packet-size 500
$telnet1 set interval 0.005
$telnet1 attach-agent $tcp1
$ns at 0.5 "$telnet1 start"
$ns at 2.0 "$telnet1 stop"

$ns at 5.0 "finish"
$ns run
```

OUTPUT： sudo ns a3prog1.tcl



2. Simulate an Ethernet LAN with 12 nodes consisting of multiple traffic(TCP and UDP).

- Apply different path for multiple traffic
- Apply CBR,FTP,TELNET application
- Find the number of packets dropped.
- change the shape and color of every source and destination node.
- Changethe color of packet for multiple Traffic

```
set ns [new Simulator]
$ns color 1 brown
$ns color 2 green
set nf [open out.nam w]
$ns namtrace-all $nf
set tf [open out.tr w]
$ns trace-all $tf

proc finish {} {
global ns nf tf
$ns flush-trace
close $nf
close $tf
exec nam out.nam &
exit (0)
}

set n0 [$ns node]
set n1 [$ns node]
set n2 [$ns node]
set n3 [$ns node]
set n4 [$ns node]
set n5 [$ns node]
set n6 [$ns node]
set n7 [$ns node]
set n8 [$ns node]
set n9 [$ns node]
set n10 [$ns node]
set n11 [$ns node]

$ns make-lan "$n0 $n1 $n2 $n3 $n4 $n5 $n6 $n7 $n8 $n9 $n10 $n11"  0.22Mb 10ms LL
Queue/DropTail Mac/802_3

$n0 color red
```

```
$n1 color red
$n2 color red
$n3 color red
$n4 color blue
$n5 color blue
$n6 color blue
$n7 color blue
$n8 color blue
$n9 color purple
$n10 color purple
$n11 color purple

set udp1 [new Agent/UDP]
$ns attach-agent $n0 $udp1
set null1 [new Agent/Null]
$ns attach-agent $n3 $null1
$ns connect $udp1 $null1

set tcp1 [new Agent/TCP]
$ns attach-agent $n4 $tcp1
$tcp1 set fid_ 1
set tcpsink0 [new Agent/TCPSink]
$ns attach-agent $n8 $tcpsink0
$ns connect $tcp1 $tcpsink0

set tcp2 [new Agent/TCP]
$ns attach-agent $n9 $tcp2
$tcp2 set fid_ 2
set tcpsink1 [new Agent/TCPSink]
$ns attach-agent $n11 $tcpsink1
$ns connect $tcp2 $tcpsink1

set cbr1 [new Application/Traffic/CBR]
$cbr1 set packet-size 200
$cbr1 set interval 0.005
$cbr1 attach-agent $udp1
$ns at 0.5 "$cbr1 start"
$ns at 4.0 "$cbr1 stop"

set ftp0 [new Application/FTP]
$ftp0 set packet-size 200
$ftp0 set interval 0.005
$ftp0 attach-agent $tcp1
$ns at 0.5 "$ftp0 start"
$ns at 4.0 "$ftp0 stop"
```

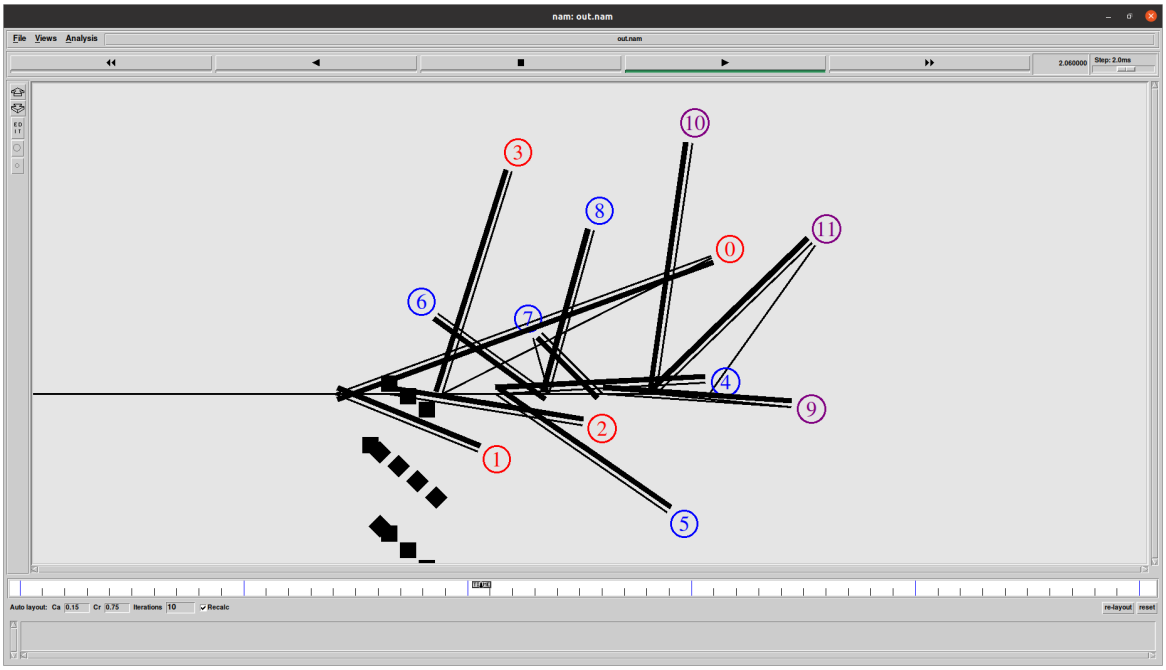
```
set telnet0 [new Application/Telnet]
$telnet0 set packet-size 200
$telnet0 set interval 0.005
$telnet0 attach-agent $tcp2
$ns at 0.5 "$telnet0 start"
$ns at 4.5 "$telnet0 stop"

$ns at 5.0  "finish"
$ns run
```

packet.c :

```
#include<stdio.h>
 void main()
 {       FILE *fp=fopen("/home/student/Documents/out.tr","r");
         char x='d';
         char c;
         int count=0;

         if(!fp)
                 printf("can't open file");

         while((c=getc(fp))!=EOF)
                  if(c=='d')
                          count++;

         printf("packet dropped %d",count);
         fclose(fp);
 }
```

OUTPUT :

gcc packet.c **sudo ns a3prog2.tcl**



(An off-Campus Institution of NITTE (DEEMED TO BE UNIVERSITY), MANGALORE)

DCN Ass3 – Error Detection

1. **Write a program for error detection code using Parity check technique.**

**************SENDER**************

Enter Message : 1011
1. ODD parity (0)
2. even parity (1)

Enter a option : 1
Message to be Transmitted : 10110

Do You Want to Introduce error(Y/N) : Y
Enter the Position : 1 (condition : position between 1 to T size)

**************RECEIVER**************

Message received at the Receiver : 00110

ERROR in MESSAGE

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

int main() {
   char msg[20];
   int option, position, count = 0, i;

   printf("************SENDER************");
   printf("\nEnter Message: ");
```

```
scanf("%s", msg);
printf("1. ODD parity (0)\t2. Even parity (1)\n");
printf("Enter an option: ");
scanf("%d", &option);
int pos = strlen(msg);
if(option == 1)
   msg[pos] = '0';
else if(option == 2)
   msg[pos] = '1';
printf("Message to be Transmitted: %s ", msg);
printf("\nDo You Want to Introduce error (Y/N): ");
char choice;
scanf(" %c", &choice);
if (choice == 'Y' || choice == 'y') {
   printf("Enter the Position (condition: position between 1 to %d size): ", pos);
   scanf("%d", &position);
   if (position > 0 && position <= pos) {
      if (msg[position - 1] == '0')
         msg[position - 1] = '1';
      else
         msg[position - 1] = '0';
   }
}

printf("\n***********RECEIVER***********\n");
printf("Message received at the Receiver: %s\n", msg);
for(i = 0; i < pos; i++) {
   if(msg[i] == '1')
      count++;
}

if(msg[pos] == '0') {
   if(count % 2 != 0)
      printf("\nMESSAGE WITHOUT ERROR");
   else
      printf("\nERROR IN MESSAGE");
}
else if(msg[pos] == '1') {
```

```
    if(count % 2 == 0)
      printf("\nMESSAGE WITHOUT ERROR\n");
    else
      printf("\nERROR IN MESSAGE\n");
  }

  return 0;
}
```

OUTPUT 1 :

**************SENDER**************
Enter Message: 01010111010
1. ODD parity (0) 2. Even parity (1)
Enter an option: 1
Message to be Transmitted: 010101110100
Do You Want to Introduce error (Y/N): y
Enter the Position (condition: position between 1 to 11 size): 4

**************RECEIVER**************
Message received at the Receiver: 010001110100
MESSAGE WITHOUT ERROR

OUTPUT 2 :

**************SENDER**************
Enter Message: 101010111
1. ODD parity (0) 2. Even parity (1)
Enter an option: 2
Message to be Transmitted: 1010101111
Do You Want to Introduce error (Y/N): y
Enter the Position (condition: position between 1 to 9 size): 3

**************RECEIVER**************
Message received at the Receiver: 1000101111
ERROR IN MESSAGE

2. Write a program for error detecting code using CRC technique

**************SENDER**************
Enter Message : 101010
Enter Pattern : 1111
(Condition : pattern size should be less than Message or else re enter pattern)

Given Message is : 101010
Given Pattern is : 1111
Message size is : 6
Pattern Size is : 4
FCS : 3bit
After Appending Q is :101010000
Remainder R is :100
Message to be Transmitted T is :101010100
Do You Want to Introduce error(Y/N) : Y
Enter the Position : 2 (condition : position between 1 to T size)

**************RECEIVER**************
Message received at the Receiver : 111010100
Remainder R is : 100
ERROR IN MESSAGE

```
#include <stdio.h>
#include <string.h>
int main()
{
 char message[50], pattern[50], msg[50], reminder[20];
 int i, j, msglen, patlen, position, error_flag = 0;

 printf("************SENDER************\n");
 printf("Enter Message : ");
 scanf("%s", message);
 printf("Enter Pattern : ");
 scanf("%s", pattern);
 msglen = strlen(message);
```

```
patlen = strlen(pattern);
if (patlen >= msglen)
{       printf("Pattern size should be less than Message size. Please re-enter the pattern.\n");
        return 0;
}
printf("\nGiven Message is: %s\n", message);
printf("Given Pattern is: %s\n", pattern);
printf("Message size is: %d\n", msglen);
printf("Pattern Size is: %d\n", patlen);
printf("FCS: %dbit\n", patlen - 1);
strcpy(msg, message);
for (i = 0; i < patlen - 1; i++)
        msg[msglen + i] = '0';
msg[msglen + i] = '\0';
printf("After Appending Q is: %s\n", msg);
for (i = 0; i < msglen; i++)
         if (msg[i] == '1')
                 for (j = 0; j < patlen; j++)
                         if (msg[i + j] == pattern[j])
                                 msg[i + j] = '0';
                         else
                                 msg[i + j] = '1';

printf("Remainder R is: %s\n", &msg[msglen]);
strcat(message, msg + msglen);
printf("Message to be Transmitted T is: %s\n", message);
char choice;
printf("\nDo You Want to Introduce error (Y/N): ");
scanf(" %c", &choice);
if (choice == 'Y' || choice == 'y')
{       printf("Enter the Position (between 1 to T size): ");
        scanf("%d", &position);
        if (position > 0 && position <= strlen(message))
                message[position - 1] = (message[position - 1] == '0') ? '1' : '0';
        else
        {       printf("Invalid position.\n");
                return 0;
        }
```

```
}

printf("\n************RECEIVER************\n");
printf("Message received at the Receiver: %s\n", message);
for (i = 0; i < msglen; i++)
        if (message[i] == '1')
                for (j = 0; j < patlen; j++)
                        if (message[i + j] == pattern[j])
                                message[i + j] = '0';
                        else
                                message[i + j] = '1';

strcpy(reminder,message+msglen);
printf("Remainder R is: %s\n", reminder);
for (i = 0; i < strlen(reminder); i++)
        if (reminder[i] == '1')
        {       error_flag = 1;
                break;
        }
if (error_flag == 1)
        printf("\nERROR IN MESSAGE\n");
else
        printf("\nNO ERROR IN MESSAGE\n");

return 0;
}
```

OUTPUT 1 :

**************SENDER**************

Enter Message : 1010111101

Enter Pattern : 111

Given Message is: 1010111101

Given Pattern is: 111

Message size is: 10

Pattern Size is: 3

FCS: 2bit

After Appending Q is: 101011110100
Remainder R is: 11
Message to be Transmitted T is: 101011110111

Do You Want to Introduce error (Y/N): y
Enter the Position (between 1 to T size): 4

**************RECEIVER**************
Message received at the Receiver: 101111110111
Remainder R is: 11

ERROR IN MESSAGE

OUTPUT 2 :

**************SENDER**************
Enter Message : 1010001101
Enter Pattern : 110101

Given Message is: 1010001101
Given Pattern is: 110101
Message size is: 10
Pattern Size is: 6
FCS: 5bit
After Appending Q is: 101000110100000
Remainder R is: 01110
Message to be Transmitted T is: 101000110101110

Do You Want to Introduce error (Y/N): n

**************RECEIVER**************
Message received at the Receiver: 101000110101110
Remainder R is: 00000

NO ERROR IN MESSAGE

DCN Ass5 - NS2-PING

1. Simulate the transmission of ping messaged over a network topology consisting of 6 nodes.

```
set ns [new Simulator]
set nf [open out.nam w]
$ns namtrace-all $nf
set tf [open out.tr w]
$ns trace-all $tf

proc finish {} {
global ns nf tf
$ns flush-trace
close $nf
close $tf
exec nam out.nam &
exit(0)
}

set n1 [$ns node]
set n2 [$ns node]
set n3 [$ns node]
set n4 [$ns node]
set n5 [$ns node]
set n6 [$ns node]

$ns duplex-link $n1 $n5 1Mb 10ms DropTail
$ns duplex-link $n2 $n3 1Mb 5ms DropTail
$ns duplex-link $n2 $n4 1Mb 5ms DropTail
$ns duplex-link $n1 $n6 1Mb 5ms DropTail
$ns duplex-link $n3 $n6 1Mb 5ms DropTail

$ns queue-limit $n1 $n5 50
$ns queue-limit $n2 $n3 50
$ns queue-limit $n2 $n4 50
$ns queue-limit $n1 $n6 50
$ns queue-limit $n3 $n6 50

Agent/Ping instproc recv {from rtt} {
$self instvar node_
```

puts "node [$node_ id] has recieved ping answer $from with round trip time $rtt ms"}

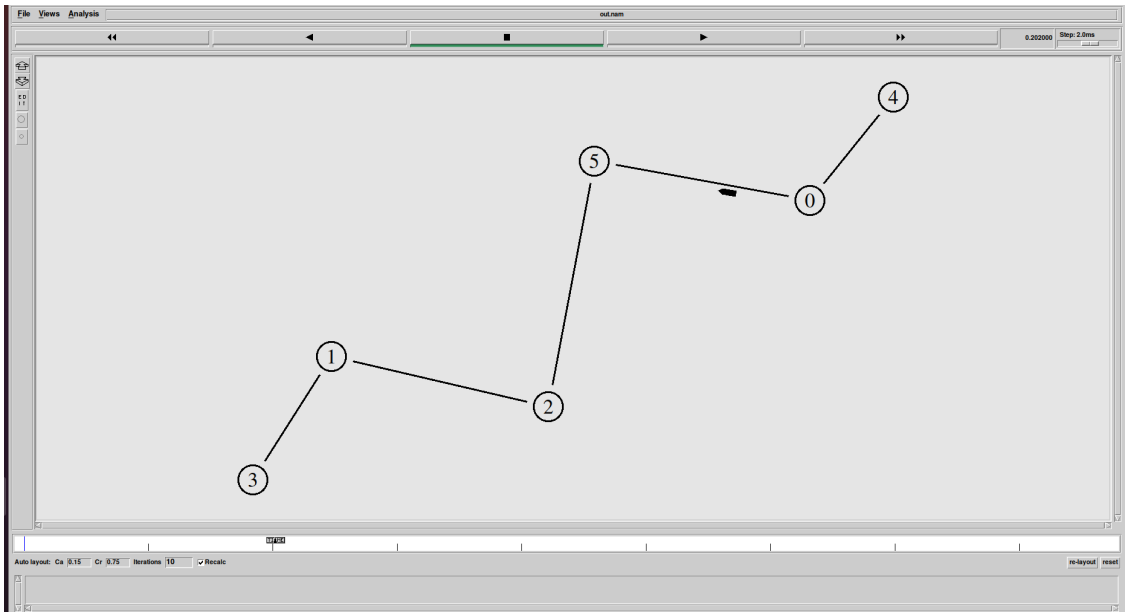
set p1 [new Agent/Ping]
$ns attach-agent $n1 $p1

set p2 [new Agent/Ping]
$ns attach-agent $n3 $p2

$ns connect $p1 $p2
$ns at 0.2 "$p1 send"
$ns at 0.6 "$p2 send"
$ns at 0.4 "$p1 send"
$ns at 0.8 "$p2 send"

$ns at 1.0 "finish"
$ns run

OUTPUT :



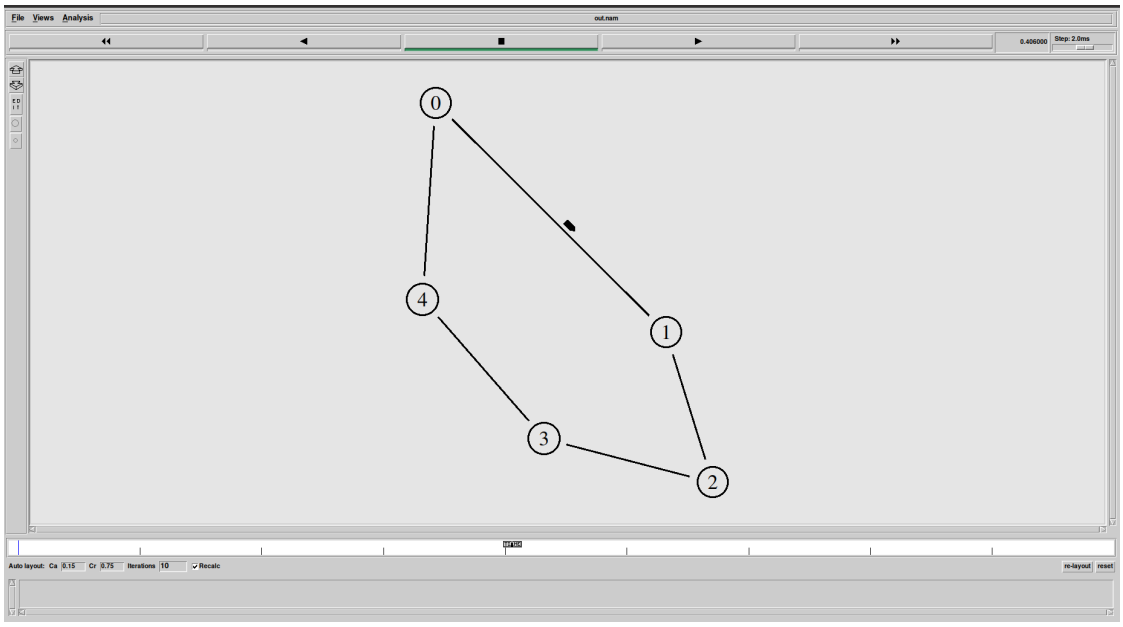
2. Write a Program to implement a Ring topology using less number of statements in TCL Language apply PING Agent

```
set ns [new Simulator]
set nf [open out.nam w]
$ns namtrace-all $nf
set tf [open out.tr w]
$ns trace-all $tf

proc finish {} {
global ns nf tf
$ns flush-trace
close $nf
close $tf
exec nam out.nam &
exit 0
}

set numnodes 5

for { set i 0 } {$i<$numnodes} {incr i} {
set nodes($i) [$ns node]
}

for { set i 0 } {$i<$numnodes} {incr i} {
set j [expr {($i+1)%$numnodes}]
$ns duplex-link $nodes($i) $nodes($j) 1Mb 10ms DropTail }

Agent/Ping instproc recv {from rtt} {
$self instvar node_
puts "node [$node_ id] has received ping answer $from with round trip time of $rtt
ms"}

set p1 [new Agent/Ping]
$ns attach-agent $nodes(0) $p1

set p2 [new Agent/Ping]
$ns attach-agent $nodes(2) $p2

$ns connect $p1 $p2

$ns at 0.2 "$p1 send"
$ns at 0.6 "$p2 send"
$ns at 0.4 "$p1 send"
```

$ns at 0.8 "$p2 send"

$ns at 1.0 "finish"
$ns run

OUTPUT :

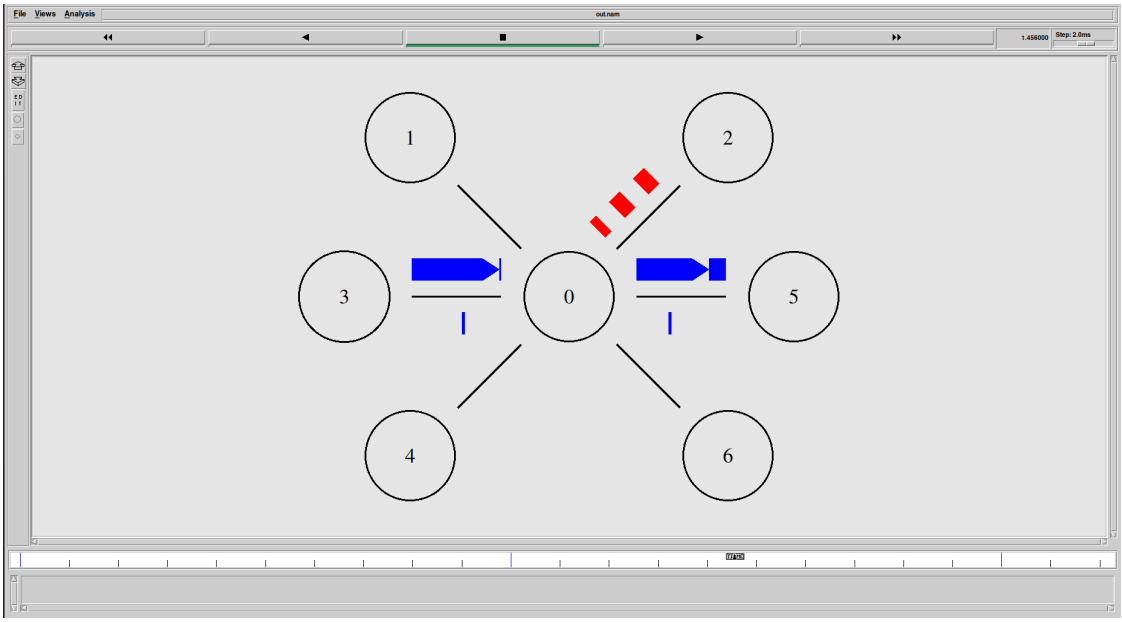


3. Write a Program to implement a Star topology using less number of statements in TCL Language using UDP and TCP Agent

```
set ns [new Simulator]
set nf [open out.nam w]
$ns namtrace-all $nf
set tf [open out.tr w]
$ns trace-all $tf

$ns color 1 red
$ns color 2 blue

proc finish {} {
global ns nf tf
$ns flush-trace
close $nf
close $tf
exec nam out.nam &
exit(0)
}

set numnodes 7

for { set i 0 } {$i<$numnodes} {incr i} {
set n($i) [$ns node]
}

for { set i 1 } {$i<7} {incr i} {
$ns duplex-link $n(0) $n($i) 1Mb 10ms DropTail }

$ns duplex-link-op $n(0) $n(1) orient left-up
$ns duplex-link-op $n(0) $n(2) orient right-up

$ns duplex-link-op $n(0) $n(3) orient left
$ns duplex-link-op $n(0) $n(4) orient left-down

$ns duplex-link-op $n(0) $n(5) orient right
$ns duplex-link-op $n(0) $n(6) orient right-down

set udp0 [new Agent/UDP]
$ns attach-agent $n(0) $udp0
$udp0 set fid_ 1
set null1 [new Agent/Null]
$ns attach-agent $n(2) $null1
$ns connect $udp0 $null1
```

```
set cbr0 [new Application/Traffic/CBR]
$cbr0 set packet-size 500
$cbr0 set interval 0.005
$cbr0 attach-agent $udp0
$ns at 0.3 "$cbr0 start"
$ns at 1.5 "$cbr0 stop"

set tcp0 [new Agent/TCP]
$ns attach-agent $n(3) $tcp0
$tcp0 set fid_ 2
set tcpsink0 [new Agent/TCPSink]
$ns attach-agent $n(5) $tcpsink0
$ns connect $tcp0 $tcpsink0

set ftp0 [new Application/FTP]
$ftp0 set packet-size 500
$ftp0 set interval 0.01
$ftp0 attach-agent $tcp0
$ns at 0.5 "$ftp0 start"
$ns at 2.0 "$ftp0 stop"

$ns at 2.5 "finish"
$ns run
```

OUTPUT :



(An off-Campus Institution of NITTE (DEEMED TO BE UNIVERSITY), MANGALORE)

DCN Ass6 – Error Correction

1. **Implement hamming distance method for error correction. Input binary data and convert input data into code words in transmitted message. Display received message and display corrected message in case of error.**

   **************SENDER**************
   Enter Data : 1011001

   Out Put
   Code Word :0110
   Transmission data T : 10101101110

   Do You Want to Introduce error(Y/N) : Y
   Enter the Position : 2 (condition : position between 1 to T size)

   **************RECEIVER**************
   Message received at the Receiver : 10101101100
   Code Word :0010 Error in 2nd position

   After Correction of Data : 10101101110

```
#include<stdio.h>
#include<string.h>
#include<math.h>

int power(int a,int b)
{       int r=1;
        for(int i=0; i<b; i++)
                r=r*a;
```

```
        return r;
}

int check(char message [],int n)
{       int len=strlen(message),flag=0,i=n-1,count=0;
        while(i<len)
        {       int k=0;
                while(k<n)
                {       if(message[i++]=='1')
                                count++;
                        k++;
                }
                k=0;
                while(k<n)
                {       i++;
                        k++;
                }
        }
        return count;
}

void main()
{       char mess[20],nmess[50],hcode[10];
        int j=0,k=0,i=0;

        printf("*** SENDER ***\n");
        printf("Enter Data:");
        scanf("%s",&mess);
        int len=strlen(mess);

        while(mess[i]!='\0')
        {       if((j+1)==power(2,k))
                {       nmess[j++]=' ';
                        nmess[j]='\0';
                        k++;
                }
                else
                {       nmess[j++]=mess[i++];
```

```
                nmess[j]='\0';
        }
}

int l=0;
j=0;
printf("\nAfter appending Empty space for message:%s\n",nmess);
len=strlen(nmess);

while(power(2,l)<len)
{       int count=check(nmess,power(2,l));
        if(count%2==0)
                hcode[j++]='0';
        else
                hcode[j++]='1';
        l++;
}

printf("\nCode Word:%s\n",hcode);
j=0,i=0,k=0;
while(nmess[j]!='\0')
{       if((j+1)==power(2,k))
        {       nmess[j]=hcode[i++];
                k++;
        }
        j++;
}

int pos;
char ch;
printf("\nTransmission Data:%s",nmess);
printf("\nDo you want to introduce error (y/n):");
scanf(" %c",&ch);

if(ch=='y')
{       printf("\nEnter the position:");
        x:
        scanf("%d",&pos);
```

```
            if(pos>len)
                goto x;
            else
                nmess[pos-1]=(nmess[pos-1]=='1')?'0':'1';
        }

        l=0,j=0;
        printf("*** RECEIVER ***\n");
        printf("Message received at the receiver:%s\n",nmess);

        while(power(2,l)<len)
        {       int count=check(nmess,power(2,l));
            if(count%2==0)
                hcode[j++]='0';
            else
                hcode[j++]='1';
            l++;
        }
        len=strlen(hcode);

        for (i = 0; i < len/2; i++)
        {       int temp = hcode[i];
            hcode[i] = hcode[len - i - 1];
            hcode[len - i - 1] = temp;
        }

        printf("Code Word:%s\n",hcode);
        int result=0;
        pos=0;

        for(i=len-1;i>=0;i--,pos++)
            if(hcode[i]=='1')
                result+=power(2,pos);

        printf("Error in %d position\n",result);
        nmess[result-1]=(nmess[result-1]=='1')?'0':'1';
        printf("After Correction of Data:%s",nmess);
}
```

OUTPUT1 :

***** SENDER *****

Enter Data:10100101111

After appending Empty space for message: 1 010 0101111

Code Word:1111

Transmission Data:111101010101111

Do you want to introduce error (y/n):y

Enter the position:3

***** RECEIVER *****

Message received at the receiver:110101010101111

Code Word0011

Error in 3 position

After Correction of Data:111101010101111

OUTPUT 2:

***** SENDER *****

Enter Data:010101011110000

After appending Empty space for message: 0 101 0101111 0000

Code Word:00010

Transmission Data:000010110101111100000

Do you want to introduce error (y/n):n

***** RECEIVER *****

Message received at the receiver:000010110101111100000

Code Word:00000

Error in 0 position

After Correction of Data:000010110101111100000

2. Write a program to implement Leaky Bucket algorithm.

Enter the Size of the Bucket : 20
Enter the out rate : 5
Enter the time Interval : 5
Enter th number of Packet : 5
Enter the time and Size of Packet 1 : 2 6
Enter the time and Size of Packet 2 : 3 10
Enter the time and Size of Packet 3 : 14 4
Enter the time and Size of Packet 4 : 16 15
Enter the time and Size of Packet 5 : 21 10

Operation	Time	Filled	Free-Space
Insert	2	6	14
Insert	3	16	4
Remove	5	11	9
Remove	10	6	14
Insert	14	10	10
Remove	15	5	15
Insert	16	20	0
Remove	20	15	5
Insert	21	overflow	
Remove	25	10	10
Remove	30	5	15
Remove	35	0	20

```
#include<stdio.h>

struct ts
{       int t, ps;
}s[20];

void main()
{       int n,outrate,time,np,i;

        printf("Enter the size of the bucket:");
        scanf("%d",&n);
        printf("Enter the outrate of the bucket:");
        scanf("%d",&outrate);
        printf("Enter the time interval of leak:");
        scanf("%d",&time);
```

```
printf("Enter the number of packets:");
scanf("%d",&np);

for(i=0;i<np;i++)
{       printf("Enter the time and size of packet%d: ",i);
        scanf("%d",&s[i].t);
        scanf("%d",&s[i].ps);
}

printf("oper  time  filled  free-space\n");
int t[30],filled[30],free[30],k=1,j=0,flag;
i=0;
do
{       flag=0;
        if(i>=np)
        {       l:
                {       printf("remove  ");
                        t[j]=time*k;
                        filled[j]=filled[j-1]-outrate;

                        printf("%d     %d     ",t[j],filled[j]);
                        k++;
                }
        }
        else if(s[i].t<time*k) //inserting
        {       printf("insert  ");
                t[j]=s[i].t;

                if(j==0 && s[i].ps<=n)
                {       filled[j]=s[i].ps;
                        printf("%d     %d     ",t[j],filled[j]);
                }
                else if((filled[j-1]+s[i].ps)<=n)
                {       filled[j]=filled[j-1]+s[i].ps;
                        printf("%d     %d     ",t[j],filled[j]);
                }
                else
                {       flag=1;
```

```
                        printf("%d     overflow    ",t[j]);
                }
        }
        else
        {       i--;
                goto l;
        }

        if(flag==0)
        {       free[j]=n-filled[j];
                printf("%d\n",free[j]);
        }
        else
        {       printf("\n");
                filled[j]=filled[j-1];
        }
        j++;
        i++;
    }while(filled[j-1]!=0);
}
```

OUTPUT :

Enter the size of the bucket:20
Enter the outrate of the bucket:5
Enter the time interval of leak:5
Enter the number of packets:5
Enter the time and size of packet0: 2 6
Enter the time and size of packet1: 3 10
Enter the time and size of packet2: 14 4
Enter the time and size of packet3: 16 15
Enter the time and size of packet4: 21 10
oper time filled free-space
insert 2 6 14
insert 3 16 4
remove 5 11 9
remove 10 6 14
insert 14 10 10
remove 15 5 15
insert 16 20 0
remove 20 15 5
insert 21 overflow
remove 25 10 10
remove 30 5 15
remove 35 0 20

NITTE (Deemed to be University) | NMAM INSTITUTE OF TECHNOLOGY

(An off-Campus Institution of NITTE (DEEMED TO BE UNIVERSITY), MANGALORE)

DCN Ass7 – Shortest Path – Non Adaptive

1.Write a program to find the shortest path using Dijkstra's Algorithm

Enter a number of Node : 4

Enter distance to A to A : 0
Enter distance to A to B : 10
Enter distance to A to C : 0
Enter distance to A to D : 0
Enter distance to B to A : 10
Enter distance to B to B : 0
Enter distance to B to C : 3
Enter distance to B to D : 20
Enter distance to C to A : 0
Enter distance to C to B :3
Enter distance to C to C :0
Enter distance to C to D :10
Enter distance to D to A :0
Enter distance to D to B :20
Enter distance to D to C :10
Enter distance to D to D :0

Enter starting Node – A
Distance of Node B – 10
Route A -> B
Distance of Node C – 13
Route A -> B→ C
Distance of Node D – 23
Route A -> B ->C->D

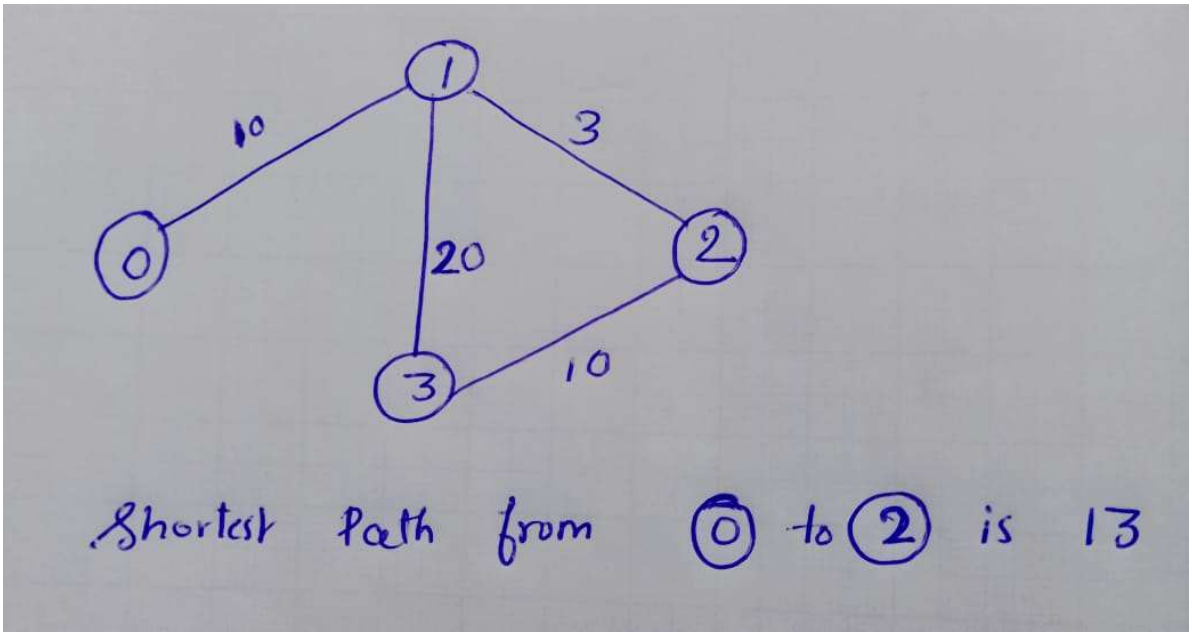
Enter a Destination Node : C
The shortest route is - A -> B →C
Distance is 13

```
#include<stdio.h>
#define INFINITY 9999
#define MAX 100

void Dijkstra(int cost[MAX][MAX], int n, int start, int destination)
{       int distance[MAX], pred[MAX];
        int visited[MAX], count, mindistance, nextnode, i, j;

        for (i = 0; i < n; i++)// Creating cost matrix
                for (j = 0; j < n; j++)
                        if (cost[i][j] == 0)
                                cost[i][j] = INFINITY;
                        else
                                cost[i][j] = cost[i][j];

        for (i = 0; i < n; i++)
        {       distance[i] = cost[start][i];
                pred[i] = start;
                visited[i] = 0;
        }

        distance[start] = 0;
        visited[start] = 1;
        count = 1;

        while (count < n-1)
        {       mindistance = INFINITY;

                for (i = 0; i < n; i++)
                        if (distance[i] < mindistance && !visited[i])
                        {       mindistance = distance[i];
                                nextnode = i;
                        }

                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++;
        }

        for (i = 0; i < n; i++)     // Printing the distance of each node from the source
                if (i != start)
                {       printf("Distance of Node %d: %d\n", i, distance[i]);
                        printf("Route: %d", start);
                        int path = pred[i];

                        while (path != start)
                        {
                                printf(" --> %d ", path);
                                path = pred[path];
                        }
                        printf(" --> %d\n", i);
                }

        // Printing the shortest path
        printf("Shortest distance from source %d to destination %d: %d\n", start, destination,
        distance[destination]);
        printf("Shortest path: %d ", start);
        int path = pred[destination];

        while (path != start)
        {       printf(" --> %d ", path);
                path = pred[path];
        }
        printf(" --> %d\n", destination);
}

int main()
{
        int cost[MAX][MAX], i, j, n, u, destination;
        printf("Enter total number of nodes in Network : ");
        scanf("%d", &n);
```

```
        for (i = 0; i < n; i++)
                for (j = 0; j < n; j++)
                        if (i > j)
                        {       printf("Distance between %d and %d: %d\n", i, j, cost[j][i]);
                                cost[i][j] = cost[j][i];
                        }
                        else
                        {       printf("Distance between %d and %d:", i, j);
                                scanf("%d", &cost[i][j]);
                        }

        printf("Enter the starting node:");
        scanf("%d", &u);

        printf("Enter the destination node:");
        scanf("%d", &destination);

        Dijkstra(cost, n, u, destination);
        return 0;
}
```

OUTPUT 1:



Enter total number of nodes in Network : 4

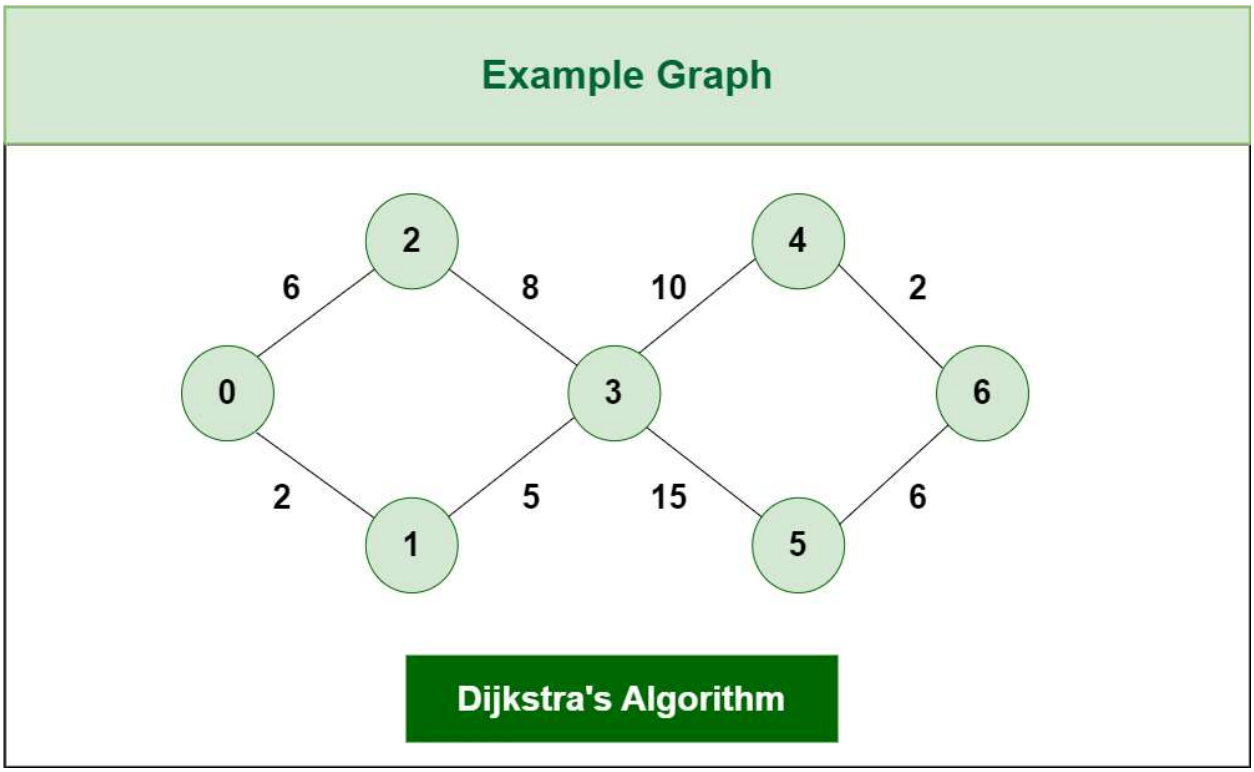
Distance between 0 and 0:0
Distance between 0 and 1:10
Distance between 0 and 2:0
Distance between 0 and 3:0
Distance between 1 and 0: 10
Distance between 1 and 1:0
Distance between 1 and 2:3
Distance between 1 and 3:20
Distance between 2 and 0: 0
Distance between 2 and 1: 3
Distance between 2 and 2:0
Distance between 2 and 3:10
Distance between 3 and 0: 0
Distance between 3 and 1: 20
Distance between 3 and 2: 10
Distance between 3 and 3:0

Enter the starting node:0
Enter the destination node:2

Distance of Node 1: 10
Route: 0 --> 1
Distance of Node 2: 13
Route: 0 --> 1 --> 2
Distance of Node 3: 23
Route: 0 --> 2 --> 1 --> 3

Shortest distance from source 0 to destination 2: 13
Shortest path: 0 --> 1 --> 2

OUTPUT 2:



So, the Shortest Distance from the Source Vertex is 19 which is optimal one

Enter total number of nodes in Network : 7

Distance between 0 and 0:0
Distance between 0 and 1:2
Distance between 0 and 2:6
Distance between 0 and 3:0
Distance between 0 and 4:0
Distance between 0 and 5:0
Distance between 0 and 6:0
Distance between 1 and 0: 2
Distance between 1 and 1:0
Distance between 1 and 2:0
Distance between 1 and 3:5
Distance between 1 and 4:0
Distance between 1 and 5:0

Distance between 1 and 6:0
Distance between 2 and 0: 6
Distance between 2 and 1: 0
Distance between 2 and 2:0
Distance between 2 and 3:8
Distance between 2 and 4:0
Distance between 2 and 5:0
Distance between 2 and 6:0
Distance between 3 and 0: 0
Distance between 3 and 1: 5
Distance between 3 and 2: 8
Distance between 3 and 3:0
Distance between 3 and 4:10
Distance between 3 and 5:15
Distance between 3 and 6:0
Distance between 4 and 0: 0
Distance between 4 and 1: 0
Distance between 4 and 2: 0
Distance between 4 and 3: 10
Distance between 4 and 4:0
Distance between 4 and 5:0
Distance between 4 and 6:2
Distance between 5 and 0: 0
Distance between 5 and 1: 0
Distance between 5 and 2: 0
Distance between 5 and 3: 15
Distance between 5 and 4: 0
Distance between 5 and 5:0
Distance between 5 and 6:6
Distance between 6 and 0: 0
Distance between 6 and 1: 0
Distance between 6 and 2: 0
Distance between 6 and 3: 0
Distance between 6 and 4: 2
Distance between 6 and 5: 6
Distance between 6 and 6:0

Enter the starting node:0
Enter the destination node:6

Distance of Node 1: 2
Route: 0 --> 1
Distance of Node 2: 6
Route: 0 --> 2
Distance of Node 3: 7
Route: 0 --> 1 --> 3
Distance of Node 4: 17
Route: 0 --> 3 --> 1 --> 4
Distance of Node 5: 22
Route: 0 --> 3 --> 1 --> 5
Distance of Node 6: 19
Route: 0 --> 4 --> 3 --> 1 --> 6

Shortest distance from source 0 to destination 6: 19
Shortest path: 0 --> 4 --> 3 --> 1 --> 6

(An off-Campus Institution of NITTE (DEEMED TO BE UNIVERSITY), MANGALORE)

DCN Ass8 – Bellman-Ford Distance Vector Routing

1. Write a program to construct distance vector table for all the router of a given network using Bellman-Ford Distance Vector Routing protocol method.

Example:

Enter the number of nodes : 4

Enter the cost matrix:

	1	2	3	4
1	0	3	5	999
2	3	0	999	1
3	5	4	0	2
4	999	1	2	0

Intial Table

Router1

Destination	Distance	Next Hop
1	0	1
2	3	2

3	5	3
4	999	-

Similarly construct for Router 2, 3 ,4.

Updated table

Router 1

Destination	Distance	Next Hop
1	0	1
2	3	2
3	5	3
4	4	2

Router 2

Destination	Distance	Next Hop
1	3	1
2	0	2
3	3	4

4	1	4

Router 3

Destination	Distance	Next Hop
1	5	1
2	3	4
3	0	3
4	2	4

Router 4

Destination	Distance	Next Hop
1	4	2
2	1	2
3	2	3
4	0	4

```
#include <stdio.h>

struct node
{       unsigned dist[20];
        unsigned from[20];
}rt[10];

int main()
{       int dmat[20][20],n, i, j, k, count = 0;
        printf("Enter the number of nodes: ");
        scanf("%d", &n);

        printf("\nEnter the cost matrix:(Assign value 999 if there is no direct connection)\n");
        for (i = 0; i < n; i++)
                for (j = 0; j < n; j++)
                {       scanf("%d", &dmat[i][j]);
                        dmat[i][i] = 0;
                        rt[i].dist[j] = dmat[i][j];
                        rt[i].from[j] = j;
                }

        printf("\nInitial Tables:\n");
        for (i = 0; i < n; i++)
        {       printf("\nRouter %d:\n", i + 1);
                printf("Destination\tDistance\tNext Hop\n");

                for (j = 0; j < n; j++)
                        if(rt[i].dist[j]==999)
                                printf("%d\t\t%d\t\t-\n", j + 1, rt[i].dist[j]);
                        else
                                printf("%d\t\t%d\t\t%d\n", j + 1, rt[i].dist[j], rt[i].from[j] + 1);
        }

        do
        {       count = 0;
                for (i = 0; i < n; i++)
                        for (j = 0; j < n; j++)
                                for (k = 0; k < n; k++)
```

```
                    if (rt[i].dist[j] > dmat[i][k] + rt[k].dist[j])
                    {       rt[i].dist[j] = rt[i].dist[k] + rt[k].dist[j];
                            rt[i].from[j] = k;
                            count++;
                    }
        }while (count != 0);

        printf("\nUpdated Tables:\n");
        for (i = 0; i < n; i++)
        {       printf("\nRouter %d:\n", i + 1);
                printf("Destination\tDistance\tNext Hop\n");
                for (j = 0; j < n; j++)
                        printf("%d\t\t%d\t\t%d\n", j + 1, rt[i].dist[j], rt[i].from[j] + 1);
        }
        printf("\n");
        return 0;
}
```

OUTPUT :

```
Enter the number of nodes: 4
Enter the cost matrix:(Assign value 999 if there is no direct connection)
0 3 5 999
3 0 999 1
5 4 0 2
999 1 2 0

Initial Tables:

Router 1:
Destination     Distance        Next Hop
1               0               1
2               3               2
3               5               3
4               999             -

Router 2:
```

Destination	Distance	Next Hop
1	3	1
2	0	2
3	999	-
4	1	4

Router 3:

Destination	Distance	Next Hop
1	5	1
2	4	2
3	0	3
4	2	4

Router 4:

Destination	Distance	Next Hop
1	999	-
2	1	2
3	2	3
4	0	4

Updated Tables:

Router 1:

Destination	Distance	Next Hop
1	0	1
2	3	2
3	5	3
4	4	2

Router 2:

Destination	Distance	Next Hop
1	3	1
2	0	2
3	3	4
4	1	4

Router 3:

Destination	Distance	Next Hop

1	**5**	**1**
2	**3**	**4**
3	**0**	**3**
4	**2**	**4**

Router 4:

Destination	**Distance**	**Next Hop**
1	**4**	**2**
2	**1**	**2**
3	**2**	**3**
4	**0**	**4**

DCN Ass9 – Socket Programming

1. Write a Program to Demonstrate Client/Server Model using Socket Programming

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

int main()
{       /* Define the IP (Internet Protocol) address and port number, that would be
        used to create a socket. Here, we are using the localhost address for both the
        server and the client.*/
        char *ip = "127.0.0.1";
        int port = 5566;

        int server_sock, client_sock;          //define the variables
        struct sockaddr_in server_addr, client_addr;
        socklen_t addr_size;
        char buffer[1024];
        int n;server_sock = socket(AF_INET, SOCK_STREAM, 0);
        if (server_sock < 0)
        {       perror("[-]Socket error");
                exit(1);
        }
        printf("[+]TCP server socket created.\n");

        /* Here, we initialize the server address by providing the required IP and port
        number. The server keeps all the address information for both the server and
        the client in the sockaddr_in struct.*/
        memset(&server_addr, '\0', sizeof(server_addr));
        server_addr.sin_family = AF_INET;
        server_addr.sin_port = port;
        server_addr.sin_addr.s_addr = inet_addr(ip);

        //Binding the socket descriptor with the server address information.
        n = bind(server_sock, (struct sockaddr*)&server_addr,sizeof(server_addr));
        if (n < 0)
        {       perror("[-]Bind error");
                exit(1);
```

```
        }
        printf("[+]Bind to the port number: %d\n", port);

        //Now, we listen for incoming connections from the clients.
        listen(server_sock, 5);
        printf("Listening...\n");

        /* the server handles only one client at a time. So, only one client would
        communicate and the rest would have to wait for the communication to get
        completed.*/
        char c[200];
        while(1)
        {       addr_size = sizeof(client_addr);
                client_sock = accept(server_sock, (struct sockaddr*)&client_addr,
                &addr_size);
                printf("[+]Client connected.\n");
                while(1)
                {       bzero(buffer, 1024);
                        recv(client_sock, buffer, sizeof(buffer), 0);
                        printf("Client: %s\n", buffer);
                        bzero(buffer, 1024);

                        printf("Enter the message : ");
                        fgets(c,200,stdin);
                        strcpy(buffer, c);
                        printf("Server: %s\n", buffer);
                        send(client_sock, buffer, strlen(buffer), 0);
                }
                close(client_sock);
                printf("[+]Client disconnected.\n\n");
        }
}



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

int main()
{       /*NOTE: Make sure that the IP address and the port number for the client
        are the same as the server*/
```

```
char *ip = "127.0.0.1", buffer[1024];
int port = 5566,sock,n;
struct sockaddr_in addr;
socklen_t addr_size;

/*We start by creating a TCP (Transmission Control Protocol) socket. The
socket would beused to connect to the server and start communication.*/
sock = socket(AF_INET, SOCK_STREAM, 0);
if (sock < 0)
{       perror("[-]Socket error");
        exit(1);
}
printf("[+]TCP server socket created.\n");

//provide the required IP address and the port number to the required data
structures.
memset(&addr, '\0', sizeof(addr));
addr.sin_family = AF_INET;
addr.sin_port = port;
addr.sin_addr.s_addr = inet_addr(ip);

//We send a connection request to the server and wait for the server to accept
the request.
connect(sock, (struct sockaddr*)&addr, sizeof(addr));
printf("Connected to the server.\n");

//We send a message to the server and wait for the reply.
bzero(buffer, 1024);
char k[200];
while(1)
{       printf("Enter the message : ");
        fgets(k,200,stdin);
        strcpy(buffer, k);
        printf("Client: %s\n", buffer);
        send(sock, buffer, strlen(buffer), 0);

        /* We receive the reply from the server and print it on the console.*/
        bzero(buffer, 1024);
        recv(sock, buffer, sizeof(buffer), 0);
        printf("Server: %s\n", buffer);
}

//At last, we close the connection from the server.
close(sock);
printf("Disconnected from the server.\n");
```

```
    return 0;
}
```

OUTPUT :

Terminal 1 :

```
adminmca@MCA-Lab1:~/Documents/Manish NNM23MC073/Ass9$ gcc server.c -o
server
adminmca@MCA-Lab1:~/Documents/Manish NNM23MC073/Ass9$ ./server
[+]TCP server socket created.
[+]Bind to the port number: 5566
Listening...
[+]Client connected.
Client: Hii MAnish here

Enter the message : Hllo Manish
Server: Hllo Manish

Client: How are you server

Enter the message : fine.. what about youuuu??
Server: fine.. what about youuuu??
```

Terminal 2 :

```
adminmca@MCA-Lab1:~/Documents/Manish NNM23MC073/Ass9$ gcc client.c -o
client
adminmca@MCA-Lab1:~/Documents/Manish NNM23MC073/Ass9$ ./client
[+]TCP server socket created.
Connected to the server.
Enter the message : Hii MAnish here
Client: Hii MAnish here

Server: Hllo Manish

Enter the message : How are you server
Client: How are you server

Server: fine.. what about youuuu??

Enter the message :
```