|  |  |  |  |
| --- | --- | --- | --- |
| **List Of Experiments** | | | |
| **Sl.No** | **Title** | **CO’s** | **Page No.** |
| **PART-A: Simulation Experiments using NCTUNS Tool** | | | |
| **1** | Implement a point to point Network with 4 nodes and duplex links between them. Analyze the network performance by setting the queue size &  Varying the bandwidth. | **1** |  |
| **2** | Implement 4 node point to point networks with links n0-n2, n1-n2 and n2-n3. Apply TCP agent between n0-n3 and UDP agent between n1-n3. Apply relevant applications over TCP & UDP  Agents changing the parameters & determine the number of packets sent by TCP/UDP | **1** |  |
| **3** | Implement Ethernet LAN using n (6-10) nodes. Compare the throughput by changing the error rate  And data rate. | **1** |  |
| **4** | Implement Ethernet LAN using n nodes and assign  Multiple traffic to the nodes and obtain congestion window for different sources/destinations. | **2** |  |
| **5** | Implement ESS with transmission nodes in wireless  LAN and obtain the performance parameters. | **2** |  |
| **PART-B:Implementthefollowingin C/C++** | | | |
| **1** | Write a program for a HLDC frame to perform the  following:1.Bit Stuffing.2.Character stuffing | **3** |  |
| **2** | Write a program for a distance vector algorithm to  Find suitable path for transmission | **3** |  |
| **3** | Implement Dijkstra’s algorithm to compute the  Shortest routing path. | **3** |  |
| **4** | For a given data, use CRC-CCITT polynomial to obtain CRC code. Verify the program for the cases.  1.WithoutErrorand 2.WithError | **4** |  |
| **5** | Implement stopandwaitProtocol | **4** |  |
| **6** | Writeaprogramfor congestioncontrolusingleaky  bucket algorithm. | **4** |  |

# PART-A

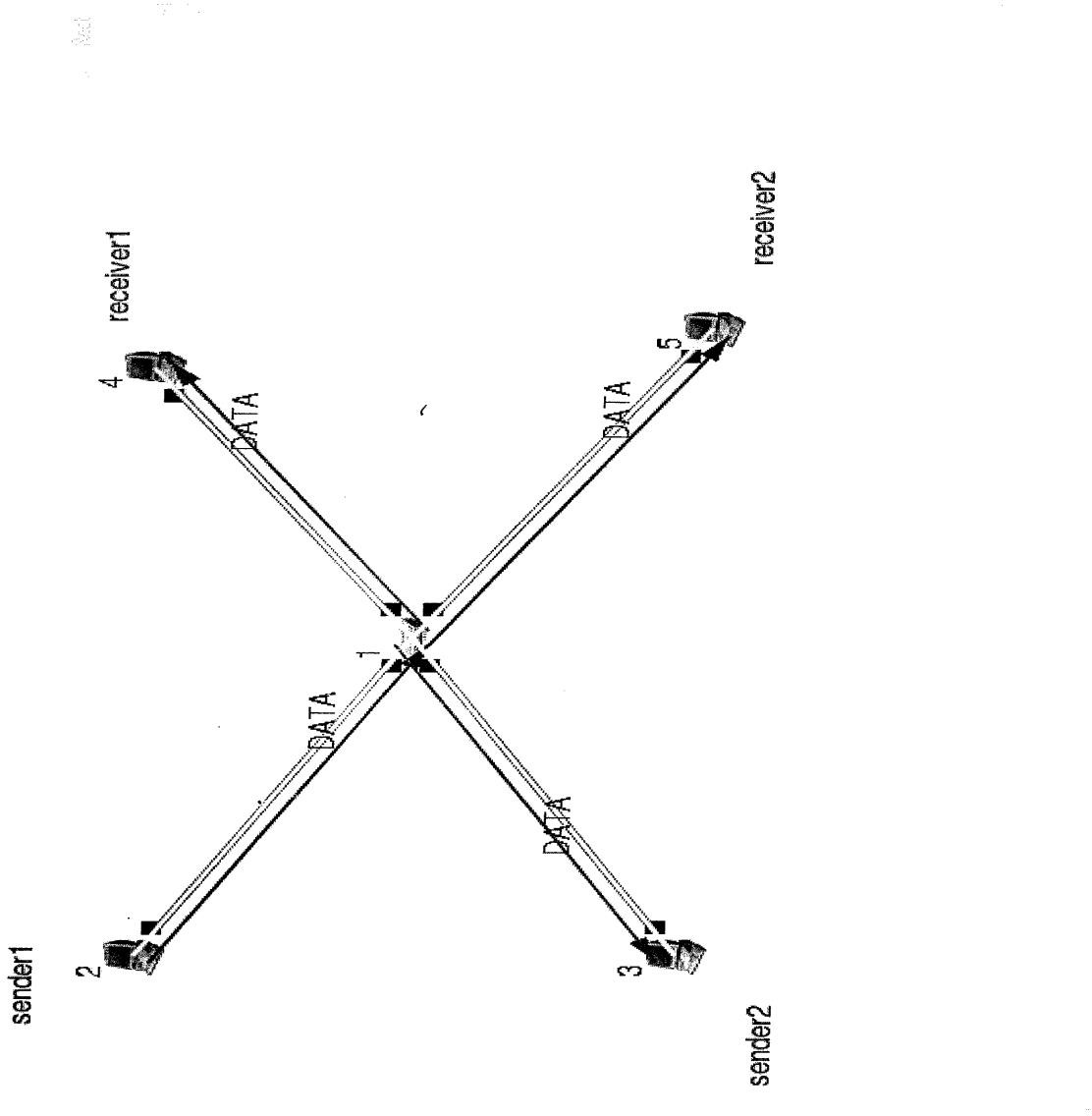
## Simulation experiments using

## NCTUNS tool

### Experiment-1

Implement a point to point Network with 4 nodes and duplex links between them. Analyze the network performance by setting the queue size & varying the bandwidth.

#### Topology:



UDP Command for Sender:

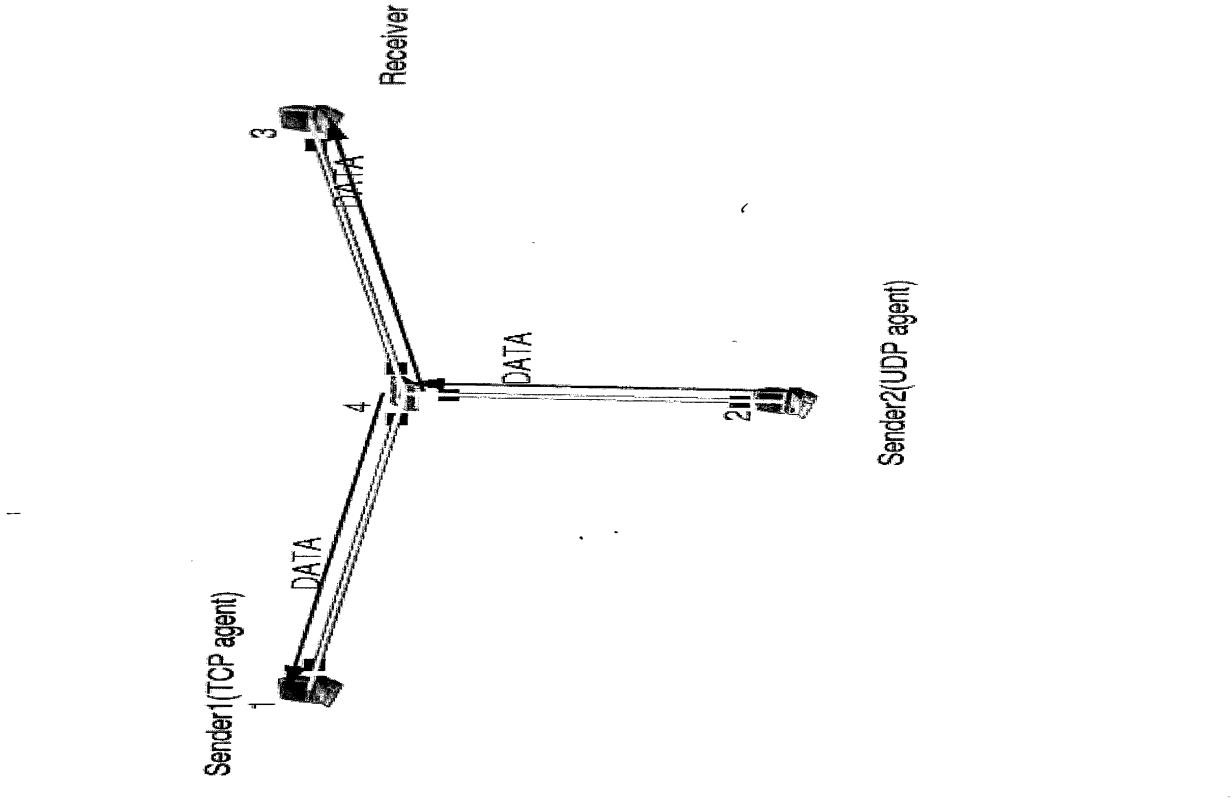
stg-u10241001.0.1.3 (receiver IP address)

TCP Command for Receiver: rtg–u– wlog1

Parameters: Throughput, Drop and Collision of Outgoing and Incoming Packets

### Experiment-2

Implement4 node point to point networks with links n0-n2, n1-n2 and n2-n3. Apply TCP agent between n0-n3 and UDP agent between n1-n3.Apply relevant applications over TCP&UDP



agents changing the parameters & determine the number of packets sent by TCP/UDP

UDP and TCP Command for Sender:

stg - u 1024 100 1.0.1.3 (receiver IP address) stcp-p2000-l10241.0.1.3(receiver address)

UDP and TCP Command for Receiver: rtg – u– w log1

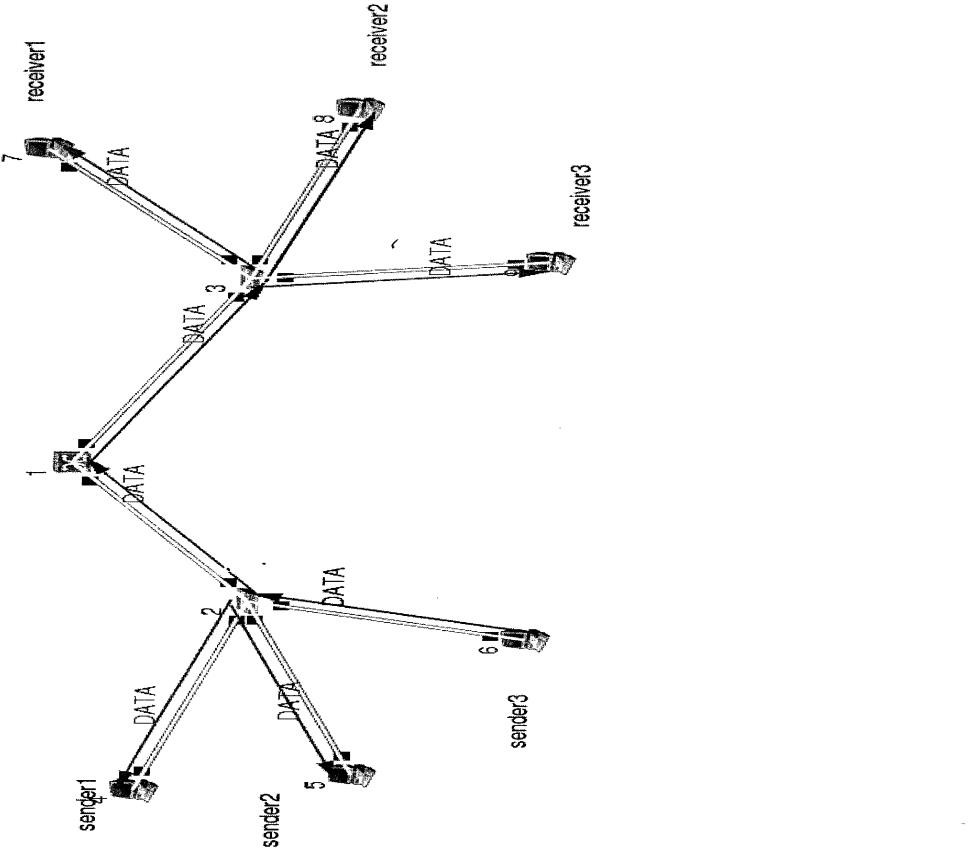
rtcp- p2000- l1024

Parameters: Throughput of Outgoing and Incoming Packets

### Experiment-3

Implement Ethernet LAN using n(6-10) nodes. Compare the throughput by changing the error rate and data rate.

#### Topology:



TCP Command for Receiver:

stcp-p2000-l10241.0.1.3(receiver address) TCP Command for Sender:

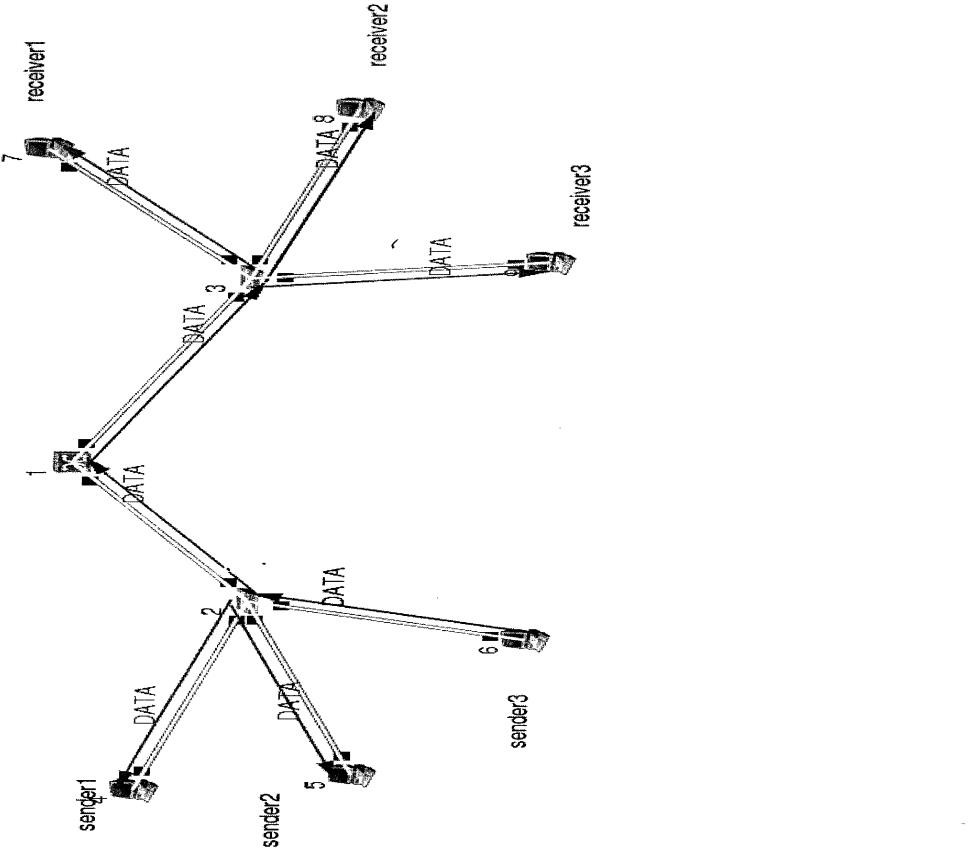
rtcp- p2000- l1024

Double click on receiver link and change BER to 0.00001,run again Parameters: Throughput of Outgoing Packets

### Experiment-4

Implement Ethernet LAN using n nodes and assign multiple traffic to the nodes and obtain congestion window for different sources/destinations.

#### Topology:



TCP Command for Receiver:

stcp-p2000-l10241.0.1.3 (receiver address) TCP Command for Sender:

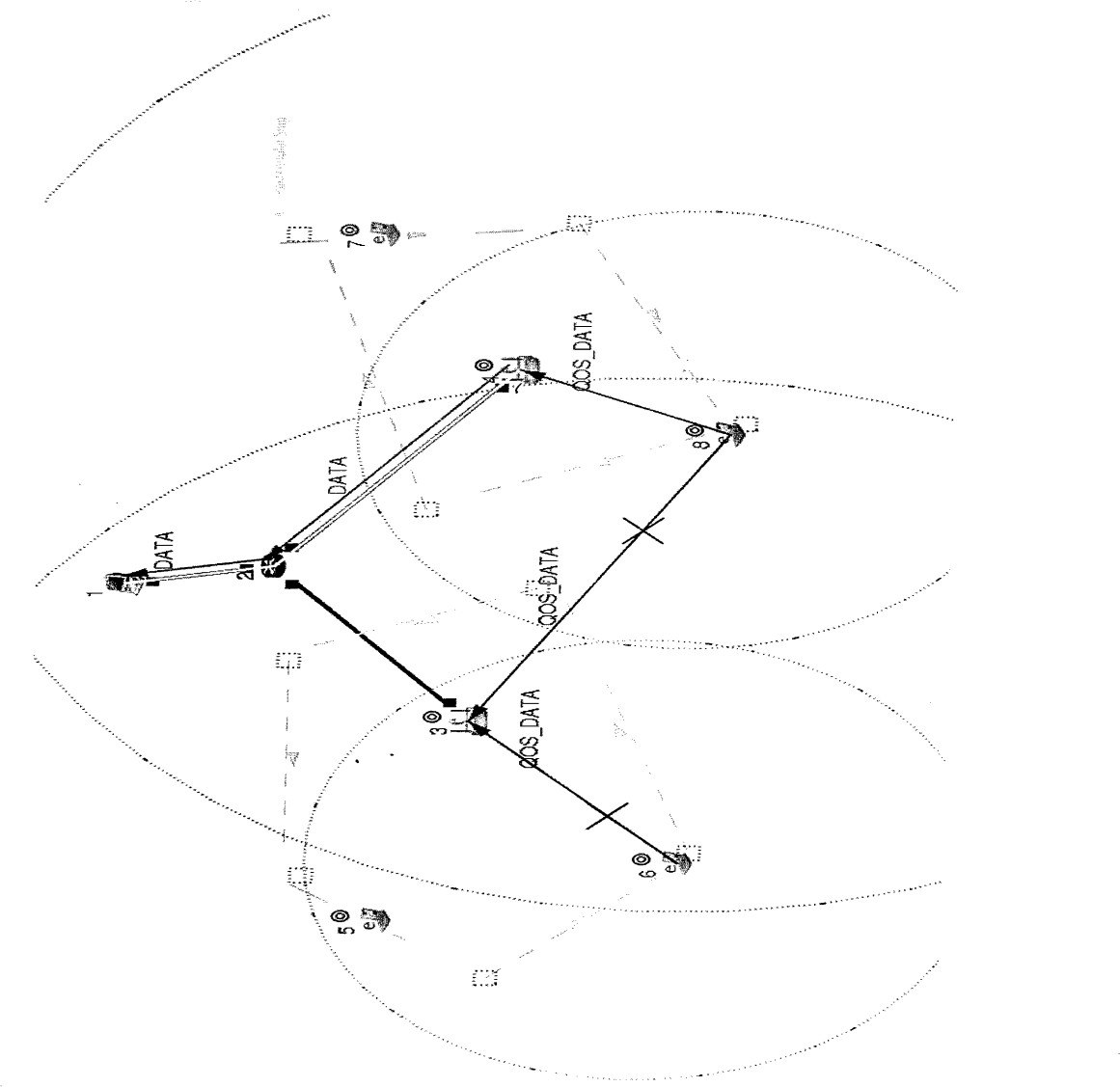
rtcp- p2000- l1024

Double click on receiver link and change BER to 0.00001,run again Parameters: Receiver side Collision Packets and Drop Packets

### Experiment-5

Implement ESS with transmission nodes in wireless LAN and obtain the performance parameters.

#### Topology:



MobileHost1

ttcp–t–u–s–p30001.0.1.1 Mobile Host 2

ttcp–t–u–r–p3000

# PART-B

## Implement the following in C/C++

### Experimentno.:-1.(a)

**Program for Bit-stuffing :**

#include<conio.h> #include<stdio.h> #include<string.h>

void main()

{

char s[100], d[100]="";

int i, j, count; clrscr();

printf(" \nEnter the string 0's&1's:"); gets(s);

strcat(d,"01111110");

for(i=0,j=8; i<strlen(s); i++,j++)

{

if(s[i]=='0')

{

d[j]=s[i]; count=0;

}

else

{

count++; d[j]=s[i];

}

if(count==5)

{ d[++j]='0';

}

}

printf("\n\nThe Stuffed String is:");

puts (strcat ( d, "01111110" ) ; getch();

}

**Result:-**

Enter the string 0’s &1’s:-

0111111111110

The Stuffed string is:-01111110011111011111011001111110

### Experiment number.:-1.(b)

**Aim:Write a program for a HLDC frame to perform the following Character stuffing.**

#include<stdio.h>

#include<conio.h>

#include<string.h>

void main( )

{

char s[100], d[100]="";

int i, j;

clrscr( );

printf("\n Enter the character:");

gets(s);

strcat(d,"ECEDEL"); for(i=0,j=6;i<strlen(s);i++,j++)

{

if(s[i]=='e'&&s[i+1]=='c'&&s[i+2]=='e')

{

strcat(d,"eceECE"); i=i+3;

j=j+6; d[j]=s[i];

}

else

{

d[j]=s[i];

//i++;

//j++;

}

}

printf("\n\nTheStuffedStringis:");

puts(strcat(d,"ECEDEL "));

getch( );

}

Result:

Enter the character: welcomeece

The stuffed string is:- ECEDELwelcomeeceECEECEDEL

Enter the character: welcomeeae

The stuffed string is:- ECEDELwelcomeeaeECEDEL

### Experiment no.:-2

**Aim:write a program for distance vector algorithm to find suitable path for transmission Program:**

#include<stdio.h> struct node

{

unsigned dist[20]; unsignedfrom[20];

}rt[10];

int main()

{

int costmat [20][20], source, desti;

int nodes, i, j, k, count=0;

printf("\n Enter the number of nodes:"); scanf("%d",&nodes);

printf("\n Enter the cost matrix :\n"); for(i=0; i<nodes; i++)

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

{

scanf("%d", &costmat[i][j]); costmat[i][i]=0; rt[i].dist[j]=costmat[i][j]; rt[i].from[j]=j;

}

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

{

printf("\n\n For router%d\n",i); for(j=0;j<nodes;j++)

printf("\t \n node %d via %d Distance %d", j, rt[i].from[j], rt[i].dist[j]);

}

do

{

count=0; for(i=0;i<nodes;i++) for(j=0;j<nodes;j++)

// if(i!=j) for(k=0;k<nodes;k++)

if(rt[i].dist[j]>rt[i].dist[k]+rt[k].dist[j])

{

rt[i].dist[j]=rt[i].dist[k]+rt[k].dist[j]; rt[i].from[j]=rt[i].from[k];

count++;

}

}while(count!=0); for(i=0;i<nodes;i++)

{

printf("\n\nFor router %d \n", i+1); for(j=0;j<nodes;j++)

printf("\t\nnode%dvia%dDistance%d",j+1,rt[i].from[j]+1,rt[i].dist[j]);

}

printf("\n\n");

}

**Result:**

Enterthenumberofnodes:3Enterth e cost matrix:

|  |  |  |
| --- | --- | --- |
| 0 | 5 | 3 |
| 5 | 0 | 1 |
| 3 | 1 | 0 |

Node0via0distance3 Node1via1distance1 Node2via2 distance0

### Forrouter1

Node1via1distance0 Node2via3distance4 Node3via3 distance3

### Forrouter2

Node1via3distance4 Node2via2distance0 Node3via3 distance1

### Forrouter3

Node1via1distance3 Node2via2distance1 Node3via3 distance0

### Experiment no.:-3

**Aim: Implement Dijkstra’s algorithm to compute the shortest routing path.**

**Program:**

#include<stdio.h> #include<conio.h>

#define INFINITY 99

#define startnode

void dijkstra(int cost[10][10], int n);

int main()

{ int cost[10][10], i, j, n, u; printf("Enterno.of Nodes:"); scanf("%d",&n);

printf("\nEnter the cost matrix:\n"); for(i=0;i<n;i++)

for(j=0;j<n;j++) scanf("%d",&cost[i][j]);

dijkstra(cost, n); return 0;

}

void dijkstra(int cost[10][10], int n)

{ int distance[10], pred[10];

int visited[10], count, mindistance, nextnode, i, j;

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;

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];

printf("%d %d \n",i,distance[i]);

pred[i]= nextnode;

}

count++;

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

if(i!=startnode)

{

printf("\n Distance to node %d = %d", i, distance[i]); printf("\n Through the Path=%d", i);

j=i; do

{

j=pred[j];

printf("<-%d ",j);

}while(j!=startnode);

}

}

**Result:-**

Enter no of nodes :4

Enter the cost matrix:

|  |  |  |  |
| --- | --- | --- | --- |
| 0 | 2 | 5 | 7 |
| 2 | 0 | 5 | 99 |
| 5 | 5 | 0 | 2 |
| 7 | 99 | 2 | 0 |

Distance to model 1=2 Through the path=1<-0

Distance to model 2=5 Through the path=2<-0

Distance to model3=7 Through the path=3<-0

### Experimentno.:-4

**Aim:Forthegivendata, useCRC-CCITTpolynomialtoobtainCRCcode.Verifythe program for the cases: a) Without error and b) With error**

### Program:

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

char t[100], cs[100], g[50];

int a, i, j, N;

clrscr();

void xor1( )

{

for(j=1; j<N; j++)

cs[j] = ((cs[j]= =g[j])? '0' : '1');

}

void crc( )

{

for(i=0; i<N; i++) cs[i]=t[i];

do

{ if(cs[0]=='1')

xor1();

for(j=0; j<N-1; j++)

cs[j]=cs[j+1];

cs[j]=t[i++];

}

while(i<=a+N-1);

}

int main()

{

printf("\n Enter data: "); scanf("%s", t);

printf("Enter the generating polynomial data:"); scanf("%s",g);

N=strlen(g); a=strlen(t);

if( ((N-1)<a) && (g[0]=='1') && (g[N-1]=='1') )

{

for(i=a; i<a+N-1; i++) t[i]='0';

printf("\nModified data is:%s ",t); crc( );

printf("\nChecksum is: %s ",cs);

for(i=a; i<a+N-1; i++)

t[i]=cs[i-a];

printf("\nTransmitting code word is: %s", t); printf("\n Enter received message "); scanf("%s", t);

crc( );

for(i=0; (i<N-1) && (cs[i]!='1'); i++);

if(i<N-1)

printf("\n With Error\n"); else

printf("\n Without error\n");

}

else

printf("wrong generating polynomial\n"); return 0;

}

**Result:-**

Example1:

Example2:

Enter data: - 11110010 Generating polynomial:-1011 Modified data is:-11110010000

Check Sum:- 010

Transmission codeword:-11110010010 Enter Received Message: 11110010010 Without Error

Enter data: - 11110010 Generating polynomial:-1011 Modified data is:-11110010000

Check Sum:- 010

Transmission codeword:-11110010010 Enter Received Message: 11100010010 With Error

### Experiment no.:-5 Aim: Implementation of Stop and Wait Protocol.

**Program:**

#include<stdio.h> #include<stdlib.h> #include<conio.h> void main()

{

int i, n, r, a;

clrscr();

printf("\n\n enter the number of packets:"); scanf("%d", &n);

for(i=1; i<=n; i++)

{

printf("\n\n the packet sent is:%d", i); r=(rand()+1)%2;

if(r==1)

{

a=rand()%2; if(a==1)

printf("\n\n acknowlegement number:%d",i+1); else

{

printf("\n\n no acknowlegement number:%d",i+1);

i--;

}

}

else

{

printf("\n\n timeout, resend"); i--;

}

}

getch ();

}

**Result:-**

Enter the number of packets:2

The packet sent is:1

No acknowledgement number:2 The packet sent is: 1 Acknowledgement number: 2 The packet sent is: 2

Time out, resend Thepacketsentis: 2

No acknowledgement number: 3

The packet sent is:2

Time out, resend The packet sent is:2

No acknowledgement number: 3 The packet sent is: 2 Acknowledgement number: 3

### Experiment no.:- 6

**Aim: Write a program for congestion control using leaky bucket algorithm.** **Program:**

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

#define MIN(x,y)(x>y)?y:x

int main()

{

int orate,drop=0,cap,x,count=0, inp[10]={0}, i=0, nsec,ch;

printf("\n Enter bucket size : ");

scanf("%d", &cap);

printf("\n enter output rate:");

scanf("%d", &orate);

do

{

printf("\n enter number of packets coming at second %d:", i+1); scanf("%d", &inp[i]);

i++;

printf("\nenter1 to contiue or 0 to quit ");

scanf("%d",&ch);

}

while(ch); nsec=i;

printf("\nsecond\trecieved\tsent\tdropped\tremained\n");

for(i=0;count || i<nsec;i++)

{

printf("%d",i+1);

printf("\t%d\t ",inp[i]);

printf("\t%d\t",MIN((inp[i]+count),orate)); if((x=inp[i]+count-orate)>0)

{

if(x>cap)

{

count=cap; drop=x-cap;

}

else

{

count=x; drop=0;

}

}

else

{

drop=0; count=0;

}

printf("\t%d\t %d\n",drop,count);

}

return0;

}

### Result:-

enter the size of bucket:5

enter the output rate:2

enter number of packets coming at second1:10

enter1 to continue or 0 to quit 1

entre number of packets coming at second 2: 2

enter1to continueor0 to quit.........................................0

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Time(sec)** | **Receive** | **Sent** | **Dropped** | **Remained** |
| 1 | 10 | 2 | 3 | 5 |
| 2 | 2 | 2 | 0 | 5 |
| 3 | 0 | 2 | 0 | 3 |
| 4 | 0 | 2 | 0 | 1 |
| 5 | 0 | 1 | 0 | 0 |