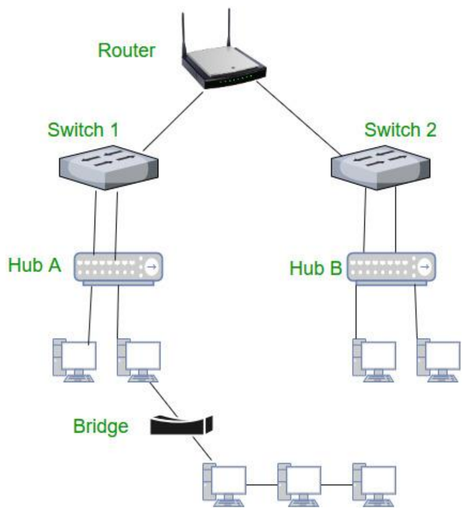
COMPUTER NETWORKS LAB

Experiment : 1

Aim: Algorithm to implementstudy of Network devices in detail and connect the computers in Local Area Network.

Description:
Network devices, also known as networking hardware, are physical devices that allow hardware on a computer network to communicate and interact with one another. For example Repeater, Hub, Bridge, Switch, Routers, Gateway, Brouter, and NIC, etc.



Hub – A hub is a basically multi-port repeater. A hub connects multiple wires coming from different branches, for example, the connector in star topology which connects different stations. Hubs cannot filter data, so data packets are sent to all connected devices.
Bridge – A bridge operates at the data link layer. A bridge is a repeater, with add on the functionality of filtering content by reading the MAC addresses of the source and destination. It is also used for interconnecting two LANs working on the same protocol.
Switch – A switch is a multiport bridge with a buffer and a design that can boost its efficiency(a large number of ports imply less traffic) and performance. A switch is a data link layer device.

Experiment : 2

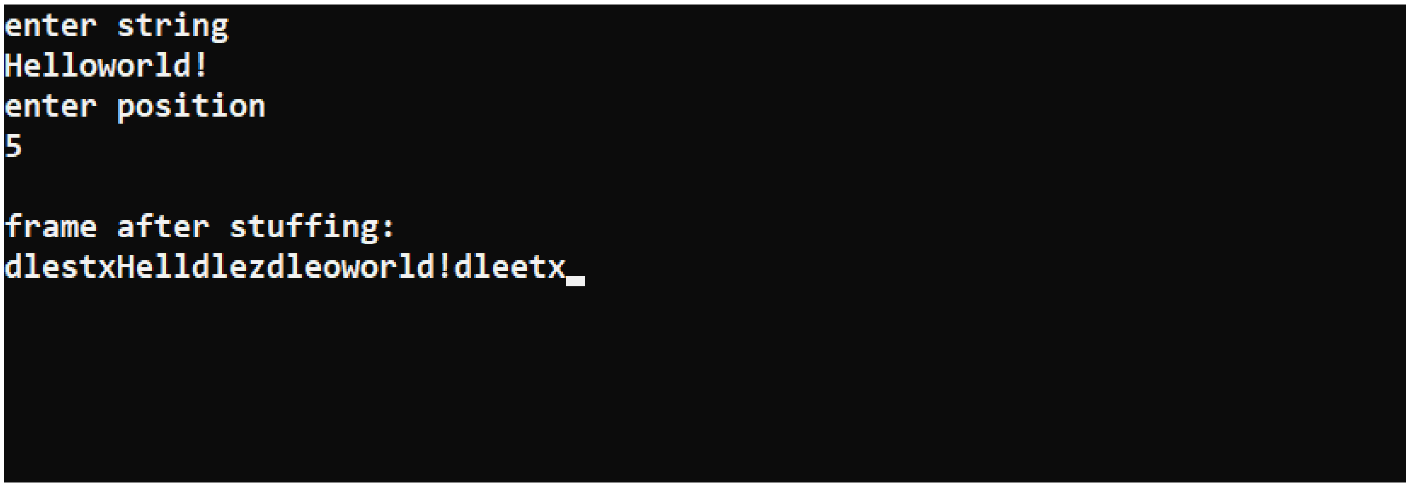
Aim: Algorithm to implement the data link layer framing methods such as i) Character stuffing ii) bit stuffing

(i). Character stuffing

Program:

```
#include<stdio.h>
#include<conio.h>
#include<string.h>
void main()
        {
        int i=0,j=0,n,pos;
        char a[20],b[50],ch=’z’;
        clrscr();
        printf("enter string\n");
        scanf("%s",&a);
        n=strlen(a);
        printf("enter position\n");
        scanf("%d",&pos);
        if(pos>n)
                {
                printf("invalid position, Enter again less than %d:",n);
                scanf("%d",&pos);
                }
        b[0]='d';
        b[1]='l';
        b[2]='e';
        b[3]='s';
        b[4]='t';
        b[5]='x';
        j=6;
        while(i<n)
                {
                if(i==pos-1)
                        {
                        b[j]='d';
                        b[j+1]='l';
                        b[j+2]='e';
                        b[j+3]=ch;
                        b[j+4]='d';
                        b[j+5]='l';
                        b[j+6]='e';
                        j=j+7;
                        }
                if(a[i]=='d' && a[i+1]=='l' && a[i+2]=='e')
                        {
                        b[j]='d';
                        b[j+1]='l';
                        b[j+2]='e';
                        j=j+3;
                        }
                b[j]=a[i];
                i++;
                j++;
                }
        b[j]='d';
        b[j+1]='l';
        b[j+2]='e';
        b[j+3]='e';
        b[j+4]='t';
        b[j+5]='x';
        b[j+6]='\0';
        printf("\nframe after stuffing:\n");
        printf("%s",b);
        getch();
        }
```

Output:



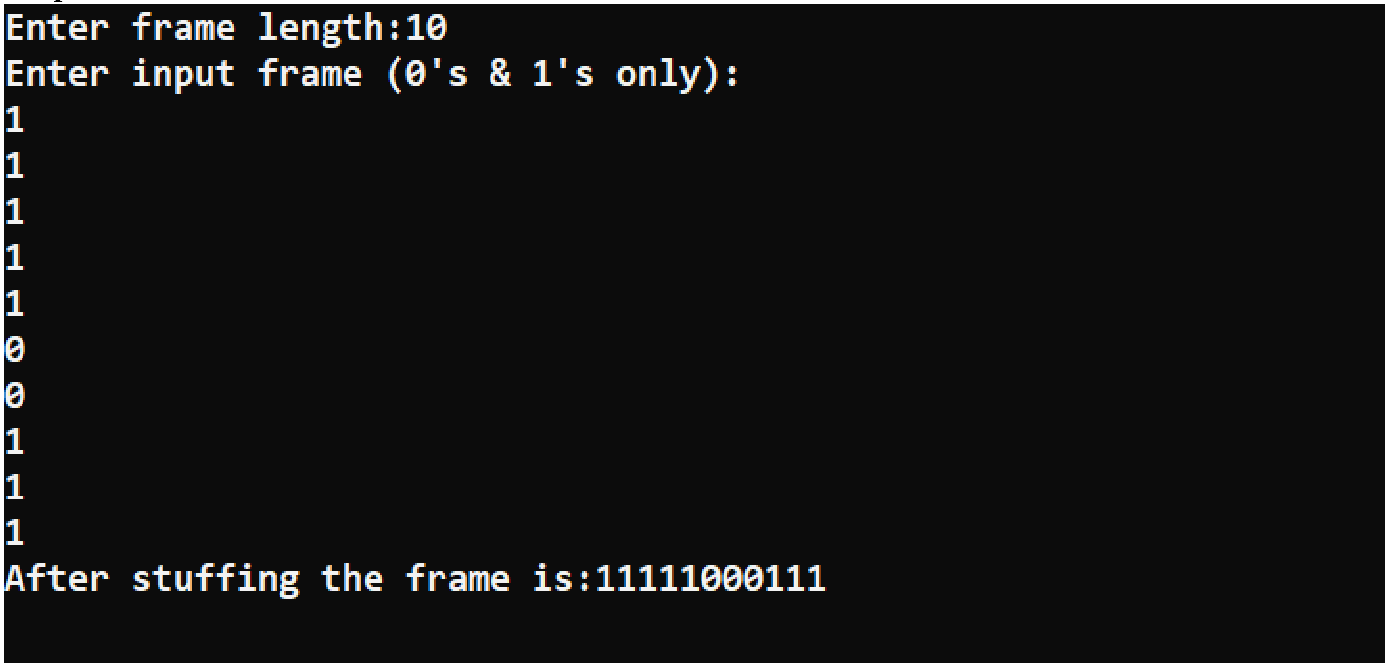
```
enter string
Helloworld!
enter position
5

frame after stuffing:
dlestxHelldlezdleoworld!dleetx
```

(ii). Bit Stuffing

Program:

```
#include<stdio.h>
#include<conio.h>
#include<string.h>
void main()
        {
        int a[20],b[30],i,j,k,count,n;
        clrscr();
        printf("Enter frame length:");
        scanf("%d",&n);
        printf("Enter input frame (0's & 1's only):");
        for(i=0;i<n;i++)
                scanf("%d",&a[i]);
        i=0;
        count=1;
         j=0;
        while(i<n)
                {
                if(a[i]==1)
                        {
                        b[j]=a[i];
                        for(k=i+1;a[k]==1 && k<n &&count<5;k++)
                                { j++;b[j]=a[k];
                                count++;
                                if(count==5)
                                        {
                                        j++;
                                        b[j]=0;
                                        }
                                i=k;
                                }
                        }
                else
                        {
                        b[j]=a[i];
                        }
                i++;
                j++;
                }
    printf("After stuffing the frame is:");
for(i=0;i<j;i++)
      printf("%d",b[i]);
getch();
}
```

Output:



```
Enter frame length:10
Enter input frame (0's & 1's only):
1
1
1
1
1
0
0
1
1
1
After stuffing the frame is:11111000111
```

Experiment : 3

Aim: Algorithm to implementdata link layer farming method checksum

Program:

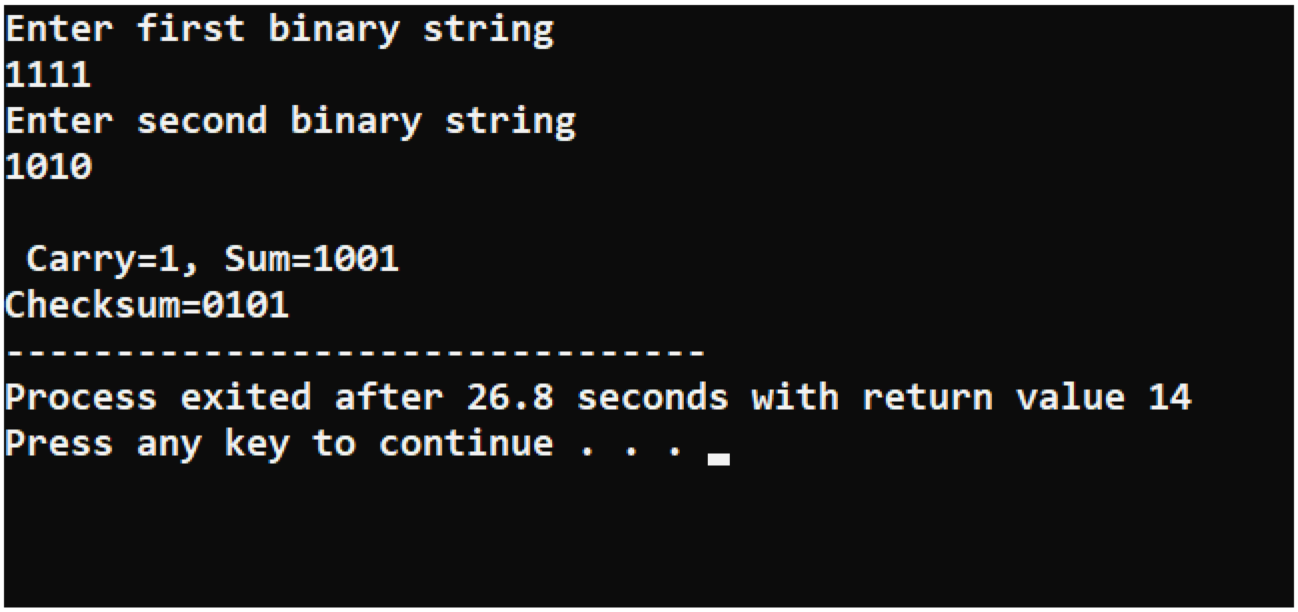
```
#include<stdio.h>
#include<string.h>
int main()
{
        char a[20],b[20];
        char sum[20],complement[20];
        int i,length;
        printf("Enter first binary string\n");
        scanf("%s",a);
        printf("Enter second binary string\n");
        scanf("%s",b);
        if(strlen(a)==strlen(b))
        {
                length=strlen(a);
                char carry='0';
                for(i=length-1;i>=0;i--)
                {
                        if(a[i]=='0'&&b[i]=='0'&&carry=='0')
                        {
                                sum[i]='0';
                                carry='0';
                        }
                        else if(a[i]=='0'&&b[i]=='0'&&carry=='1')
                        {
                                sum[i]='1';
                                carry='0';
```

```
}
else if(a[i]=='0'&&b[i]=='1'&&carry=='0')
{
        sum[i]='1';
        carry='0';
}
else if(a[i]=='0'&&b[i]=='1'&&carry=='1')
{
        sum[i]='0';
        carry='1';
}
else if(a[i]=='1'&&b[i]=='0'&&carry=='0')
{
        sum[i]='1';
        carry='0';
}
else if(a[i]=='1'&&b[i]=='0'&&carry=='1')
{
        sum[i]='0';
        carry='1';
}
else if(a[i]=='1'&&b[i]=='1'&&carry=='0')
{
        sum[i]='0';
        carry='1';
}
else if(a[i]=='1'&&b[i]=='1'&&carry=='1')
{
        sum[i]='1';
        carry='1';
```

```
                }
                else
                        break;
        }
        printf("\n Carry=%c, Sum=%s",carry,sum);
        for(i=length-1;i>0;i--)
        {
                if(sum[i]=='1'&& carry=='1')
                {
                        sum[i]='0';
                        carry='1';
                }
                else if(sum[i]=='0'&& carry=='1')
                {
                        sum[i]='1';
                        break;
                }
                else
                        break;
        }
        for(i=0;i<length;i++)
        {
                if(sum[i]=='0')
                        complement[i]='1';
                else
                        complement[i]='0';
        }
        printf("\nChecksum=%s",complement);
}
else
```

```
        {
                printf("\n Wrong input Strings");
        }
}
```

Output:



```
Enter first binary string
1111
Enter second binary string
1010

 Carry=1, Sum=1001
Checksum=0101
--------------------------------
Process exited after 26.8 seconds with return value 14
Press any key to continue . . .
```

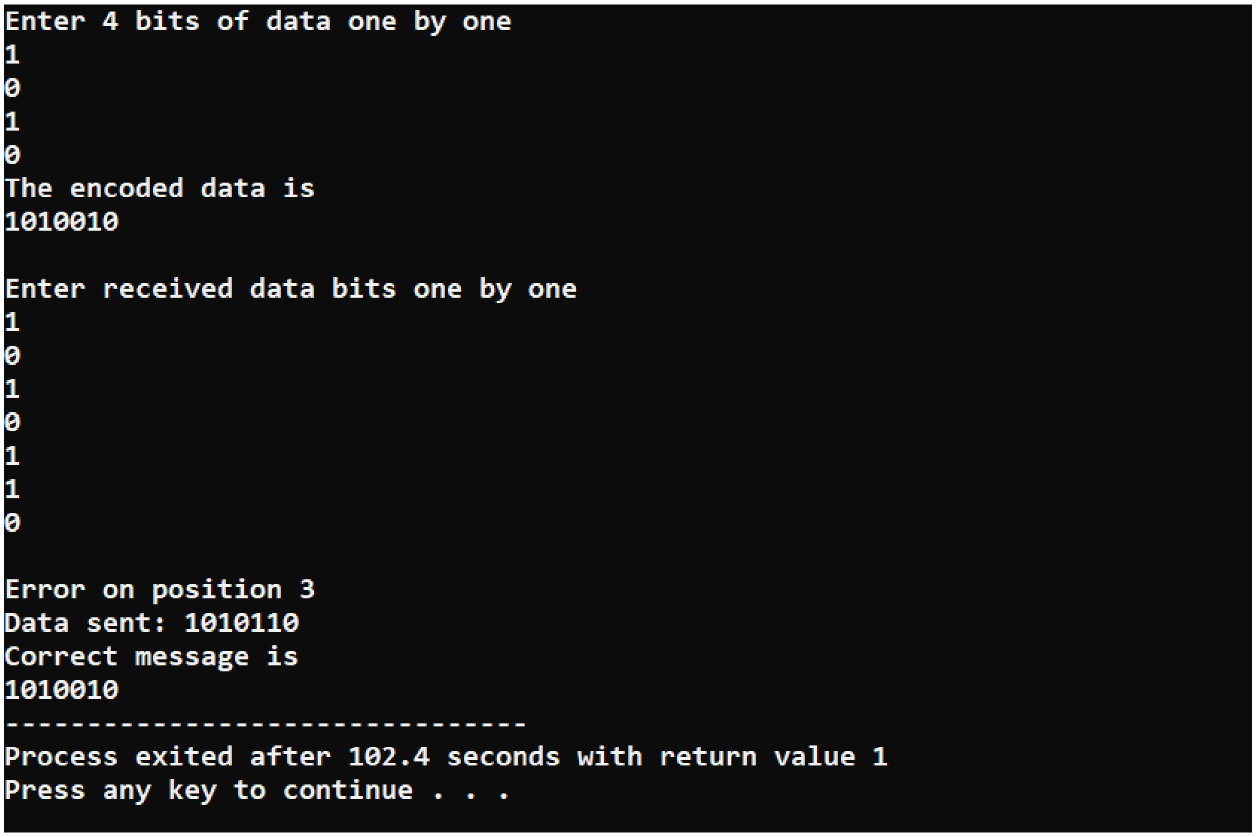
Experiment : 4

Aim: Algorithm to implement Hamming Code generation for error detection and correction

Program:

```
#include<stdio.h>
void main()
{
int data[10];
int dataatrec[10],c,c1,c2,c3,i;
printf("Enter 4 bits of data one by one\n");
scanf("%d",&data[0]);
scanf("%d",&data[1]);
scanf("%d",&data[2]);
scanf("%d",&data[4]);
//calculation of even parity
data[6]=data[0]^data[2]^data[4];
data[5]=data[0]^data[1]^data[4];
data[3]=data[0]^data[1]^data[2];
printf("The encoded data is \n");
for(i=0;i<7;i++)

printf("%d",data[i]);


printf("\n\nEnter received data bits one by one\n");
for(i=0;i<7;i++)
scanf("%d",&dataatrec[i]);
c1=dataatrec[6]^dataatrec[4]^dataatrec[2]^dataatrec[0];
c2=dataatrec[5]^dataatrec[4]^dataatrec[1]^dataatrec[0];
c3=dataatrec[3]^dataatrec[2]^dataatrec[1]^dataatrec[0];
c=c3*4+c2*2+c1;
if(c==0)
{
printf("\nNo error while transmission of data\n");
}
else
{
printf("\nError on position %d",c);
printf("\nData sent: ");
for(i=0;i<7;i++)

printf("%d",dataatrec[i]);
printf("\nCorrect message is \n");
//if erroneous bit is 0 we complement it else vice versa
if(dataatrec[7-c]==0)
dataatrec[7-c]=1;
else
dataatrec[7-c]=0;
for(i=0;i<7;i++)
{
printf("%d",dataatrec[i]);
}
}
}
```

Output:



```
Enter 4 bits of data one by one
1
0
1
0
The encoded data is
1010010

Enter received data bits one by one
1
0
1
0
1
1
0

Error on position 3
Data sent: 1010110
Correct message is
1010010
--------------------------------
Process exited after 102.4 seconds with return value 1
Press any key to continue . . .
```

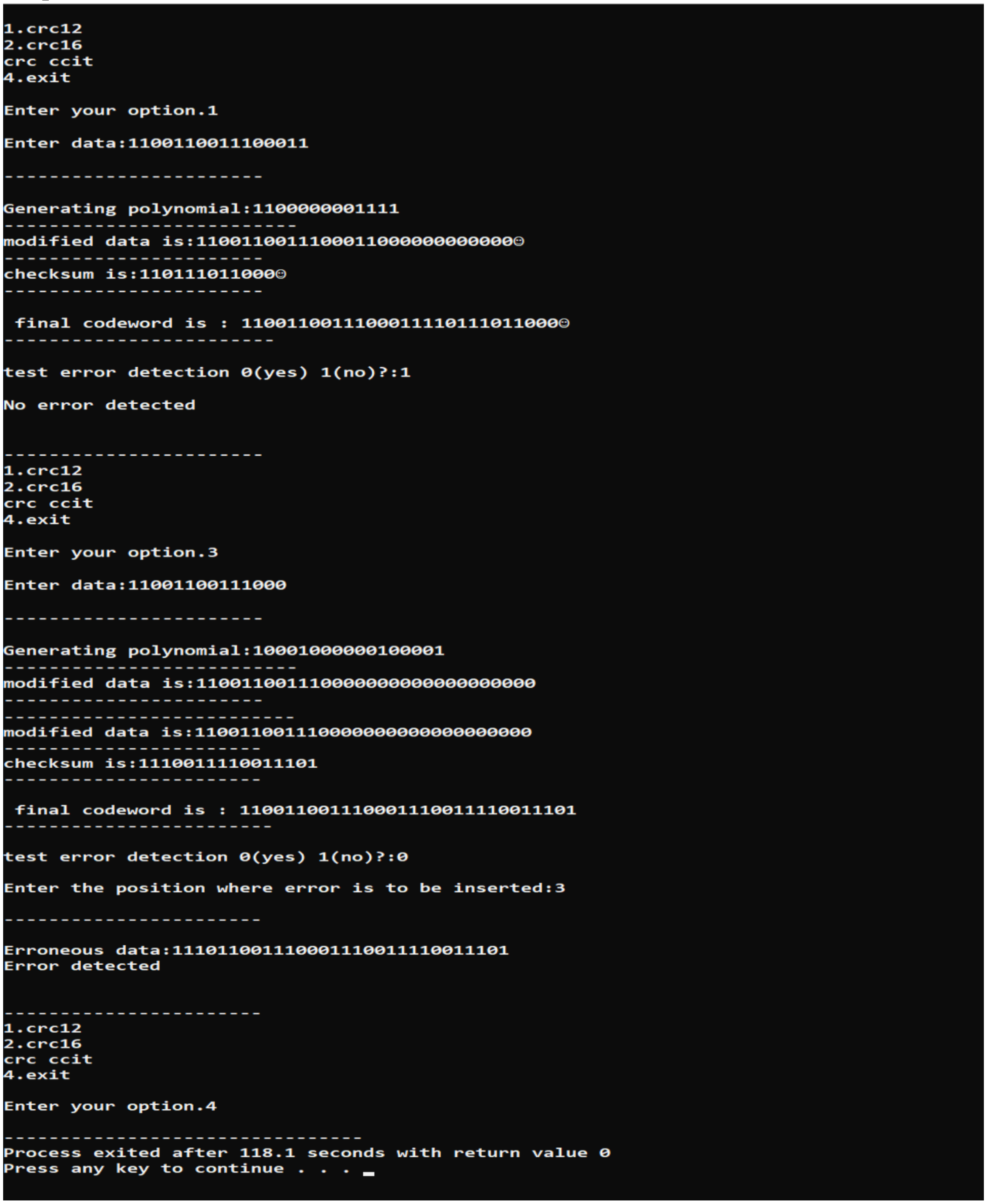
Experiment : 5

Aim: Algorithm to implement data set of characters the three CRC polynomials – CRC 12, CRC 16 and CRC CCIP.

Program:

```
#include <stdio.h>
#include <string.h>
#define N strlen(g)
char t[28],cs[28],g[28];
int a,e,c,b;
void xor()
{
        for(c=1;c<N;c++)
                cs[c]=((cs[c]==g[c])?'0':'1');
}
void crc()
{
        for(e=0;e<N;e++)
                cs[e]=t[e];
        do
        {
                if(cs[0]=='1')
                        xor();
                for(c=0;c<N-1;c++)
                        cs[c]=cs[c+1];
                cs[c]=t[e++];
        }while(e<=a+N-1);
}

int main()
{
  int flag=0;
  do
  {
        printf("\n1.crc12\n2.crc16\ncrc ccit\n4.exit\n\nEnter your option.");
        scanf("%d",&b);
        switch(b)
        {
                case 1:strcpy(g,"1100000001111");
                break;
                case 2:strcpy(g,"11000000000000101");
                break;
                case 3:strcpy(g,"10001000000100001");
                break;
                case 4:return 0;
        }
    printf("\nEnter data:");
        scanf("%s",t);
        //printf("\nN value is %d",N);
        printf("\n----------------------\n");
        printf("\nGenerating polynomial:%s",g);
        a=strlen(t);
        for(e=a;e<a+N-1;e++)
                t[e]='0';
        printf("\n-------------------------\n");

printf("modified data is:%s",t);
        printf("\n----------------------\n");
        crc();
        printf("checksum is:%s",cs);
        for(e=a;e<a+N-1;e++)
                t[e]=cs[e-a];
        printf("\n----------------------\n");
        printf("\n final codeword is : %s",t);
        printf("\n-----------------------\n");
        printf("\ntest error detection 0(yes) 1(no)?:");
        scanf("%d",&e);
        if(e==0)
        {
                do
                {
                        printf("\nEnter the position where error is to be inserted:");
                        scanf("%d",&e);
                }while(e==0||e>a+N-1);
                t[e-1]=(t[e-1]=='0')?'1':'0';
                printf("\n----------------------\n");
                printf("\nErroneous data:%s\n",t);
        }
        crc();
        for(e=0;(e<N-1)&&(cs[e]!='1');e++);
                if(e<N-1)
                        printf("Error detected\n\n");
                else
                        printf("\nNo error detected \n\n");
        printf("\n----------------------");

  }while(flag!=1);
}
```

Output:



```
1.crc12
2.crc16
crc ccit
4.exit

Enter your option.1

Enter data:1100110011100011

-----------------------

Generating polynomial:1100000001111
--------------------------
modified data is:110011001110001100000000000☺
-----------------------
checksum is:110111011000☺
-----------------------

 final codeword is : 1100110011100011110111011000☺
------------------------

test error detection 0(yes) 1(no)?:1

No error detected


-----------------------
1.crc12
2.crc16
crc ccit
4.exit

Enter your option.3

Enter data:1100110011100O

-----------------------

Generating polynomial:10001000000100001
--------------------------
modified data is:11001100111000000000000000000
-----------------------
--------------------------
modified data is:11001100111000000000000000000
-----------------------
checksum is:1110011110011101
-----------------------

 final codeword is : 11001100111000111001111 0011101
------------------------

test error detection 0(yes) 1(no)?:0

Enter the position where error is to be inserted:3

-----------------------

Erroneous data:11101100111000111001111 0011101
Error detected


-----------------------
1.crc12
2.crc16
crc ccit
4.exit

Enter your option.4

--------------------------------
Process exited after 118.1 seconds with return value 0
Press any key to continue . . . _
```

Experiment : 6

Aim: Algorithmto implement Sliding window protocol for Goback N

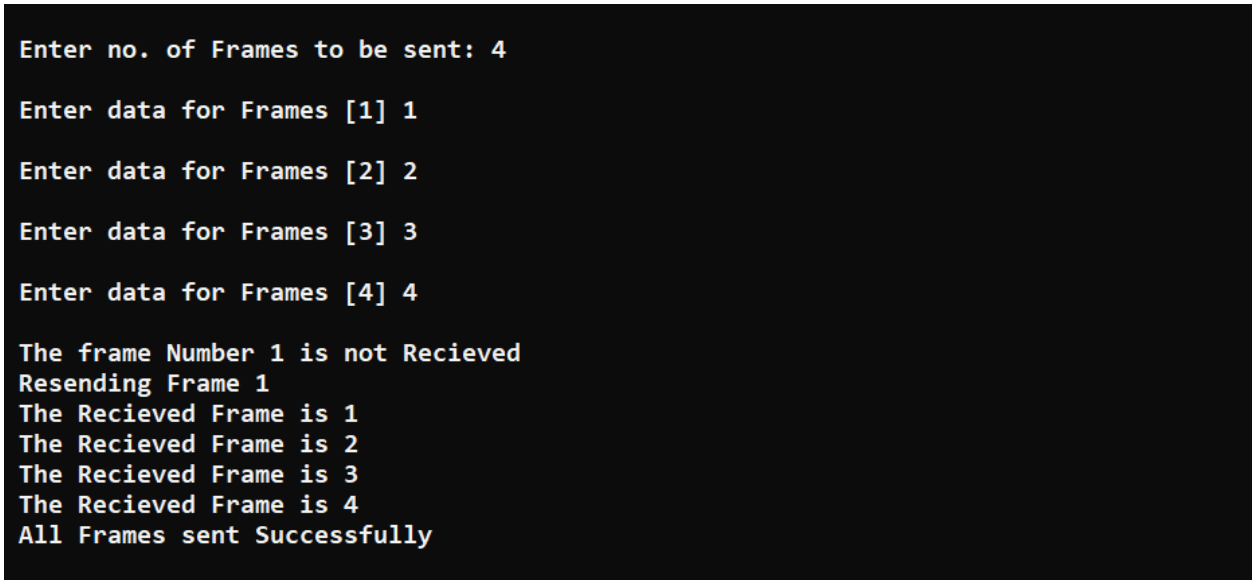
Program:

```
#include <stdio.h>
#include<stdlib.h>
#include<math.h>
int n,r;
struct frame
{
   char ack;
   int data;
}frm[10];

int sender(void);
void recvack(void);
void resend_gb(void);

int main()
{
   int c;
   sender();
   recvack();
   resend_gb();
   printf("\n All Frames sent Successfully\n");
}

int sender()
```

```
{
   int i;
   printf("\n Enter no. of Frames to be sent: ");
   scanf("%d",&n);
   for(i=1;i<=n;i++)
   {
      printf("\n Enter data for Frames [%d] ", i);
      scanf("%d",&frm[i].data);
      frm[i].ack='y';
   }
   return 0;
}

void recvack()
{
   int i;
   //rand();
   r = rand()%n;
   frm[r].ack='n';
   for(i=1;i<=n;i++)
   {
      if(frm[i].ack=='n')
         printf("\n The frame Number %d is not Recieved",r);
   }
}
void resend_gb()
{
   int i;
```

```
    printf("\n Resending Frame %d ", r);
    for(i=r;i<=n;i++)
    {
        sleep(2);
        frm[i].ack='y';
        printf("\n The Recieved Frame is %d",frm[i].data);
    }
}
```

Output:



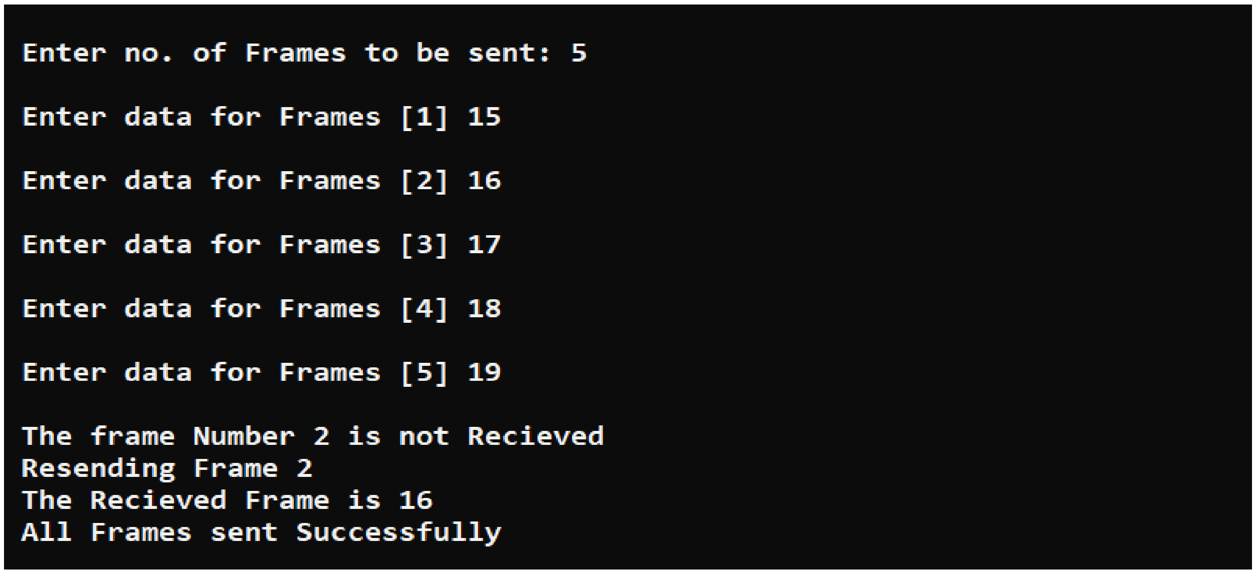
```
Enter no. of Frames to be sent: 4

Enter data for Frames [1] 1

Enter data for Frames [2] 2

Enter data for Frames [3] 3

Enter data for Frames [4] 4

The frame Number 1 is not Recieved
Resending Frame 1
The Recieved Frame is 1
The Recieved Frame is 2
The Recieved Frame is 3
The Recieved Frame is 4
All Frames sent Successfully
```

Experiment : 7

Aim: Algorithm to implement Sliding window protocol for Selective repeat.

Program:

```
#include <stdio.h>
#include<stdlib.h>
#include<math.h>
#include<unistd.h>
int n,r;
struct frame
{
    char ack;
    int data;
}frm[10];
int sender(void);
void recvack(void);
void resend_sr(void);
int main()
{
    sender();
    recvack();
    resend_sr();
    printf("\n All Frames sent Successfully\n");
}

int sender()
{
    int i;
    printf("\n Enter no. of Frames to be sent: ");
    scanf("%d",&n);
    for(i=1;i<=n;i++)
    {
        printf("\n Enter data for Frames [%d] ", i);
        scanf("%d",&frm[i].data);
        frm[i].ack='y';
    }
    return 0;
}
void recvack()
{
    int i;
    rand();
    r = rand()%n;
    frm[r].ack='n';
    for(i=1;i<=n;i++)
    {
        if(frm[i].ack=='n')
            printf("\n The frame Number %d is not Recieved",r);
    }
}
void resend_sr()
{
    printf("\n Resending Frame %d ", r);
    sleep(2);
    frm[r].ack='y';
    printf("\n The Recieved Frame is %d",frm[r].data);
}
```
OUTPUT:



```
Enter no. of Frames to be sent: 5

Enter data for Frames [1] 15

Enter data for Frames [2] 16

Enter data for Frames [3] 17

Enter data for Frames [4] 18

Enter data for Frames [5] 19

The frame Number 2 is not Recieved
Resending Frame 2
The Recieved Frame is 16
All Frames sent Successfully
```

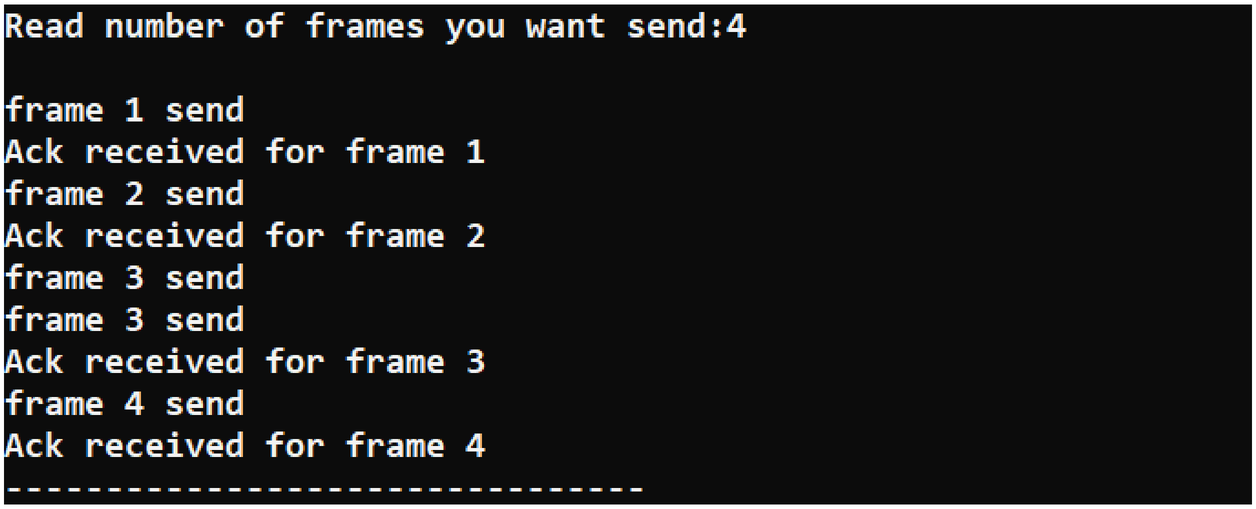
Experiment : 8

Aim: Algorithm to implement Stop and Wait Protocol

Program:

```
int main()
{
        int n,i,wait;
        printf("Read number of frames you want send:");
        scanf("%d",&n);
        for(i=1;i<=n;i++)
        {
                printf("\nframe %d send",i);
                wait=rand()%8;
                sleep(5);
                if(wait>5)
                {
                        i=i-1;
                  continue;
                }
                else
                        printf("\nAck received for frame %d",i);
        }
}
```

Output:



```
Read number of frames you want send:4

frame 1 send
Ack received for frame 1
frame 2 send
Ack received for frame 2
frame 3 send
frame 3 send
Ack received for frame 3
frame 4 send
Ack received for frame 4
--------------------------------
```

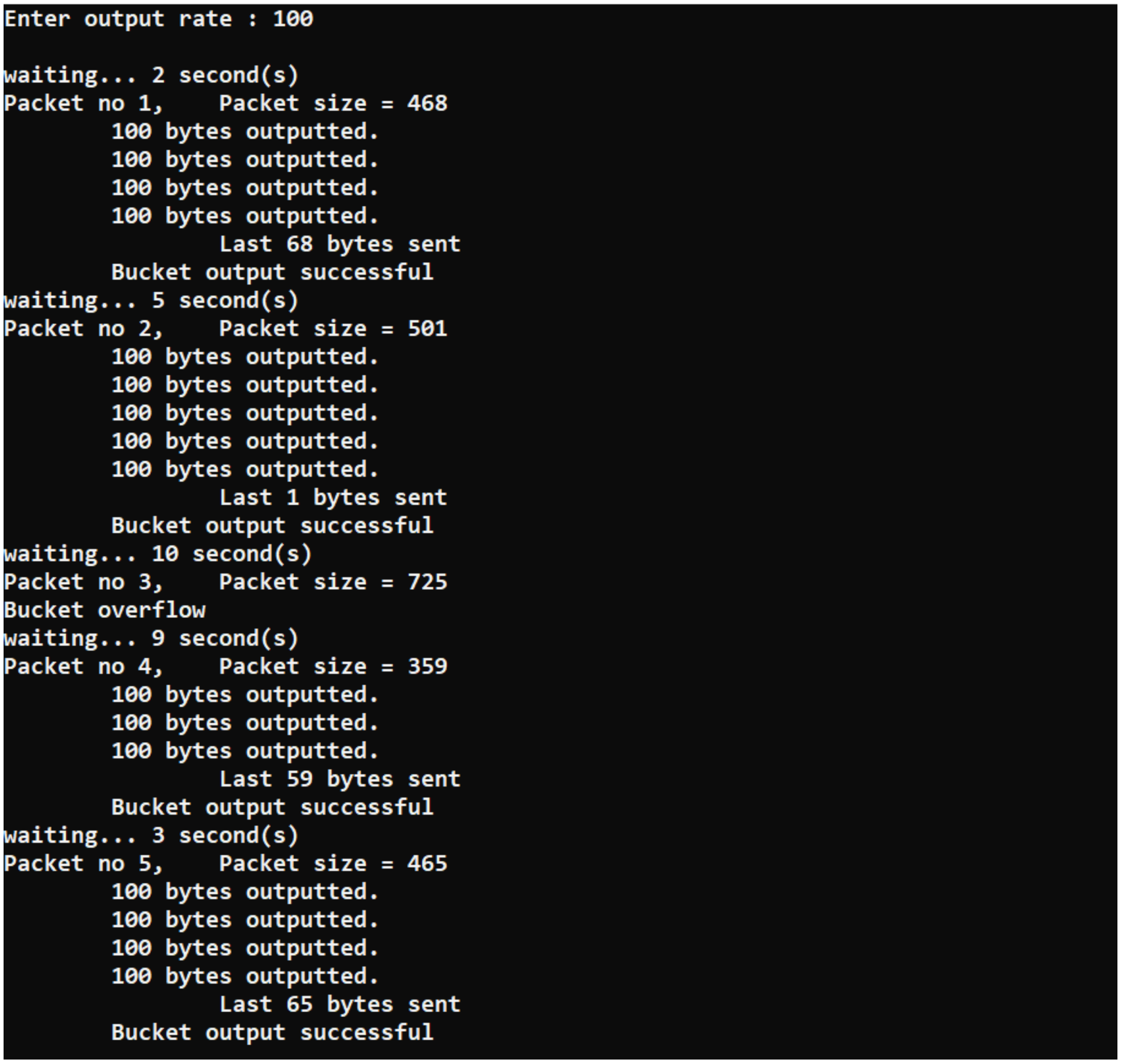
Experiment : 9

Aim: Algorithm to implement congestion control using leaky bucket algorithm

Program:

```
#include<stdio.h>
#include<unistd.h>
#define bucketSize 512
void bktInput(int packetSize,int output)
{
        if(packetSize>bucketSize)
                printf("\nBucket overflow");
        else
        {
                sleep(2);
                while(packetSize>output)
                {
                        printf("\n\t%d bytes outputted.",output);
                        packetSize=packetSize-output;
                        sleep(2);
                }
                if(packetSize>0)
                        printf("\n\t\tLast %d bytes sent\t",packetSize);
                printf("\n\tBucket output successful");
        }
}
int main()
{
        int op,i,pktSize,time;
        //randomize();
        printf("Enter output rate : ");
        scanf("%d",&op);
        for(i=1;i<=5;i++)
        {
                time=rand()%10+1;
                printf("\nwaiting... %d second(s)",time);
                sleep(time);
                //printf("\nwaited\n");
                pktSize=rand()%1000+1;
                printf("\nPacket no %d,\tPacket size = %d",i,pktSize);
                bktInput(pktSize,op);
        }
        return 0;
}
```

Output:



```
Enter output rate : 100

waiting... 2 second(s)
Packet no 1,    Packet size = 468
        100 bytes outputted.
        100 bytes outputted.
        100 bytes outputted.
        100 bytes outputted.
                Last 68 bytes sent
        Bucket output successful
waiting... 5 second(s)
Packet no 2,    Packet size = 501
        100 bytes outputted.
        100 bytes outputted.
        100 bytes outputted.
        100 bytes outputted.
        100 bytes outputted.
                Last 1 bytes sent
        Bucket output successful
waiting... 10 second(s)
Packet no 3,    Packet size = 725
Bucket overflow
waiting... 9 second(s)
Packet no 4,    Packet size = 359
        100 bytes outputted.
        100 bytes outputted.
        100 bytes outputted.
                Last 59 bytes sent
        Bucket output successful
waiting... 3 second(s)
Packet no 5,    Packet size = 465
        100 bytes outputted.
        100 bytes outputted.
        100 bytes outputted.
        100 bytes outputted.
                Last 65 bytes sent
        Bucket output successful
```

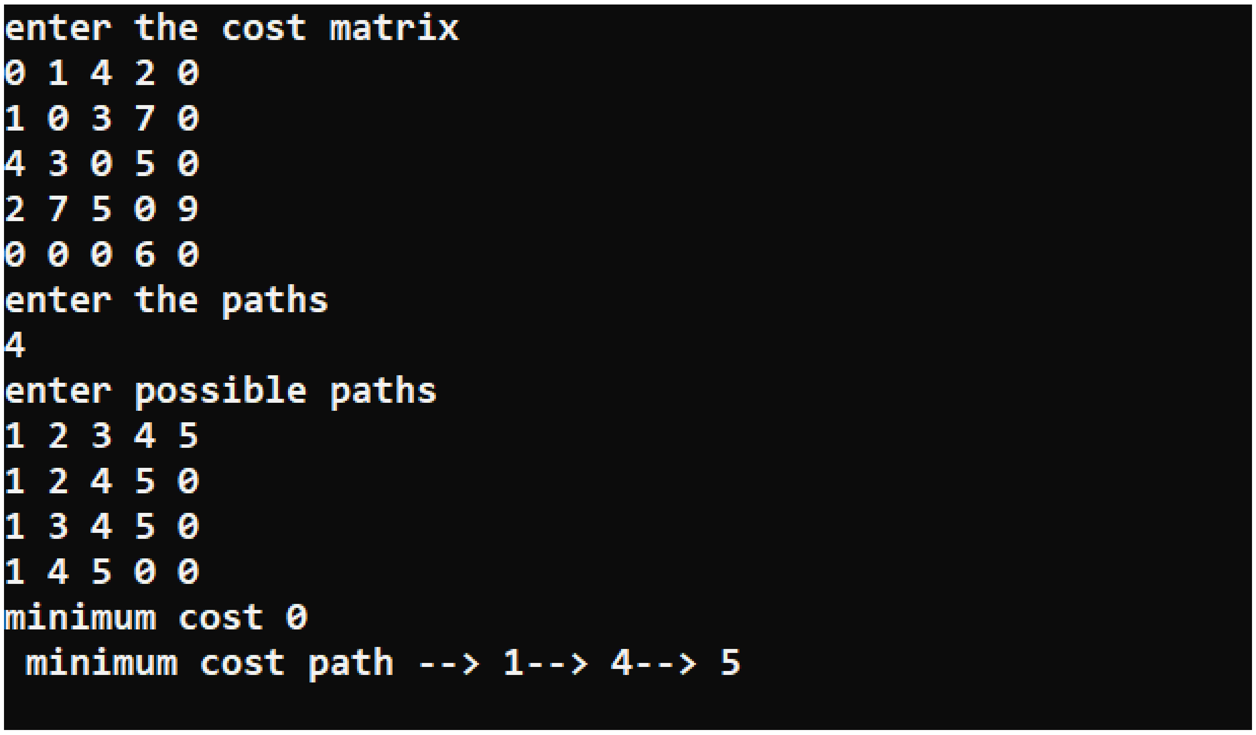
Experiment : 10

Aim: Algorithm to implement Dijkstra‘s algorithm to compute the Shortest path through a graph

Program:

```
#include<stdio.h>
#include<conio.h>
void main()
  {
  int path[5][5],i,j,min,a[5][5],p,st=1,ed=5,stp,edp,t[5],index;
  printf("enter the cost matrix\n");
  for(i=0;i<5;i++)
    for(j=0;j<5;j++)
      scanf("%d",&a[i][j]);
  printf("enter the paths\n");
  scanf("%d",&p);
  printf("enter possible paths\n");
  for(i=0;i<p;i++)
    for(j=0;j<5;j++)
      scanf("%d",&path[i][j]);
  for(i=0;i<p;i++)
    {
    t[i]=0;stp=st;
    for(j=0;j<5;j++)
      {
      edp=path[i][j+1];

      t[i]=t[i]+a[stp][edp];
      if(edp==ed)
        break;
      else
        stp=edp;
      }
    }
  min=t[st];
  index=st;
  for(i=0;i<p;i++)
    {
    if(min>t[i])
      {
      min=t[i];
      index=i;
      }
    }
  printf("minimum cost %d",min);
  printf("\n minimum cost path ");
  for(i=0;i<5;i++)
    {
    printf("--> %d",path[index][i]);
    if(path[index][i]==ed)
      break;
    }
  getch();
  }
```

Output:



```
enter the cost matrix
0 1 4 2 0
1 0 3 7 0
4 3 0 5 0
2 7 5 0 9
0 0 0 6 0
enter the paths
4
enter possible paths
1 2 3 4 5
1 2 4 5 0
1 3 4 5 0
1 4 5 0 0
minimum cost 0
 minimum cost path --> 1--> 4--> 5
```

Experiment : 11

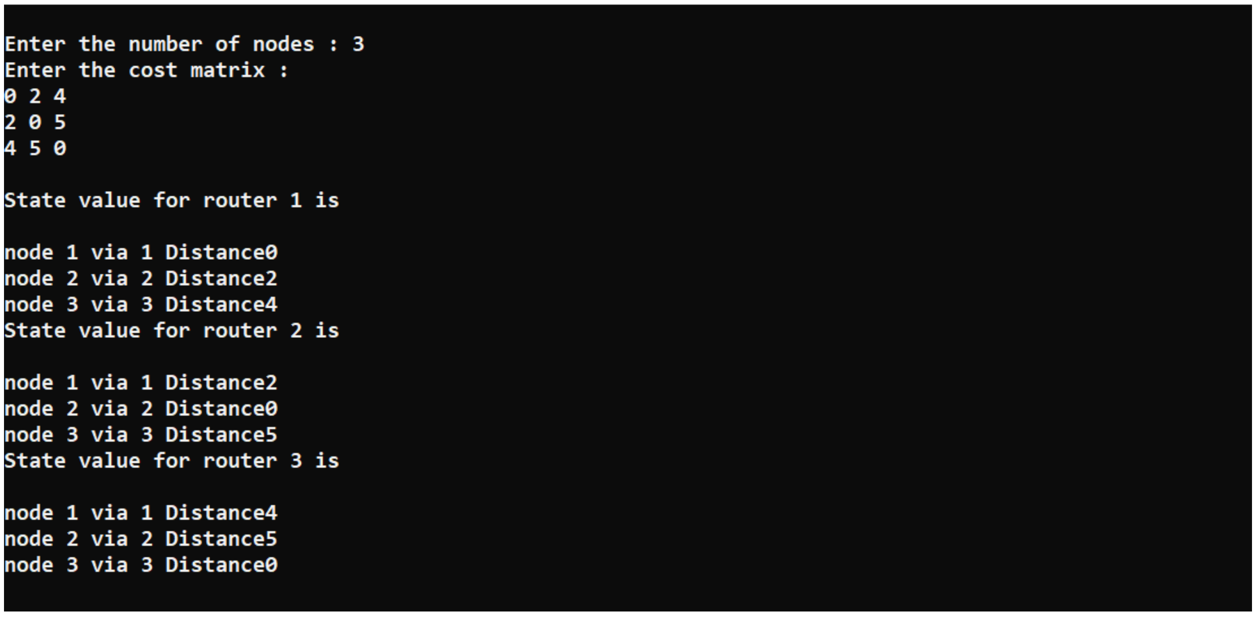
Aim: Algorithm to implementDistance vector routing algorithm by obtaining routing table at each node

Program:

```
#include<stdio.h>
#include<conio.h>
struct node
 {
 unsigned dist[20];
 unsigned from[20];
 }rt[10];
void main()
 {
 int dmat[20][20];
 int n,i,j,k,count=0;
 printf("\nEnter the number of nodes : ");
 scanf("%d",&n);

 printf("Enter the cost matrix :\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;
   }
  }
 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);
      for(i=0;i<n;i++)
        {
        printf("\nState value for router %d is \n",i+1);
        for(j=0;j<n;j++)
          {
           printf("\nnode %d via %d Distance%d",j+1,rt[i].from[j]+1,rt[i].dist[j]);
          }
        }
      printf("\n");
      }
```

Output:



```
Enter the number of nodes : 3
Enter the cost matrix :
0 2 4
2 0 5
4 5 0

State value for router 1 is

node 1 via 1 Distance0
node 2 via 2 Distance2
node 3 via 3 Distance4
State value for router 2 is

node 1 via 1 Distance2
node 2 via 2 Distance0
node 3 via 3 Distance5
State value for router 3 is

node 1 via 1 Distance4
node 2 via 2 Distance5
node 3 via 3 Distance0
```

Experiment : 12

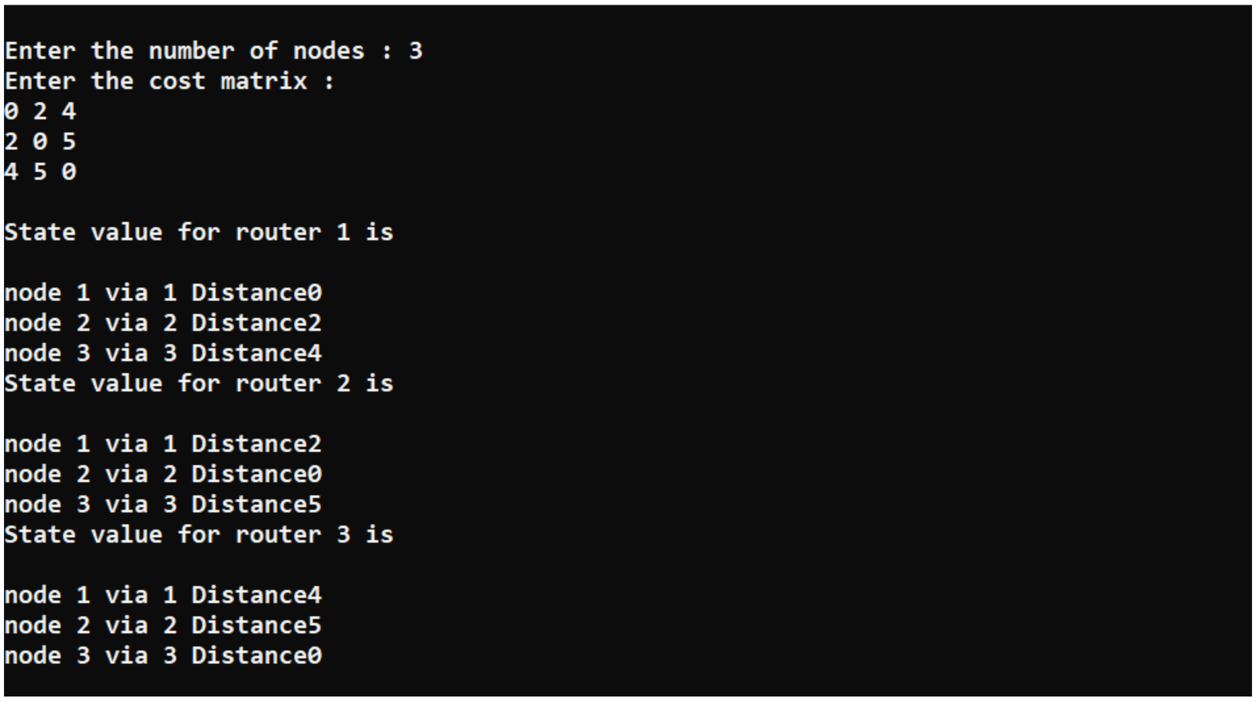
Aim: Algorithm to implement Broadcast tree by taking subnet of hosts

Program:

```
#include<stdio.h>
#include<conio.h>
struct node
  {
  unsigned dist[20];
  unsigned from[20];
  }rt[10];
void main()
  {
  int dmat[20][20];
  int n,i,j,k,count=0;
  printf("\nEnter the number of nodes : ");
  scanf("%d",&n);
  printf("Enter the cost matrix :\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;
      }


    }
  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);
       for(i=0;i<n;i++)
        {
        printf("\nState value for router %d is \n",i+1);
        for(j=0;j<n;j++)
          {
           printf("\nnode %d via %d Distance%d",j+1,rt[i].from[j]+1,rt[i].dist[j]);
          }
        }
       printf("\n");
       }
```

Output:



```
Enter the number of nodes : 3
Enter the cost matrix :
0 2 4
2 0 5
4 5 0

State value for router 1 is

node 1 via 1 Distance0
node 2 via 2 Distance2
node 3 via 3 Distance4
State value for router 2 is

node 1 via 1 Distance2
node 2 via 2 Distance0
node 3 via 3 Distance5
State value for router 3 is

node 1 via 1 Distance4
node 2 via 2 Distance5
node 3 via 3 Distance0
```