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

```
#include<iostream>
#include<bits/stdc++.h>
using namespace std;
char m[50],g[50],r[50],q[50],temp[50];
void shiftleft();
void calrem(){
            int i,j;
   for(i=1;i<=16;i++)
      r[i-1] =((int)temp[i]-48)^((int)g[i]-48)+48;
}
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];
   cout<<"Intermediate remainder :";
   for(i=0;i<n-16;i++)
 {
      if(r[0]=='1')
{
         q[i]='1';
         calrem();
          }
      else{
         q[i]='0';
         shiftleft();
          }
      r[16]=m[17+i];
      r[17]='\0';
      cout<<"REMAINDER "<<i<<" :"<<r<<endl;
      for(j=0;j<=17;j++)
         temp[j]=r[j];
          }
   q[n-16]='\0';
}
void shiftleft(){
   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';
}
int main(){
  int n,i=0;
  char ch;
  int flag=0;
  cout<<"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';

  //divisor
  for(i=0;i<16;i++)
    g[i]='0';
  g[0]=g[4]=g[11]=g[16]='1';
  g[17]='\0';

  cout<<"Generator :"<<g<<endl;
  crc(n);
  cout<<"Quotient :"<<q<<endl;
  caltrans(n);
  cout<<"Transmitted frame :"<<m;
  char d;
  cout<<endl<<"Do you want to change transmitted frame?(y,n";
  cin>>d;
if(d=='y'){
    cout<<"Enter transmitted frame: ";
    cin>>m;
}
  cout<<"CRC Checking"<<endl;
  crc(n);
  for(i=0;i<16;i++)
{
      if(r[i]!='0')
      flag =1;
 }
      if(flag==1)
        cout<<"Error during transmission";
      else
      cout<<"correct";
}
```

OUTPUT:

```
PS C:\CN_PROGRAM> cd "c:\CN_PROGRAM\" ; if ($?) { g++ crc.cpp -o crc } ; if ($?) { .\crc }
Enter the frame bits: 100100
Generator :10001000000100001
Intermediate remainder :REMAINDER 0 :00110000001000010
REMAINDER 1 :01100000010000100
REMAINDER 2 :11000000100001000
REMAINDER 3 :10010001001010010
REMAINDER 4 :00110010011100110
REMAINDER 5 :0110010011100110
Quotient :100110
Transmitted frame :10010001100100111100110
Do you want to change transmitted frame?(y,n)n
CRC Checking
Intermediate remainder :REMAINDER 0 :00110011000001100
REMAINDER 1 :01100110000011000
REMAINDER 2 :11001100000110001
REMAINDER 3 :10001000000100001
REMAINDER 4 :00000000000000000
REMAINDER 5 :0000000000000000
correct
```

2.Write a program for distance vector algorithm to find suitable path for Transmission.

```
#include <iostream>
#include <stdio.h>
using namespace std;
struct router{
   int dist[10];
   int next[10];
} router[10];

int main(){
   int no;
   cout << "Enter number of router : " ;
   cin >> no;
   cout << "Enter adjacency matrix : " ;
   int vt[no][no];
          for (int i = 0; i < no; i++){
          for (int j = 0; j < no; j++){
         cin >> router[i].dist[j];
         router[i].next[j] = j;
          }
      cout ;
          }
          for (int i = 0; i < no; i++){
          for (int j = 0; j < no; j++){
```

```
            for (int k = 0; k < no; k++){
              if (router[i].dist[j] > router[i].dist[k] + router[k].dist[j]) {
                  router[i].dist[j] = router[i].dist[k] + router[j].dist[k];
                  router[i].next[j] = k;
            }
            }
            }
            }
            for (int i = 0; i < no; i++){
        cout << "Router info for router: " << i + 1 << endl;
        cout << "Dest\tNext Hop\tDist" << endl;
        for (int j = 0; j < no; j++)
           printf("%d\t%d\t\t%d\n", j + 1, router[i].next[j] + 1, router[i].dist[j]);
            }
    return 0;
}
```

OUTPUT:

```
PS C:\CN_PROGRAM> cd "c:\CN_PROGRAM\" ; if ($?) { g++ distancevector.cpp -o distancevector } ; if ($?) { .\distancevector }
Enter number of router : 7
Enter adjacency matrix :
0 2 99 3 99 99 99
2 0 5 99 4 99 99
99 5 0 99 99 4 3
3 99 99 0 5 99 99
99 4 99 5 0 2 99
99 99 4 99 2 0 1
99 99 3 99 99 1 0
Router info for router: 1
Dest    Next Hop        Dist
1       1               0
2       2               2
3       2               7
4       4               3
5       2               6
6       5               8
7       6               9
Router info for router: 2
Dest    Next Hop        Dist
1       1               2
2       2               0
3       3               5
4       1               5
5       5               4
6       5               6
7       6               7
Router info for router: 3
7       7               3
Router info for router: 4
Dest    Next Hop        Dist
1       1               3
2       1               5
3       1               10
4       4               0
5       5               5
6       5               7
7       6               8
```

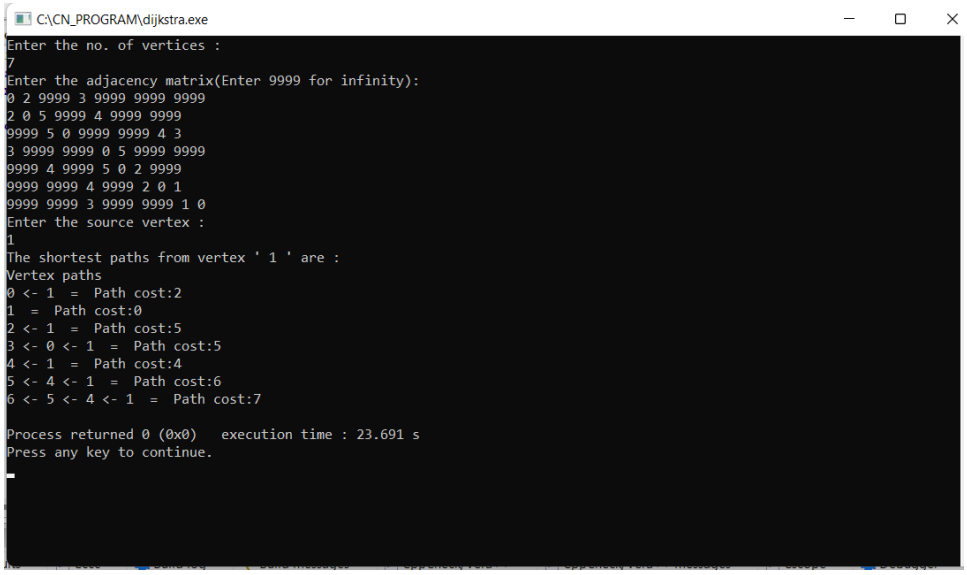
```
Router info for router: 4
Dest    Next Hop        Dist
1       1               3
2       1               5
3       1               10
4       4               0
5       5               5
6       5               7
7       6               8
Router info for router: 5
Dest    Next Hop        Dist
1       2               6
2       2               4
3       6               6
4       4               5
5       5               0
6       6               2
7       6               3
Router info for router: 6
Dest    Next Hop        Dist
1       5               8
2       5               6
3       3               4
4       5               7
5       5               2
6       6               0
7       7               1
Router info for router: 7
Dest    Next Hop        Dist
1       6               9
2       6               7
3       3               3
4       6               8
5       6               3
6       6               1
7       7               0
```

3. Implement Dijkstra's algorithm to compute the shortest path for a given Topology.

```
#include <iostream>
using namespace std;
int a[30][30], source, dist[30], path[30];
void dijkstar(int a[][30], int n){
   int visited[n];
          for (int i = 0; i < n; i++){
      dist[i] = a[source][i];
      path[i] = source;
      visited[i] = 0;
          }
   visited[source] = 1;
          for (int c = 0; c < n; c++){
      int min = 999, u;
          for (int j = 0; j < n; j++){
          if (dist[j] < min && visited[j] != 1){
            min = dist[j];
            u = j;
          }
          }
      visited[u] = 1;
          for (int i = 0; i < n; i++){
          if (min + a[u][i] < dist[i]){
            dist[i] = min + a[u][i];
            path[i] = u;
          }
          }
          }
}
int main(){
```

```
    int n;
    cout << "Enter the no. of vertices :" << endl;
    cin >> n;
    cout << "Enter the adjacency matrix(Enter 9999 for infinity):" << endl;
            for (int i = 0; i < n; i++){
            for (int j = 0; j < n; j++){
          cin >> a[i][j];
            }
            }
    cout << "Enter the source vertex :" << endl;
    cin >> source;
    cout << "The shortest paths from vertex ' " << source << " ' are :" << endl;
    cout << "Vertex paths" << endl;
    dijkstar(a, n);
            for (int i = 0; i < n; i++){
       int k = i;
            while (k != source){
          cout << k << " <- ";
          k = path[k];
            }
       cout << source << "  =  ";
       cout << "Path cost:" << dist[i] << endl;
            }
    return 0;
}
```

OUTPUT:



4. Write a program for congestion control using Leaky bucket algorithm.

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

```
#include <unistd.h>
#define NOF_PACKETS 5
int main(){
   int packet_sz[NOF_PACKETS], i, clk, b_size, o_rate, p_sz_rm = 0, p_sz, p_time, op;
   for (i = 0; i < NOF_PACKETS; ++i)
      packet_sz[i] = rand() % 100;
   for (i = 0; i < NOF_PACKETS; ++i)
      printf("\npacket[%d]:%d bytes\t", i, packet_sz[i]);
   printf("\nEnter the Output rate:");
   scanf("%d", &o_rate);
   printf("Enter the Bucket Size:");
   scanf("%d", &b_size);
          for (i = 0; i < NOF_PACKETS; ++i){
      if ((packet_sz[i] + p_sz_rm) > b_size)
          if (packet_sz[i] > b_size) /*compare the packet siz with bucket size*/
            printf("\n\nIncoming packet size (%dbytes) is Greater than bucket capacity (%dbytes)-PACKET
REJECTED", packet_sz[i], b_size);
         else
            printf("\n\nBucket capacity exceeded-PACKETS REJECTED!!");
         else{
         p_sz_rm += packet_sz[i];
         printf("\n\nIncoming Packet size: %d", packet_sz[i]);
         printf("\nBytes remaining to Transmit: %d", p_sz_rm);
         //p_time = random() * 10;
         //printf("\nTime left for transmission: %d units", p_time);
         //for(clk = 10; clk <= p_time; clk += 10)
           while (p_sz_rm > 0){
            sleep(1);
            if (p_sz_rm) {
                  if (p_sz_rm <= o_rate) /*packet size remaining          comparing with output rate*/
               op = p_sz_rm, p_sz_rm = 0;
              else
                 op = o_rate, p_sz_rm -= o_rate;
              printf("\nPacket of size %d Transmitted", op);
              printf("----Bytes Remaining to Transmit: %d", p_sz_rm);
            }
             else{
              printf("\nNo packets to transmit!!");
            }
            }
            }
            }
}
```

OUTPUT:

```
PS C:\CN_PROGRAM> cd "c:\CN_PROGRAM\" ; if ($?) { gcc leaky_bucket.c -o leaky_bucket } ; if ($?) { .\leaky_bucket }

packet[0]:41 bytes
packet[1]:67 bytes
packet[2]:34 bytes
packet[3]:0 bytes
packet[4]:69 bytes
Enter the Output rate:40
Enter the Bucket Size:80


Incoming Packet size: 41
Bytes remaining to Transmit: 41
Packet of size 40 Transmitted----Bytes Remaining to Transmit: 1
Packet of size 1 Transmitted----Bytes Remaining to Transmit: 0

Incoming Packet size: 67
Bytes remaining to Transmit: 67
Packet of size 40 Transmitted----Bytes Remaining to Transmit: 27
Packet of size 27 Transmitted----Bytes Remaining to Transmit: 0

Incoming Packet size: 34
Bytes remaining to Transmit: 34
Packet of size 34 Transmitted----Bytes Remaining to Transmit: 0

Incoming Packet size: 0
Bytes remaining to Transmit: 0

Incoming Packet size: 69
Bytes remaining to Transmit: 69
Packet of size 40 Transmitted----Bytes Remaining to Transmit: 29
Packet of size 29 Transmitted----Bytes Remaining to Transmit: 0
PS C:\CN_PROGRAM>
```

5.Using TCP/IP sockets, write a client-server program to make client sending the file name and the server to send back the contents of the requested file if Present.

Server.py

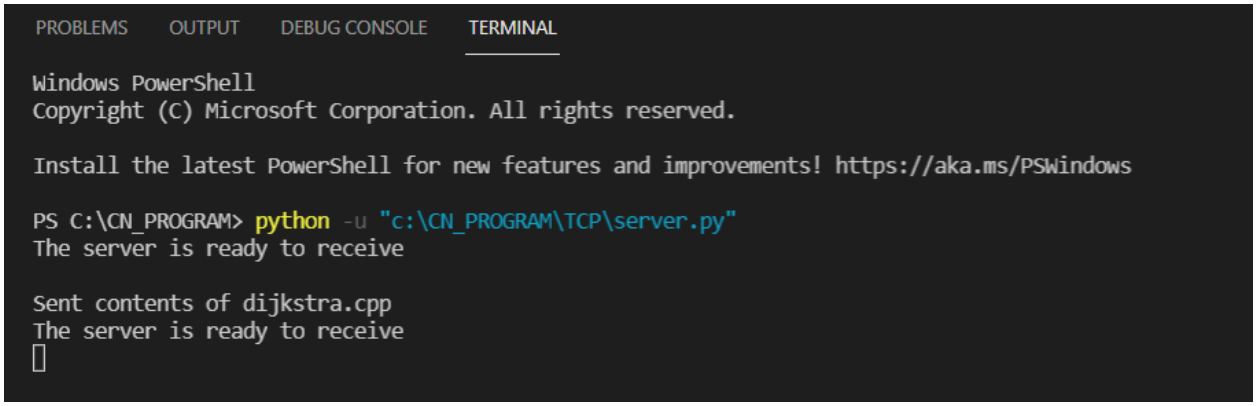
```
from socket import *
serverName = "127.0.0.1"
serverPort = 12000
serverSocket = socket(AF_INET, SOCK_STREAM)
serverSocket.bind((serverName, serverPort))
serverSocket.listen(1)
while 1:
    print("The server is ready to receive")
    connectionSocket, addr = serverSocket.accept()
    sentence = connectionSocket.recv(1024).decode()

    file = open(sentence, "r")
    l = file.read(1024)

    connectionSocket.send(l.encode())
    print('\nSent contents of ' + sentence)
    file.close()
    connectionSocket.close()
```

Client.py

```
from socket import *
serverName = '127.0.0.1'
serverPort = 12000
clientSocket = socket(AF_INET, SOCK_STREAM)
clientSocket.connect((serverName, serverPort))
sentence = input("\nEnter file name: ")

clientSocket.send(sentence.encode())
filecontents = clientSocket.recv(1024).decode()
print('\nFrom Server:\n')
print(filecontents)
clientSocket.close()
```

OUTPUT:

```
PS C:\CN_PROGRAM> cd TCP
PS C:\CN_PROGRAM\TCP> python client.py

Enter file name: dijkstra.cpp

From Server:

#include <iostream>
using namespace std;
int a[30][30], source, dist[30], path[30];

void dijkstar(int a[][30], int n)
{
    int visited[n];
    for (int i = 0; i < n; i++)
        visited[i] = 0;
    }
    visited[source] = 1;
    for (int c = 0; c < n; c++)
    {
        int min = 999, u;
        for (int j = 0; j < n; j++)
        {
            if (dist[j] < min && visited[j] != 1)
            {
                min = dist[j];
                u = j;
            }
        }
        visited[u] = 1;
        for (int i = 0; i < n; i++)
        {
            if (min + a[u][i] < dist[i])
            {
                dist[i] = min + a[u][i];
                path[i] = u;
            }
        }
    }
}
int main()
{
    int n;
    cout << "Enter the no. of vertices :" << endl;
```



6.Using UDP sockets, write a client-server program to make client sending the file name and the server to send back the contents of the requested file if Present.

ClientUDP.py

```
from socket import *
serverName = "127.0.0.1"
serverPort = 12000
clientSocket = socket(AF_INET, SOCK_DGRAM)

sentence = input("\nEnter file name:  ")

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

filecontents, serverAddress = clientSocket.recvfrom(2048)
print('\nReply from Server:\n')
print(filecontents.decode("utf-8"))
# for i in filecontents:
# print(str(i), end = '')
clientSocket.close()
```

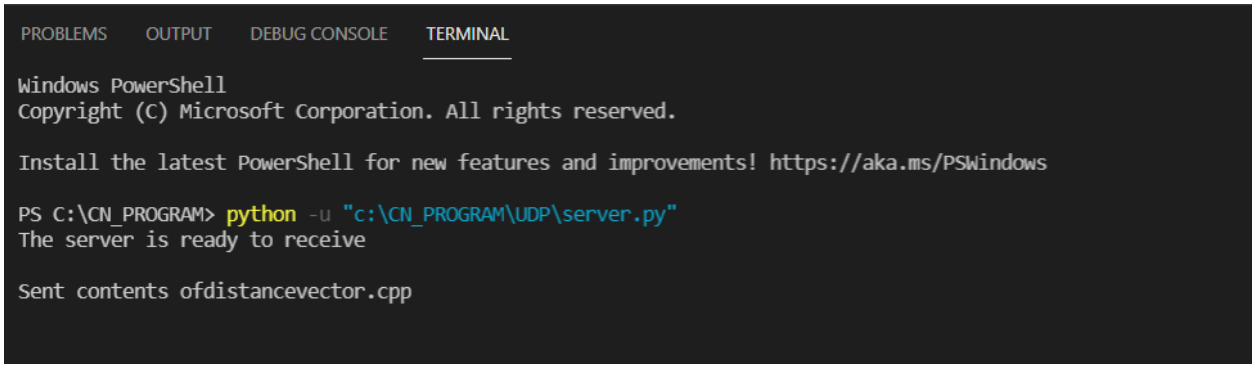
ServerUDP.py

```
from socket import *
serverPort = 12000
serverSocket = socket(AF_INET, SOCK_DGRAM)
serverSocket.bind(("127.0.0.1", serverPort))
print ("The server is ready to receive")
while 1:
  sentence, clientAddress = serverSocket.recvfrom(2048)
  sentence = sentence.decode("utf-8")
```

```
  file=open(sentence,"r")
  l=file.read(2048)
  serverSocket.sendto(bytes(l,"utf-8"),clientAddress)
  print ('\nSent contents of', end =")
  print (sentence)
# for i in sentence:
# print (str(i), end = &#39;&#39;)
  file.close()
```

OUTPUT:

```
PROBLEMS    OUTPUT    DEBUG CONSOLE    TERMINAL

Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

Install the latest PowerShell for new features and improvements! https://aka.ms/PSWindows

PS C:\CN_PROGRAM> cd UDP
PS C:\CN_PROGRAM\UDP> python client.py

Enter file name:  distancevector.cpp

Reply from Server:

#include <iostream>
#include <stdio.h>
using namespace std;
struct router
{
    int dist[10];
    int next[10];
} router[10];

int main()
        for (int j = 0; j < no; j++)
        {
            for (int k = 0; k < no; k++)
            {
                if (router[i].dist[j] > router[i].dist[k] + router[k].dist[j])
                {
                    router[i].dist[j] = router[i].dist[k] + router[j].dist[k];
                    router[i].next[j] = k;
                }
            }
        }
    }

    for (int i = 0; i < no; i++)
    {
        cout << "Router info for router: " << i + 1 << endl;
        cout << "Dest\tNext Hop\tDist" << endl;
        for (int j = 0; j < no; j++)
            printf("%d\t%d\t\t%d\n", j + 1, router[i].next[j] + 1, router[i].dist[j]);
    }
    return 0;
}
```


PROBLEMS
OUTPUT
DEBUG CONSOLE
TERMINAL
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.
Install the latest PowerShell for new features and improvements! https://aka.ms/PSWindows
PS C:\CN_PROGRAM> python -u "c:\CN_PROGRAM\UDP\server.py"
The server is ready to receive
Sent contents ofdistancevector.cpp