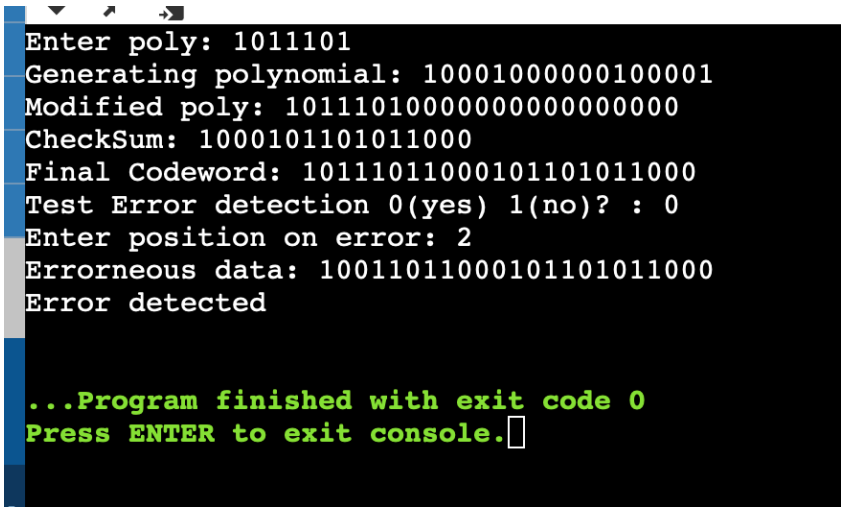
1. Write a program for error detecting code using CRC-CCITT (16 bits).

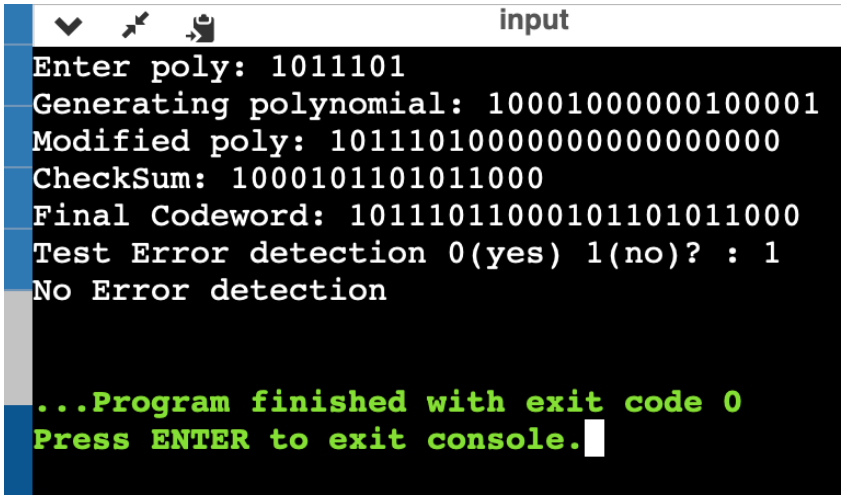
CODE:

```
import java.util.*;

public class CRC{

   public static int n;

   public static void main(String[] args){
      Scanner in=new Scanner(System.in);
      CRC ob=new CRC();

      String code, copy, rec,zero="0000000000000000";

      System.out.print("Enter poly: ");
      code=in.nextLine();

      System.out.println("Generating polynomial: 10001000000100001");

      n=code.length();
      copy=code;
      code+=zero;

      System.out.println("Modified poly: "+code);

      code=ob.divide(code);

      System.out.println("CheckSum: "+code.substring(n));

      copy=copy.substring(0,n)+code.substring(n);

      System.out.println("Final Codeword: "+copy);

      System.out.print("Test Error detection 0(yes) 1(no)? : ");
      int choice = in.nextInt();

      if(choice == 0){
         System.out.print("Enter position on error: ");
         int errorPos = in.nextInt();

         if(copy.charAt(errorPos) == '1')
            copy = copy.substring(0,errorPos) + "0" + copy.substring(errorPos+1);
         else
            copy = copy.substring(0,errorPos) + "1" + copy.substring(errorPos+1);

         System.out.println("Errorneous data: "+copy);

         System.out.println("Error detected");
      }
      else
         System.out.println("No Error detection");
   }

   public String divide(String s){
      int i,j;
      char x;
      String div="10001000000100001";
```

```
        for(i=0;i<n;i++){
            x=s.charAt(i);

            for(j=0;j<17;j++){
                if(x=='1'){
                    if(s.charAt(i+j)!=div.charAt(j))
                        s=s.substring(0,i+j)+"1"+s.substring(i+j+1);
                    else
                        s=s.substring(0,i+j)+"0"+s.substring(i+j+1);
                }
            }
        }

        return s;
    }
}
```

OUTPUT :





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

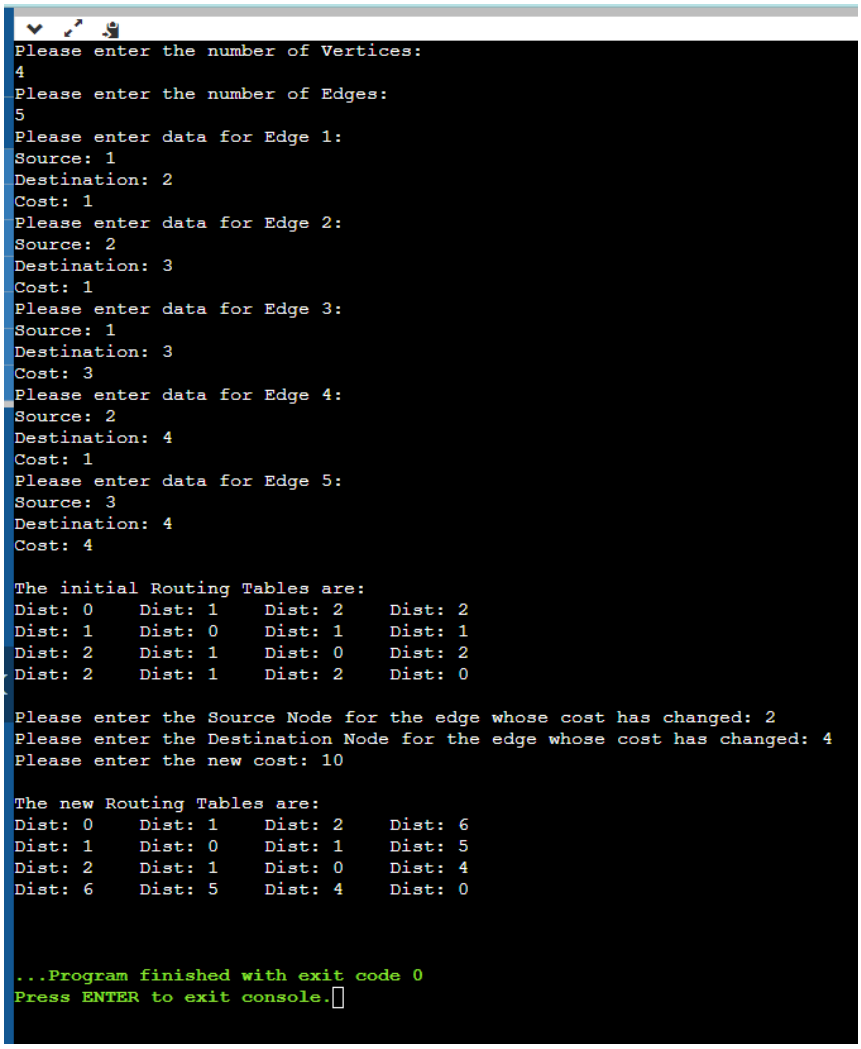
CODE:

```
import java.io.*;
public class Main
{
 static int graph[][];
 static int via[][];
 static int rt[][];
 static int v;
 static int e;

 public static void main(String args[]) throws IOException
 {
  BufferedReader br = new BufferedReader(new InputStreamReader(System.in));

  System.out.println("Please enter the number of Vertices: ");
  v = Integer.parseInt(br.readLine());

  System.out.println("Please enter the number of Edges: ");
  e = Integer.parseInt(br.readLine());

  graph = new int[v][v];
  via = new int[v][v];
  rt = new int[v][v];
  for(int i = 0; i < v; i++)
   for(int j = 0; j < v; j++)
   {
    if(i == j)
     graph[i][j] = 0;
    else
     graph[i][j] = 9999;
   }

  for(int i = 0; i < e; i++)
  {
   System.out.println("Please enter data for Edge " + (i + 1) + ":");
   System.out.print("Source: ");
   int s = Integer.parseInt(br.readLine());
   s--;
   System.out.print("Destination: ");
   int d = Integer.parseInt(br.readLine());
   d--;
   System.out.print("Cost: ");
   int c = Integer.parseInt(br.readLine());
   graph[s][d] = c;
   graph[d][s] = c;
  }

  dvr_calc_disp("The initial Routing Tables are: ");

  System.out.print("Please enter the Source Node for the edge whose cost has changed: ");
  int s = Integer.parseInt(br.readLine());
  s--;
  System.out.print("Please enter the Destination Node for the edge whose cost has changed: ");
  int d = Integer.parseInt(br.readLine());
  d--;
  System.out.print("Please enter the new cost: ");
  int c = Integer.parseInt(br.readLine());
```

```
 graph[s][d] = c;
 graph[d][s] = c;

 dvr_calc_disp("The new Routing Tables are: ");
}

static void dvr_calc_disp(String message)
{
 System.out.println();
 init_tables();
 update_tables();
 System.out.println(message);
 print_tables();
 System.out.println();
}

static void update_table(int source)
{
 for(int i = 0; i < v; i++)
 {
  if(graph[source][i] != 9999)
  {
   int dist = graph[source][i];
   for(int j = 0; j < v; j++)
   {
    int inter_dist = rt[i][j];
    if(via[i][j] == source)
     inter_dist = 9999;
    if(dist + inter_dist < rt[source][j])
    {
     rt[source][j] = dist + inter_dist;
     via[source][j] = i;
    }
   }
  }
 }
}

static void update_tables()
{
 int k = 0;
 for(int i = 0; i < 4*v; i++)
 {
  update_table(k);
  k++;
  if(k == v)
   k = 0;
 }
}

static void init_tables()
{
 for(int i = 0; i < v; i++)
 {
  for(int j = 0; j < v; j++)
  {
   if(i == j)
   {
    rt[i][j] = 0;
    via[i][j] = i;
```

```
    }
    else
    {
     rt[i][j] = 9999;
     via[i][j] = 100;
    }
   }
  }
 }

 static void print_tables()
 {
  for(int i = 0; i < v; i++)
  {
   for(int j = 0; j < v; j++)
   {
    System.out.print("Dist: " + rt[i][j] + "    ");
   }
   System.out.println();
  }
 }

}
```

OUTPUT :



```
Please enter the number of Vertices:
4
Please enter the number of Edges:
5
Please enter data for Edge 1:
Source: 1
Destination: 2
Cost: 1
Please enter data for Edge 2:
Source: 2
Destination: 3
Cost: 1
Please enter data for Edge 3:
Source: 1
Destination: 3
Cost: 3
Please enter data for Edge 4:
Source: 2
Destination: 4
Cost: 1
Please enter data for Edge 5:
Source: 3
Destination: 4
Cost: 4

The initial Routing Tables are:
Dist: 0    Dist: 1    Dist: 2    Dist: 2
Dist: 1    Dist: 0    Dist: 1    Dist: 1
Dist: 2    Dist: 1    Dist: 0    Dist: 2
Dist: 2    Dist: 1    Dist: 2    Dist: 0

Please enter the Source Node for the edge whose cost has changed: 2
Please enter the Destination Node for the edge whose cost has changed: 4
Please enter the new cost: 10

The new Routing Tables are:
Dist: 0    Dist: 1    Dist: 2    Dist: 6
Dist: 1    Dist: 0    Dist: 1    Dist: 5
Dist: 2    Dist: 1    Dist: 0    Dist: 4
Dist: 6    Dist: 5    Dist: 4    Dist: 0



...Program finished with exit code 0
Press ENTER to exit console.
```

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

CODE:

```
import java.util.*;

class Edge{
    int src, dest, w;
    public Edge(int src, int dest, int w){
        this.src = src;
        this.dest = dest;
        this.w = w;
    }
}

class Node {
    int vertex, w;
    public Node(int vertex, int w) {
        this.vertex = vertex;
        this.w = w;
    }
}


class Graph{
    List<List<Edge>> edgeList = null;
    Graph(List<Edge> edges, int N){
        edgeList = new ArrayList<>();
        for (int i = 0; i < N; i++) {
            edgeList.add(new ArrayList<>());
        }
        for (Edge edge: edges){
            edgeList.get(edge.src).add(edge);
        }
    }
}

class Dijkstra{
    private static void getPath(int[] prev, int i, List<Integer> route){
        if (i >= 0){
            getPath(prev, prev[i], route);
            route.add(i);
        }
    }

    public static void getShortestPath(Graph graph, int src, int N){
        PriorityQueue<Node> minHeap;
        minHeap = new PriorityQueue<>(Comparator.comparingInt(node -> node.w));
        minHeap.add(new Node(src, 0));
        List<Integer> dist = new ArrayList<>(Collections.nCopies(N, Integer.MAX_VALUE));
        dist.set(src, 0);
        boolean[] done = new boolean[N];
        done[src] = true;
        int[] prev = new int[N];
        prev[src] = -1;
        List<Integer> route = new ArrayList<>();
        while (!minHeap.isEmpty()){
            Node node = minHeap.poll();
            int u = node.vertex;
            for (Edge edge: graph.edgeList.get(u)){
```

```
            int v = edge.dest;
            int w = edge.w;
            if (!done[v] && (dist.get(u) + w) < dist.get(v)){
                dist.set(v, dist.get(u) + w);
                prev[v] = u;
                minHeap.add(new Node(v, dist.get(v)));
            }
        }
        done[u] = true;
    }

    for(int i = 1; i < N; ++i){
        if (i != src && dist.get(i) != Integer.MAX_VALUE) {
            getPath(prev, i, route);
            System.out.printf("Route is %d => %d and min cost = %d and path is %s\n",
                    src, i, dist.get(i), route);
            route.clear();
        }
    }
}

public static void main(String[] args){
    Scanner s = new Scanner(System.in);
    List<Edge> edges = new ArrayList<>();
    System.out.println("Enter number of vertices");
    int n = s.nextInt();
    System.out.println("Enter the adjacency weighted matrix");
    int[][] mat = new int[n][n];
    for(int i=0; i<n; i++){
        for(int j=0; j<n; j++){
            mat[i][j] = s.nextInt();
        }
    }

    for(int i=0; i<n; i++){
        for(int j=0; j<n; j++){
            if(i == j) continue;
            if(mat[i][j] != -1){
                edges.add(new Edge(i, j, mat[i][j]));
            }
        }
    }
    Graph graph = new Graph(edges, n);
    int src = 0;
    getShortestPath(graph, src, n);
    s.close();
}
}
```

OUTPUT :

```
Enter number of vertices
5
Enter the adjacency weighted matrix
-1 10 -1 -1 3
-1 -1 2 -1 4
-1 -1 -1 9 -1
-1 -1 7 -1 -1
-1 1 8 2 -1
Route is 0 => 1 and min cost = 4 and path is [0, 4, 1]
Route is 0 => 2 and min cost = 6 and path is [0, 4, 1, 2]
Route is 0 => 3 and min cost = 5 and path is [0, 4, 3]
Route is 0 => 4 and min cost = 3 and path is [0, 4]


...Program finished with exit code 0
Press ENTER to exit console.
```

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

CODE:

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

#define NOF_PACKETS 5
/*
int rand (int a)
{
   int rn = (random() % 10) % a;
   return  rn == 0 ? 1 : rn;
}
*/
/*
 #include <stdlib.h>

      long int random(void);


The random() function uses a nonlinear additive feedback random number generator employing a default ta-
      ble of size 31 long integers to return successive pseudo-random numbers in the range from 0 to RAND_MAX.
      The period of this random number generator is very large, approximately 16 * ((2^31) - 1).
*/
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] = random() % 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 :

```
packet[0]:83 bytes
packet[1]:86 bytes
packet[2]:77 bytes
packet[3]:15 bytes
packet[4]:93 bytes
Enter the Output rate:30
Enter the Bucket Size:85


Incoming Packet size: 83
Bytes remaining to Transmit: 83
Packet of size 30 Transmitted----Bytes Remaining to Transmit: 53
Packet of size 30 Transmitted----Bytes Remaining to Transmit: 23
Packet of size 23 Transmitted----Bytes Remaining to Transmit: 0

Incoming packet size (86bytes) is Greater than bucket capacity (85bytes)-PACKET REJECTED

Incoming Packet size: 77
Bytes remaining to Transmit: 77
Packet of size 30 Transmitted----Bytes Remaining to Transmit: 47
Packet of size 30 Transmitted----Bytes Remaining to Transmit: 17
Packet of size 17 Transmitted----Bytes Remaining to Transmit: 0

Incoming Packet size: 15
Bytes remaining to Transmit: 15
Packet of size 15 Transmitted----Bytes Remaining to Transmit: 0

Incoming packet size (93bytes) is Greater than bucket capacity (85bytes)-PACKET REJECTED

...Program finished with exit code 0
Press ENTER to exit console.
```

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.

CODE:

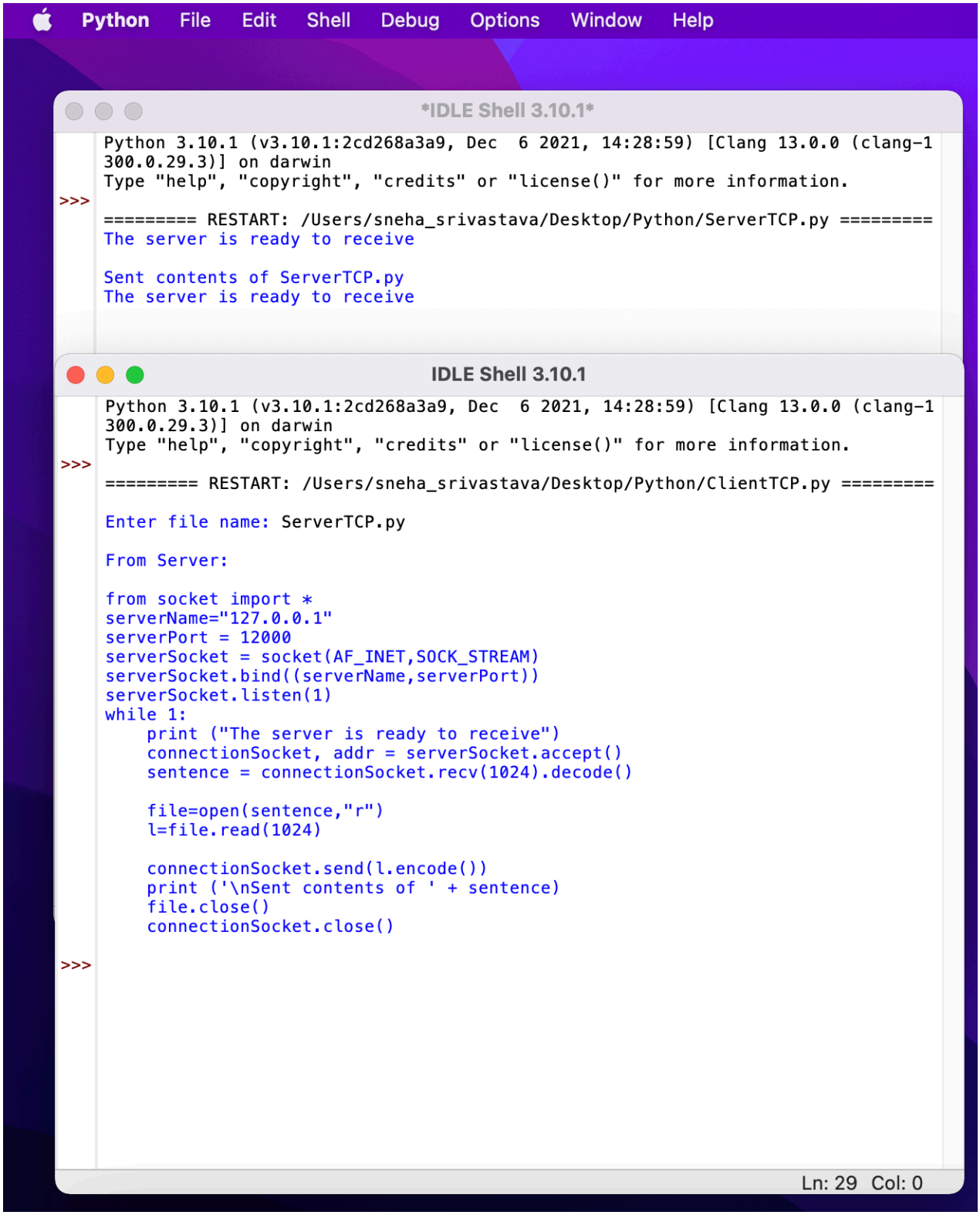
ClientTCP.py

```
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()
```

ServerTCP.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()
```

OUTPUT :


Python File Edit Shell Debug Options Window Help
IDLE Shell 3.10.1
Python 3.10.1 (v3.10.1:2cd268a3a9, Dec 6 2021, 14:28:59) [Clang 13.0.0 (clang-1300.0.29.3)] on darwin
Type "help", "copyright", "credits" or "license()" for more information.
>>>
========= RESTART: /Users/sneha_srivastava/Desktop/Python/ServerTCP.py =========
The server is ready to receive
Sent contents of ServerTCP.py
The server is ready to receive
IDLE Shell 3.10.1
Python 3.10.1 (v3.10.1:2cd268a3a9, Dec 6 2021, 14:28:59) [Clang 13.0.0 (clang-1300.0.29.3)] on darwin
Type "help", "copyright", "credits" or "license()" for more information.
>>>
========= RESTART: /Users/sneha_srivastava/Desktop/Python/ClientTCP.py =========
Enter file name: ServerTCP.py
From Server:
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()
>>>
Ln: 29 Col: 0

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.

CODE:

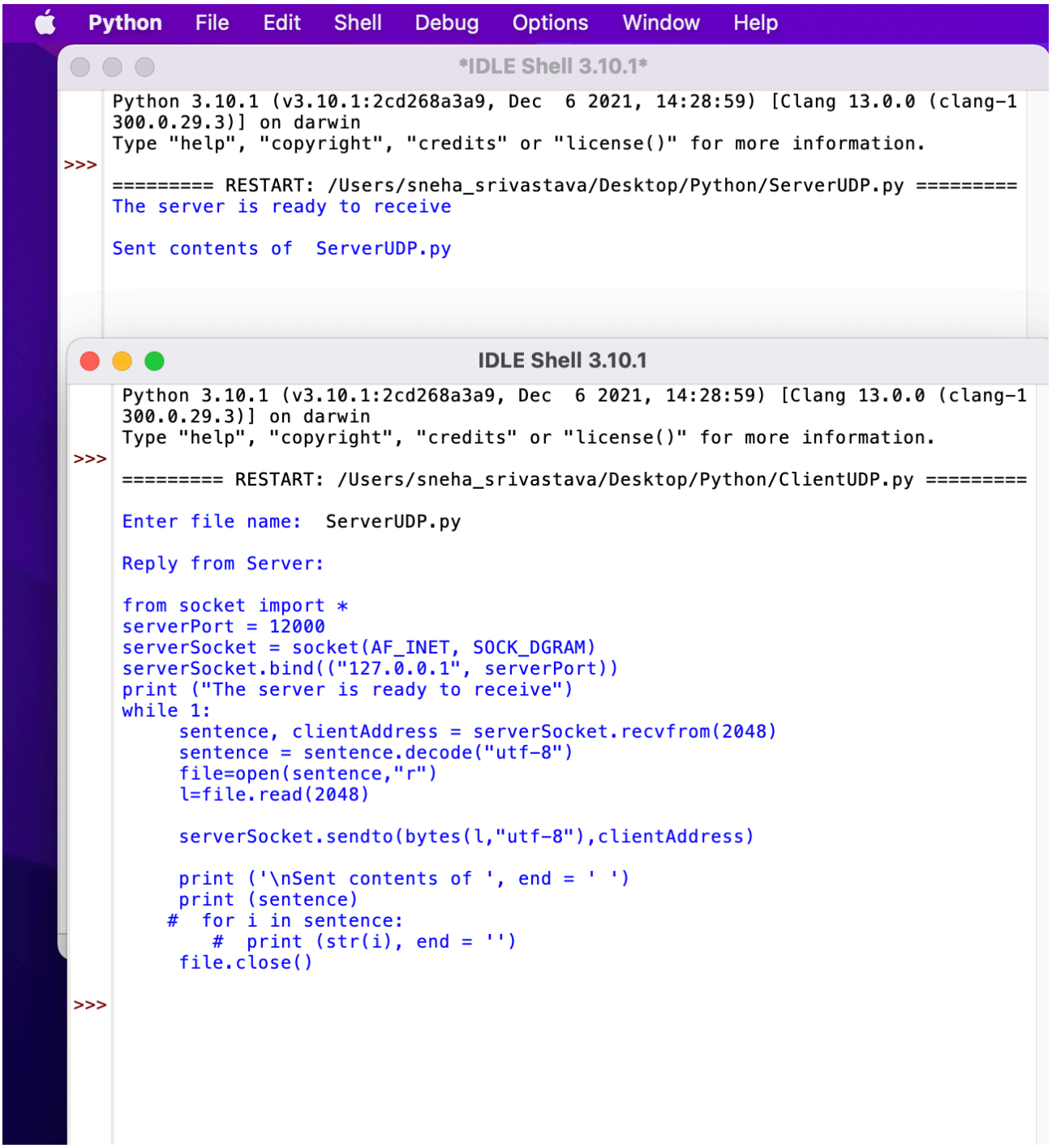
ClientUDP.py

```
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()
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 = '')
    file.close()
```

OUTPUT :


Python File Edit Shell Debug Options Window Help
IDLE Shell 3.10.1
Python 3.10.1 (v3.10.1:2cd268a3a9, Dec 6 2021, 14:28:59) [Clang 13.0.0 (clang-1300.0.29.3)] on darwin
Type "help", "copyright", "credits" or "license()" for more information.
>>>
========= RESTART: /Users/sneha_srivastava/Desktop/Python/ServerUDP.py =========
The server is ready to receive
Sent contents of ServerUDP.py
IDLE Shell 3.10.1
Python 3.10.1 (v3.10.1:2cd268a3a9, Dec 6 2021, 14:28:59) [Clang 13.0.0 (clang-1300.0.29.3)] on darwin
Type "help", "copyright", "credits" or "license()" for more information.
>>>
========= RESTART: /Users/sneha_srivastava/Desktop/Python/ClientUDP.py =========
Enter file name: ServerUDP.py
Reply from Server:
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 = '')
file.close()
>>>