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

```
#include <iostream>
#include <string.h>
using namespace std;

int crc(char *ip, char *op, char *poly, int mode)
{
   strcpy(op, ip);
   if (mode) {
      for (int i = 1; i < strlen(poly); i++)
         strcat(op, "0");
   cout << "modified input" << op <<endl;
   }
   for (int i = 0; i < strlen(ip); i++) {
      if (op[i] == '1') {
         for (int j = 0; j < strlen(poly); j++) {
            if (op[i + j] == poly[j])
               op[i + j] = '0';
            else
               op[i + j] = '1';
         }
      }
   }
   for (int i = 0; i < strlen(op); i++)
      if (op[i] == '1')
         return 0;
   return 1;
}
int main()
{
   char ip[50], op[50], recv[50];
   char poly[] = "10001000000100001";
   int choice;
   cout << "Enter the input message in binary:";
   cin >> ip;
```

```
    cout << "generated polynomial is" << poly <<endl;
    crc(ip, op, poly, 1);
    cout<<"The checksum is:"<<op+strlen(ip)<<endl;
    cout << "The transmitted message is: " << ip << op + strlen(ip) << endl;
    cout << "do you want to test error" << endl;
    cin >> choice;
    if(choice == 1)
    {
        int pos,n;
        char cp[50];
        strcmp(cp, op);
            cout<<"Enter the position where to insert error bit"<<endl;
            cin>>pos;
            cout << "enter bit you wanted to insert" <<endl;
            cin >> n;
            cp[pos]=n;
            if(!strcmp(op, cp))
                {
                    cout << "No error"<<endl;
                }
            else
                {
                    cout << "Error occured"<<endl;
                }
            return 0;
        }
        else{ cout << ""<<endl;}
    cout << "Enter the recevied message in binary" << endl;
    cin >> recv;
    if (crc(recv, op, poly, 0))
      cout << "No error in data" << endl;
    else
      cout << "Error in data transmission has occurred" << endl;
    return 0;
}
```

C:\Users\User\Desktop\crc.exe

```
Enter the input message in binary:1011101
generated polynomial is10001000000100001
modified input10111010000000000000000
The checksum is:1000101101011000
The transmitted message is: 10111011000101101011000
do you want to test error
1
Enter the position where to insert error bit
3
enter bit you wanted to insert
0
Error occured

--------------------------------
Process exited after 48.98 seconds with return value 0
Press any key to continue . . .
```

C:\Users\User\Desktop\crc.exe

```
Enter the input message in binary:1011101
generated polynomial is10001000000100001
modified input10111010000000000000000
The checksum is:1000101101011000
The transmitted message is: 10111011000101101011000
do you want to test error
0

Enter the recevied message in binary
10111011000101101011001
Error in data transmission has occurred

--------------------------------
Process exited after 20.38 seconds with return value 0
Press any key to continue . . .
```

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

```
#include <bits/stdc++.h>

using namespace std;

#define MAX 10

int n;

class router {

char adj_new[MAX], adj_old[MAX];

int table_new[MAX], table_old[MAX];

 public:

 router( ){

for(int i=0;i<MAX;i++) table_old[i]=table_new[i]=99;

 }

void copy( ){

for(int i=0;i<n;i++) {

 adj_old[i] =adj_new[i];

 table_old[i]=table_new[i];

 }

 }

int equal( ) {

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

if(table_old[i]!=table_new[i]||adj_new[i]!=adj_old[i])return 0;

return 1;

 }
```

```
void input(int j) {
 cout<<"Enter 1 if the corresponding router is adjacent to router"
<<(char)('A'+j)<<" else enter 99: "<<endl<<" ";
for(int i=0;i<n;i++)
if(i!=j) cout<<(char)('A'+i)<<" ";
 cout<<"\nEnter matrix:";
for(int i=0;i<n;i++) {
if(i==j)
 table_new[i]=0;
else
 cin>>table_new[i];
 adj_new[i]= (char)('A'+i);
 }
 cout<<endl;
 }
void display(){
 cout<<"\nDestination Router: ";
for(int i=0;i<n;i++) cout<<(char)('A'+i)<<" ";
 cout<<"\nOutgoing Line: ";
for(int i=0;i<n;i++) cout<<adj_new[i]<<" ";
 cout<<"\nHop Count: ";
for(int i=0;i<n;i++) cout<<table_new[i]<<" ";
 }
void build(int j) {
```

```
for(int i=0;i<n;i++)
for(int k=0;(i!=j)&&(k<n);k++)
if(table_old[i]!=99)
if((table_new[i]+table_new[k])<table_new[k]) {
 table_new[k]=table_new[i]+table_new[k];
 adj_new[k]=(char)('A'+i);
 }
 }
} r[MAX];
void build_table( ) {
int i=0, j=0;
while(i!=n) {
for(i=j;i<n;i++) {
 r[i].copy();
 r[i].build(i);
 }
for(i=0;i<n;i++)
if(!r[i].equal()) {
 j=i;
break;
 }
 }
}
int main() {
```

```
 cout<<"Enter the number the routers(<"<<MAX<<"): "; cin>>n;
for(int i=0;i<n;i++) r[i].input(i);
 build_table();
for(int i=0;i<n;i++) {
 cout<<"Router Table entries for router "<<(char)('A'+i)<<":-";
 r[i].display();
 cout<<endl<<endl;
 }
}
```

C:\Users\User\Desktop\distance.exe

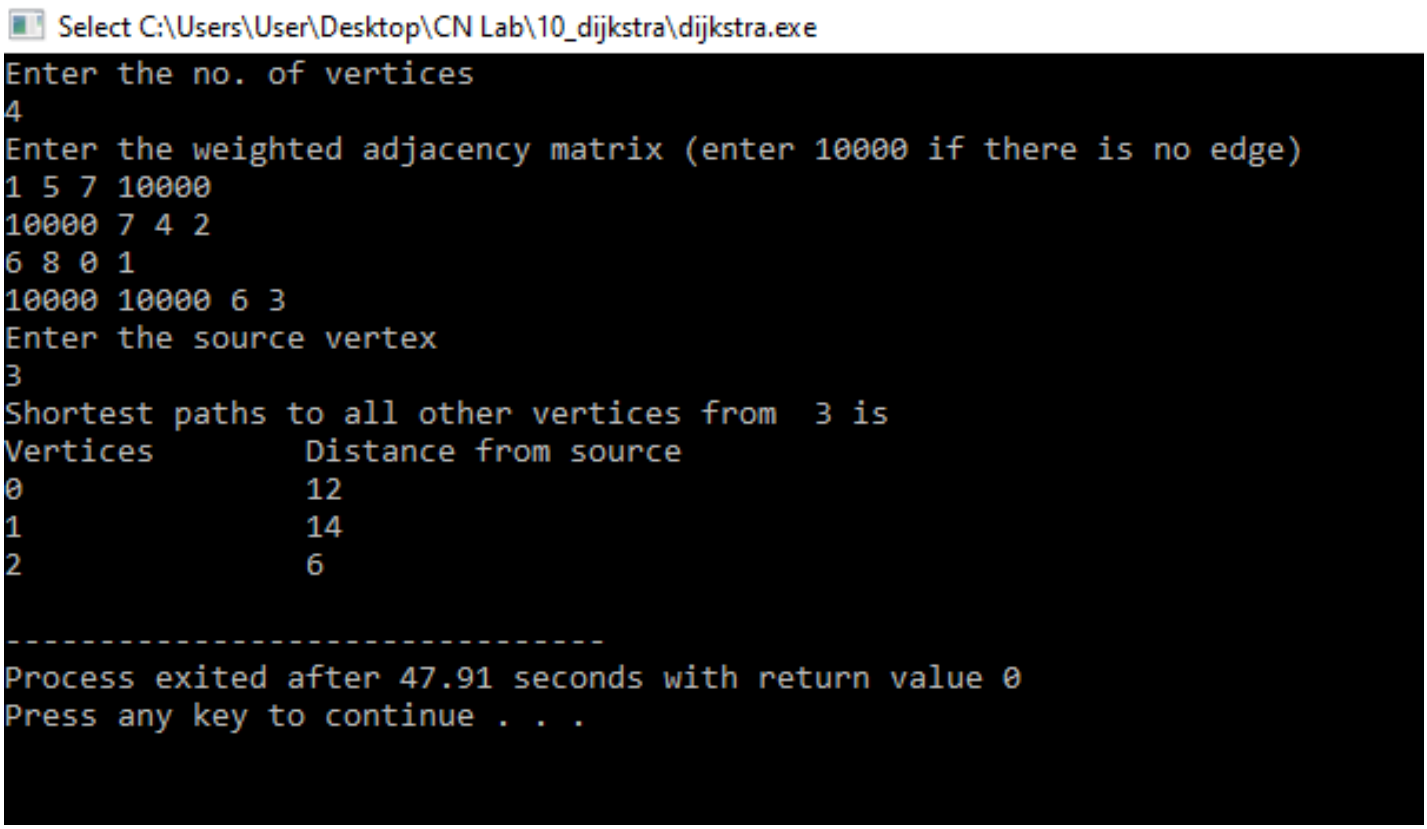
```
Enter the number the routers(<10): 5
Enter 1 if the corresponding router is adjacent to routerA else enter 99:
 B C D E
Enter matrix:1 1 99 99

Enter 1 if the corresponding router is adjacent to routerB else enter 99:
 A C D E
Enter matrix:1 99 99 99

Enter 1 if the corresponding router is adjacent to routerC else enter 99:
 A B D E
Enter matrix:1 99 1 1

Enter 1 if the corresponding router is adjacent to routerD else enter 99:
 A B C E
Enter matrix:99 99 1 99

Enter 1 if the corresponding router is adjacent to routerE else enter 99:
 A B C D
Enter matrix:99 99 1 99

Router Table entries for router A:-
Destination Router: A B C D E
Outgoing Line: A B C D E
Hop Count: 0 1 1 99 99

Router Table entries for router B:-
Destination Router: A B C D E
Outgoing Line: A B C D E
Hop Count: 1 0 99 99 99

Router Table entries for router C:-
Destination Router: A B C D E
Outgoing Line: A B C D E
Hop Count: 1 99 0 1 1

Router Table entries for router D:-
Destination Router: A B C D E
Outgoing Line: A B C D E
Hop Count: 99 99 1 0 99

Router Table entries for router E:-
Destination Router: A B C D E
Outgoing Line: A B C D E
Hop Count: 99 99 1 99 0


--------------------------------
Process exited after 41.26 seconds with return value 0
Press any key to continue . . .
```

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

```
#include<iostream>
#include<climits>
using namespace std;
int a[30][30],n;
int minimum(int visited[],int dist[])
{
	int mindis=10000, mini;
	for(int i=0;i<n;i++)
	{
		if(!visited[i] && dist[i]<mindis)
		{
			mindis=dist[i];
			mini=i;
		}
	}
	return mini;
}

void dijkstra(int src)
{
	int dist[n],visited[n];

	for(int i=0;i<n;i++)
	{
```

```
            dist[i]=10000;
            visited[i]=0;
        }
        dist[src]=0;
        for(int i=0;i<n-1;i++)
        {
            int u=minimum(visited,dist);
            visited[u]=1;
            for(int v=0;v<n;v++)
            {
                if(!visited[v] && a[u][v]!=10000 && dist[u]!=10000 &&
(dist[u]+a[u][v])<dist[v])
                    dist[v]=dist[u]+a[u][v];
            }
        }
        cout<<"Shortest paths to all other vertices from  "<<src<<" is "<<endl;
        cout<<"Vertices\tDistance from source"<<endl;
        for(int i=0;i<n;i++)
        {
            if(i!=src)
                cout<<i<<"\t\t"<<dist[i]<<endl;
        }
}

int main()
{
        cout<<"Enter the no. of vertices"<<endl;
```

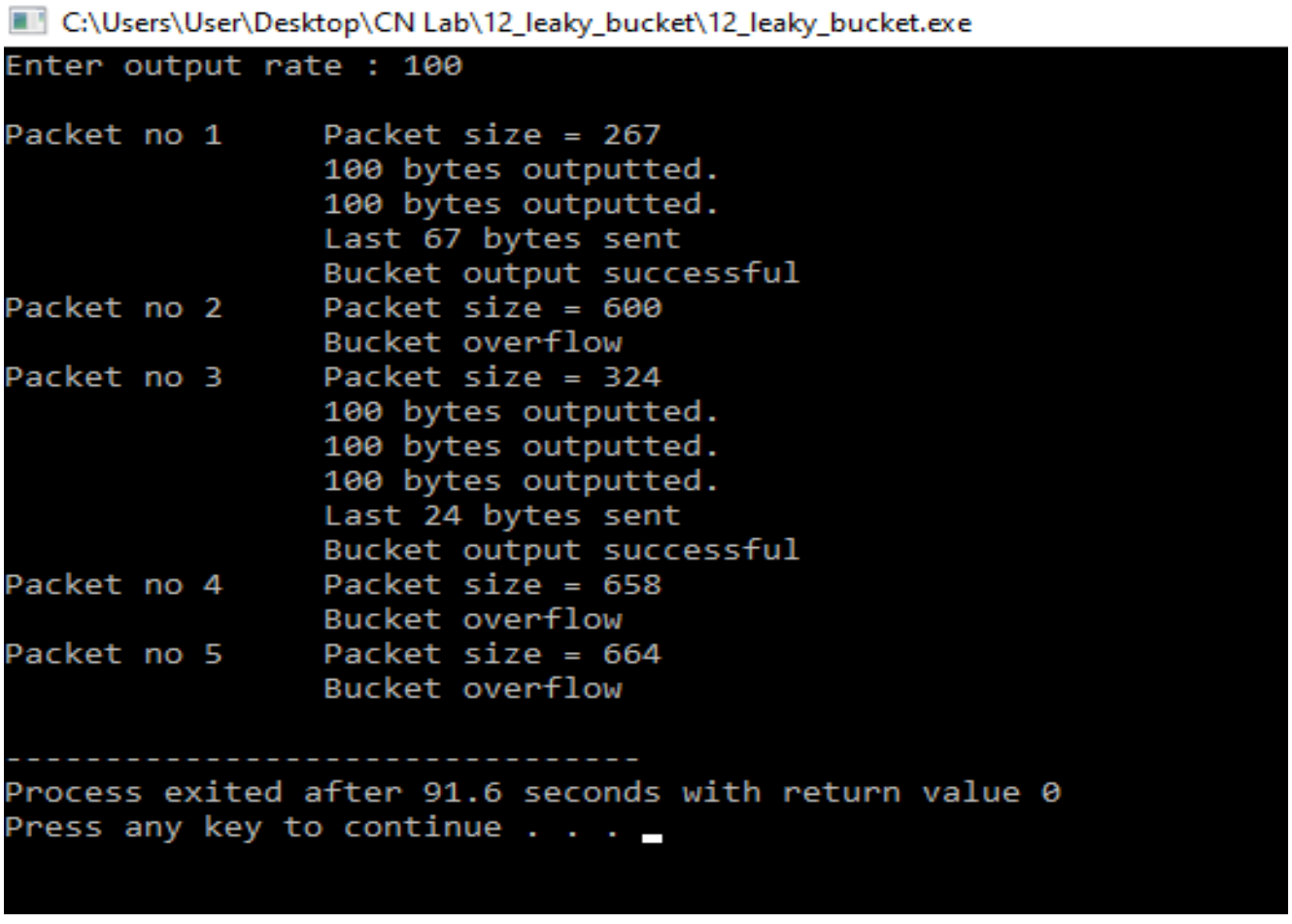
```
    cin>>n;
    cout<<"Enter the weighted adjacency matrix (enter 10000 if there is no edge)"<<endl;
    for(int i=0;i<n;i++)
    {
        for(int j=0;j<n;j++)
            cin>>a[i][j];
    }
    int src;
    cout<<"Enter the source vertex"<<endl;
    cin>>src;
    dijkstra(src);
    return 0;
}
```



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

```
#include<bits/stdc++.h>
#include<unistd.h>
using namespace std;
#define bucketSize 500

void bucketInput(int a,int b)
{
	if(a > bucketSize)
		cout<<"\n\t\tBucket overflow";
	else{
		sleep(5);
		while(a > b){
			cout<<"\n\t\t"<<b<<" bytes outputted.";
			a-=b;
			sleep(5);
		}
		if(a > 0)
			cout<<"\n\t\tLast "<<a<<" bytes sent\t";
		cout<<"\n\t\tBucket output successful";
	}
}
int main()
{
	int op,pktSize;
	cout<<"Enter output rate : ";
```

```
        cin>>op;
        for(int i=1;i<=5;i++)
        {
                sleep(rand()%10);
                pktSize=rand()%700;
                cout<<"\nPacket no "<<i<<"\tPacket size = "<<pktSize;
                bucketInput(pktSize,op);
        }
        cout<<endl;
        return 0;
}
```



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.

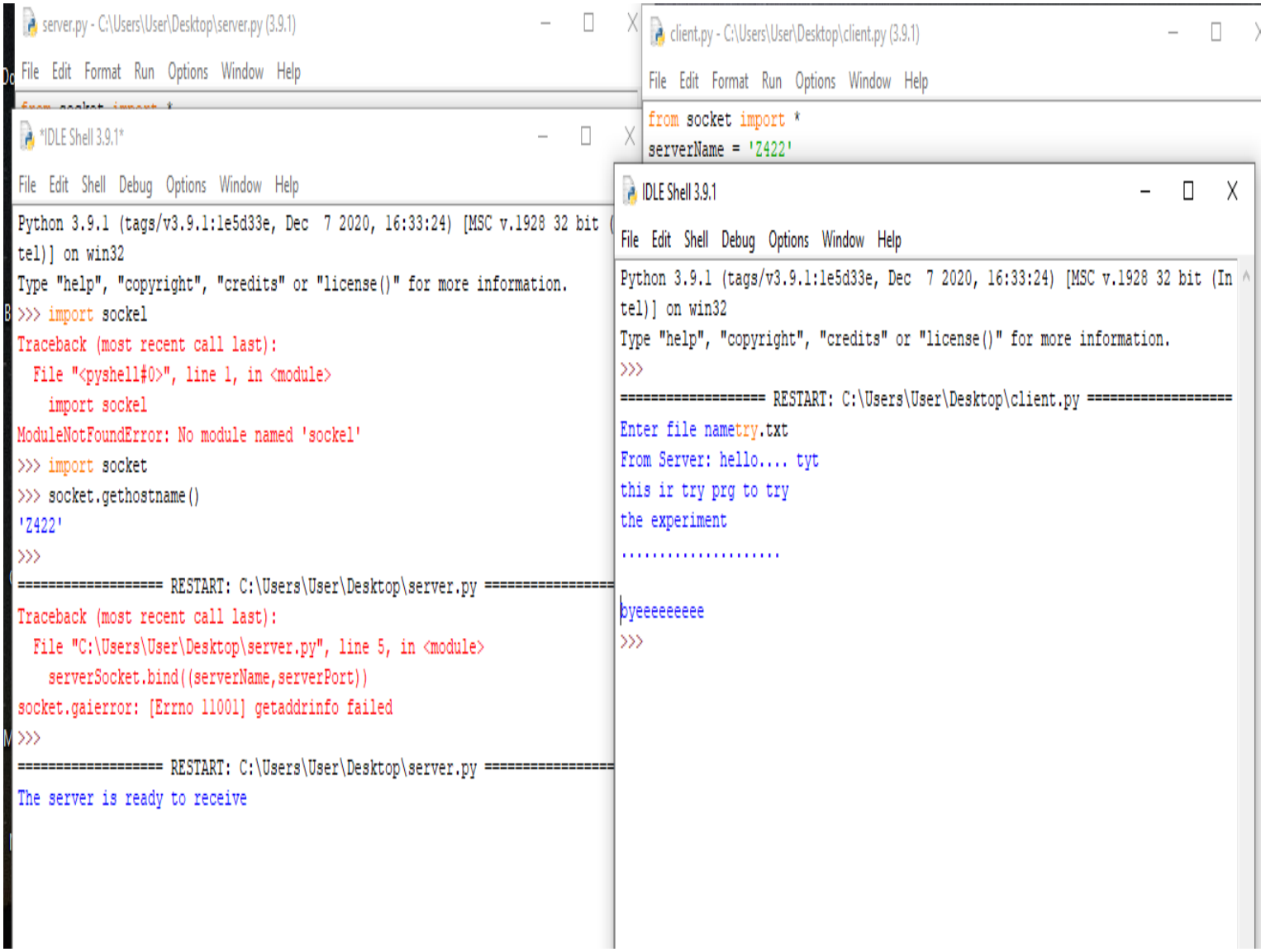
TCPClient.py :

```
from socket import *
serverName = 'Z422'
serverPort = 12001
clientSocket = socket(AF_INET, SOCK_STREAM)
clientSocket.connect((serverName,serverPort))
sentence = input("Enter file name")


clientSocket.send(sentence.encode())
filecontents = clientSocket.recv(1024).decode()
print ('From Server:', filecontents)
clientSocket.close()
```

TCPServer.py :

```
from socket import *
serverName='Z422'
serverPort = 12001
serverSocket = socket(AF_INET,SOCK_STREAM)
serverSocket.bind((serverName,serverPort))
serverSocket.listen(1)
print ("The server is ready to receive")
while 1:
   connectionSocket, addr = serverSocket.accept()
   sentence = connectionSocket.recv(1024).decode()
```

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


connectionSocket.send(l.encode())
file.close()
connectionSocket.close()
```



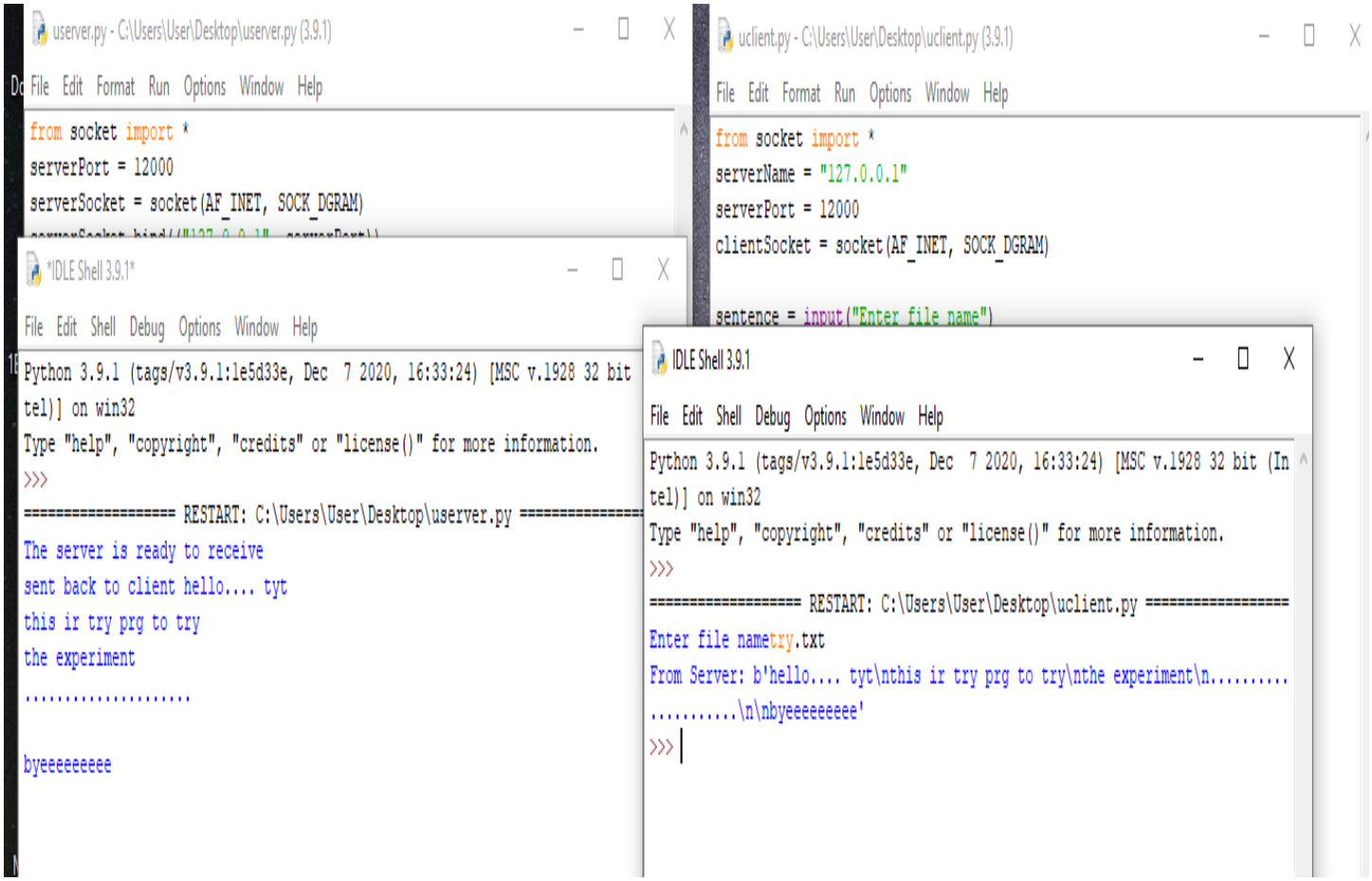
6. Using UDP sockets, write a client-server program to make client sending the file name and the server to send back the contents of the requested file if present.

UDPClient.py:

```
from socket import *
serverName = "127.0.0.1"
serverPort = 12000
clientSocket = socket(AF_INET, SOCK_DGRAM)
sentence = input("Enter file name")
clientSocket.sendto(bytes(sentence,"utf-8"),(serverName, serverPort))
filecontents,serverAddress = clientSocket.recvfrom(2048)
print ('From Server:', filecontents)
clientSocket.close()
```

UDPServer.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)
    file=open(sentence,"r")
    l=file.read(2048)
    serverSocket.sendto(bytes(l,"utf-8"),clientAddress)
    print("sent back to client",l)
file.close()
```


userver.py - C:\Users\User\Desktop\userver.py (3.9.1)
File Edit Format Run Options Window Help
from socket import *
serverPort = 12000
serverSocket = socket(AF_INET, SOCK_DGRAM)
IDLE Shell 3.9.1
File Edit Shell Debug Options Window Help
Python 3.9.1 (tags/v3.9.1:1e5d33e, Dec 7 2020, 16:33:24) [MSC v.1928 32 bit
tel)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
========== RESTART: C:\Users\User\Desktop\userver.py ==========
The server is ready to receive
sent back to client hello.... tyt
this ir try prg to try
the experiment
.....
byeeeeeeeee
uclient.py - C:\Users\User\Desktop\uclient.py (3.9.1)
File Edit Format Run Options Window Help
from socket import *
serverName = "127.0.0.1"
serverPort = 12000
clientSocket = socket(AF_INET, SOCK_DGRAM)
sentence = input("Enter file name")
IDLE Shell 3.9.1
File Edit Shell Debug Options Window Help
Python 3.9.1 (tags/v3.9.1:1e5d33e, Dec 7 2020, 16:33:24) [MSC v.1928 32 bit (In
tel)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
========== RESTART: C:\Users\User\Desktop\uclient.py ==========
Enter file nametry.txt
From Server: b'hello.... tyt\nthis ir try prg to try\nthe experiment\n.....
.....\n\nbyeeeeeeeee'
>>>