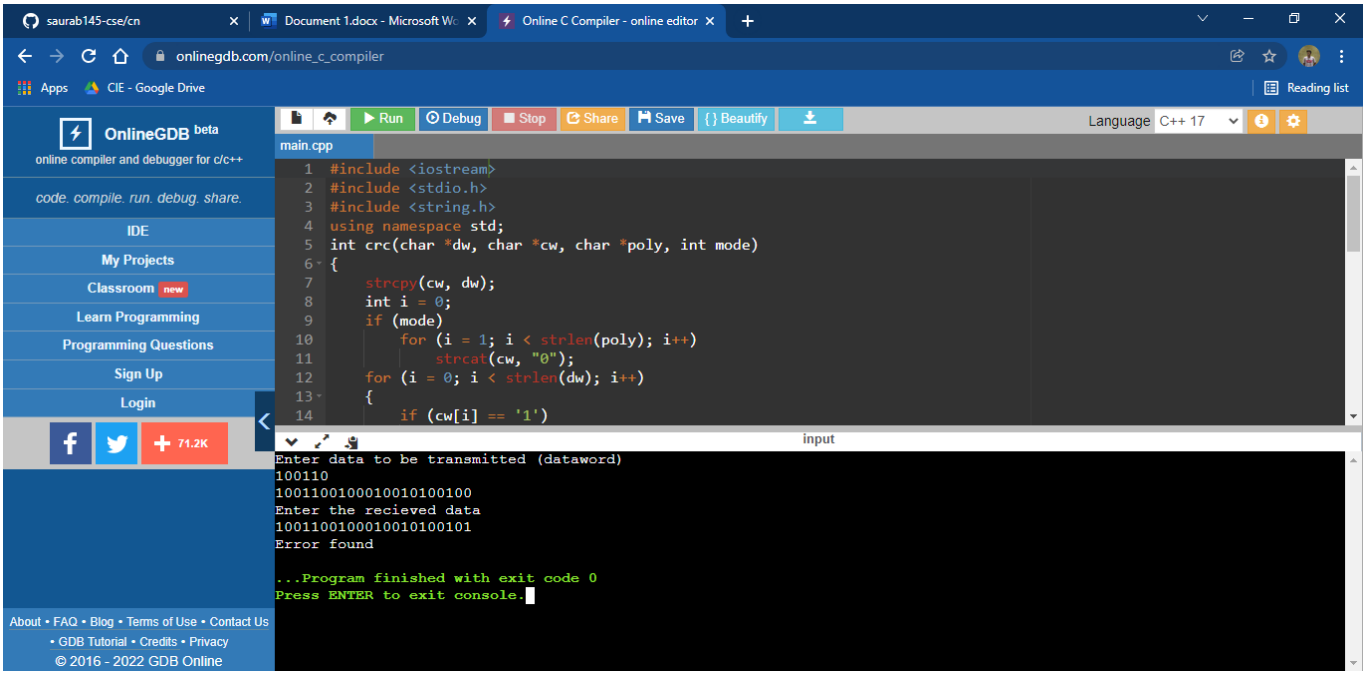
CN Lab Programs:-

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

```
#include <iostream>
#include <stdio.h>
#include <string.h>
using namespace std;
int crc(char *dw, char *cw, char *poly, int mode)
{
   strcpy(cw, dw);
   int i = 0;
   if (mode)
      for (i = 1; i < strlen(poly); i++)
         strcat(cw, "0");
   for (i = 0; i < strlen(dw); i++)
   {
      if (cw[i] == '1')
         for (int j = 0; j < strlen(poly); j++)
         {
            if (cw[i + j] == poly[j])
               cw[i + j] = '0';
            else
               cw[i + j] = '1';
         }
   }
   for (i = 0; i < strlen(cw); i++)
      if (cw[i] == '1')
         return 0;
```

```
    return 1;
}
int main()
{
    char dw[50], cw[100];
    char poly[] = "10001000000100001";
    cout << "Enter data to be transmitted (dataword)" << endl;
    cin >> dw;
    crc(dw, cw, poly, 1);
    cout << dw << cw + strlen(dw) << endl;
    cout << "Enter the recieved data" << endl;
    cin >> cw;
    if (crc(dw, cw, poly, 0))
        cout << "No error in tranmission" << endl;
    else
        cout << "Error found";
    return 0;
}
```

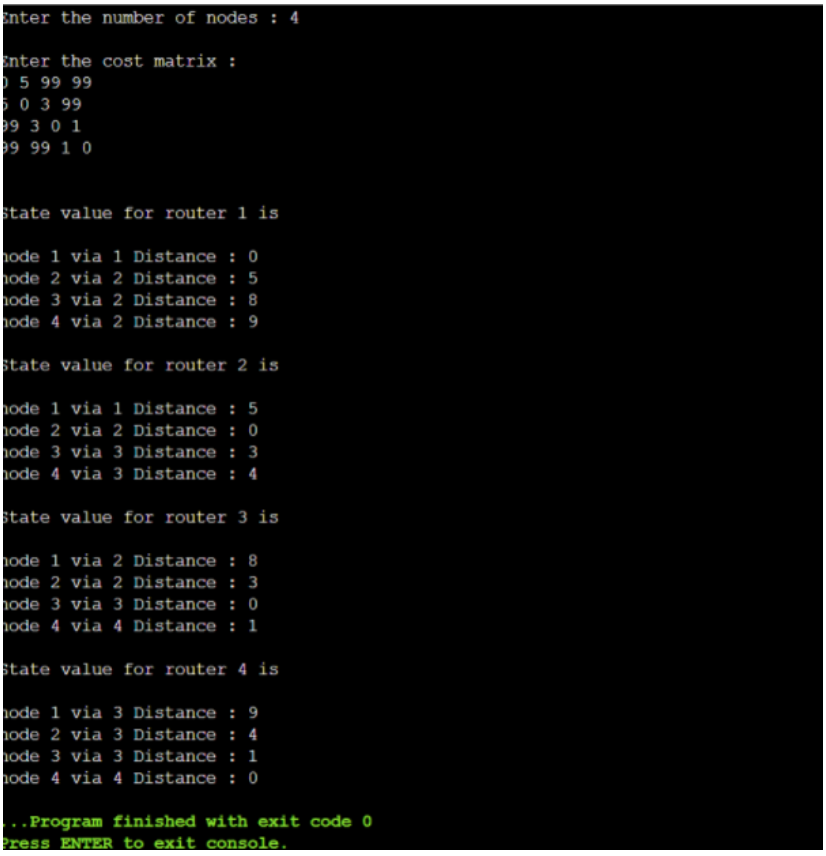


7) Write a program for distance vector algorithm to find suitable path for Transmission.

```
#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 : " << endl;
   cin >> no;
```

```
cout << "Enter adjacency matrix : " << endl;
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 << endl;
}
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;
}
```

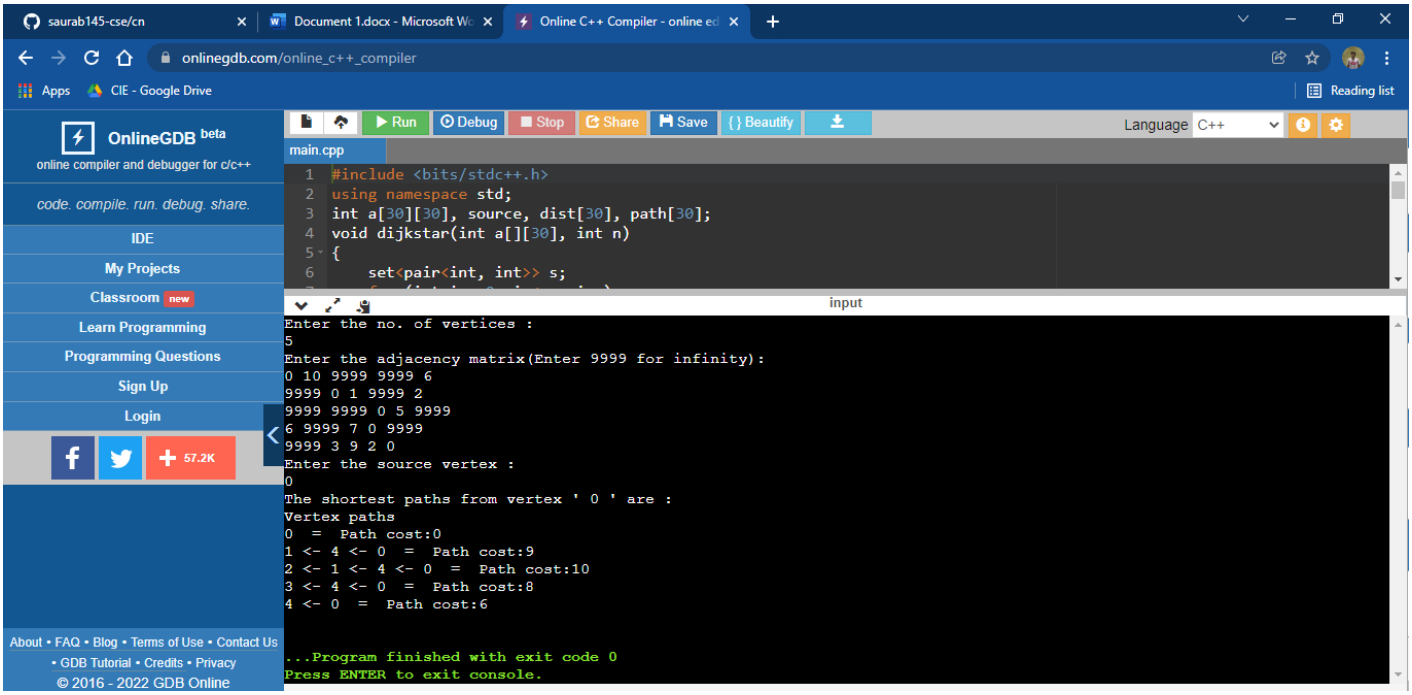


8) Implement Dijkstra's algorithm to compute the shortest path for a given topology.

```
#include <bits/stdc++.h>
using namespace std;
int a[30][30], source, dist[30], path[30];
```

```
void dijkstar(int a[][30], int n)
{
   set<pair<int, int>> s;
   for (int i = 0; i < n; i++)
   {
      dist[i] = a[source][i];
      path[i] = source;
      s.insert({dist[i], i});
   }
   while (!s.empty())
   {
      pair<int, int> t = *s.begin();
      s.erase(s.begin());
      for (int i = 0; i < n; i++)
      {
         if (dist[i] > t.first + a[t.second][i])
         {
            dist[i] = dist[t.second] + a[t.second][i];
            path[i] = t.second;
            s.insert({dist[i], i});
         }
      }
   }
}
int main()
{
   int n;
   cout << "Enter the no. of vertices :" << endl;
   cin >> n;
```

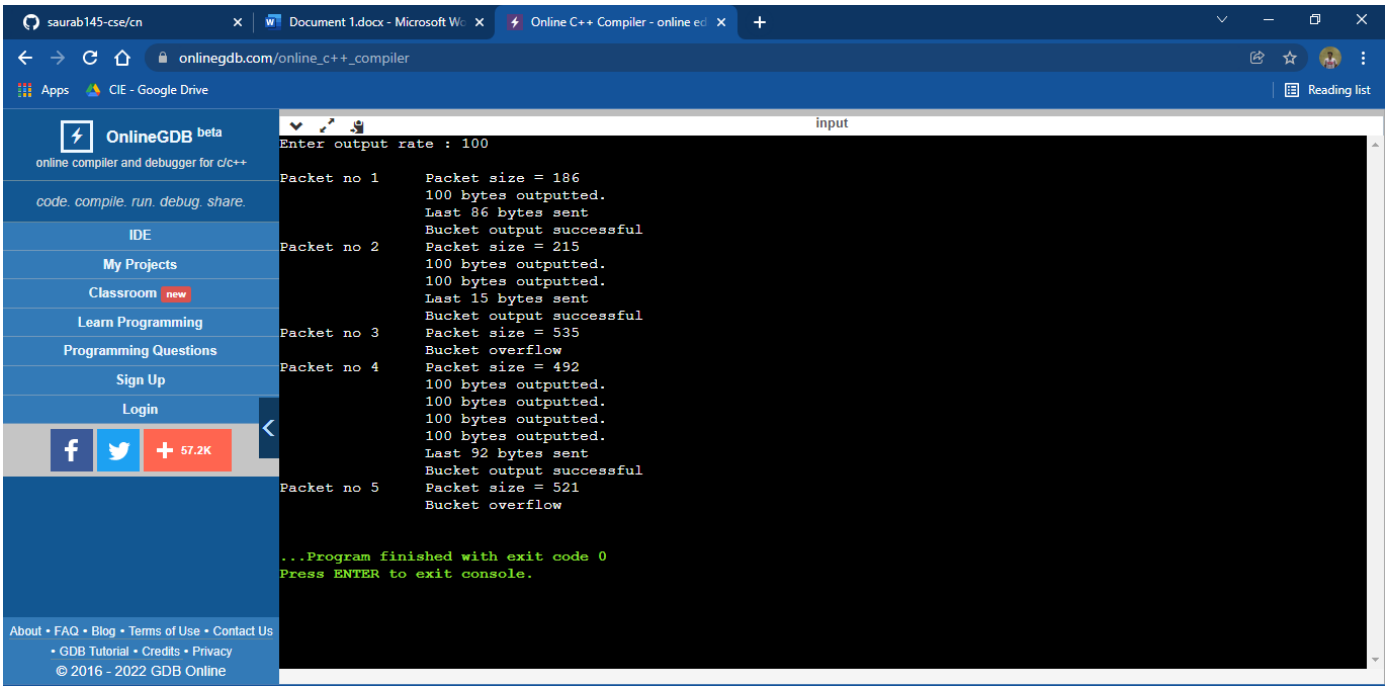
```
    cout << "Enter the adjacency matrix(Enter 9999 for infinity):" << endl;
    for (int i = 0; i < n; i++)
    {
        for (int j = 0; j < n; j++)
        {
            cin >> a[i][j];
        }
    }
    cout << "Enter the source vertex :" << endl;
    cin >> source;
    cout << "The shortest 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;
}
```



9) Write a program for congestion control using Leaky bucket algorithm.

```
#include <bits/stdc++.h>
#include <Windows.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;
}
```



10):- 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.

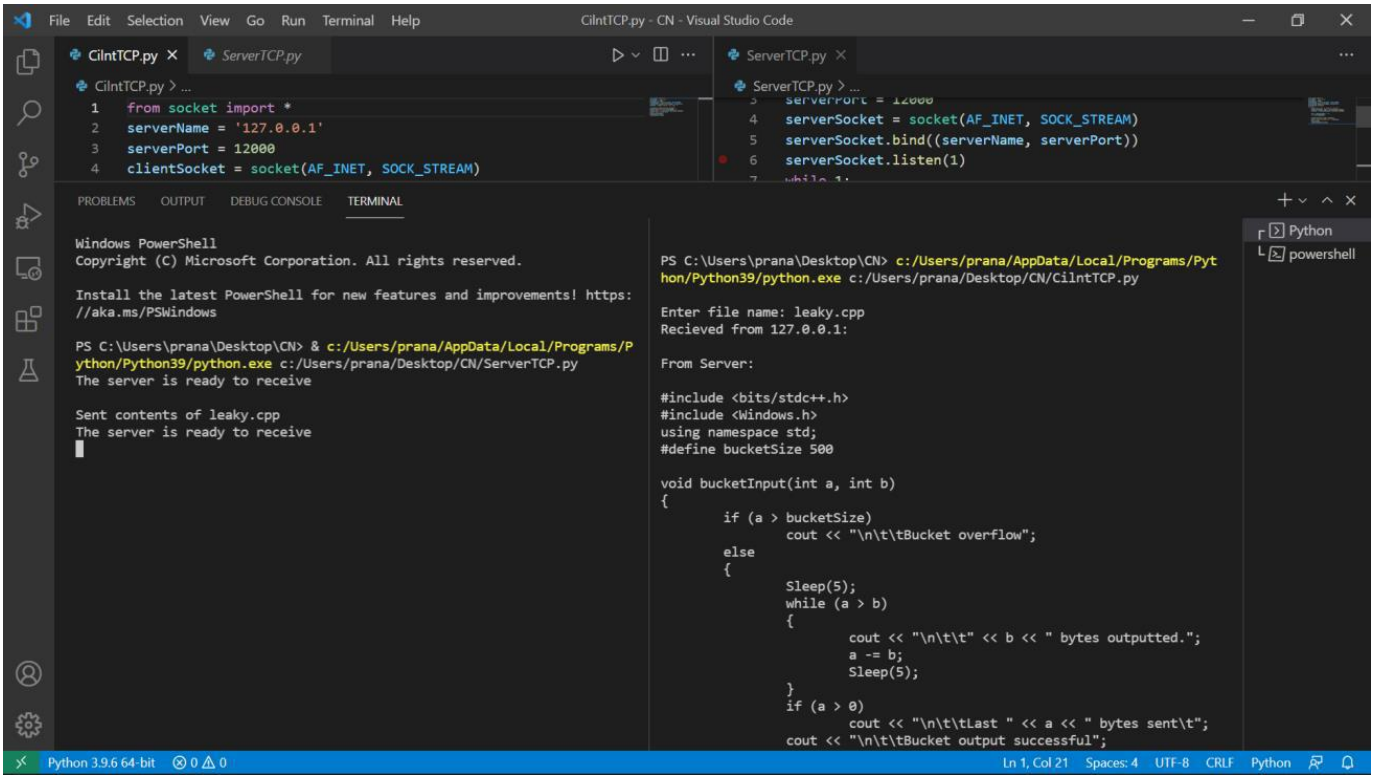
Server:

```
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:

```
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())
print(f"Recieved from {serverName}: ")
filecontents = clientSocket.recv(1024).decode()
print('\nFrom Server:\n')
print(filecontents)
clientSocket.close()
```



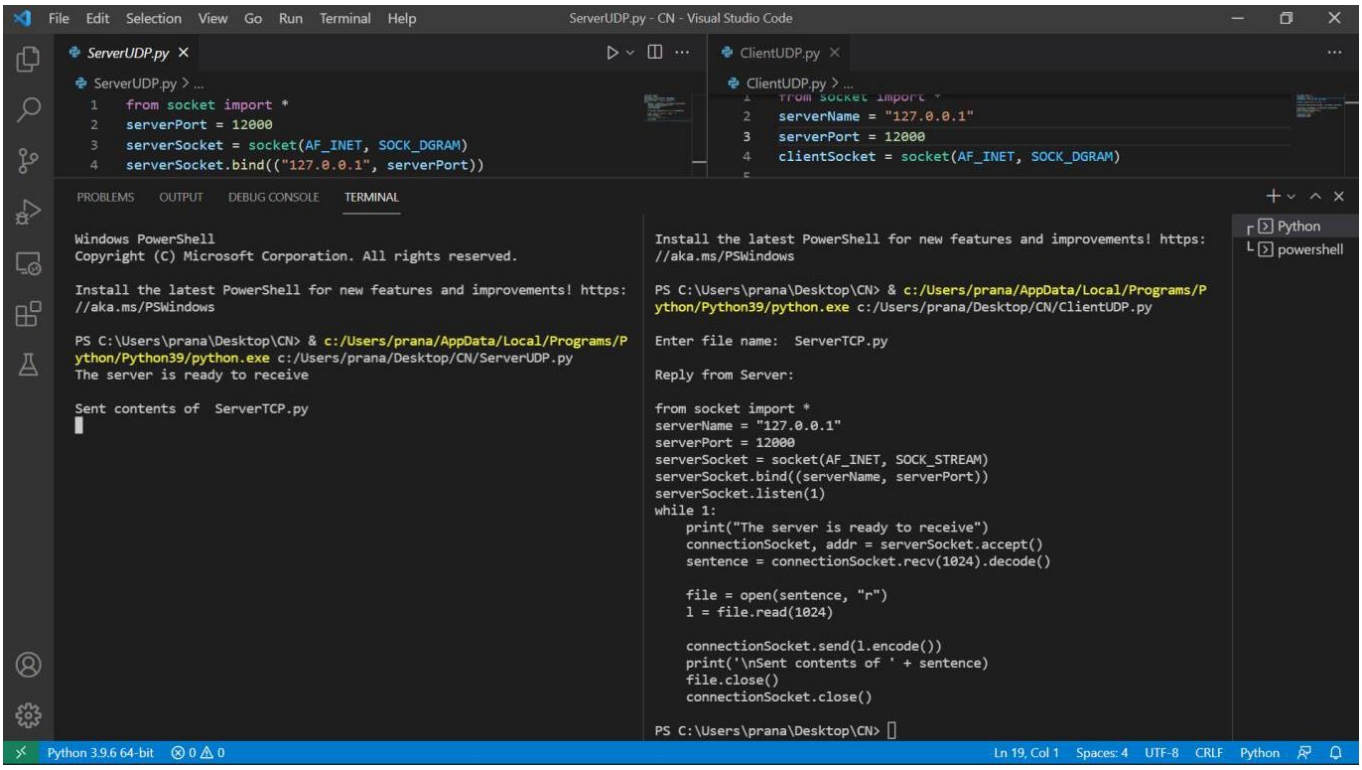
11) Q:- Using UDP sockets, write a client-server program to make client sending the file name and the server to send back the contents of the requested file if Present.

```
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()
```

Client:-

```
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()
```


File Edit Selection View Go Run Terminal Help
ServerUDP.py - CN - Visual Studio Code
ServerUDP.py
ClientUDP.py
from socket import *
serverPort = 12000
serverSocket = socket(AF_INET, SOCK_DGRAM)
serverSocket.bind(("127.0.0.1", serverPort))
serverName = "127.0.0.1"
serverPort = 12000
clientSocket = socket(AF_INET, SOCK_DGRAM)
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:\Users\prana\Desktop\CN> & c:/Users/prana/AppData/Local/Programs/Python/Python39/python.exe c:/Users/prana/Desktop/CN/ServerUDP.py
The server is ready to receive
Sent contents of ServerTCP.py
Install the latest PowerShell for new features and improvements! https://aka.ms/PSWindows
PS C:\Users\prana\Desktop\CN> & c:/Users/prana/AppData/Local/Programs/Python/Python39/python.exe c:/Users/prana/Desktop/CN/ClientUDP.py
Enter file name: ServerTCP.py
Reply 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()
PS C:\Users\prana\Desktop\CN>
Python
powershell
Python 3.9.6 64-bit
Ln 19, Col 1 Spaces: 4 UTF-8 CRLF Python