PYTHON EXPERIMENT 6

Name : Gini Chacko
Roll : 8942
Class : SE Comps B

Aim: To implement following programs in Python

1. WAP to copy a file from client to server using tcp socket

CODE:
A.] For Server

```
import socket

IP = socket.gethostbyname(socket.gethostname())
PORT = 4455
ADDR = (IP, PORT)
SIZE = 1024
FORMAT = "utf-8"

def main():
    print("[STARTING] Server is starting.")
    """ Staring a TCP socket. """
    server = socket.socket(socket.AF_INET, socket.SOCK_STREAM)

    """ Bind the IP and PORT to the server. """
    server.bind(ADDR)

    """ Server is listening, i.e., server is now waiting for the client to connec
    ted. """
    server.listen()
    print("[LISTENING] Server is listening.")

    while True:
        """ Server has accepted the connection from the client. """
        conn, addr = server.accept()
        print(f"[NEW CONNECTION] {addr} connected.")
```

```
        """ Receiving the filename from the client. """
        filename = conn.recv(SIZE).decode(FORMAT)
        print(f"[RECV] Receiving the filename.")
        file = open(filename, "w")
        conn.send("Filename received.".encode(FORMAT))

        """ Receiving the file data from the client. """
        data = conn.recv(SIZE).decode(FORMAT)
        print(f"[RECV] Receiving the file data.")
        file.write(data)
        conn.send("File data received".encode(FORMAT))

        """ Closing the file. """
        file.close()

        """ Closing the connection from the client. """
        conn.close()
        print(f"[DISCONNECTED] {addr} disconnected.")

if __name__ == "__main__":
    main()
```

B.] For Client

```
import socket

IP = socket.gethostbyname(socket.gethostname())
PORT = 4455
ADDR = (IP, PORT)
FORMAT = "utf-8"
SIZE = 1024

def main():
    """ Staring a TCP socket. """
    client = socket.socket(socket.AF_INET, socket.SOCK_STREAM)

    """ Connecting to the server. """
    client.connect(ADDR)

    """ Opening and reading the file data. """
    file = open("data/file_send.txt", "r")
    data = file.read()
```

```
    """ Sending the filename to the server. """
    client.send("file_recieved.txt".encode(FORMAT))
    msg = client.recv(SIZE).decode(FORMAT)
    print(f"[SERVER]: {msg}")

    """ Sending the file data to the server. """
    client.send(data.encode(FORMAT))
    msg = client.recv(SIZE).decode(FORMAT)
    print(f"[SERVER]: {msg}")

    """ Closing the file. """
    file.close()

    """ Closing the connection from the server. """
    client.close()


if __name__ == "__main__":
    main()
```

OUTPUT:

```
PS C:\GINI\ENGG\2nd Year\Sem 4\Python\LABS\EXP 6\File Transfer> & C:/ProgramData/Anaconda
3/envs/pip_env/python.exe "c:/GINI/ENGG/2nd Year/Sem 4/Python/LABS/EXP 6/File Transfer/se
rver.py"
[STARTING] Server is starting.
[LISTENING] Server is listening.
[NEW CONNECTION] ('192.168.56.1', 51901) connected.
[RECV] Receiving the filename.
[RECV] Receiving the file data.
[DISCONNECTED] ('192.168.56.1', 51901) disconnected.

PS C:\GINI\ENGG\2nd Year\Sem 4\Python\LABS\EXP 6\File Transfer>  & C:/ProgramData/Anacond
a3/envs/pip_env/python.exe "c:/GINI/ENGG/2nd Year/Sem 4/Python/LABS/EXP 6/File Transfer/c
lient.py"
[SERVER]: Filename received.
[SERVER]: File data received
```

file_recieved.txt

```
1   Hello, This is Gini Chacko.
2
3   Poem...
4
5   But soft what light through yonder window breaks
6   It is the east and Juliet is the sun
7   Arise fair sun and kill the envious moon
8   Who is already sick and pale with grief
```

2. WAP to develop chat application

CODE:

A.] For Server

```
import socket
def server_program():
    # get the hostname
    host = socket.gethostname()
    print(host)
    port = 5000  # initiate port no above 1024

    server_socket = socket.socket()  # get instance
    # look closely. The bind() function takes tuple as argument
    server_socket.bind((host, port))  # bind host address and port together

    # configure how many client the server can listen simultaneously
    server_socket.listen(2)
    conn, address = server_socket.accept()  # accept new connection
    print("Connection from: " + str(address))

    while True:
        # receive data stream. it won't accept data packet greater than 1024 byte
    s
        data = conn.recv(1024).decode()
        if not data:
            # if data is not received break
            break
        print("from connected user: " + str(data))
        data = input(' -> ')
        conn.send(data.encode())  # send data to the client

    conn.close()  # close the connection


if __name__ == '__main__':
    server_program()
```

B.] For Client

```python
import socket

def client_program():
    host = socket.gethostname()  # as both code is running on same pc
    port = 5000  # socket server port number

    client_socket = socket.socket()  # instantiate
    client_socket.connect((host, port))  # connect to the server

    message = input(" -> ")  # take input

    while message.lower().strip() != 'bye':
        client_socket.send(message.encode())  # send message
        data = client_socket.recv(1024).decode()  # receive response

        print('Received from server: ' + data)  # show in terminal

        message = input(" -> ")  # again take input

    client_socket.close()  # close the connection


if __name__ == '__main__':
    client_program()
```

OUTPUT:

```
PS C:\Users\Chacko> & python "c:/GINI/ENGG/2nd Year/Sem 4/Python/LABS/EXP 6/server_chat.p
y"
PM-CHACKO
Connection from: ('192.168.56.1', 51831)
from connected user: Hi
 -> How are you?
from connected user: I am fine. What's your name?
 -> I am Gini Chacko
from connected user: Ok. Take care
 -> You too
PS C:\Users\Chacko> & python "c:/GINI/ENGG/2nd Year/Sem 4/Python/LABS/EXP 6/chat_client.p
y"
 -> Hi
Received from server: How are you?
 -> I am fine. What's your name?
Received from server: I am Gini Chacko
 -> Ok. Take care
Received from server: You too
 -> Bye
```

POSTLAB QUESTIONS:

1) Differentiate between Connection oriented and connection less service.

Ans:

S.NO	Connection-oriented Service	Connection-less Service
1.	Connection-oriented service is related to the telephone system.	Connection-less service is related to the postal system.
2.	Connection-oriented service is preferred by long and steady communication.	Connection-less Service is preferred by bursty communication.
3.	Connection-oriented Service is necessary.	Connection-less Service is not compulsory.
4.	Connection-oriented Service is feasible.	Connection-less Service is not feasible.
5.	In connection-oriented Service, Congestion is not possible.	In connection-less Service, Congestion is possible.
6.	Connection-oriented Service gives the guarantee of reliability.	Connection-less Service does not give the guarantee of reliability.
7.	In connection-oriented Service, Packets follow the same route.	In connection-less Service, Packets do not follow the same route.
8.	Connection-oriented Services requires a bandwidth of high range.	Connection-less Service requires a bandwidth of low range.

2) Write a program to demonstrate how to set UDP connection?
Ans:

UDP Server using Python

CODE:

```
import socket
localIP     = "127.0.0.1"
localPort   = 20001
bufferSize  = 1024

msgFromServer       = "Hello UDP Client"
bytesToSend         = str.encode(msgFromServer)

# Create a datagram socket
UDPServerSocket =
     socket.socket(family=socket.AF_INET,type=socket.SOCK_DGRAM)

# Bind to address and ip
UDPServerSocket.bind((localIP, localPort))

print("UDP server up and listening")

# Listen for incoming datagrams
while(True):
   bytesAddressPair = UDPServerSocket.recvfrom(bufferSize)
   message = bytesAddressPair[0]
   address = bytesAddressPair[1]
   clientMsg = "Message from Client:{}".format(message)
   clientIP  = "Client IP Address:{}".format(address)
   print(clientMsg)
   print(clientIP)

   # Sending a reply to client
   UDPServerSocket.sendto(bytesToSend, address)
```

OUTPUT:

UDP server up and listening

Message from Client:b"Hello UDP Server"

Client IP Address:("127.0.0.1", 51696)

UDP Client using Python

CODE:

```
import socket
msgFromClient       = "Hello UDP Server"
bytesToSend         = str.encode(msgFromClient)
serverAddressPort   = ("127.0.0.1", 20001)
bufferSize          = 1024

# Create a UDP socket at client side
UDPClientSocket =
    socket.socket(family=socket.AF_INET,type=socket.SOCK_DGRAM)

# Send to server using created UDP socket
UDPClientSocket.sendto(bytesToSend, serverAddressPort)
msgFromServer = UDPClientSocket.recvfrom(bufferSize)
msg = "Message from Server {}".format(msgFromServer[0])
print(msg)
```

OUTPUT:

Message from Server b"Hello UDP Client"