26/10/24

PRACTICAL - 02 (CO)

AIM:

Implement echo client server using TCP/UDP sockets.

CLIENT:

```
import socket
import time
def pingServer(host='127.0.0.1', port=12345)
    with socket.socket(socket.AF_INET, socket.SOCK_DGRAM) as s:
        try:
            s.sendto(b'Hello', (host, port))
            print("Message sent to server")
        except socket.timeout:
            print("Request timed out");
if __name__ == "__main__":
    ping_server()
```

SERVER:

```
import socket
def start_server(host='127.0.0.1', port=12345):
    with socket.socket(socket.AF_INET, socket.SOCK_DGRAM) as s:
        s.bind((host, port))
        print(f"UDP server running on {host}:{port}")
        while True:
            data, addr = s.recvfrom(1024)
            print(f'Received message from {addr}: {data.decode()}")
if __name__ == "__main__":
    start_server()
```

26/10/24

OUTPUT:

python receiver.py
UDP server running on 127.0.0.1 : 12345
Received message from (127.0.0.1. 59250)
Python client.py
Received reply from server : Hello client

RESULT:

Thus program to implement echo client server using TCP / UDP sockets is successfully executed.

26/10/24

PRACTICAL - 12 (b)

AIM:

Implement chat client server using TCP / UDP sockets.

CHAT server.py:

```
import socket
def receiver():
    port = 12345
    host = '127.0.0.1'
    with socket.socket('socket.AF_INET, socket_SOCK_DGRAM) as s:
        s.bind ((host, port))
while (True):
    d, add = s.recvfrom ('1024')
    print ("clien")
    a = input (" enter Reply")
    s.sendto (a.encoded.add)
    if (a == 'end'):
        break
        exit
    rev()
```

Receiver.py:

```
import socket
import time
def receiver(a):
    host = '127.0.0.1'
    while (True):
        a = input ("enter Message")
        if (a == 'end'):
            receiver (0)
            break
        else :
            receiver (a)
```

using

26/10/24

OUTPUT:

python server.py

client {'hi'}

client {'How are you'}

Enter reply I'm fine

Python receiver.py

Enter message: Hi

{'Hello'}

Enter message: How are you

{'I am fine'}

RESULT:

Thus the client server using TCP/UDP is executed and output is verified successfully.