EXNO: 12
DATE: 18/10/24

Practical - 12A:

AIM:-

a] Implement echo client server TCP/UDP sockets.

ALGORITHM:-

TCP Echo client server Algorithm (short):-

Server(TCP):
1. create and bind a TCP socket to host and port
2. Listen for connections and accept a client
3. Receive data from the client
4. Send the same data back (echo).
5. close the client connection and continue.

client(TCP):
1. create a TCP socket and connect to server
2. Send data to the server.
3. Receive echoed data from the server
4. close the socket.

client:-

```
import socket
import time
def ping_server(host='127.0.0.1', port=12345):
    with socket.socket(socket.AF_INET, socket.
                       SOCK_DIAGRAM) as s:
        try:
            s.sendto(b'Hello', (host, port))
        except s.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_DIAGRAM) 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().
```

OUTPUT.

```
Python server.py
UDP server running on 127.0.0.1:12345
Received message from ('127.0.0.1', 53009): Ping

Python client.py
Received Ping from ('127.0.0.1', 12345). in
0.0 seconds.
```

RESULT:

Thus echo client server using TCP/UDP socket is implemented and executed successfully.