

## **EXPERIMENT – 12**

**AIM:** - a) Implement echo client server using TCP/UDP sockets.

**CODE:** -

```
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}")

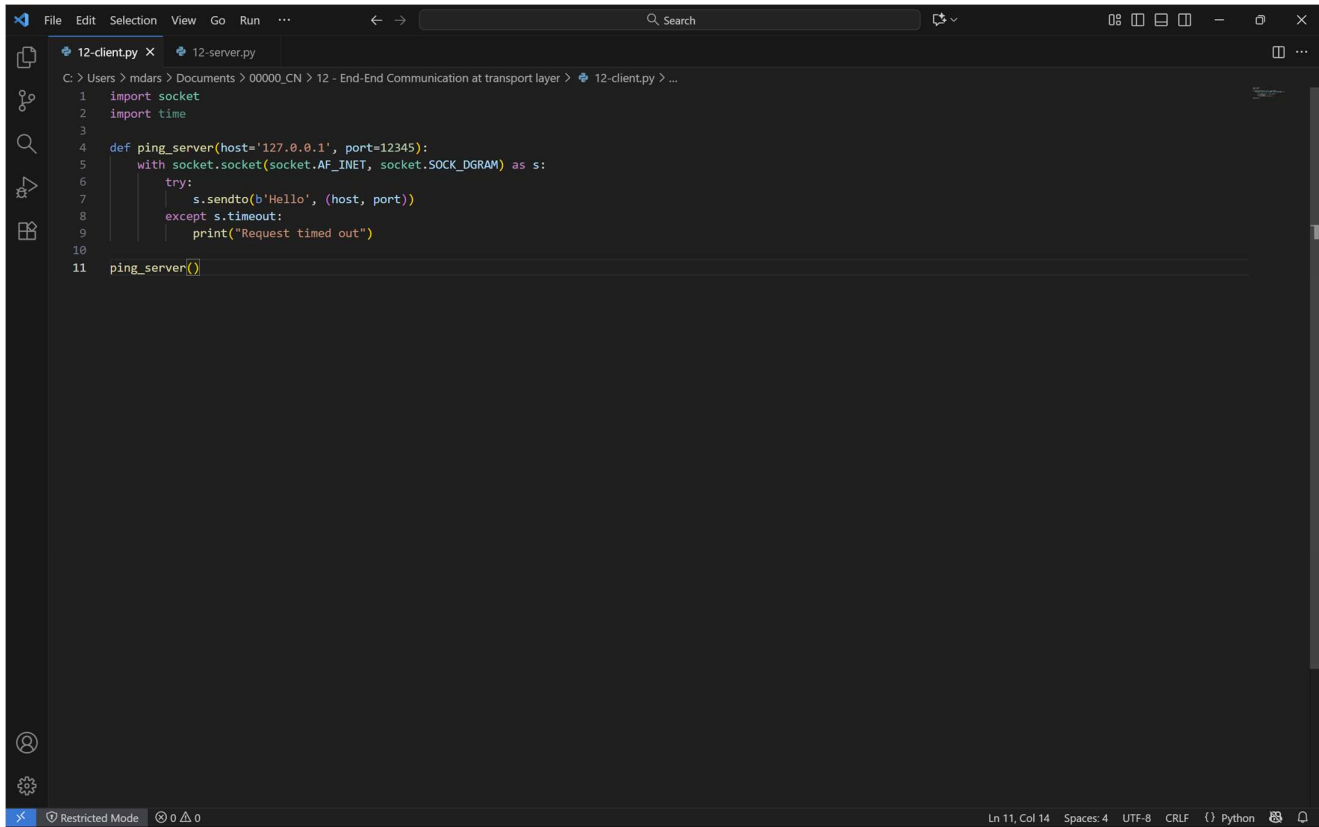
        while True:
            data, addr = s.recvfrom(1024)
            print(f"Received message from {addr}: {data.decode()}")

import socket
import time

def ping_server(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))
        except s.timeout:
            print("Request timed out")

ping_server()
```

## OUTPUT: -



The image shows a screenshot of a Visual Studio Code editor window. The editor has a dark theme and displays a Python file named '12-client.py'. The code is as follows:

```
1 import socket
2 import time
3
4 def ping_server(host='127.0.0.1', port=12345):
5     with socket.socket(socket.AF_INET, socket.SOCK_DGRAM) as s:
6         try:
7             s.sendto(b'Hello', (host, port))
8         except s.timeout:
9             print("Request timed out")
10
11 ping_server()
```

The editor's interface includes a menu bar at the top with 'File', 'Edit', 'Selection', 'View', 'Go', 'Run', and '...'. A search bar is located to the right of the menu. On the left side, there is a sidebar with icons for Explorer, Search, Run and Debug, and Extensions. The status bar at the bottom indicates 'Restricted Mode', '0 0 0', and 'Ln 11, Col 14 Spaces: 4 UTF-8 CRLF Python'.

## RESULT: -

Implement echo client server using TCP/UDP sockets.

**AIM:** - b) Implement chat client server using TCP/UDP sockets.

**CODE:** -

```
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}")

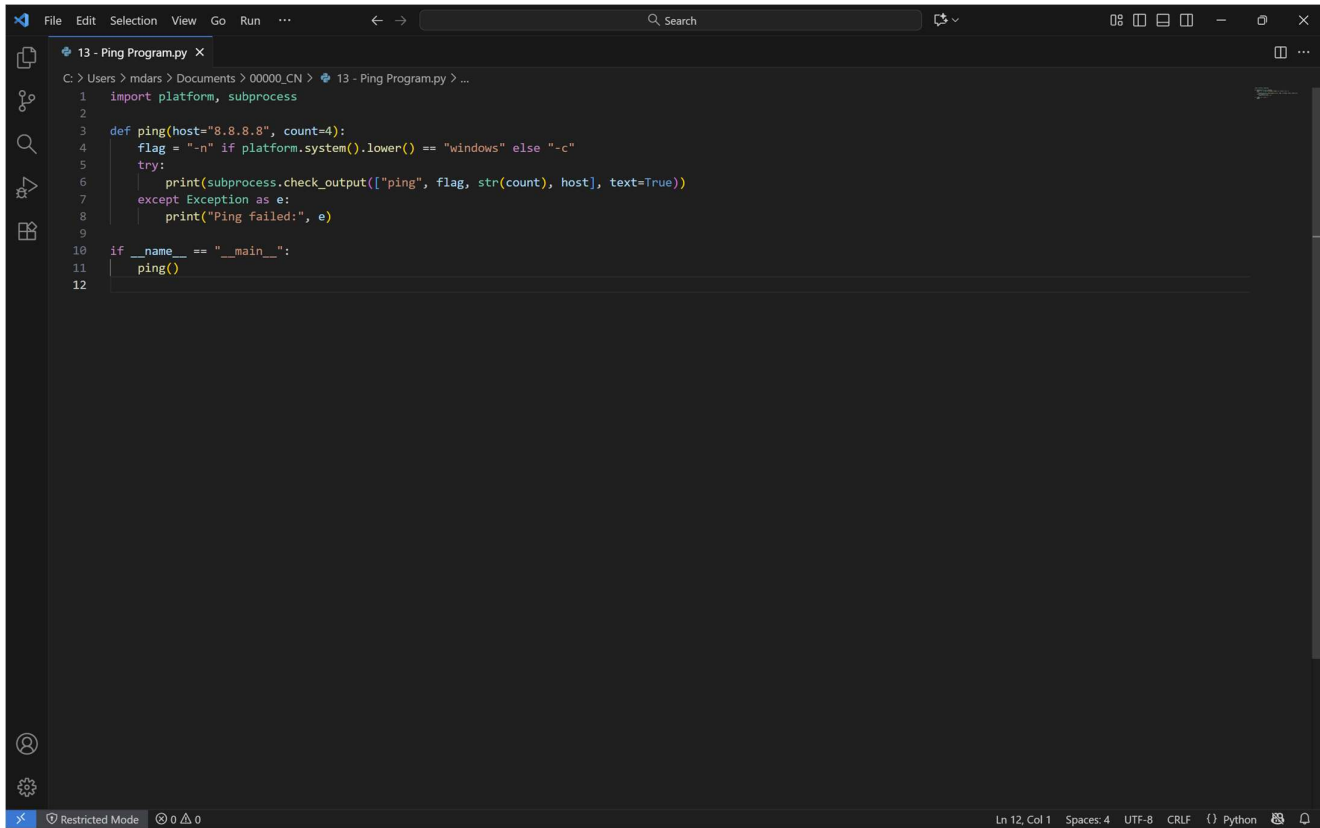
        while True:
            data, addr = s.recvfrom(1024)
            print(f"Received message from {addr}: {data.decode()}")

import socket
import time

def ping_server(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))
        except s.timeout:
            print("Request timed out")

ping_server()
```

## OUTPUT: -



The screenshot shows a Python IDE window titled "13 - Ping Program.py". The code is as follows:

```
1 import platform, subprocess
2
3 def ping(host="8.8.8.8", count=4):
4     flag = "-n" if platform.system().lower() == "windows" else "-c"
5     try:
6         print(subprocess.check_output(["ping", flag, str(count), host], text=True))
7     except Exception as e:
8         print("Ping failed:", e)
9
10 if __name__ == "__main__":
11     ping()
12
```

The IDE interface includes a menu bar (File, Edit, Selection, View, Go, Run, ...), a search bar, and a sidebar with icons for Explorer, Search, Run and Debug, and Source Control. The status bar at the bottom indicates "Ln 12, Col 1", "Spaces: 4", "UTF-8", "CRLF", and "Python".

## RESULT: -

Implement chat client server using TCP/UDP sockets.

