Assignment 9

Implementation of TCP/UDP Socket Programming

NAME: Shirish Manoj Bobde

Reg. No.: 812

Roll No.: ECE/21152

Problem Statement 1

Write a TCP/UDP socket program (in C/C++/Java/Python) to establish a connection between client and server. The server should act as a network device maintaining an ARP table. Implement ARP request and reply functionality.  
Display appropriate messages indicating the ARP request and response. Test your program with multiple clients requesting ARP resolution for different IP addresses.

Codes:

**Server:**

import socket

SERVER\_IP = '127.0.0.1'

SERVER\_PORT = 12345

server\_socket = socket.socket(socket.AF\_INET, socket.SOCK\_STREAM)

server\_socket.bind((SERVER\_IP, SERVER\_PORT))

server\_socket.listen(5)

print(f"Server listening on {SERVER\_IP}:{SERVER\_PORT}")

arp\_table = {}

def handle\_arp\_request(ip\_address):

    if ip\_address in arp\_table:

        return arp\_table[ip\_address]

    else:

        return "IP address not found in ARP table"

def handle\_client\_connection(client\_socket):

    while True:

        data = client\_socket.recv(1024).decode()

        if not data:

            break

        if data.startswith("ARP\_REQUEST"):

            ip\_address = data.split()[1]

            mac\_address = handle\_arp\_request(ip\_address)

            response = f"ARP\_RESPONSE {mac\_address}"

            client\_socket.send(response.encode())

    client\_socket.close()

while True:

    client\_socket, client\_address = server\_socket.accept()

    print(f"Connection from {client\_address[0]}:{client\_address[1]}")

    client\_handler = threading.Thread(target=handle\_client\_connection, args=(client\_socket,))

    client\_handler.start()

**Client:**

import socket

def main():

    server\_host = "127.0.0.1"

    server\_port = 8888

    client\_socket = socket.socket(socket.AF\_INET, socket.SOCK\_STREAM)

    client\_socket.connect((server\_host, server\_port))

    while True:

        ip\_address = input("Enter IP address (type 'exit' to quit): ")

        if ip\_address.lower() == 'exit':

            break

        client\_socket.send(ip\_address.encode())

        mac\_address = client\_socket.recv(1024).decode()

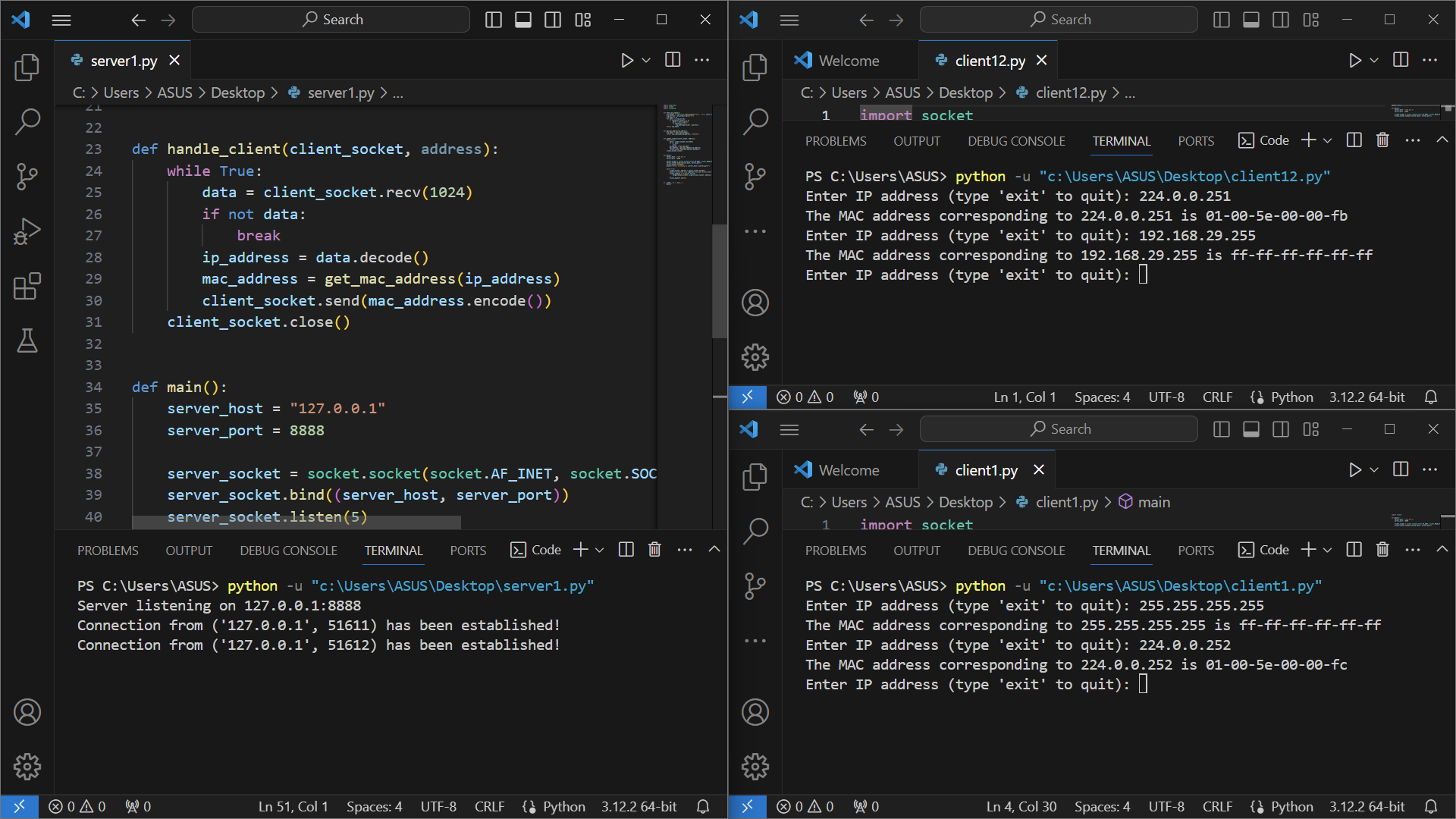
        print(f"The MAC address corresponding to {ip\_address} is {mac\_address}")

    client\_socket.close()

if \_\_name\_\_ == "\_\_main\_\_":

    main()

Output:



Problem Statement 2

Write a TCP/UDP socket program (in C/C++/Java/Python) to establish a connection between client and server. The server should act as a network device maintaining a RARP table mapping MAC addresses to IP addresses. Implement RARP request and reply functionality. Display appropriate messages indicating the RARP request and response. Test your program with multiple clients requesting RARP resolution for different MAC addresses.

Code:

**Server**

import subprocess

import socket

import threading

def fetch\_arp\_table():

    arp\_output = subprocess.check\_output(["arp", "-a"], text=True)

    arp\_entries = arp\_output.split("\n")

    arp\_table = {}

    for entry in arp\_entries:

        if len(entry.strip()) > 0:

            parts = entry.split()

            if len(parts) == 3:

                arp\_table[parts[1]] = parts[0]

    return arp\_table

def get\_ip\_address(mac\_address):

    arp\_table = fetch\_arp\_table()

    return arp\_table.get(mac\_address, "Unknown")

def handle\_client(client\_socket, address):

    while True:

        data = client\_socket.recv(1024)

        if not data:

            break

        mac\_address = data.decode()

        ip\_address = get\_ip\_address(mac\_address)

        client\_socket.send(ip\_address.encode())

    client\_socket.close()

def main():

    server\_host = "127.0.0.1"

    server\_port = 8889

    server\_socket = socket.socket(socket.AF\_INET, socket.SOCK\_STREAM)

    server\_socket.bind((server\_host, server\_port))

    server\_socket.listen(5)

    print(f"Server listening on {server\_host}:{server\_port}")

    while True:

        client\_socket, address = server\_socket.accept()

        print(f"Connection from {address} has been established!")

        client\_handler = threading.Thread(

            target=handle\_client, args=(client\_socket, address)

        )

        client\_handler.start()

if \_\_name\_\_ == "\_\_main\_\_":

    main()

**Client**

import socket

def main():

    server\_host = "127.0.0.1"

    server\_port = 8889

    client\_socket = socket.socket(socket.AF\_INET, socket.SOCK\_STREAM)

    client\_socket.connect((server\_host, server\_port))

    while True:

        mac\_address = input("Enter MAC address (type 'exit' to quit): ")

        if mac\_address.lower() == "exit":

            break

        client\_socket.send(mac\_address.encode())

        ip\_address = client\_socket.recv(1024).decode()

        print(f"The IP address corresponding to {mac\_address} is {ip\_address}")

    client\_socket.close()

if \_\_name\_\_ == "\_\_main\_\_":

    main()

Output:

