

Distributed Systems

Lab 4

Name – Khushi Israni

Registration no. - 210905156

Sem – VI

Section – A

Roll No. - 25

Ex1)

The screenshot shows the Visual Studio Code editor with the 'EXPLORER' sidebar on the left. The 'EXAMPLES' folder is expanded, showing 'ex1client.py' and 'ex1server.py'. The 'ex1server.py' file is open in the editor, displaying the following Python code:

```
1 import socket
2 host = socket.gethostname()
3 port = 12345
4 s = socket.socket()
5 s.bind((host,port))
6 s.listen(5)
7 conn, addr = s.accept()
8 print("Got connection from", addr[0], "(", addr[1], ")")
9 print("Thank you for connecting")
10 while True:
11     data = conn.recv(1024)
12     if not data: break
13     conn.sendall(data)
14 conn.close()
```

The 'TERMINAL' panel at the bottom shows the execution of the server script:

```
khushiisrani@Khushis-MacBook-Air examples % python3 ex1server.py
Got connection from 127.0.0.1 ( 54536 )
Thank you for connecting
khushiisrani@Khushis-MacBook-Air examples %
```

The status bar at the bottom indicates the file is at 'Ln 14, Col 13' with 'Spaces: 4', 'UTF-8' encoding, and 'LF' line endings. The Python version is '3.11.4 64-bit'.

The screenshot shows the Visual Studio Code editor with the 'EXPLORER' sidebar on the left. The 'EXAMPLES' folder is expanded, showing 'ex1client.py' and 'ex1server.py'. The 'ex1client.py' file is open in the editor, displaying the following Python code:

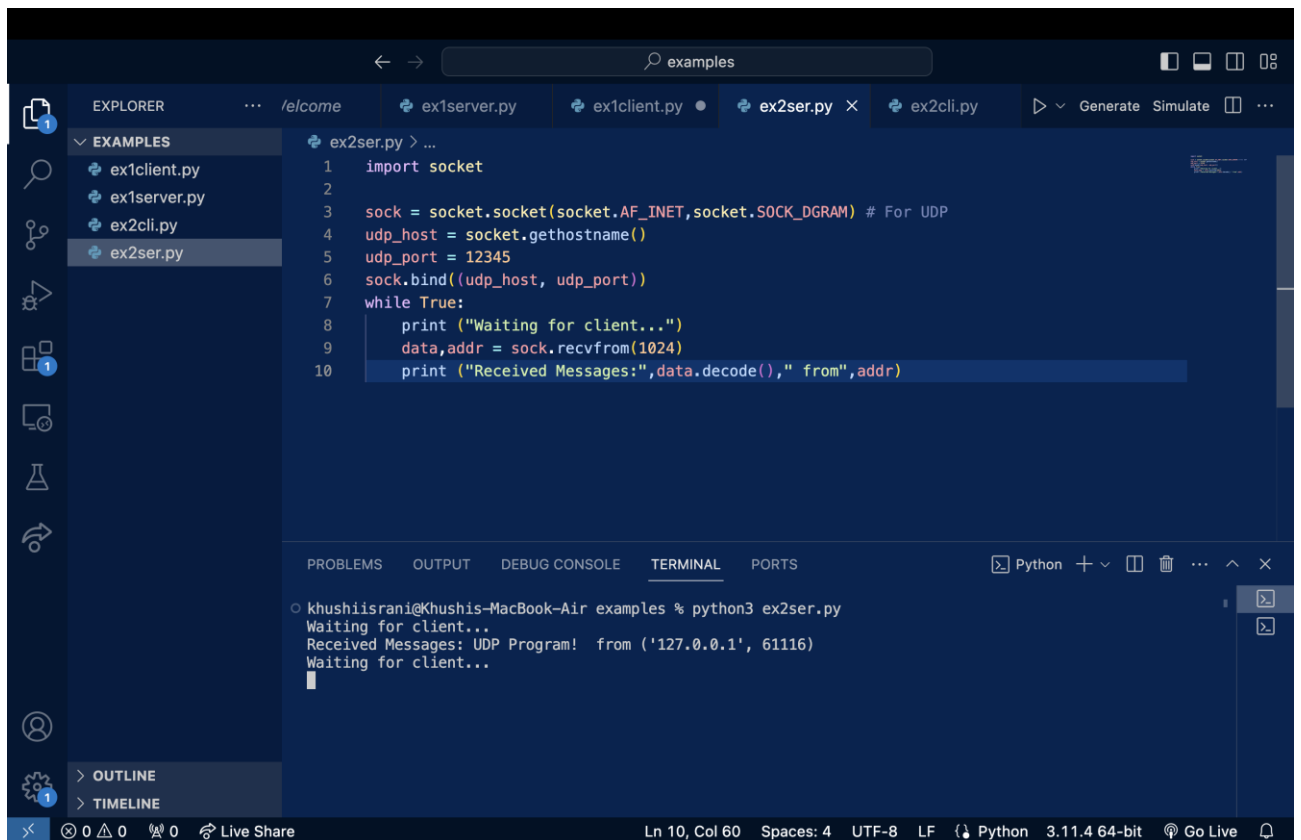
```
1 import socket
2 host = socket.gethostname()
3 port = 12345
4 s = socket.socket()
5 s.connect((host,port))
6 s.sendall(b"Welcome user!")
7 data = s.recv(1024)
8 s.close()
9 print(repr(data))
```

The 'TERMINAL' panel at the bottom shows the execution of the client script:

```
khushiisrani@Khushis-MacBook-Air examples % python3 ex1client.py
b'Welcome user!'
khushiisrani@Khushis-MacBook-Air examples %
```

The status bar at the bottom indicates the file is at 'Ln 9, Col 18' with 'Spaces: 4', 'UTF-8' encoding, and 'LF' line endings. The Python version is '3.11.4 64-bit'.

Ex2)



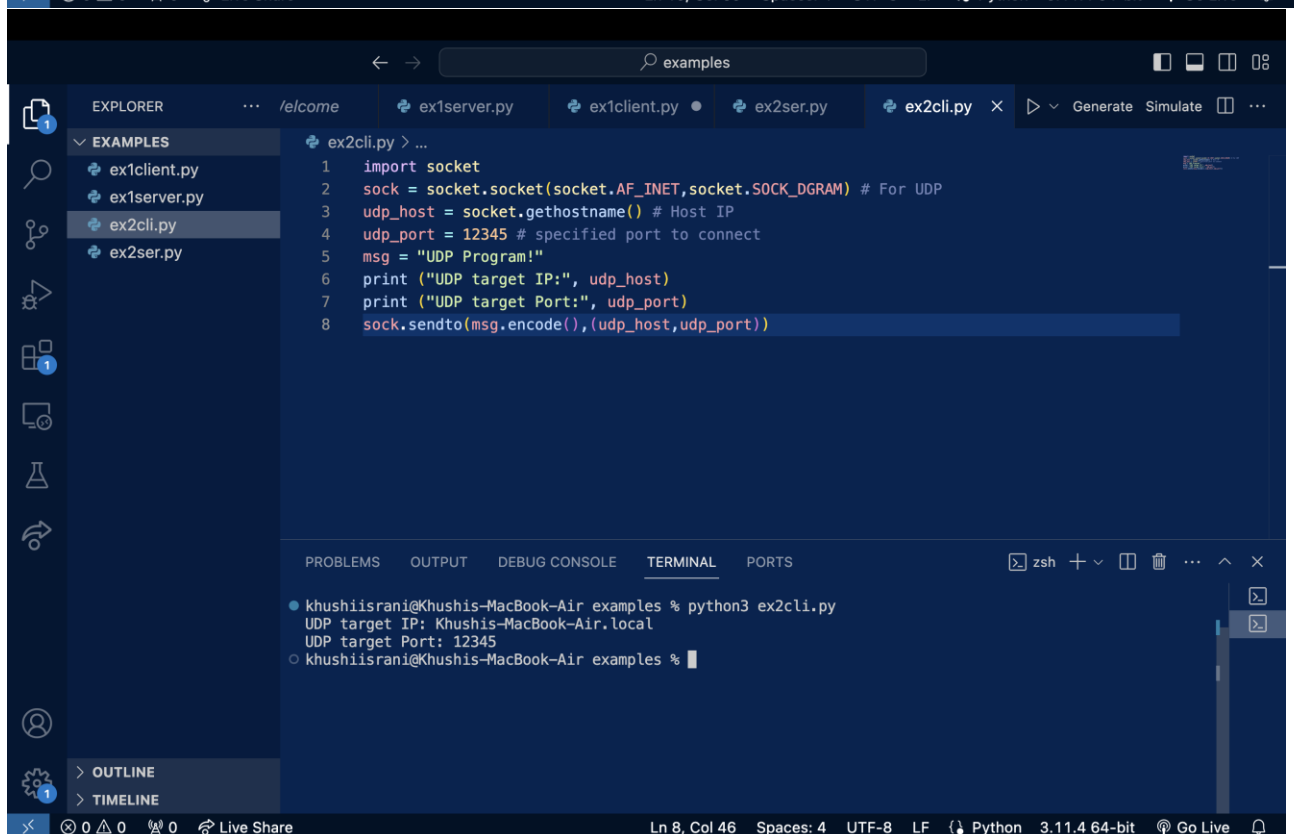
The screenshot shows the Visual Studio Code editor with the file explorer on the left displaying a folder named 'examples' containing files: ex1client.py, ex1server.py, ex2cli.py, and ex2ser.py. The main editor window shows the code for ex2ser.py:

```
1 import socket
2
3 sock = socket.socket(socket.AF_INET, socket.SOCK_DGRAM) # For UDP
4 udp_host = socket.gethostname()
5 udp_port = 12345
6 sock.bind((udp_host, udp_port))
7 while True:
8     print ("Waiting for client...")
9     data, addr = sock.recvfrom(1024)
10    print ("Received Messages:", data.decode(), " from", addr)
```

The terminal at the bottom shows the command `python3 ex2ser.py` being executed, with the following output:

```
Waiting for client...
Received Messages: UDP Program! from ('127.0.0.1', 61116)
Waiting for client...
```

The status bar at the bottom indicates the file is at line 10, column 60, with 4 spaces, UTF-8 encoding, and LF line endings. The Python version is 3.11.4 64-bit.



The screenshot shows the Visual Studio Code editor with the file explorer on the left displaying the same 'examples' folder. The main editor window shows the code for ex2cli.py:

```
1 import socket
2 sock = socket.socket(socket.AF_INET, socket.SOCK_DGRAM) # For UDP
3 udp_host = socket.gethostname() # Host IP
4 udp_port = 12345 # specified port to connect
5 msg = "UDP Program!"
6 print ("UDP target IP:", udp_host)
7 print ("UDP target Port:", udp_port)
8 sock.sendto(msg.encode(), (udp_host, udp_port))
```

The terminal at the bottom shows the command `python3 ex2cli.py` being executed, with the following output:

```
UDP target IP: Khushis-MacBook-Air.local
UDP target Port: 12345
```

The status bar at the bottom indicates the file is at line 8, column 46, with 4 spaces, UTF-8 encoding, and LF line endings. The Python version is 3.11.4 64-bit.

Ex3)

The screenshot shows the Visual Studio Code editor with the file explorer on the left displaying a folder named 'EXAMPLES' containing several Python files: ex1client.py, ex1server.py, ex2cli.py, ex2ser.py, ex3cli.py, and ex3ser.py. The 'ex3ser.py' file is selected and open in the main editor. The code in ex3ser.py is a Python server script that listens on port 2053 for incoming connections. It prints the client's address and enters a loop where it receives data, prints it, prompts the user to enter a message, and sends it back to the client. The terminal at the bottom shows the command 'python3 ex3ser.py' being executed, with output indicating a connection from '127.0.0.1' and the exchange of 'Hello, world' and 'Hiii'.

```
1 import socket
2 HOST = '127.0.0.1' # Standard loopback interface address (localhost)
3 PORT = 2053 # Port to listen on (non-privileged ports are > 1023)
4 with socket.socket(socket.AF_INET, socket.SOCK_STREAM) as s:
5     s.bind((HOST, PORT))
6     s.listen()
7     conn, addr = s.accept()
8     with conn:
9         print('Connected by', addr)
10        while True:
11            data = conn.recv(1024)
12            if data:
13                print("Client: ",data.decode())
14                data = input("Enter message to client:");
15                if not data:
16                    break;
17                # sending message as bytes to client.
18                conn.sendall(bytearray(data, 'utf-8'));
19 conn.close()
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

Python + Python 3.11.4 64-bit Go Live

khushiisrani@Khushis-MacBook-Air examples % python3 ex3ser.py
Connected by ('127.0.0.1', 55452)
Client: Hello, world
Enter message to client:Hiii
Enter message to client:

The screenshot shows the Visual Studio Code editor with the file explorer on the left displaying the same 'EXAMPLES' folder. The 'ex3cli.py' file is now selected and open in the main editor. The code in ex3cli.py is a Python client script that connects to the server at '127.0.0.1' on port 2053. It sends the message 'Hello, world' and receives 'Received Connection' and 'Server: Hiii' in response. The terminal at the bottom shows the command 'python3 ex3cli.py' being executed, with output indicating the connection and the received data.

```
1 import socket
2 HOST = '127.0.0.1' # The server's hostname or IP address
3 PORT = 2053 # The port used by the server
4 with socket.socket(socket.AF_INET, socket.SOCK_STREAM) as s:
5     s.connect((HOST, PORT))
6     s.sendall(b'Hello, world')
7     data = s.recv(1024)
8     print('Received Connection')
9     print('Server:', data.decode())
10
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

zsh + Python 3.11.4 64-bit Go Live

khushiisrani@Khushis-MacBook-Air examples % python3 ex3cli.py
Received Connection
Server: Hiii
khushiisrani@Khushis-MacBook-Air examples %

Ex4)

The screenshot shows the Visual Studio Code interface with the Explorer sidebar on the left displaying a folder named 'EXAMPLES'. The file 'ex4ser.py' is selected. The main editor window shows the code for 'ex4ser.py' with line numbers 6 through 18. The code is a Python script that sets up a server socket, binds it to a host and port (9991), listens for connections, and when a connection is accepted, it prints the connection details, gets the current time, sends it to the client, and closes the connection.

```
6 host = socket.gethostname()
7 port = 9991
8 # bind to the port
9 serversocket.bind((host, port))
10 # queue up to 5 requests
11 serversocket.listen(5)
12 while True:
13     # establish a connection
14     clientsocket,addr = serversocket.accept()
15     print("Got a connection from %s" % str(addr))
16     currentTime = time.ctime(time.time()) + "\r\n"
17     clientsocket.send(currentTime.encode('ascii'))
18     clientsocket.close()
```

Below the editor, the TERMINAL panel shows the command 'python3 ex4ser.py' being executed, resulting in the output: 'Got a connection from ('127.0.0.1', 55672)'.

The screenshot shows the Visual Studio Code interface with the Explorer sidebar on the left displaying a folder named 'EXAMPLES'. The file 'ex4cli.py' is selected. The main editor window shows the code for 'ex4cli.py' with line numbers 1 through 13. The code is a Python script that creates a client socket, connects it to the server (localhost, 9991), receives data from the server, prints it, and closes the connection.

```
1 #client.py
2 import socket
3 # create a socket object
4 s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
5 # get local machine name
6 host = socket.gethostname()
7 port = 9991
8 # connection to hostname on the port.
9 s.connect((host, port))
10 # Receive no more than 1024 bytes
11 tm = s.recv(1024)
12 print(' Current time from Sever :', tm.decode())
13 s.close()
```

Below the editor, the TERMINAL panel shows the command 'python3 ex4cli.py' being executed, resulting in the output: 'Current time from Sever : Sun Feb 4 18:01:51 2024'.

Ex5)

Server program

```

# server.py

import socket

HOST = '127.0.0.1' # Standard loopback interface address (localhost)

PORT = 12345 # Port to listen on (non-privileged ports are > 1023)

s = socket.socket()

s.bind((HOST, PORT))

s.listen()

print("\nWaiting for incoming connections...\n")

conn, addr = s.accept()

print("Received connection from ", addr[0], "(" , addr[1], ") \n")

s_name = conn.recv(1024)

s_name = s_name.decode()

print(s_name, "has connected to the chat room\nEnter [e] to exit chat room\n")

name = input(str("Enter your name: "))

conn.send(name.encode())

while True:

    message = input(str("Me : "))

    if message == "[e]":

        message = "Left chat room!"

        conn.send(message.encode())

        print("\n")

        break

    conn.send(message.encode())

    message = conn.recv(1024)

    message = message.decode()

    print(s_name, ":", message)

```

Client program

```

import socket

HOST = '127.0.0.1' # Standard loopback interface address (localhost)

PORT = 12345 # Port to listen on (non-privileged ports are > 1023)

s = socket.socket()

name = input(str("\nEnter your name: "))

print("\nTrying to connect to ", HOST, "(", PORT, ")\n")

s.connect((HOST, PORT))

print("Connected...\n")

s.send(name.encode())

s_name = s.recv(1024)

s_name = s_name.decode()

print(s_name, "has joined the chat room\nEnter [e] to exit chat room\n")

while True:

    message = s.recv(1024)

    message = message.decode()

    print(s_name, ":", message)

    message = input(str("Me : "))

    if message == "[e]":

        message = "Left chat room!"

        s.send(message.encode())

        print("\n")

        break

    s.send(message.encode())

```


The screenshot shows a code editor with a file explorer at the top displaying several Python files: ex3ser.py, ex3cli.py, ex4ser.py, ex4cli.py, ex5ser.py, and ex5cli.py. The active file is ex5cli.py, which contains the following code:

```
10 s_name = s.recv(1024)
11 s_name = s_name.decode()
12 print(s_name, "has joined the chat room\nEnter [e] to exit chat room\n")
13 while True:
14     message = s.recv(1024)
15     message = message.decode()
16     print(s_name, ":", message)
17     message = input(str("Me : "))
18     if message == "[e]":
19         message = "Left chat room!"
20     s.send(message.encode())
```

Below the code editor is a terminal window. The terminal shows the execution of the program:

```
khushiisrani@Khushis-MacBook-Air examples % python3 ex5cli.py
Enter your name: khushi
Trying to connect to 127.0.0.1 ( 12345 )
Connected...
saoni has joined the chat room
Enter [e] to exit chat room
saoni : hello
Me : hi
saoni : bye
Me : bye
saoni : Left chat room!
Me : [e]
```

The terminal window also shows the command prompt and the file name: khushiisrani@Khushis-MacBook-Air examples %

Ex6)

Server program

```
import socket
```

```
import os
```

```
from _thread import *
```

```
ServerSocket = socket.socket()
```

```
host = '127.0.0.1'
```

```
port = 11596
```

```
ThreadCount = 0
```

```
try:
```

```
ServerSocket.bind((host, port))
```

```
except socket.error as e:
```

```
print(str(e))
```

```
print("Waiting for a Connection..")
```

```
ServerSocket.listen(5)
```

```
def threaded_client(connection):  
    connection.send(str.encode('Welcome to the Server'))  
  
    while True:  
        data = connection.recv(2048)  
        print('Received from client : ' + str(ThreadCount)+data.decode())  
        Inputs = input('Server Says: ')  
        if not data:  
            break  
        connection.sendall(Inputs.encode())  
        connection.close()  
  
    while True:  
        Client, address = ServerSocket.accept()  
        print('Connected to: ' + address[0] + ':' + str(address[1]))  
        start_new_thread(threaded_client, (Client, ))  
  
        ThreadCount += 1  
        print('Thread Number: ' + str(ThreadCount))  
  
ServerSocket.close()
```

