Sohag University
Faculty of Engineering

LAB 5

By: Zaynab Al-Ariny

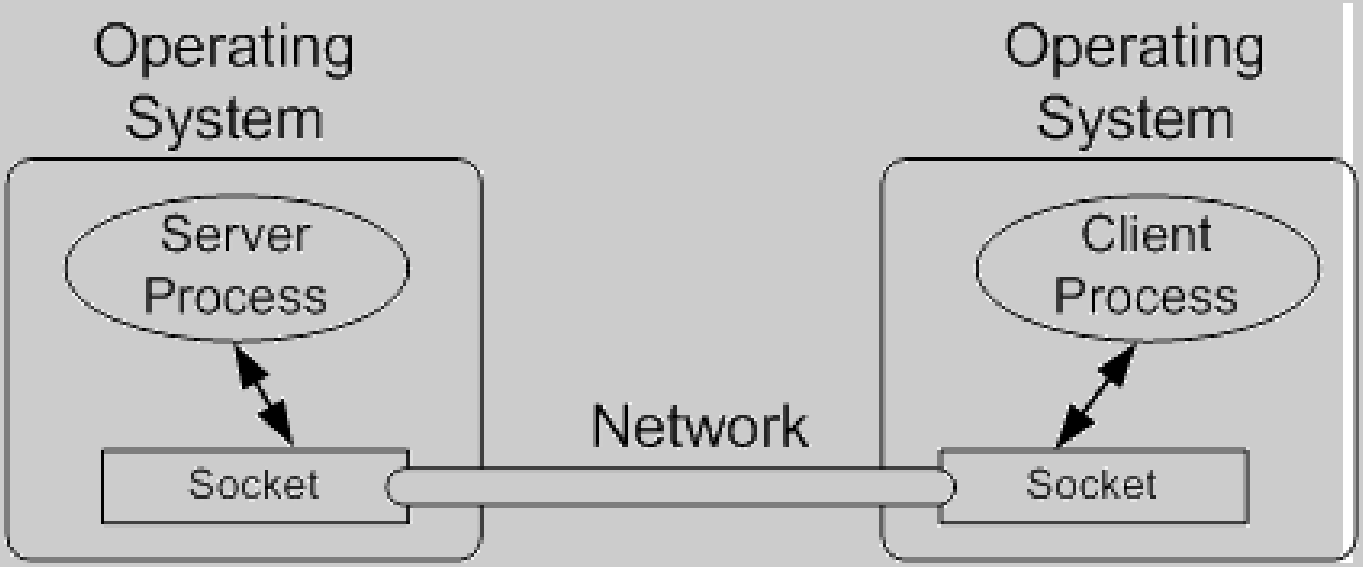
Objectives:

→ Build socket client/server application.
→ Investigate TCP and UDP protocols

Network Socket

- What is Socket?

→ A socket is one of the most fundamental technologies of computer networking.
→ A socket is one end-point of a two-way communication link between two programs running on the network.
→ A server is a process that performs some functions on request from a client.



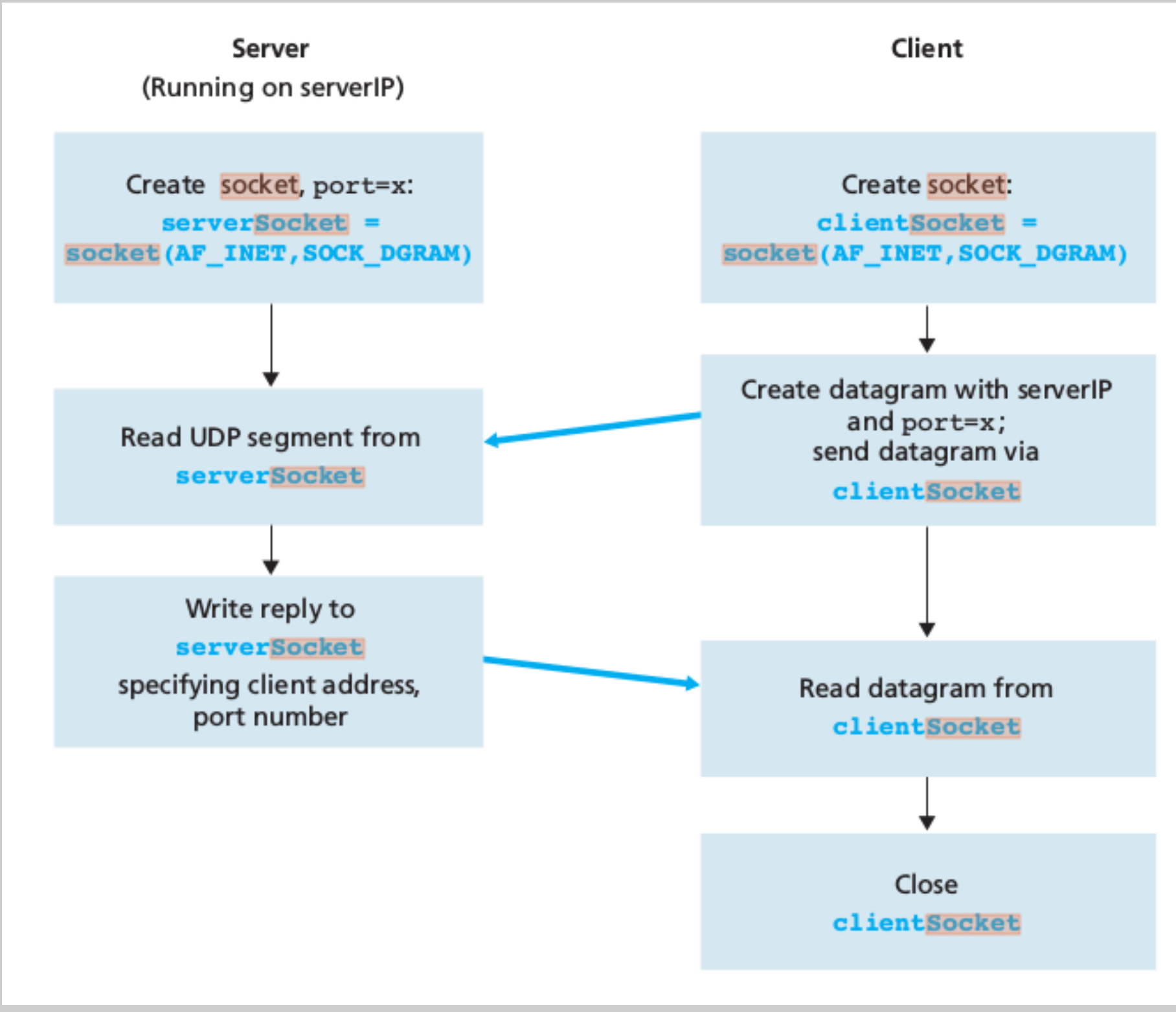
→ We'll use the following simple client-server application to demonstrate socket programming for both UDP and TCP:

1. The client reads a line of characters (data) from its keyboard and sends the data to the server.
2. The server receives the data and converts the characters to uppercase.
3. The server sends the modified data to the client.
4. The client receives the modified data and displays the line on its screen.

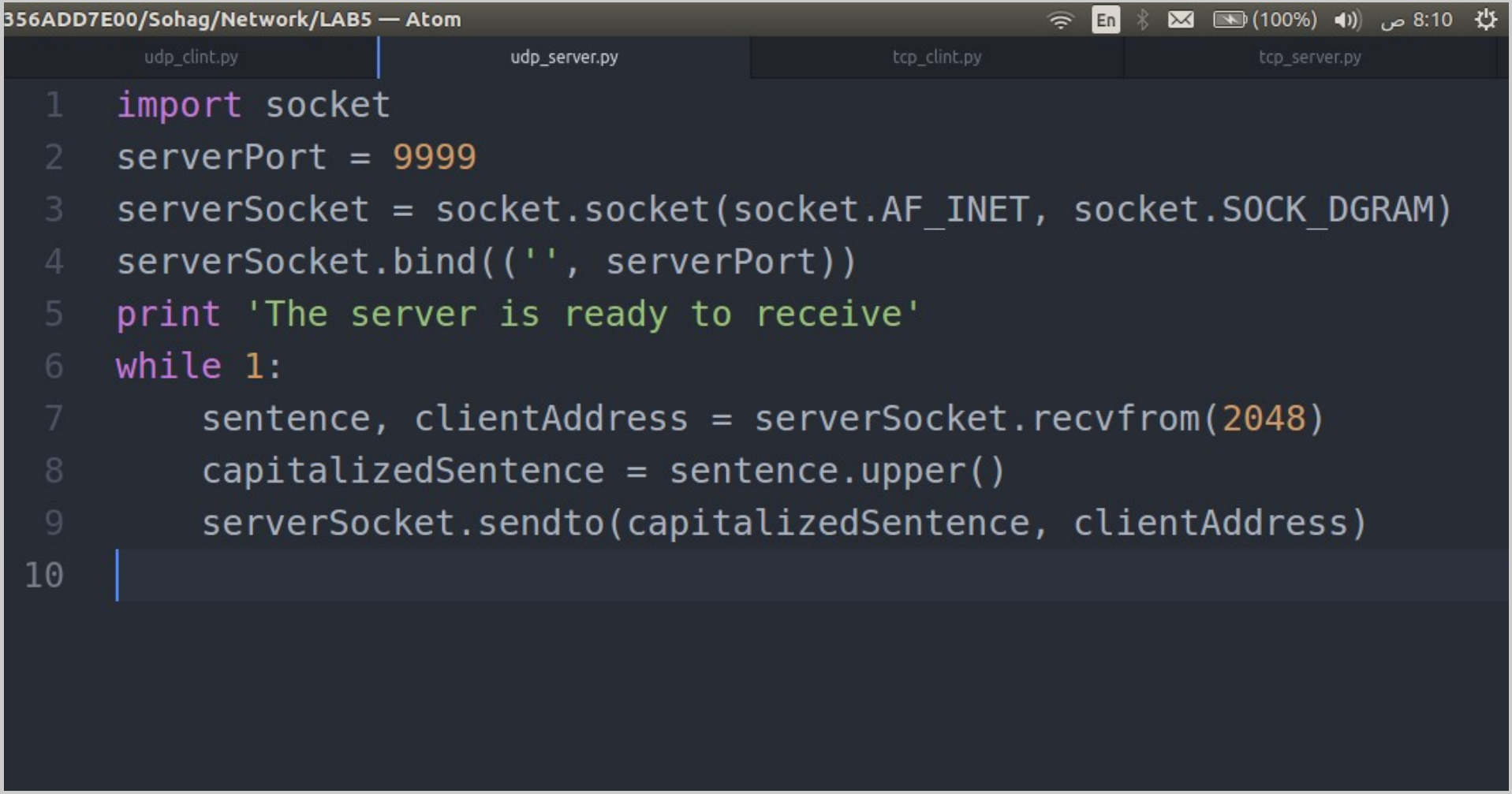
UDP Socket APP

UDP Socket:

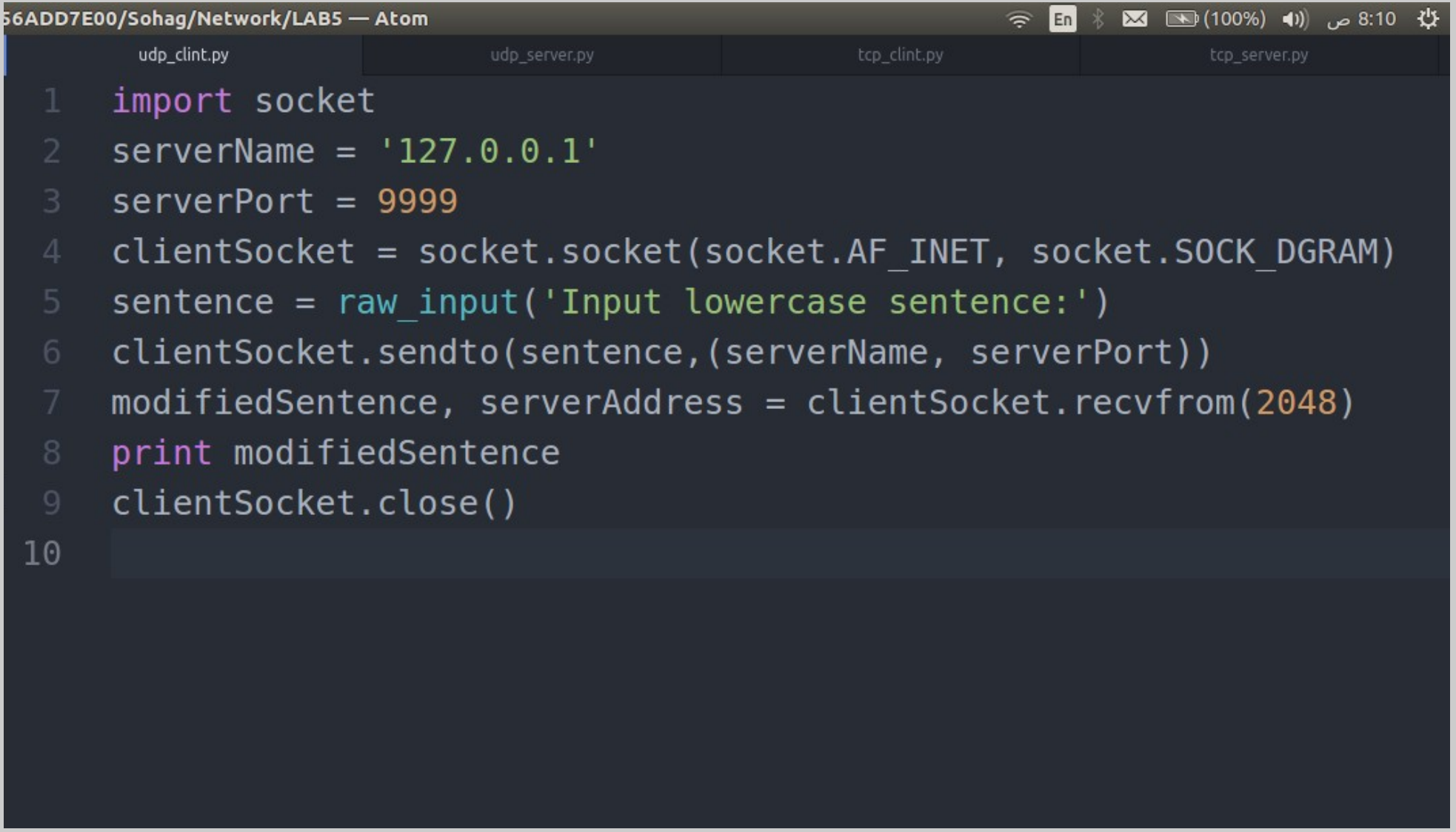
- → unreliable transfer
- → no handshaking before sending data
- → transmitted data may be lost or received out-of-order


Server
(Running on serverIP)
Client
Create socket, port=x:
serverSocket = socket(AF_INET,SOCK_DGRAM)
Create socket:
clientSocket = socket(AF_INET,SOCK_DGRAM)
Create datagram with serverIP and port=x; send datagram via clientSocket
Read UDP segment from serverSocket
Write reply to serverSocket specifying client address, port number
Read datagram from clientSocket
Close clientSocket

UDP Server


356ADD7E00/Sohag/Network/LAB5 — Atom
En
(100%)
8:10 ص
udp_clint.py
udp_server.py
tcp_clint.py
tcp_server.py
1 import socket
2 serverPort = 9999
3 serverSocket = socket.socket(socket.AF_INET, socket.SOCK_DGRAM)
4 serverSocket.bind(('', serverPort))
5 print 'The server is ready to receive'
6 while 1:
7 sentence, clientAddress = serverSocket.recvfrom(2048)
8 capitalizedSentence = sentence.upper()
9 serverSocket.sendto(capitalizedSentence, clientAddress)
10

UDP Clint

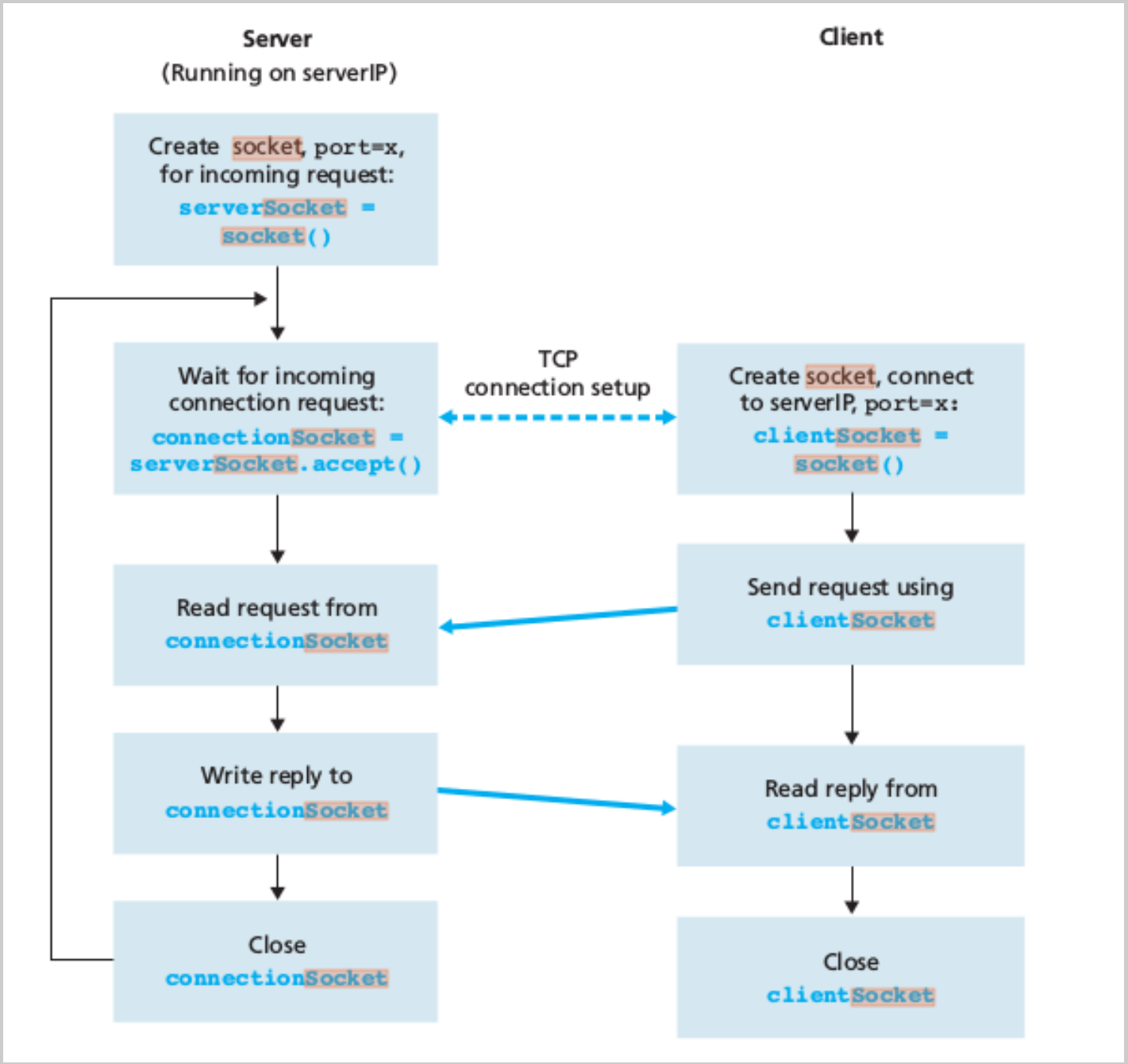

56ADD7E00/Sohag/Network/LAB5 — Atom
En
(100%)
8:10 ص
udp_clint.py
udp_server.py
tcp_clint.py
tcp_server.py
1 import socket
2 serverName = '127.0.0.1'
3 serverPort = 9999
4 clientSocket = socket.socket(socket.AF_INET, socket.SOCK_DGRAM)
5 sentence = raw_input('Input lowercase sentence:')
6 clientSocket.sendto(sentence,(serverName, serverPort))
7 modifiedSentence, serverAddress = clientSocket.recvfrom(2048)
8 print modifiedSentence
9 clientSocket.close()
10

TCP Socket APP

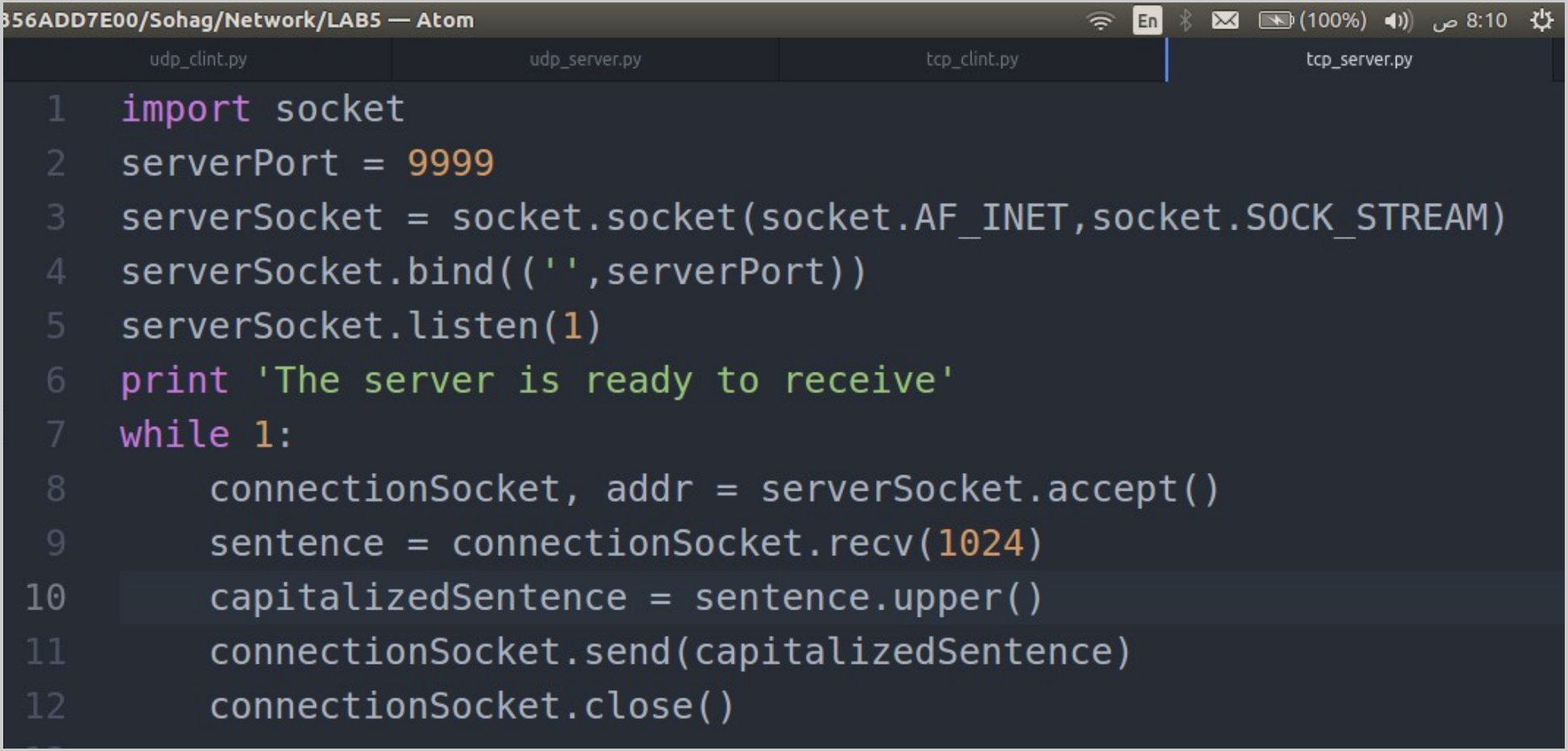
TCP Socket:

- → Reliable transfer
- → Handshaking needed before sending data
- → Provide in-order byte-stream transfer

client-server application using TCP



TCP Server


356ADD7E00/Sohag/Network/LAB5 — Atom
En
(100%)
ص 8:10
udp_clint.py
udp_server.py
tcp_clint.py
tcp_server.py
1 import socket
2 serverPort = 9999
3 serverSocket = socket.socket(socket.AF_INET,socket.SOCK_STREAM)
4 serverSocket.bind(('',serverPort))
5 serverSocket.listen(1)
6 print 'The server is ready to receive'
7 while 1:
8 connectionSocket, addr = serverSocket.accept()
9 sentence = connectionSocket.recv(1024)
10 capitalizedSentence = sentence.upper()
11 connectionSocket.send(capitalizedSentence)
12 connectionSocket.close()

TCP Clint


6ADD7E00/Sohag/Network/LAB5 — Atom
En
(100%)
8:15 ص
udp_clint.py
udp_server.py
tcp_clint.py
tcp_server.py
1 import socket
2 serverName = '127.0.0.1'
3 serverPort = 9999
4 clientSocket = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
5 clientSocket.connect((serverName,serverPort))
6 sentence = raw_input('Input lowercase sentence:')
7 clientSocket.send(sentence)
8 modifiedSentence = clientSocket.recv(1024)
9 print 'From Server:', modifiedSentence
10 clientSocket.close()
11

Let's have fun

1. check that you have Python installed on your machine. If NOT, then install it.
2. Download socket apps examples from here.
3. work in pairs, test the provided UDP client/server apps.
4. work in pairs, test the provided TCP client/server programs.
5. Capture some TCP and UDP packets using wireshark. Notice the difference between them?
5. download codeA and codeB from here, test and answer the following:
    a. which one is the code of a client? And which one is a server?
    b. Does it use a UDP or a TCP socket?
    c. If it use UDP, Edit it to use TCP.

Bonus !

What is the difference between Unicast, Broadcast and Multicast? Give real life applications for each type.

Try to creating a simple Chat application implementing Multicasting.