CLIENT SIDE:

import socket

from os import system, name

import threading

import time

#To clear screen

def clear():

# for windows

if name == 'nt':

\_ = system('cls')

# for mac and linux(here, os.name is 'posix')

else:

\_ = system('clear')

#Thread handler for receiving

def receive\_server(mainSocket):

while True:

message = mainSocket.recv(1024).decode()

#time.sleep(1)

if message == "Exit":

return

print("\n<", socket.gethostbyname(socket.gethostname()), "> ", message, sep = "")

#Main function

clear()

mainSocket = socket.socket(socket.AF\_INET, socket.SOCK\_STREAM, 0)

host = socket.gethostname()

port = 12345

mainSocket.connect((host, port))

message = mainSocket.recv(1024).decode()

print(message)

t = []

t.append(threading.Thread(target=receive\_server, args=(mainSocket, )))

t[0].start()

while True:

reply = input()

print("\n<You>", reply)

mainSocket.send(reply.encode())

if reply == "Exit" or reply == "exit":

break

t[0].join()

mainSocket.close()

SERVER SIDE:  
import socket

from os import system, name

import threading

#To clear screen

def clear():

# for windows

if name == 'nt':

\_ = system('cls')

# for mac and linux(here, os.name is 'posix')

else:

\_ = system('clear')

#Function to broadcast message

def broadcast(message, connectSocket):

for clients in client\_list:

if clients != connectSocket:

clients.send(message.encode())

#Function to delete a client

def remove\_client(connectSocket):

if connectSocket in client\_list:

client\_list.remove(connectSocket)

#Client thread handler

def client\_thread(connectSocket, addr):

connectSocket.send("Welcome to this chat room! (Send 'Exit' to leave)".encode())

while True:

message = connectSocket.recv(1024).decode()

#print(client\_list)

if message == "Exit" or message == "exit":

message\_to\_send = "<" + addr[0] + "> " + "left the chat"

broadcast(message\_to\_send, connectSocket)

print("\n<", addr[0], "> left the chat", sep = "")

connectSocket.send("Exit".encode())

remove\_client(connectSocket)

connectSocket.close()

return

print("\n<", addr[0], "> ", message, sep ="")

message\_to\_send = "<" + addr[0] + "> " + message

broadcast(message, connectSocket)

#Function to reuse inactive threads

def check\_thread(connectSocket, addr, t, i):

if i > 0:

for thread in t:

if not thread.isAlive():

thread = threading.Thread(target=client\_thread, args=(connectSocket, addr))

thread.start()

return(t, i)

t.append(threading.Thread(target=client\_thread, args=(connectSocket, addr)))

t[i].start()

i = i + 1

return(t, i)

#Main function

clear()

print("Chat Room Server\n")

mainSocket = socket.socket(socket.AF\_INET, socket.SOCK\_STREAM, 0)

host = socket.gethostname()

port = 12345

mainSocket.bind((host, port))

client\_list = []

no\_of\_clients = 0

mainSocket.listen(10)

i = 0

t = []

while True:

connectSocket, addr = mainSocket.accept()

client\_list.append(connectSocket)

message\_to\_send = "<" + addr[0] + "> joined the chat"

broadcast(message\_to\_send, connectSocket)

print("\n", addr, "joined the chat")

(t, i) = check\_thread(connectSocket, addr, t, i)

print("Thread: ", i)

connectSocket.close()

mainSocket.close()

OUTPUT:  
