Fuat Yiğit Koçyiğit – 16429085948

CMPE472 Section 1

08.12.2022

TED UNIVERSITY

CMPE 472

Programming Assignment 2:

UDP Pinger

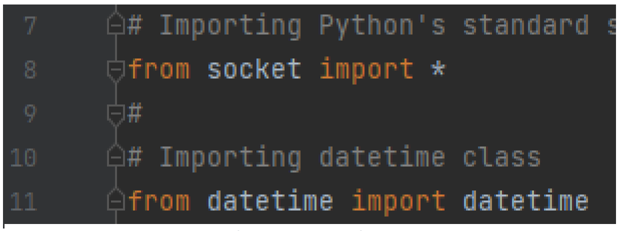
Assignment Summary

In this assignment, I learned the basics of socket programming for UDP connections in Python. I learned how to send and receive datagram packets using UDP sockets and how to set a proper socket timeout.

The web server I created with python is able send messages to the PingerServer and wait until the timeout value finishes. It calculates the RTT value if response is reached and prints the ping informations, else, it prints the timeout message.

Assignment Steps

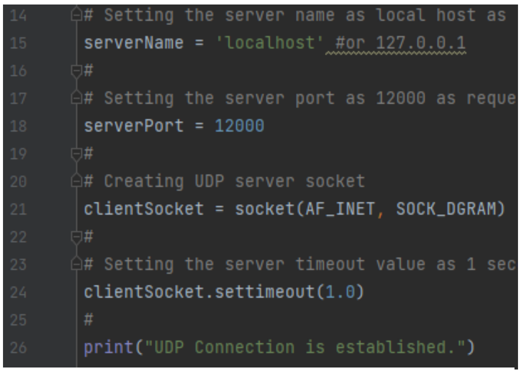
1. In order to create my client server program, I firstly needed to import the required modules from the Python library which are socket and datetime (this is for RTT calculation).



```
# Importing Python's standard s
from socket import *
#
# Importing datetime class
from datetime import datetime
```

Required imports in the program

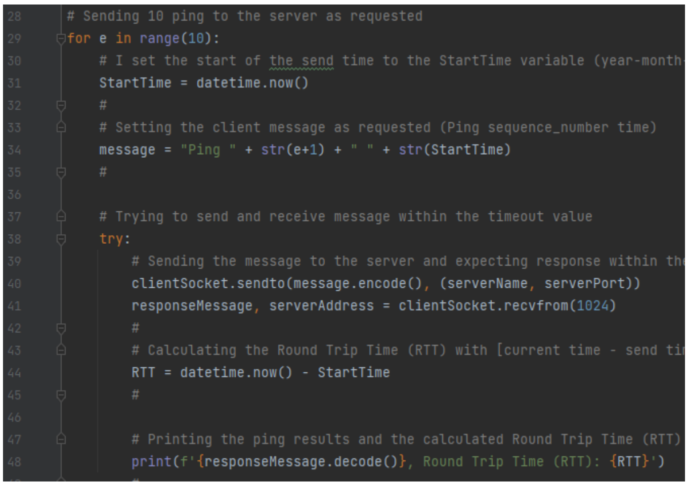
2. As the second step, I set the name and port number of the server. After that, I established the UDP socket connection with the "***clientSocket = socket(AF_INET, SOCK_DGRAM)***" and set the timeout value as 1 second with "***clientSocket.settimeout(1.0)***".



```
# Setting the server name as local host as
serverName = 'localhost' #or 127.0.0.1
#
# Setting the server port as 12000 as reque
serverPort = 12000
#
# Creating UDP server socket
clientSocket = socket(AF_INET, SOCK_DGRAM)
#
# Setting the server timeout value as 1 sec
clientSocket.settimeout(1.0)
#
print("UDP Connection is established.")
```

Setting the UDP connection in the program

3. In the third step, I created a for loop that will repeat 10 times (since the requested pin number is 10). I saved the sending time and created the client message. For loop firstly tries to send and receive message within the timeout value. If it gets the response, saves the current time and calculates the Round Trip Time (RTT) with **[current time - send time]**. After that, the ping results and the calculated RTT printed to console.



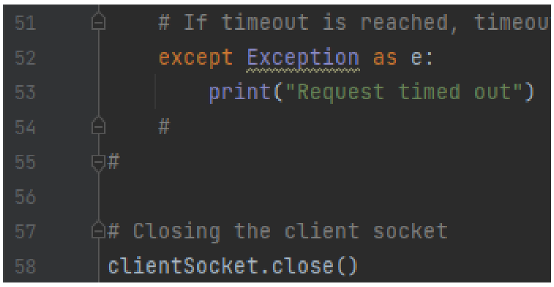
```
# Sending 10 ping to the server as requested
for e in range(10):
    # I set the start of the send time to the StartTime variable (year-month-
    StartTime = datetime.now()
    #
    # Setting the client message as requested (Ping sequence_number time)
    message = "Ping " + str(e+1) + " " + str(StartTime)
    #

    # Trying to send and receive message within the timeout value
    try:
        # Sending the message to the server and expecting response within the
        clientSocket.sendto(message.encode(), (serverName, serverPort))
        responseMessage, serverAddress = clientSocket.recvfrom(1024)
        #
        # Calculating the Round Trip Time (RTT) with [current time - send ti
        RTT = datetime.now() - StartTime
        #

        # Printing the ping results and the calculated Round Trip Time (RTT)
        print(f'{responseMessage.decode()}, Round Trip Time (RTT): {RTT}')
```

The for loop for 10 pins and calculation of the RTT value in the program

4. If the message couldn't receive a response within the timeout value, the timeout message will be printed and for loop skips to the next ping. When the 10 pings are completed, client socket gets closed and program ends.



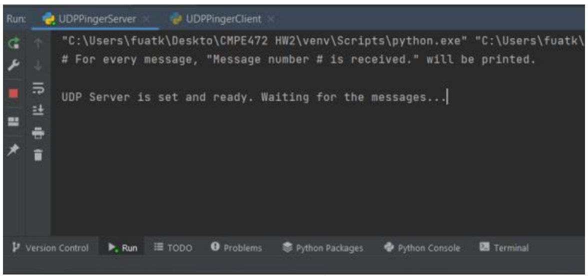
```
    # If timeout is reached, timeou
    except Exception as e:
        print("Request timed out")
    #
#

# Closing the client socket
clientSocket.close()
```

The except case of for loop in the program and the client socket

Screenshots

1. By running the ***“UDPPingerServer.py”***, we can see that (in ***Screenshot 1***) the console is verifying that **server is ready and working, waiting for client messages**.

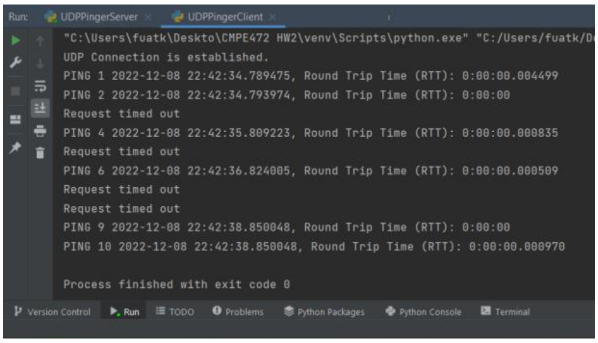


```
"C:\Users\fuatk\Deskto\CMPE472 HW2\venv\Scripts\python.exe" "C:\Users\fuatk\
# For every message, "Message number # is received." will be printed.

UDP Server is set and ready. Waiting for the messages...
```

Screenshot 1

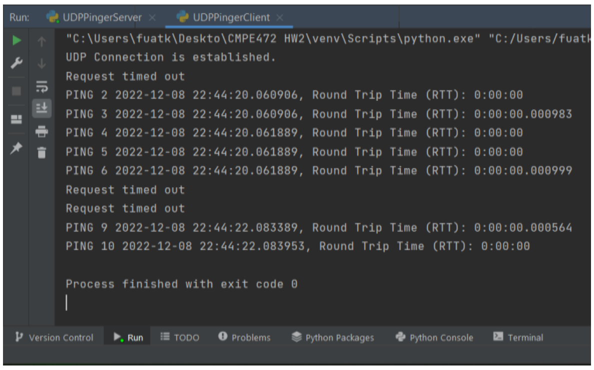
2. By running the ***“UDPClientServer.py”***, we can see that (in ***Screenshot 2 and Screenshot 3***) the console is verifying that **client send total of 10 pings**, **if the waiting time exceeded 1 second, it printed timeout message** and **passed to the next ping** (we can confirm this by looking at the time of PING 1 and PING 5 in Screenshot 1, there are 1*3 = 3 second between them). In every ping, my program **calculated the RTT value and printed**. In the Screenshot 2, the *PING 3, 5, 7 and 8* is lost. In Screenshot 3, the *PING 1, 7 and 8* is lost.



```
"C:\Users\fuatk\Deskto\CMPE472 HW2\venv\Scripts\python.exe" "C:/Users/fuatk/D
UDP Connection is established.
PING 1 2022-12-08 22:42:34.789475, Round Trip Time (RTT): 0:00:00.004499
PING 2 2022-12-08 22:42:34.793974, Round Trip Time (RTT): 0:00:00
Request timed out
PING 4 2022-12-08 22:42:35.809223, Round Trip Time (RTT): 0:00:00.000835
Request timed out
PING 6 2022-12-08 22:42:36.824005, Round Trip Time (RTT): 0:00:00.000509
Request timed out
Request timed out
PING 9 2022-12-08 22:42:38.850048, Round Trip Time (RTT): 0:00:00
PING 10 2022-12-08 22:42:38.850048, Round Trip Time (RTT): 0:00:00.000970

Process finished with exit code 0
```

Screenshot 2

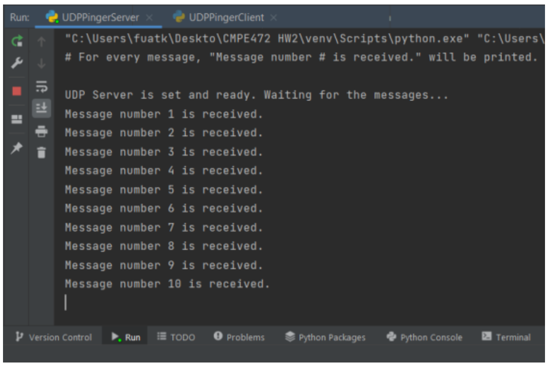


```
"C:\Users\fuatk\Deskto\CMPE472 HW2\venv\Scripts\python.exe" "C:/Users/fuatk
UDP Connection is established.
Request timed out
PING 2 2022-12-08 22:44:20.060906, Round Trip Time (RTT): 0:00:00
PING 3 2022-12-08 22:44:20.060906, Round Trip Time (RTT): 0:00:00.000983
PING 4 2022-12-08 22:44:20.061889, Round Trip Time (RTT): 0:00:00
PING 5 2022-12-08 22:44:20.061889, Round Trip Time (RTT): 0:00:00
PING 6 2022-12-08 22:44:20.061889, Round Trip Time (RTT): 0:00:00.000999
Request timed out
Request timed out
PING 9 2022-12-08 22:44:22.083389, Round Trip Time (RTT): 0:00:00.000564
PING 10 2022-12-08 22:44:22.083953, Round Trip Time (RTT): 0:00:00

Process finished with exit code 0
```

Screenshot 3

3. After we run the ***“UDPClientServer.py”*** for 1 time, we can see that the ***“UDPPingerServer.py”***s informed us about **10 new message receival**. This also proves that **both servers work without any problem**.



```
"C:\Users\fuatk\Deskto\CMPE472 HW2\venv\Scripts\python.exe" "C:\Users\
# For every message, "Message number # is received." will be printed.

UDP Server is set and ready. Waiting for the messages...
Message number 1 is received.
Message number 2 is received.
Message number 3 is received.
Message number 4 is received.
Message number 5 is received.
Message number 6 is received.
Message number 7 is received.
Message number 8 is received.
Message number 9 is received.
Message number 10 is received.
```

Screenshot 4