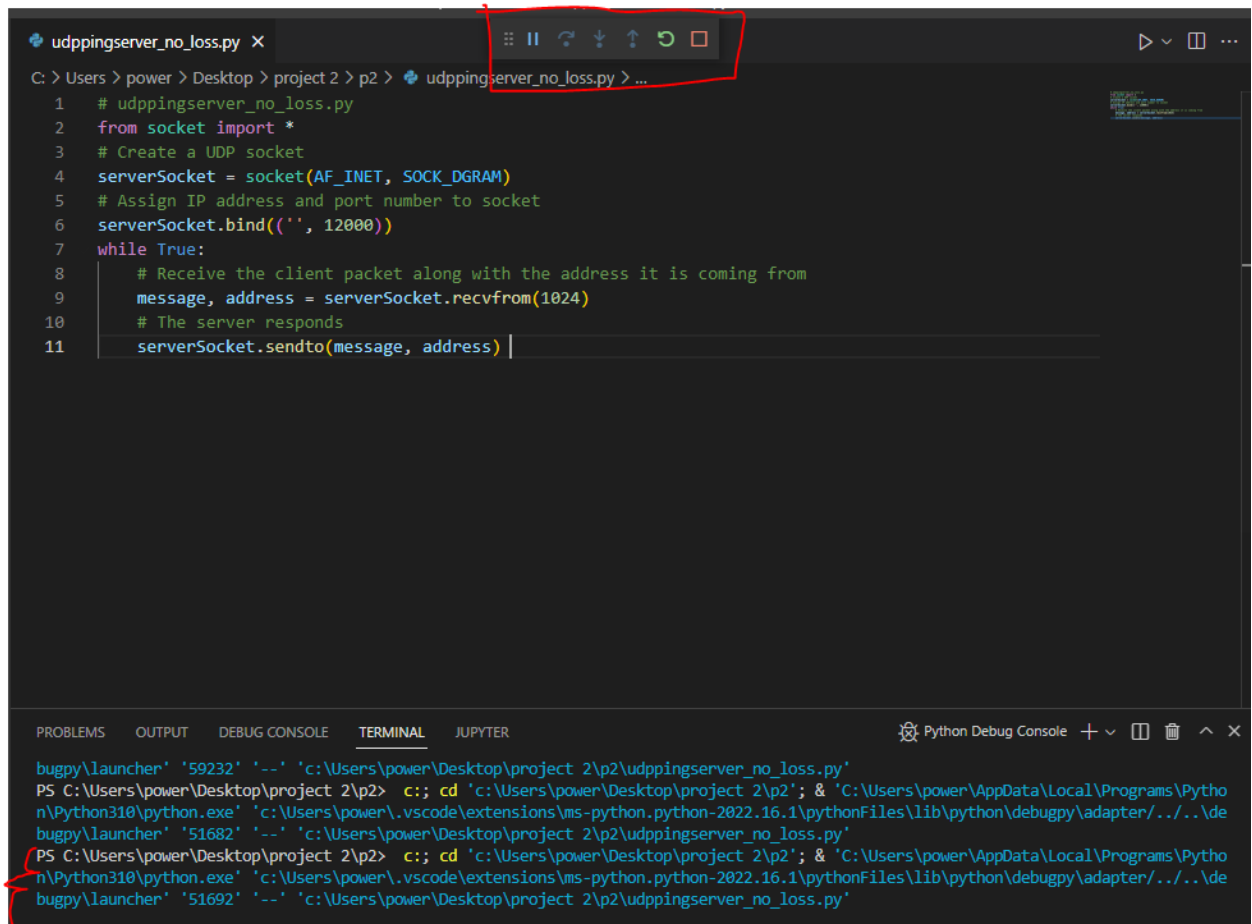


Project 2

Part 1 – UDP Pinger with No Delay and No Loss

1. My UDP Pinger will capture the current system time and pass it into the message sending to the server, it then proceeds to make a try section for receiving a ping back from the server. Should the try succeed, then it will receive an encoded message and address, and mark the time when this packet was received. The difference in these times results in our RTT value for that packet. It is by this point that the min, max, and avg RTT values are updated for later calculation and printing. Should the Pinger fail to receive a packet via timeout, it will then state that the packet was lost and increment that counter for a later calculation. At the end of the program we display all calculations made throughout the process.
2. To specify a timeout value I used the line `clientSocket.settimeout(1.0)`
3. To run my code you must have the sample server running in the background and then proceed to run the client program after.



The screenshot shows a VS Code editor window with a file named `udppingserver_no_loss.py`. The code is a Python script that creates a UDP socket, binds it to `('', 12000)`, and enters a `while True` loop to receive and respond to client packets. The terminal at the bottom shows the command prompt running the script, with the output indicating the script is running successfully.

```
udppingserver_no_loss.py X
C: > Users > power > Desktop > project 2 > p2 > udppingserver_no_loss.py > ...

1 # udppingserver_no_loss.py
2 from socket import *
3 # Create a UDP socket
4 serverSocket = socket(AF_INET, SOCK_DGRAM)
5 # Assign IP address and port number to socket
6 serverSocket.bind(('', 12000))
7 while True:
8     # Receive the client packet along with the address it is coming from
9     message, address = serverSocket.recvfrom(1024)
10    # The server responds
11    serverSocket.sendto(message, address) |
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL JUPYTER Python Debug Console

```
buggy\launcher' '59232' '--' 'c:\Users\power\Desktop\project 2\p2\udppingserver_no_loss.py'
PS C:\Users\power\Desktop\project 2\p2> c:; cd 'c:\Users\power\Desktop\project 2\p2'; & 'C:\Users\power\AppData\Local\Programs\Python\Python310\python.exe' 'c:\Users\power\.vscode\extensions\ms-python.python-2022.16.1\pythonFiles\lib\python\debugpy\adapter\..\de
bugpy\launcher' '51682' '--' 'c:\Users\power\Desktop\project 2\p2\udppingserver_no_loss.py'
PS C:\Users\power\Desktop\project 2\p2> c:; cd 'c:\Users\power\Desktop\project 2\p2'; & 'C:\Users\power\AppData\Local\Programs\Python\Python310\python.exe' 'c:\Users\power\.vscode\extensions\ms-python.python-2022.16.1\pythonFiles\lib\python\debugpy\adapter\..\de
bugpy\launcher' '51692' '--' 'c:\Users\power\Desktop\project 2\p2\udppingserver_no_loss.py'
```

^ server given code running

```
PS C:\Users\power\Desktop\project 2\p2> c:: cd 'c:\Users\power\Desktop\project 2\p2'; & 'C:\Users\power\AppData\Local\Programs\Python\Python310\python.exe'
esktop\project 2\p2\mg1.py'
Mina 1: server reply: Mina 1: Sat Oct 22 19:10:01 2022, RTT = 1.0 ms
Mina 2: server reply: Mina 2: Sat Oct 22 19:10:01 2022, RTT = 1.0 ms
Mina 3: server reply: Mina 3: Sat Oct 22 19:10:01 2022, RTT = 0.0 ms
Mina 4: server reply: Mina 4: Sat Oct 22 19:10:01 2022, RTT = 1.0 ms
Mina 5: server reply: Mina 5: Sat Oct 22 19:10:01 2022, RTT = 0.0 ms
Mina 6: server reply: Mina 6: Sat Oct 22 19:10:01 2022, RTT = 1.0 ms
Mina 7: server reply: Mina 7: Sat Oct 22 19:10:01 2022, RTT = 0.0 ms
Mina 8: server reply: Mina 8: Sat Oct 22 19:10:01 2022, RTT = 1.0 ms
Mina 9: server reply: Mina 9: Sat Oct 22 19:10:01 2022, RTT = 0.0 ms
Mina 10: server reply: Mina 10: Sat Oct 22 19:10:01 2022, RTT = 1.0 ms
Min RTT = 0.0 ms
Max RTT = 1.0 ms
Avg RTT = 0.55 ms
Packet lost = 0.0 %
PS C:\Users\power\Desktop\project 2\p2> c:: cd 'c:\Users\power\Desktop\project 2\p2'; & 'C:\Users\power\AppData\Local\Programs\Python\Python310\python.exe'
'c:\Users\power\.vscode\extensions\ms-python.python-2022.16.1\pythonFiles\lib\python\debugpy\adapter\..\..\debugpy\launcher' '51630' '--' 'c:\Users\power\D
esktop\project 2\p2\mg1.py'
Mina 1: server reply: Mina 1: Sat Oct 22 19:11:49 2022, RTT = 1.0 ms
Mina 2: server reply: Mina 2: Sat Oct 22 19:11:49 2022, RTT = 0.0 ms
Mina 3: server reply: Mina 3: Sat Oct 22 19:11:49 2022, RTT = 0.0 ms
Mina 4: server reply: Mina 4: Sat Oct 22 19:11:49 2022, RTT = 0.0 ms
Mina 5: server reply: Mina 5: Sat Oct 22 19:11:49 2022, RTT = 0.0 ms
Mina 6: server reply: Mina 6: Sat Oct 22 19:11:49 2022, RTT = 1.0 ms
Mina 7: server reply: Mina 7: Sat Oct 22 19:11:49 2022, RTT = 0.0 ms
Mina 8: server reply: Mina 8: Sat Oct 22 19:11:49 2022, RTT = 0.0 ms
Mina 9: server reply: Mina 9: Sat Oct 22 19:11:49 2022, RTT = 1.0 ms
Mina 10: server reply: Mina 10: Sat Oct 22 19:11:49 2022, RTT = 0.0 ms
Min RTT = 0.0 ms
Max RTT = 1.0 ms
Avg RTT = 0.27 ms
Packet lost = 0.0 %
PS C:\Users\power\Desktop\project 2\p2> |
```

^Sample output of the client program running while the server is running

4.

```
from logging import exception
from socket import *
import time

# Establish host information
recieve_port = 1024

# Establish server information
remote_host = 'localhost'
remote_port = 12000

# Create our socket for sending and recieveing messages from server
clientSocket = socket(AF_INET, SOCK_DGRAM)
clientSocket.settimeout(1.0)
clientSocket.bind((remote_host, recieve_port))

# Set up varriables we will be using
num_pings = 50
ping_num = 1
min_rtt = 0
max_rtt = 0
```

```

avg_rtt = 0
packets_dropped = 0.0
total_packets = 0.0
packet_loss_rate = 0.0

# Loop for 10 pings
while ping_num <= num_pings:
    # Create a message to output later
    initial_time = time.time()
    message = 'Mina ' + str(ping_num) + ': ' + time.ctime(initial_time)
    clientSocket.sendto(message.encode(),(remote_host, remote_port))
    try:
        recieved_message, server_address = clientSocket.recvfrom(remote_port)
        post_time = time.time()
        rtt = (post_time - initial_time)* 1000
        print('Mina ' + str(ping_num) + ': server reply: ' +
            recieved_message.decode() + ', RTT = ' + str(round(rtt, 2)) + ' ms')
        avg_rtt += rtt
        total_packets += 1
        if rtt < min_rtt or min_rtt == 0:
            min_rtt = rtt
        if rtt > max_rtt or max_rtt == 0:
            max_rtt = rtt
    except:
        print('Packet lost')
        packets_dropped += 1
    ping_num += 1

# Out of loop printing
print('Min RTT = ' + str(round(min_rtt, 2)) + ' ms')
print('Max RTT = ' + str(round(max_rtt, 2)) + ' ms')
avg_rtt = str(round((avg_rtt / total_packets), 2))
print('Avg RTT = ' + avg_rtt + ' ms')
packet_loss_rate = str(round((packets_dropped / ping_num), 2) * 100)
print('Packet lost = ' + packet_loss_rate + ' %')

clientSocket.close()

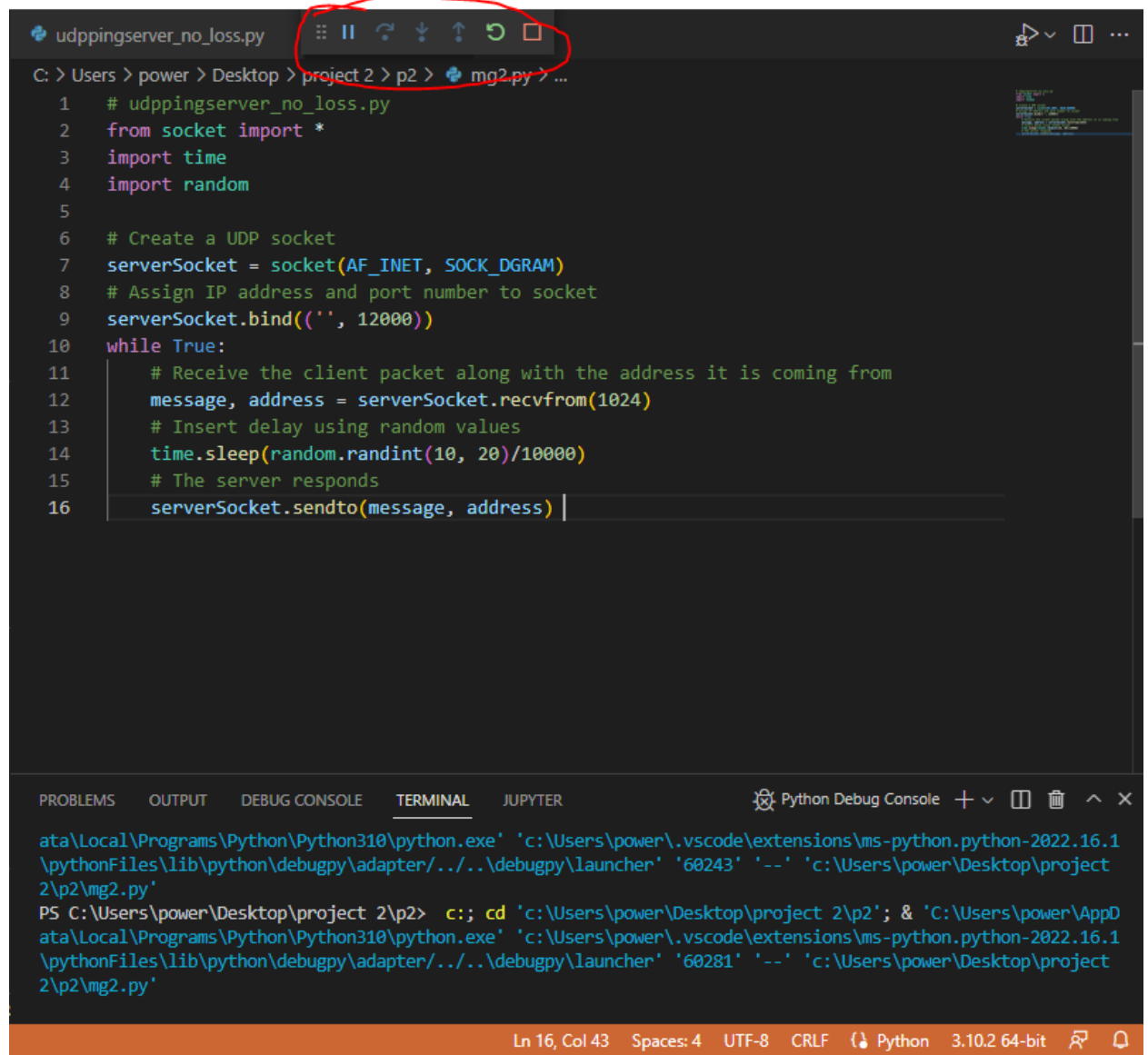
```

Part 2 – UDP Pinger No Loss, with Delays

1. I added the libraries time and random. And I did a function call in between receiving and sending information from the server back to the client. I called time.sleep and generated a random

number between 10 and 20 and divided it by 10000 to change the value into milliseconds.
`time.sleep(random.randint(10, 20)/10000)`

2. Make sure the server is running (mg2.py) and then run mg1.py.



The screenshot shows a VS Code editor window with a Python file named `udppingserver_no_loss.py`. The code is a UDP server that receives packets and responds with a delay. The terminal shows the command to run the script and the output of the command.

```
udppingserver_no_loss.py
C: > Users > power > Desktop > project 2 > p2 > mg2.py > ...
1  # udppingserver_no_loss.py
2  from socket import *
3  import time
4  import random
5
6  # Create a UDP socket
7  serverSocket = socket(AF_INET, SOCK_DGRAM)
8  # Assign IP address and port number to socket
9  serverSocket.bind(('', 12000))
10 while True:
11     # Receive the client packet along with the address it is coming from
12     message, address = serverSocket.recvfrom(1024)
13     # Insert delay using random values
14     time.sleep(random.randint(10, 20)/10000)
15     # The server responds
16     serverSocket.sendto(message, address)
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL JUPYTER Python Debug Console + - [] {} ^ x

```
ata\Local\Programs\Python\Python310\python.exe' 'c:\Users\power\.vscode\extensions\ms-python.python-2022.16.1
\pythonFiles\lib\python\debugpy\adapter\..\..\debugpy\launcher' '60243' '--' 'c:\Users\power\Desktop\project
2\p2\mg2.py'
PS C:\Users\power\Desktop\project 2\p2> c:; cd 'c:\Users\power\Desktop\project 2\p2'; & 'C:\Users\power\AppData
ata\Local\Programs\Python\Python310\python.exe' 'c:\Users\power\.vscode\extensions\ms-python.python-2022.16.1
\pythonFiles\lib\python\debugpy\adapter\..\..\debugpy\launcher' '60281' '--' 'c:\Users\power\Desktop\project
2\p2\mg2.py'
```

Ln 16, Col 43 Spaces: 4 UTF-8 CRLF Python 3.10.2 64-bit

```

Mina 1: server reply: Mina 1: Sat Oct 22 20:37:55 2022, RTT = 14.01 ms
Mina 2: server reply: Mina 2: Sat Oct 22 20:37:55 2022, RTT = 16.02 ms
Mina 3: server reply: Mina 3: Sat Oct 22 20:37:55 2022, RTT = 15.01 ms
Mina 4: server reply: Mina 4: Sat Oct 22 20:37:55 2022, RTT = 16.02 ms
Mina 5: server reply: Mina 5: Sat Oct 22 20:37:55 2022, RTT = 16.01 ms
Mina 6: server reply: Mina 6: Sat Oct 22 20:37:55 2022, RTT = 16.01 ms
Mina 7: server reply: Mina 7: Sat Oct 22 20:37:55 2022, RTT = 16.02 ms
Mina 8: server reply: Mina 8: Sat Oct 22 20:37:55 2022, RTT = 16.01 ms
Mina 9: server reply: Mina 9: Sat Oct 22 20:37:55 2022, RTT = 16.02 ms
Mina 10: server reply: Mina 10: Sat Oct 22 20:37:55 2022, RTT = 15.01 ms
Min RTT = 0 ms
Max RTT = 16.02 ms
Avg RTT = 14.19 ms
Packet lost = 0.0 %
PS C:\Users\power\Desktop\project_2>

```

3.

```

# udppingserver_no_loss.py

from socket import *
import time
import random

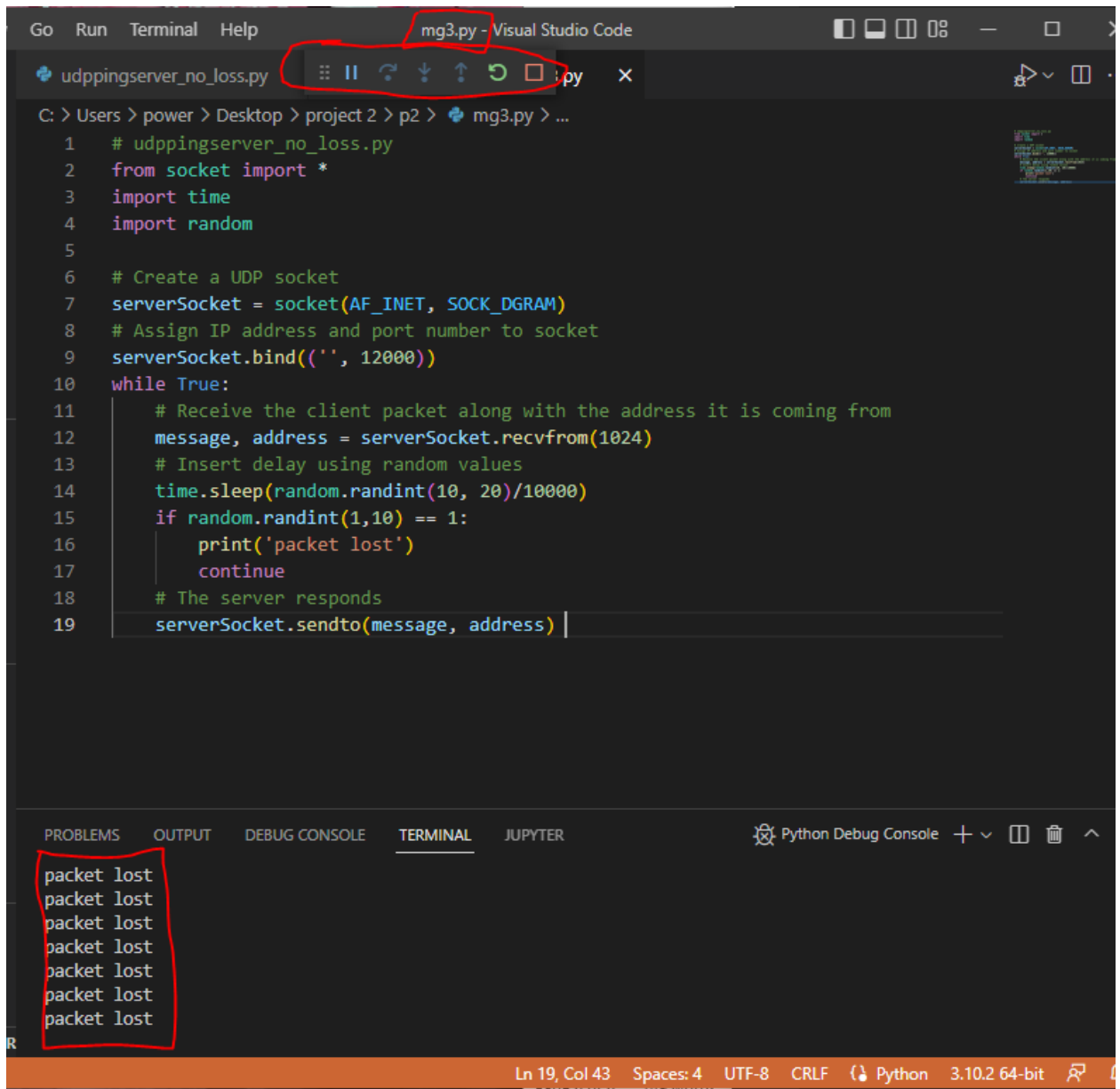
# Create a UDP socket
serverSocket = socket(AF_INET, SOCK_DGRAM)
# Assign IP address and port number to socket
serverSocket.bind('', 12000)
while True:
    # Receive the client packet along with the address it is coming from
    message, address = serverSocket.recvfrom(1024)
    # Insert delay using random values
    time.sleep(random.randint(10, 20)/10000)
    # The server responds
    serverSocket.sendto(message, address)

```

Part 3 – UDP Pinger with Delays and Packet Losses

1. My UDP server will sleep for a random time frame between 10-20ms then will randomly generate a number between 1-10, and if that random number is 1 (signifying a 10% chance 1/10) it will print 'packet lost' on the terminal of the server and continue to looking for a receive ping without issuing a .sendto call. On the client side, it waits until the except is thrown, it then will print 'packet lost' on the terminal showing and skip to the next ping. E.g. ping 1, packet lost, ping 3

2. Run the mg3.py and then run mg1.py while mg3.py is running in the back ground. If you want to change the number of pings that show up, you will need to change a value in mg1.py The variable 'num_pings' found on line 18 is responsible for the number of pings that are issued in the program. From part 1 and 2 it was 10, and part 3 it is 50.



The screenshot shows the Visual Studio Code interface. The top toolbar has a red circle around the 'Run and Debug' icon (a play button with a bug). The editor window displays the file 'udppingserver_no_loss.py' with the following code:

```
1 # udppingserver_no_loss.py
2 from socket import *
3 import time
4 import random
5
6 # Create a UDP socket
7 serverSocket = socket(AF_INET, SOCK_DGRAM)
8 # Assign IP address and port number to socket
9 serverSocket.bind(('', 12000))
10 while True:
11     # Receive the client packet along with the address it is coming from
12     message, address = serverSocket.recvfrom(1024)
13     # Insert delay using random values
14     time.sleep(random.randint(10, 20)/10000)
15     if random.randint(1,10) == 1:
16         print('packet lost')
17         continue
18     # The server responds
19     serverSocket.sendto(message, address)
```

The bottom panel shows the 'TERMINAL' tab with the following output:

```
packet lost
packet lost
packet lost
packet lost
packet lost
packet lost
packet lost
```

The status bar at the bottom indicates 'Ln 19, Col 43', 'Spaces: 4', 'UTF-8', 'CRLF', 'Python 3.10.2 64-bit'.


```
Mina 1: server reply: Mina 1: Sat Oct 22 20:59:59 2022, RTT = 3.0 ms
Mina 2: server reply: Mina 2: Sat Oct 22 20:59:59 2022, RTT = 16.01 ms
Mina 3: server reply: Mina 3: Sat Oct 22 20:59:59 2022, RTT = 15.01 ms
Mina 4: server reply: Mina 4: Sat Oct 22 20:59:59 2022, RTT = 16.01 ms
Mina 5: server reply: Mina 5: Sat Oct 22 20:59:59 2022, RTT = 16.01 ms
Mina 6: server reply: Mina 6: Sat Oct 22 20:59:59 2022, RTT = 16.01 ms
Mina 7: server reply: Mina 7: Sat Oct 22 20:59:59 2022, RTT = 16.01 ms
Mina 8: server reply: Mina 8: Sat Oct 22 20:59:59 2022, RTT = 16.01 ms
Packet lost
Mina 10: server reply: Mina 10: Sat Oct 22 21:00:00 2022, RTT = 16.01 ms
Mina 11: server reply: Mina 11: Sat Oct 22 21:00:00 2022, RTT = 16.01 ms
Mina 12: server reply: Mina 12: Sat Oct 22 21:00:00 2022, RTT = 15.01 ms
Mina 13: server reply: Mina 13: Sat Oct 22 21:00:00 2022, RTT = 15.01 ms
Packet lost
Packet lost
Mina 16: server reply: Mina 16: Sat Oct 22 21:00:02 2022, RTT = 16.01 ms
Mina 17: server reply: Mina 17: Sat Oct 22 21:00:02 2022, RTT = 16.01 ms
Mina 18: server reply: Mina 18: Sat Oct 22 21:00:02 2022, RTT = 16.01 ms
Mina 19: server reply: Mina 19: Sat Oct 22 21:00:02 2022, RTT = 15.96 ms
Packet lost
Mina 21: server reply: Mina 21: Sat Oct 22 21:00:03 2022, RTT = 16.02 ms
Mina 22: server reply: Mina 22: Sat Oct 22 21:00:03 2022, RTT = 16.01 ms
Mina 23: server reply: Mina 23: Sat Oct 22 21:00:03 2022, RTT = 15.29 ms
Mina 24: server reply: Mina 24: Sat Oct 22 21:00:03 2022, RTT = 16.02 ms
Mina 25: server reply: Mina 25: Sat Oct 22 21:00:03 2022, RTT = 16.01 ms
Mina 26: server reply: Mina 26: Sat Oct 22 21:00:03 2022, RTT = 16.01 ms
Mina 27: server reply: Mina 27: Sat Oct 22 21:00:03 2022, RTT = 16.01 ms
Mina 28: server reply: Mina 28: Sat Oct 22 21:00:03 2022, RTT = 16.02 ms
Mina 29: server reply: Mina 29: Sat Oct 22 21:00:03 2022, RTT = 16.01 ms
Mina 30: server reply: Mina 30: Sat Oct 22 21:00:03 2022, RTT = 16.01 ms
Mina 31: server reply: Mina 31: Sat Oct 22 21:00:03 2022, RTT = 16.01 ms
Mina 32: server reply: Mina 32: Sat Oct 22 21:00:03 2022, RTT = 16.01 ms
Mina 33: server reply: Mina 33: Sat Oct 22 21:00:03 2022, RTT = 16.02 ms
Packet lost
Mina 35: server reply: Mina 35: Sat Oct 22 21:00:04 2022, RTT = 16.01 ms
Mina 36: server reply: Mina 36: Sat Oct 22 21:00:04 2022, RTT = 16.11 ms
Mina 37: server reply: Mina 37: Sat Oct 22 21:00:04 2022, RTT = 16.02 ms
Mina 38: server reply: Mina 38: Sat Oct 22 21:00:04 2022, RTT = 16.01 ms
Mina 39: server reply: Mina 39: Sat Oct 22 21:00:04 2022, RTT = 16.01 ms
Mina 40: server reply: Mina 40: Sat Oct 22 21:00:04 2022, RTT = 16.01 ms
Packet lost
Mina 42: server reply: Mina 42: Sat Oct 22 21:00:05 2022, RTT = 16.01 ms
Mina 43: server reply: Mina 43: Sat Oct 22 21:00:05 2022, RTT = 15.01 ms
Mina 44: server reply: Mina 44: Sat Oct 22 21:00:05 2022, RTT = 16.01 ms
Mina 45: server reply: Mina 45: Sat Oct 22 21:00:05 2022, RTT = 16.01 ms
Mina 46: server reply: Mina 46: Sat Oct 22 21:00:05 2022, RTT = 16.01 ms
Mina 47: server reply: Mina 47: Sat Oct 22 21:00:05 2022, RTT = 16.01 ms
Mina 48: server reply: Mina 48: Sat Oct 22 21:00:05 2022, RTT = 16.01 ms
Mina 49: server reply: Mina 49: Sat Oct 22 21:00:05 2022, RTT = 16.01 ms
Mina 50: server reply: Mina 50: Sat Oct 22 21:00:05 2022, RTT = 16.01 ms
Min RTT = 3.0 ms
Max RTT = 16.11 ms
Avg RTT = 15.61 ms
Packet lost = 12.0 %
D:\C:\Users\pavon\Desktop\project_2\p2\
```


3.

```
# udppingserver_no_loss.py

from socket import *
import time
import random

# Create a UDP socket
serverSocket = socket(AF_INET, SOCK_DGRAM)
# Assign IP address and port number to socket
serverSocket.bind('', 12000)
while True:
    # Receive the client packet along with the address it is coming from
    message, address = serverSocket.recvfrom(1024)
    # Insert delay using random values
    time.sleep(random.randint(10, 20)/10000)
    if random.randint(1,10) == 1:
        print('packet lost')
        continue
    # The server responds
    serverSocket.sendto(message, address)
```

Part 4: HeartBeat Monitor Using Python

1. Run mg4c.py and mg4s.py, if you want multiple clients then you have to make multiple mg4c.py copies and have them run at the same time.

2.

```
Server recieved: Mina client2: sent heartbeat to server Sat Oct 22 22:58:56 2022
heartbeat recieved 20 seconds ago
Server recieved: Mina client1: sent heartbeat to server Sat Oct 22 22:59:03 2022
heartbeat recieved 6 seconds ago
Server recieved: Mina client1: sent heartbeat to server Sat Oct 22 22:59:15 2022
heartbeat recieved 12 seconds ago
Server recieved: Mina client1: sent heartbeat to server Sat Oct 22 22:59:25 2022
heartbeat recieved 10 seconds ago
Server recieved: Mina client2: sent heartbeat to server Sat Oct 22 22:59:31 2022
heartbeat recieved 6 seconds ago
Server recieved: Mina client1: sent heartbeat to server Sat Oct 22 22:59:48 2022
heartbeat recieved 17 seconds ago
Server recieved: Mina client2: sent heartbeat to server Sat Oct 22 23:00:07 2022
heartbeat recieved 20 seconds ago
Server recieved: Mina client1: sent heartbeat to server Sat Oct 22 23:00:10 2022
heartbeat recieved 2 seconds ago
No heartbeat after 20 seconds. Server quits.
Server stops
PS C:\Users\power\Desktop\project 2\p2>
```

```
\pythonFiles\lib\python\debugpy\adapter\..\..\debugpy\launcher' '51667' '-
2\p2\mg4c.py'
Mina client2: sent heartbeat to server Sat Oct 22 22:58:56 2022
Mina client1: sent heartbeat to server Sat Oct 22 22:59:03 2022
Mina client1: sent heartbeat to server Sat Oct 22 22:59:15 2022
Mina client1: sent heartbeat to server Sat Oct 22 22:59:25 2022
Mina client2: sent heartbeat to server Sat Oct 22 22:59:31 2022
PS C:\Users\power\Desktop\project 2\p2>
```

3.

```
from logging import exception

from socket import *
import time
import random

# Establish host information
recieve_port = 1024

# Establish server information
remote_host = 'localhost'
remote_port = 12000
```

```

# Create our socket for sending and receiving messages from server
clientSocket = socket(AF_INET, SOCK_DGRAM)
clientSocket.settimeout(20.0)
clientSocket.bind((remote_host, receive_port))

# Continuously send out messages with a 2-25 second delay
while True:
    delay = random.randint(2,25)
    time.sleep(delay)
    # Create a message to output later
    message = 'Mina client1: sent heartbeat to server ' + time.ctime()
    clientSocket.sendto(message.encode(),(remote_host, remote_port))
    try:
        received_message, server_address = clientSocket.recvfrom(remote_port)
        print(received_message.decode())
    except:
        print('Server has closed')
        break
clientSocket.close()

```

```

from socket import *
import time
import random

# Create a UDP socket
serverSocket = socket(AF_INET, SOCK_DGRAM)
# Assign IP address and port number to socket
serverSocket.bind(('localhost', 12000))
serverSocket.settimeout(20.0)

while True:
    # Receive the client packet along with the address it is coming from
    try:
        current_time = time.time()
        message, address = serverSocket.recvfrom(1024)
        receive_time = time.time()
        serverSocket.sendto(message,('localhost', 1024))
        timelapse = receive_time - current_time
        # The server responds
        print('Server received: ' + message.decode())
        print('heartbeat received ' + str(round(timelapse)) + ' seconds ago')
    except:
        print('No heartbeat after 20 seconds. Server quits.')
        print('Server stops')

```

```
        break  
serverSocket.close()
```