Assignment-3 CS2007 (21BCS064)

UDPPingerServer.py

```
# UDPPingerServer.py
# We will need the following module to generate randomized lost packets
import random

import socket

# Create a UDP socket
# Notice the use of SOCK_DGRAM for UDP packets

SERVER = socket.gethostbyname(socket.gethostname())

PORT = 20001

HEADER = 1024

ADDR = (SERVER, PORT)

serverSocket = socket.socket(socket.AF_INET, socket.SOCK_DGRAM)

# Assign IP address and port number to socket

serverSocket.bind(ADDR)

print("UDP server up and listening")


while True:
    # Generate random number in the range of 0 to 10

    rand = random.randint(0, 10)

    # Receive the client packet along with the address it is coming from

    message, address = serverSocket.recvfrom(HEADER)

    print(f"Message from Client : {message}        rand_value : {rand}")

    # Capitalize the message from the client

    message = message.upper()

    # If rand is less is than 4, we consider the packet lost and do not
respond
```

```
    if rand < 4:
        continue

    # Otherwise, the server responds

    serverSocket.sendto(message, address)
```

UDPClient.py

```
import socket

import time

SERVER = socket.gethostbyname(socket.gethostname())

PORT = 20001

HEADER = 1024

ADDR = (SERVER, PORT)

# Create a UDP socket at client side

client = socket.socket(socket.AF_INET, socket.SOCK_DGRAM)

client.settimeout(1)

# Send to server using created UDP socket

rtts = []

packet_loss_rate = 0

for i in range(1, 11):

    send_time = time.time()

    msgFromClient = f"PING sequence:{i} {send_time}"

    bytesToSend = str.encode(msgFromClient)

    client.sendto(bytesToSend, ADDR)

    try:
        msgFromServer = client.recvfrom(HEADER)

        recv_time = time.time()
```

```
        rtts.append(recv_time-send_time)
        msg = f"Message from Server {msgFromServer[0]}        Round Trip Time
: {recv_time-send_time}"

        print(msg)

    except TimeoutError:
        packet_loss_rate = packet_loss_rate+1
        print("Request Timed Out!")

print("")
print(f"Maximum Round Trip Time : {max(rtts)}")
print(f"Minimum Round Trip Time : {min(rtts)}")
print(f"Average Round Trip Time : {sum(rtts)/len(rtts)}")
print(f"Packet Loss Rate Percentage : {10*packet_loss_rate}%")
```

Client Output:

```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL

PS C:\Users\Chandrashekhar Gouda\Networking> python -u "c:\Users\Chandrashekhar Gouda\Networking\UDP\UDPClient.py"
Message from Server b'PING SEQUENCE:1 1675157390.7551756'        Round Trip Time : 0.0199995040893554 7
Message from Server b'PING SEQUENCE:2 1675157390.775175'        Round Trip Time : 0.0009999275207519531
Request Timed Out!
Message from Server b'PING SEQUENCE:4 1675157391.7778604'        Round Trip Time : 0.0
Message from Server b'PING SEQUENCE:5 1675157391.7778604'        Round Trip Time : 0.0
Message from Server b'PING SEQUENCE:6 1675157391.7778604'        Round Trip Time : 0.0
Request Timed Out!
Request Timed Out!
Request Timed Out!
Message from Server b'PING SEQUENCE:10 1675157394.8082619'        Round Trip Time : 0.0010237693786621094
Maximum Round Trip Time : 0.01999950408935547
Minimum Round Trip Time : 0.0
Average Round Trip Time : 0.0036705334981282554
Packet Loss Rate Percentage : 40%
PS C:\Users\Chandrashekhar Gouda\Networking>
```

Server Output:

```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL

PS C:\Users\Chandrashekhar Gouda\Networking> python -u "c:\Users\Chandrashekhar Gouda\Networking\UDP\UDPPingerServer.py"
UDP server up and listening
Message from Client : b'PING sequence:1 1675157390.7551756'        rand_value : 7
Message from Client : b'PING sequence:2 1675157390.775175'        rand_value : 7
Message from Client : b'PING sequence:3 1675157390.776175'        rand_value : 2
Message from Client : b'PING sequence:4 1675157391.7778604'        rand_value : 4
Message from Client : b'PING sequence:5 1675157391.7778604'        rand_value : 4
Message from Client : b'PING sequence:6 1675157391.7778604'        rand_value : 9
Message from Client : b'PING sequence:7 1675157391.7778604'        rand_value : 0
Message from Client : b'PING sequence:8 1675157392.7823315'        rand_value : 3
Message from Client : b'PING sequence:9 1675157393.7953122'        rand_value : 1
Message from Client : b'PING sequence:10 1675157394.8082619'        rand_value : 10
```

UDPHeartbeatServer.py

```
# UDPPingerServer.py
# We will need the following module to generate randomized lost packets
import random

import time

import socket

# Create a UDP socket
# Notice the use of SOCK_DGRAM for UDP packets

SERVER = socket.gethostbyname(socket.gethostname())

PORT = 20001

HEADER = 1024

ADDR = (SERVER, PORT)

serverSocket = socket.socket(socket.AF_INET, socket.SOCK_DGRAM)

# Assign IP address and port number to socket

serverSocket.bind(ADDR)

print("UDP server up and listening")


while True:
    # Generate random number in the range of 0 to 10

    rand = random.randint(0, 2)

    # Receive the client packet along with the address it is coming from

    recv_time = time.time()

    message, address = serverSocket.recvfrom(HEADER)

    print(f"Message from Client : {message}")

    # Capitalize the message from the client

    # message = message.upper()

    send_time = float(message)
```

```
    time_diff = (recv_time - send_time)+rand

    # If rand is less is than 4, we consider the packet lost and do not
respond

    # if rand < 4:
    #     continue
    if time_diff > 1:
        print(f"Message from Client : {message}     RTT : {time_diff-
1}      Client Stopped running!")
        break

    # Otherwise, the server responds

    serverSocket.sendto(message, address)
```

UDPHeartbeatClient.py

```
import socket

import time

SERVER = socket.gethostbyname(socket.gethostname())

PORT = 20001

HEADER = 1024

ADDR = (SERVER, PORT)

# Create a UDP socket at client side

client = socket.socket(socket.AF_INET, socket.SOCK_DGRAM)

client.settimeout(1)

# Send to server using created UDP socket

rtts = []

packet_loss_rate = 0

for i in range(1, 11):

    send_time = time.time()

    msgFromClient = f"{send_time}"
```

```
    bytesToSend = str.encode(msgFromClient)

    client.sendto(bytesToSend, ADDR)

    try:
        msgFromServer = client.recvfrom(HEADER)

        recv_time = time.time()

        rtts.append(recv_time-send_time)


        msg = f"Message from Server {msgFromServer[0]}        Round Trip Time
: {recv_time-send_time}"

        print(msg)

    except TimeoutError:
        packet_loss_rate = packet_loss_rate+1
        print("Stopped!")
        break
```

Client Output:

```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL

PS C:\Users\Chandrashekhar Gouda\Networking> python -u "c:\Users\Chandrashekhar Gouda\Networking\UDP\UDPHeartbeatClient.py"
Message from Server b'1675161081.5597951'       Round Trip Time : 0.014647722244262695
Message from Server b'1675161081.575441'       Round Trip Time : 0.0
Message from Server b'1675161081.5764382'       Round Trip Time : 0.0
Message from Server b'1675161081.5764382'       Round Trip Time : 0.0010023117065429688
Message from Server b'1675161081.5774405'       Round Trip Time : 0.0010004043579101562
Message from Server b'1675161081.578441'       Round Trip Time : 0.0
Stopped!
PS C:\Users\Chandrashekhar Gouda\Networking>
```

Server Output:

```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL

PS C:\Users\Chandrashekhar Gouda\Networking> python -u "c:\Users\Chandrashekhar Gouda\Networking\UDP\UDPHeartbeatServer.py"
UDP server up and listening
Message from Client : b'1675161081.5597951'
Message from Client : b'1675161081.575441'
Message from Client : b'1675161081.5764382'
Message from Client : b'1675161081.5764382'
Message from Client : b'1675161081.5774405'
Message from Client : b'1675161081.578441'
Message from Client : b'1675161081.5794404'
Message from Client : b'1675161081.5794404'     RTT : 0.9990005493164062      Client Stopped running!
PS C:\Users\Chandrashekhar Gouda\Networking>
```