**Name: Archana Purushothama**

**Id: N15100298**

**SOCKET PROGRAMMING : UDP Pinger**

1. **Python code for the UDP Server**

# UDPPingerServer.py

# We will need the following module to generate randomized lost packets

import random

from socket import \*

# Create a UDP socket

# Notice the use of SOCK\_DGRAM for UDP packets

serverSocket = socket(AF\_INET, SOCK\_DGRAM)

# Assign IP address and port number to socket

serverSocket.bind(('', 12000))

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(1024)

# 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)

1. **Python Code for UDP Client**

#UDPPingerClient.py

from socket import \*

from datetime import datetime

import time

#Create a UDP Client Socket

clientSocket = socket(AF\_INET, SOCK\_DGRAM)

host = '127.0.0.1'

port = 12000

seqNumber = 1

while(seqNumber <= 10):

pingMsg = "Ping " + str(seqNumber) + " " + str(datetime.now())

#Ping Server and start the timer

tmrStart = time.time()

print pingMsg

clientSocket.sendto(pingMsg,(host, port))

#Set the timeout to 1second

clientSocket.settimeout(1)

try:

#Receive response from the server

respMsg = clientSocket.recvfrom(1024)

tmrEnd = time.time()

reply = respMsg[0]

addr = respMsg[1]

print "Server response: "+ str(reply)

print "RTT" + str(seqNumber) + " = " + str(tmrEnd-tmrStart) + "\n"

except:

print "Request timed out!!!\n"

seqNumber+= 1

print "\nThe End!! Client Socket is closed."

clientSocket.close()

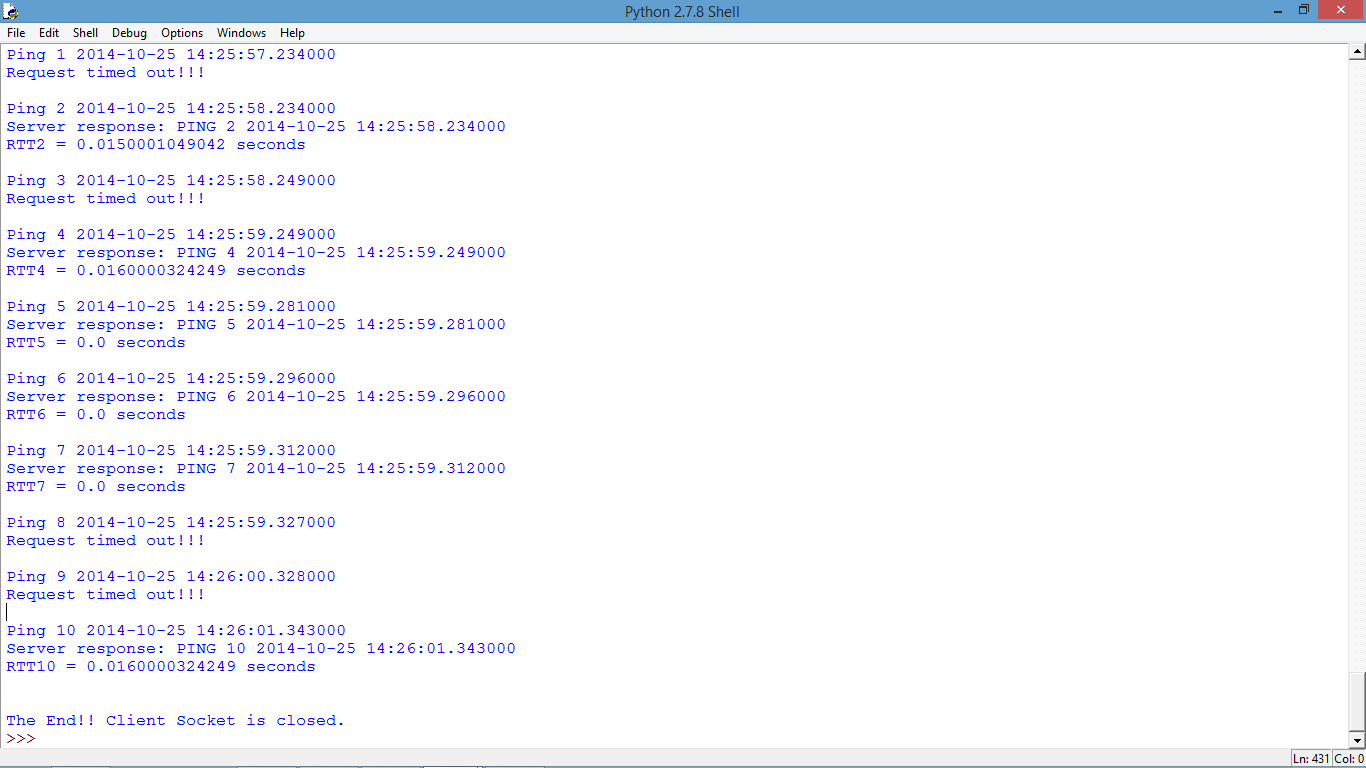


Figure 1: Output Screen of UDP Pinger.