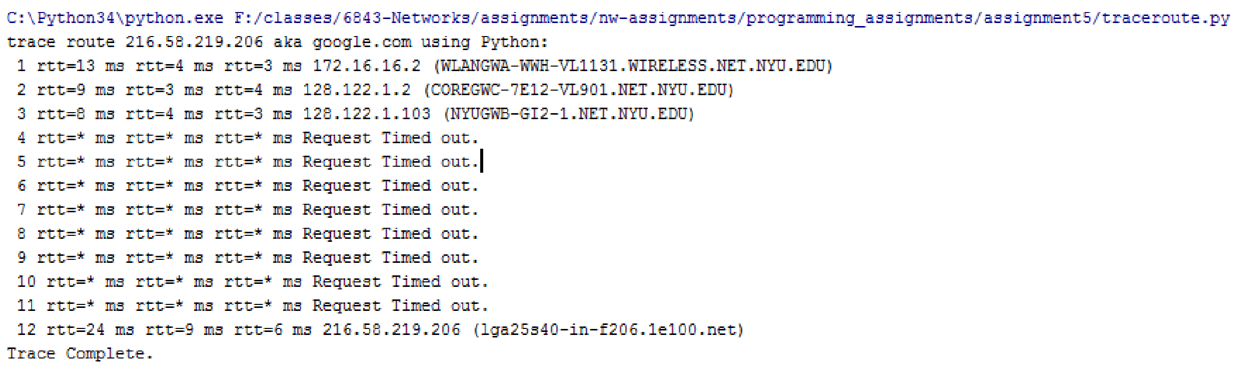
Assignment 5 – Socket Programming Assignment – ICMP Traceroute and Pinger

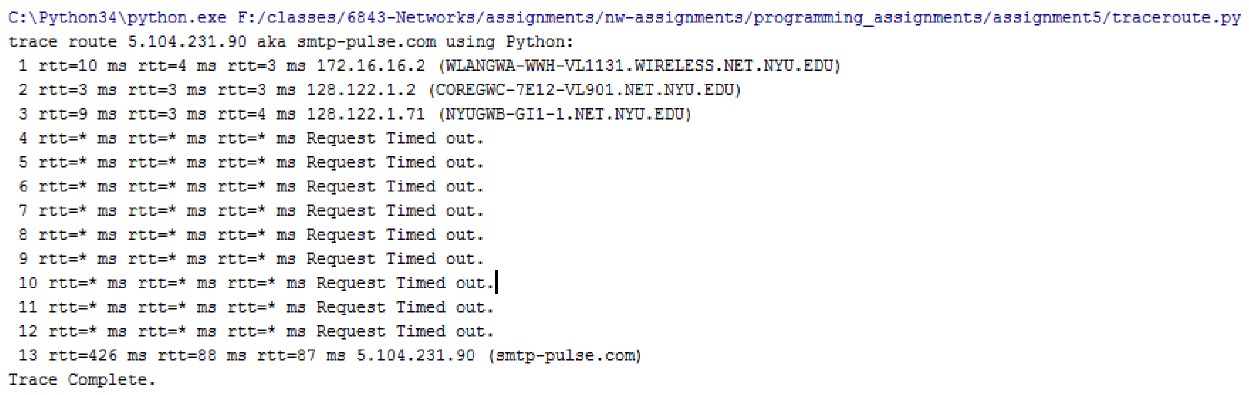
ICMP Trace Route

Output:



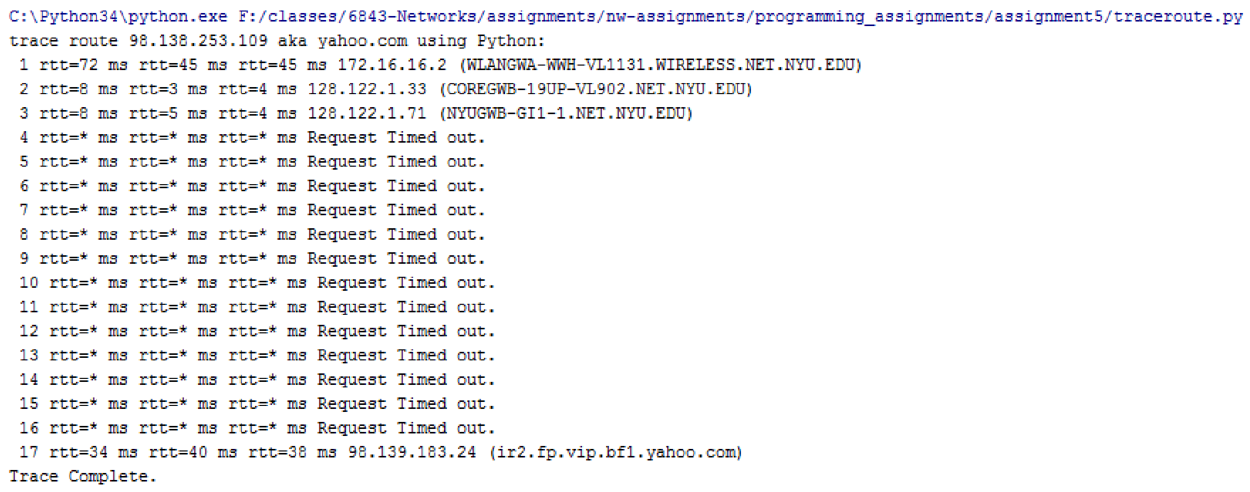
```
C:\Python34\python.exe F:/classes/6843-Networks/assignments/nw-assignments/programming_assignments/assignment5/traceroute.py
trace route 216.58.219.206 aka google.com using Python:
 1 rtt=13 ms rtt=4 ms rtt=3 ms 172.16.16.2 (WLANGWA-WWH-VL1131.WIRELESS.NET.NYU.EDU)
 2 rtt=9 ms rtt=3 ms rtt=4 ms 128.122.1.2 (COREGWC-7E12-VL901.NET.NYU.EDU)
 3 rtt=8 ms rtt=4 ms rtt=3 ms 128.122.1.103 (NYUGWB-GI2-1.NET.NYU.EDU)
 4 rtt=* ms rtt=* ms rtt=* ms Request Timed out.
 5 rtt=* ms rtt=* ms rtt=* ms Request Timed out.
 6 rtt=* ms rtt=* ms rtt=* ms Request Timed out.
 7 rtt=* ms rtt=* ms rtt=* ms Request Timed out.
 8 rtt=* ms rtt=* ms rtt=* ms Request Timed out.
 9 rtt=* ms rtt=* ms rtt=* ms Request Timed out.
 10 rtt=* ms rtt=* ms rtt=* ms Request Timed out.
 11 rtt=* ms rtt=* ms rtt=* ms Request Timed out.
 12 rtt=24 ms rtt=9 ms rtt=6 ms 216.58.219.206 (lga25s40-in-f206.1e100.net)
Trace Complete.
```

Traceroute "google.com"



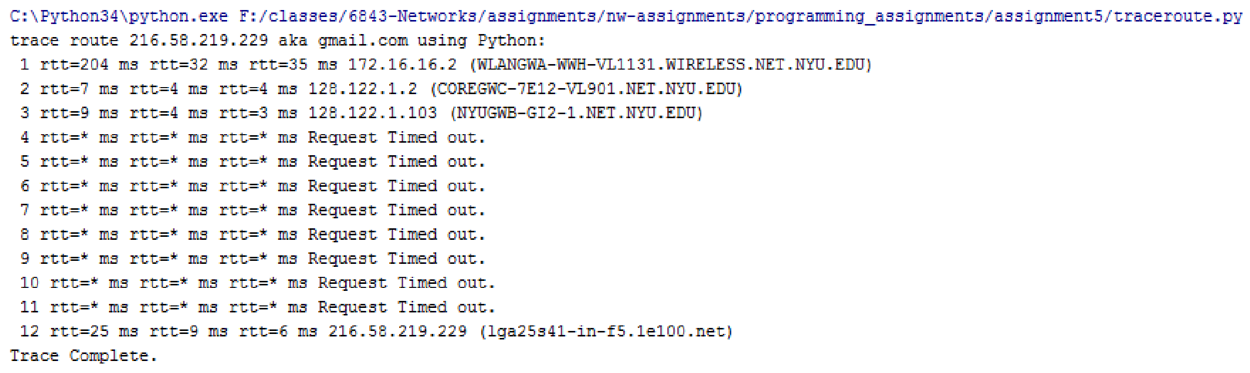
```
C:\Python34\python.exe F:/classes/6843-Networks/assignments/nw-assignments/programming_assignments/assignment5/traceroute.py
trace route 5.104.231.90 aka smtp-pulse.com using Python:
 1 rtt=10 ms rtt=4 ms rtt=3 ms 172.16.16.2 (WLANGWA-WWH-VL1131.WIRELESS.NET.NYU.EDU)
 2 rtt=3 ms rtt=3 ms rtt=3 ms 128.122.1.2 (COREGWC-7E12-VL901.NET.NYU.EDU)
 3 rtt=9 ms rtt=3 ms rtt=4 ms 128.122.1.71 (NYUGWB-GI1-1.NET.NYU.EDU)
 4 rtt=* ms rtt=* ms rtt=* ms Request Timed out.
 5 rtt=* ms rtt=* ms rtt=* ms Request Timed out.
 6 rtt=* ms rtt=* ms rtt=* ms Request Timed out.
 7 rtt=* ms rtt=* ms rtt=* ms Request Timed out.
 8 rtt=* ms rtt=* ms rtt=* ms Request Timed out.
 9 rtt=* ms rtt=* ms rtt=* ms Request Timed out.
 10 rtt=* ms rtt=* ms rtt=* ms Request Timed out.
 11 rtt=* ms rtt=* ms rtt=* ms Request Timed out.
 12 rtt=* ms rtt=* ms rtt=* ms Request Timed out.
 13 rtt=426 ms rtt=88 ms rtt=87 ms 5.104.231.90 (smtp-pulse.com)
Trace Complete.
```

Traceroute "smtp-pulse.com"



```
C:\Python34\python.exe F:/classes/6843-Networks/assignments/nw-assignments/programming_assignments/assignment5/traceroute.py
trace route 98.138.253.109 aka yahoo.com using Python:
 1 rtt=72 ms rtt=45 ms rtt=45 ms 172.16.16.2 (WLANGWA-WWH-VL1131.WIRELESS.NET.NYU.EDU)
 2 rtt=8 ms rtt=3 ms rtt=4 ms 128.122.1.33 (COREGWB-19UP-VL902.NET.NYU.EDU)
 3 rtt=8 ms rtt=5 ms rtt=4 ms 128.122.1.71 (NYUGWB-GI1-1.NET.NYU.EDU)
 4 rtt=* ms rtt=* ms rtt=* ms Request Timed out.
 5 rtt=* ms rtt=* ms rtt=* ms Request Timed out.
 6 rtt=* ms rtt=* ms rtt=* ms Request Timed out.
 7 rtt=* ms rtt=* ms rtt=* ms Request Timed out.
 8 rtt=* ms rtt=* ms rtt=* ms Request Timed out.
 9 rtt=* ms rtt=* ms rtt=* ms Request Timed out.
 10 rtt=* ms rtt=* ms rtt=* ms Request Timed out.
 11 rtt=* ms rtt=* ms rtt=* ms Request Timed out.
 12 rtt=* ms rtt=* ms rtt=* ms Request Timed out.
 13 rtt=* ms rtt=* ms rtt=* ms Request Timed out.
 14 rtt=* ms rtt=* ms rtt=* ms Request Timed out.
 15 rtt=* ms rtt=* ms rtt=* ms Request Timed out.
 16 rtt=* ms rtt=* ms rtt=* ms Request Timed out.
 17 rtt=34 ms rtt=40 ms rtt=38 ms 98.139.183.24 (ir2.fp.vip.bf1.yahoo.com)
Trace Complete.
```

Traceroute "yahoo.com"



```
C:\Python34\python.exe F:/classes/6843-Networks/assignments/nw-assignments/programming_assignments/assignment5/traceroute.py
trace route 216.58.219.229 aka gmail.com using Python:
 1 rtt=204 ms rtt=32 ms rtt=35 ms 172.16.16.2 (WLANGWA-WWH-VL1131.WIRELESS.NET.NYU.EDU)
 2 rtt=7 ms rtt=4 ms rtt=4 ms 128.122.1.2 (COREGWC-7E12-VL901.NET.NYU.EDU)
 3 rtt=9 ms rtt=4 ms rtt=3 ms 128.122.1.103 (NYUGWB-GI2-1.NET.NYU.EDU)
 4 rtt=* ms rtt=* ms rtt=* ms Request Timed out.
 5 rtt=* ms rtt=* ms rtt=* ms Request Timed out.
 6 rtt=* ms rtt=* ms rtt=* ms Request Timed out.
 7 rtt=* ms rtt=* ms rtt=* ms Request Timed out.
 8 rtt=* ms rtt=* ms rtt=* ms Request Timed out.
 9 rtt=* ms rtt=* ms rtt=* ms Request Timed out.
 10 rtt=* ms rtt=* ms rtt=* ms Request Timed out.
 11 rtt=* ms rtt=* ms rtt=* ms Request Timed out.
 12 rtt=25 ms rtt=9 ms rtt=6 ms 216.58.219.229 (lga25s41-in-f5.1e100.net)
Trace Complete.
```

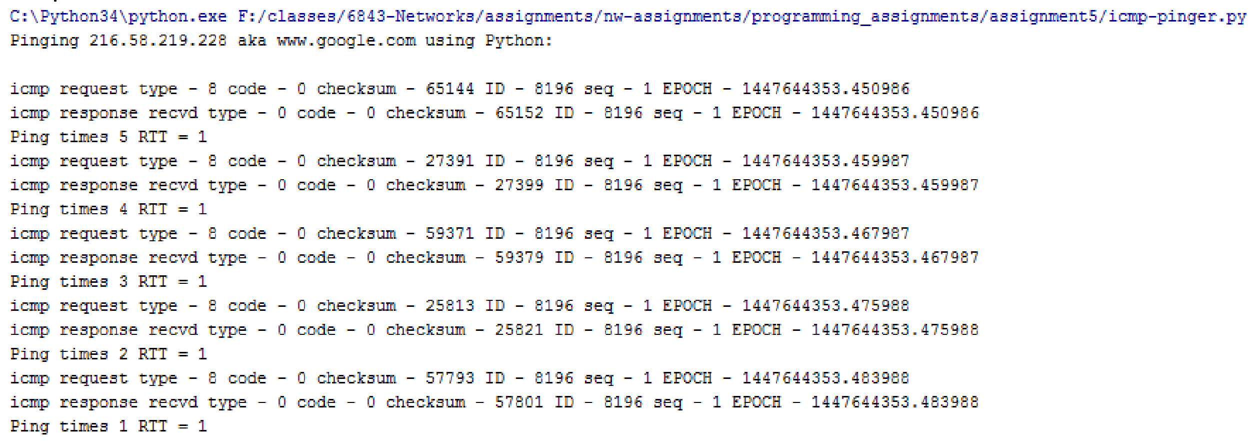
Traceroute “gmail.com”

Code – **traceroute.py**

```
from socket import *
import os
import sys
import struct
import time
import select
import binascii
import socket
import ctypes

MAX_HOPS = 30
TIMEOUT = 2.0
TRIES = 2
ICMP_ECHO_REQUEST = 8
ICMP_ECHO_REPLY = 0
ICMP_ECHO_REQUEST_CODE = 0
ICMP_ECHO_REPLY_CODE = 0

def checksum(str):
    #print(str)
    csum = 0
    countTo = (len(str) / 2) * 2
    count = 0
    while count < countTo:
        thisVal = str[count+1] * 256 + str[count]
        csum = csum + thisVal
        csum = csum & 0xffffffff
        count = count + 2
    if countTo < len(str):
        csum = csum + ord(str[len(str) - 1])
        csum = csum & 0xffffffff
        csum = (csum >> 16) + (csum & 0xffff)
    csum = csum + (csum >> 16)
    answer = ~csum
    answer = answer & 0xffff
    answer = answer >> 8 | (answer << 8 & 0xff00)
    return answer

def build_packet():
    # Header is type (8), code (8), checksum (16), id (16),
sequence (16)
    myChecksum = 0
    ID = os.getpid() & 0xFFFF
    # Make a dummy header with a 0 checksum.
    # struct -- Interpret strings as packed binary data
    header = struct.pack("bbHHh", ICMP_ECHO_REQUEST,
ICMP_ECHO_REQUEST_CODE, myChecksum, ID, 1)
    data = struct.pack("d",time.time() )
    # Calculate the checksum on the data and the dummy header.
    myChecksum = checksum(header + data)

    # Get the right checksum, and put in the header
    if sys.platform == 'darwin':
        myChecksum = socket.htons(myChecksum) & 0xffff
        #Convert 16-bit integers from host to network byte
order.
    else:
        myChecksum = socket.htons(myChecksum)
    header = struct.pack("bbHHh", ICMP_ECHO_REQUEST,
ICMP_ECHO_REQUEST_CODE, myChecksum, ID, 1)
    packet = header + data
    return packet

def get_route(hostname):
    timeLeft = TIMEOUT
    done =0
    print ("trace route " + socket.gethostbyname(hostname) + '
aka '+hostname+" using Python:")
    for ttl in range(1,MAX_HOPS+1):#xrange is removed from
python 3.0
        timeTaken =[-1,-1,-1]
        name=''
        for tries in range(0,TRIES+1):#xrange is removed from
python 3.0
            #Fill in start
            # Make a raw socket named mySocket
            destAddr = socket.gethostbyname(hostname)
            icmp = socket.getprotobyname("icmp")
            mySocket
=socket.socket(socket.AF_INET,socket.SOCK_RAW,icmp)
            mySocket.setsockopt(IPPROTO_IP, IP_TTL,
struct.pack('I', ttl))
            mySocket.settimeout(TIMEOUT)
            #Fill in end
            try:
                d = build_packet()
                mySocket.sendto(d, (hostname, 0))
                t= time.time()
                startedSelect = time.time()
                whatReady = select.select([mySocket], [], [],
timeLeft)
                howLongInSelect = (time.time() - startedSelect)
                recvPacket, addr = mySocket.recvfrom(1024)
                try:
                    host = socket.gethostbyaddr(addr[0])
                    name  = '{0} ({1})'.format(addr[0] ,
host[0])
                except Exception:
                    name  = '{0} (host name/IP not
found)'.format(addr[0])
                timeReceived = time.time()
                timeLeft = timeLeft - howLongInSelect
            except socket.timeout:
                continue
            else:
                #Fill in start
                # Fetch the icmp type from the IP packet
                icmpHeaderContent = recvPacket[20:28]
                type, code, checksum, packetID, sequence =
struct.unpack("bbHHh", icmpHeaderContent)
                #Fill in end
                if type == 11:
                    bytes = struct.calcsize("d")
                    timeSent = struct.unpack("d",
recvPacket[28:28 + bytes])[0]
                    timeTaken[tries] = (timeReceived -t)*1000
                    #print (" %d rtt=%.0f ms %s" %(ttl,
(timeReceived -t)*1000,
addr[0]),socket.gethostbyaddr(addr[0])[0])
                elif type == 3:
                    bytes = struct.calcsize("d")
                    timeSent = struct.unpack("d",
recvPacket[28:28 + bytes])[0]
                    timeTaken[tries] =(timeReceived-t)*1000
                    #print (" %d rtt=%.0f ms %s" %(ttl,
(timeReceived-t)*1000,
addr[0]),socket.gethostbyaddr(addr[0])[0])
                elif type == 0:
                    bytes = struct.calcsize("d")
                    timeSent = struct.unpack("d",
recvPacket[28:28 + bytes])[0]
                    timeTaken[tries]=(timeReceived -
timeSent)*1000
                    #print (" %d rtt=%.0f ms %s" %(ttl,
(timeReceived - timeSent)*1000,
addr[0]),socket.gethostbyaddr(addr[0])[0])
                    done = 1
                else:
                    print ("error")
                    break
            finally:
                mySocket.close()
        i=0
        a= ['*','*','*']
        if timeTaken[i] != -1:
            a[i] = int(timeTaken[i])
        if timeTaken[i+1] != -1:
            a[i+1] = int(timeTaken[i+1])
        if timeTaken[i+2] != -1:
            a[i+2] = int(timeTaken[i+2])
        if timeTaken[i] == timeTaken[i+1] == timeTaken[i+2] == -
1:
            print(" %d rtt=%s ms rtt=%s ms rtt=%s ms %s" %(ttl,
a[i],a[i+1],a[i+2] ,'Request Timed out.'))
        else:
            print(" %d rtt=%s ms rtt=%s ms rtt=%s ms %s" %(ttl,
a[i],a[i+1],a[i+2] ,name))
        if done == 1:
            print('Trace Complete.')
            return

get_route("google.com")
get_route("smtp-pulse.com")
get_route("yahoo.com")
get_route("gmail.com")
```

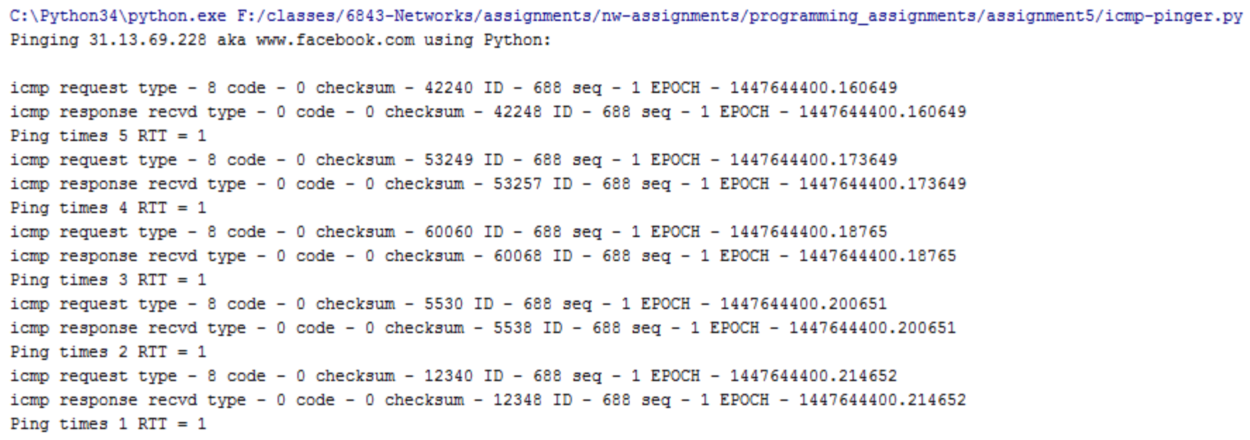
ICMP Pinger

Output:



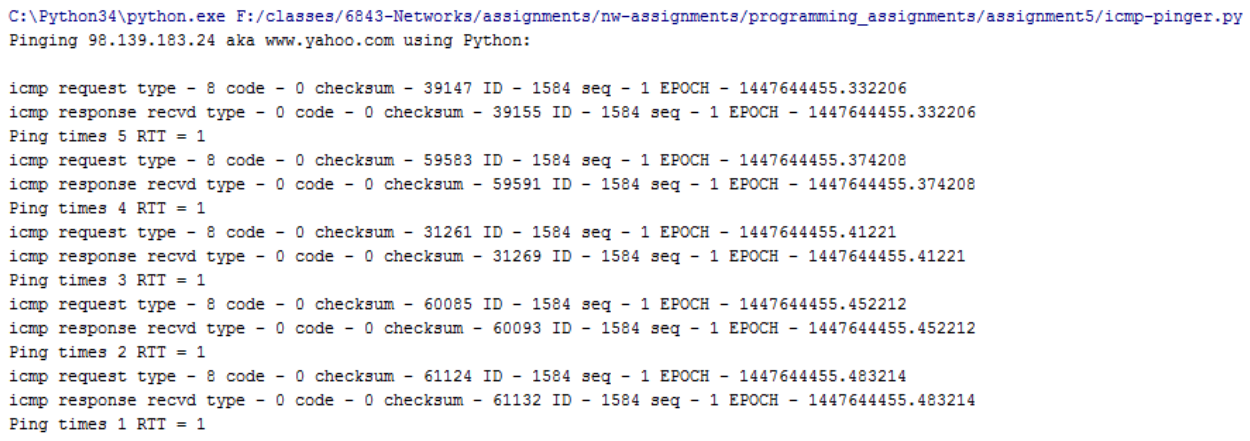
```
C:\Python34\python.exe F:/classes/6843-Networks/assignments/nw-assignments/programming_assignments/assignment5/icmp-pinger.py
Pinging 216.58.219.228 aka www.google.com using Python:

icmp request type - 8 code - 0 checksum - 65144 ID - 8196 seq - 1 EPOCH - 1447644353.450986
icmp response recvd type - 0 code - 0 checksum - 65152 ID - 8196 seq - 1 EPOCH - 1447644353.450986
Ping times 5 RTT = 1
icmp request type - 8 code - 0 checksum - 27391 ID - 8196 seq - 1 EPOCH - 1447644353.459987
icmp response recvd type - 0 code - 0 checksum - 27399 ID - 8196 seq - 1 EPOCH - 1447644353.459987
Ping times 4 RTT = 1
icmp request type - 8 code - 0 checksum - 59371 ID - 8196 seq - 1 EPOCH - 1447644353.467987
icmp response recvd type - 0 code - 0 checksum - 59379 ID - 8196 seq - 1 EPOCH - 1447644353.467987
Ping times 3 RTT = 1
icmp request type - 8 code - 0 checksum - 25813 ID - 8196 seq - 1 EPOCH - 1447644353.475988
icmp response recvd type - 0 code - 0 checksum - 25821 ID - 8196 seq - 1 EPOCH - 1447644353.475988
Ping times 2 RTT = 1
icmp request type - 8 code - 0 checksum - 57793 ID - 8196 seq - 1 EPOCH - 1447644353.483988
icmp response recvd type - 0 code - 0 checksum - 57801 ID - 8196 seq - 1 EPOCH - 1447644353.483988
Ping times 1 RTT = 1
```

ping "google.com"



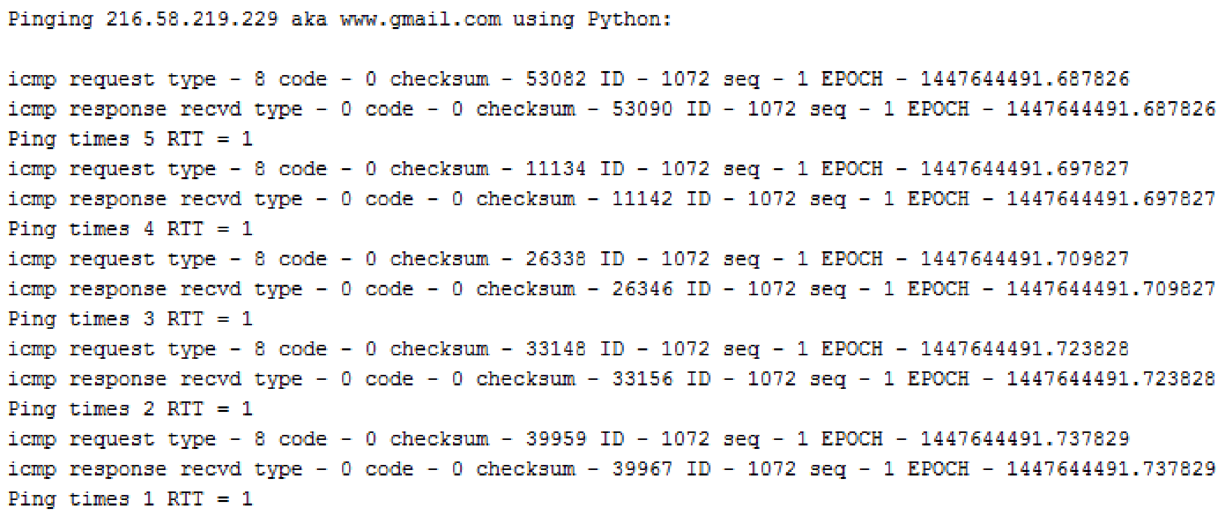
```
C:\Python34\python.exe F:/classes/6843-Networks/assignments/nw-assignments/programming_assignments/assignment5/icmp-pinger.py
Pinging 31.13.69.228 aka www.facebook.com using Python:

icmp request type - 8 code - 0 checksum - 42240 ID - 688 seq - 1 EPOCH - 1447644400.160649
icmp response recvd type - 0 code - 0 checksum - 42248 ID - 688 seq - 1 EPOCH - 1447644400.160649
Ping times 5 RTT = 1
icmp request type - 8 code - 0 checksum - 53249 ID - 688 seq - 1 EPOCH - 1447644400.173649
icmp response recvd type - 0 code - 0 checksum - 53257 ID - 688 seq - 1 EPOCH - 1447644400.173649
Ping times 4 RTT = 1
icmp request type - 8 code - 0 checksum - 60060 ID - 688 seq - 1 EPOCH - 1447644400.18765
icmp response recvd type - 0 code - 0 checksum - 60068 ID - 688 seq - 1 EPOCH - 1447644400.18765
Ping times 3 RTT = 1
icmp request type - 8 code - 0 checksum - 5530 ID - 688 seq - 1 EPOCH - 1447644400.200651
icmp response recvd type - 0 code - 0 checksum - 5538 ID - 688 seq - 1 EPOCH - 1447644400.200651
Ping times 2 RTT = 1
icmp request type - 8 code - 0 checksum - 12340 ID - 688 seq - 1 EPOCH - 1447644400.214652
icmp response recvd type - 0 code - 0 checksum - 12348 ID - 688 seq - 1 EPOCH - 1447644400.214652
Ping times 1 RTT = 1
```

ping "facebook.com"



```
C:\Python34\python.exe F:/classes/6843-Networks/assignments/nw-assignments/programming_assignments/assignment5/icmp-pinger.py
Pinging 98.139.183.24 aka www.yahoo.com using Python:

icmp request type - 8 code - 0 checksum - 39147 ID - 1584 seq - 1 EPOCH - 1447644455.332206
icmp response recvd type - 0 code - 0 checksum - 39155 ID - 1584 seq - 1 EPOCH - 1447644455.332206
Ping times 5 RTT = 1
icmp request type - 8 code - 0 checksum - 59583 ID - 1584 seq - 1 EPOCH - 1447644455.374208
icmp response recvd type - 0 code - 0 checksum - 59591 ID - 1584 seq - 1 EPOCH - 1447644455.374208
Ping times 4 RTT = 1
icmp request type - 8 code - 0 checksum - 31261 ID - 1584 seq - 1 EPOCH - 1447644455.41221
icmp response recvd type - 0 code - 0 checksum - 31269 ID - 1584 seq - 1 EPOCH - 1447644455.41221
Ping times 3 RTT = 1
icmp request type - 8 code - 0 checksum - 60085 ID - 1584 seq - 1 EPOCH - 1447644455.452212
icmp response recvd type - 0 code - 0 checksum - 60093 ID - 1584 seq - 1 EPOCH - 1447644455.452212
Ping times 2 RTT = 1
icmp request type - 8 code - 0 checksum - 61124 ID - 1584 seq - 1 EPOCH - 1447644455.483214
icmp response recvd type - 0 code - 0 checksum - 61132 ID - 1584 seq - 1 EPOCH - 1447644455.483214
Ping times 1 RTT = 1
```

ping "yahoo.com"



```
Pinging 216.58.219.229 aka www.gmail.com using Python:

icmp request type - 8 code - 0 checksum - 53082 ID - 1072 seq - 1 EPOCH - 1447644491.687826
icmp response recvd type - 0 code - 0 checksum - 53090 ID - 1072 seq - 1 EPOCH - 1447644491.687826
Ping times 5 RTT = 1
icmp request type - 8 code - 0 checksum - 11134 ID - 1072 seq - 1 EPOCH - 1447644491.697827
icmp response recvd type - 0 code - 0 checksum - 11142 ID - 1072 seq - 1 EPOCH - 1447644491.697827
Ping times 4 RTT = 1
icmp request type - 8 code - 0 checksum - 26338 ID - 1072 seq - 1 EPOCH - 1447644491.709827
icmp response recvd type - 0 code - 0 checksum - 26346 ID - 1072 seq - 1 EPOCH - 1447644491.709827
Ping times 3 RTT = 1
icmp request type - 8 code - 0 checksum - 33148 ID - 1072 seq - 1 EPOCH - 1447644491.723828
icmp response recvd type - 0 code - 0 checksum - 33156 ID - 1072 seq - 1 EPOCH - 1447644491.723828
Ping times 2 RTT = 1
icmp request type - 8 code - 0 checksum - 39959 ID - 1072 seq - 1 EPOCH - 1447644491.737829
icmp response recvd type - 0 code - 0 checksum - 39967 ID - 1072 seq - 1 EPOCH - 1447644491.737829
Ping times 1 RTT = 1
```

ping “gmail.com”

Code- **icmp-pinger.py**

```
from socket import *
import os
import sys
import struct
import time
import select
import binascii
import socket
import ctypes
import math

ICMP_ECHO_REQUEST = 8
ICMP_ECHO_REPLY = 0
ICMP_ECHO_REQUEST_CODE = 0
ICMP_ECHO_REPLY_CODE = 0

def checksum(str):
    #print(str)
    csum = 0
    countTo = (len(str) / 2) * 2
    count = 0
    while count < countTo:
        thisVal = str[count+1] * 256 + str[count]
        csum = csum + thisVal
        csum = csum & 0xffffffff
        count = count + 2
    if countTo < len(str):
        csum = csum + ord(str[len(str) - 1])
        csum = csum & 0xffffffff
        csum = (csum >> 16) + (csum & 0xffff)
    csum = csum + (csum >> 16)
    answer = ~csum
    answer = answer & 0xffff
    answer = answer >> 8 | (answer << 8 & 0xff00)
    return answer

def receiveOnePing(mySocket, ID, timeout, destAddr):
    timeLeft = timeout
    while 1:
        timeLeft = timeout
        startedSelect = time.time()
        whatReady = select.select([mySocket], [], [], timeLeft)
        howLongInSelect = (time.time() - startedSelect)
        #if whatReady[0] == []: # Timeout
            #return "#Request timed out."
        timeReceived = time.time()
        recPacket, addr = mySocket.recvfrom(1024)

        #Fill in start
        #Fetch the ICMP header from the IP packet
        psize= len(recPacket)
        #advance to 128th byte coz thats where ICMP starts
        icmp_header = struct.unpack("bbHHhd",recPacket[psize-
16:])
        print('icmp response recvd type -
'+str(icmp_header[0])+' code - '+str(icmp_header[1])+
            ' checksum - '+str(icmp_header[2])+' ID -
'+str(icmp_header[3])+' seq - '+str(icmp_header[4])+' EPOCH -
'+str(icmp_header[5]))
        #Fill in end

        timeLeft = timeLeft - howLongInSelect
        timeLeft = math.ceil(timeLeft)
        if timeLeft <= 0:
            return "#Request timed out."
        else:
            return timeLeft

def sendOnePing(mySocket, destAddr, ID):
    # Header is type (8), code (8), checksum (16), id (16),
sequence (16)
    myChecksum = 0
    # Make a dummy header with a 0 checksum.
    # struct -- Interpret strings as packed binary data
    header = struct.pack("bbHHh", ICMP_ECHO_REQUEST,
ICMP_ECHO_REQUEST_CODE, myChecksum, ID, 1)
    tim = time.time()
    data = struct.pack("d",tim )
    # Calculate the checksum on the data and the dummy header.
    myChecksum = checksum(header + data)

    # Get the right checksum, and put in the header
    if sys.platform == 'darwin':
        myChecksum = socket.htons(myChecksum) & 0xffff
        #Convert 16-bit integers from host to network byte
order.
    else:
        myChecksum = socket.htons(myChecksum)
    header = struct.pack("bbHHh", ICMP_ECHO_REQUEST,
ICMP_ECHO_REQUEST_CODE, myChecksum, ID, 1)
    packet = header + data
    print('icmp request type - '+str(ICMP_ECHO_REQUEST)+' code -
'+str(ICMP_ECHO_REQUEST_CODE)+
         ' checksum - '+str(myChecksum)+' ID - '+str(ID)+' seq
- '+str(1)+' EPOCH - '+str(tim))
    mySocket.sendto(packet, (destAddr, 1)) # AF_INET address
must be tuple, not str
    #Both LISTS and TUPLES consist of a number of objects
    #which can be referenced by their position number within the
object

def doOnePing(destAddr, timeout):
    icmp = socket.getprotobyname("icmp")
    #SOCK_RAW is a powerful socket type. For more details see:
http://sock-raw.org/papers/sock_raw
    #Fill in start
    #Create Socket here
    mySocket =
socket.socket(socket.AF_INET,socket.SOCK_RAW,icmp)
    #Fill in end
    myID = os.getpid() & 0xFFFF #Return the current process i
    sendOnePing(mySocket, destAddr, myID)
    delay = receiveOnePing(mySocket, myID, timeout, destAddr)
    mySocket.close()
    return delay

def ping(host, timeout=1):
    #timeout=1 means: If one second goes by without a reply from
the server,
    #the client assumes that either the client’s ping or the
server’s pong is lost
    dest = socket.gethostbyname(host)
    print ("Pinging " + dest + ' aka '+host+" using Python:")
    print ("")
    #Send ping requests to a server separated by approximately
one second
    i = 5
    while i :
        delay = doOnePing(dest, timeout)
        print('Ping times '+str(i)+' RTT = '+str(delay))
        i-=1
    return delay

ping("www.google.com")
ping("www.facebook.com")
ping("www.yahoo.com")
ping("www.gmail.com")
```