Procedure for running the python UDP and TCP client and server

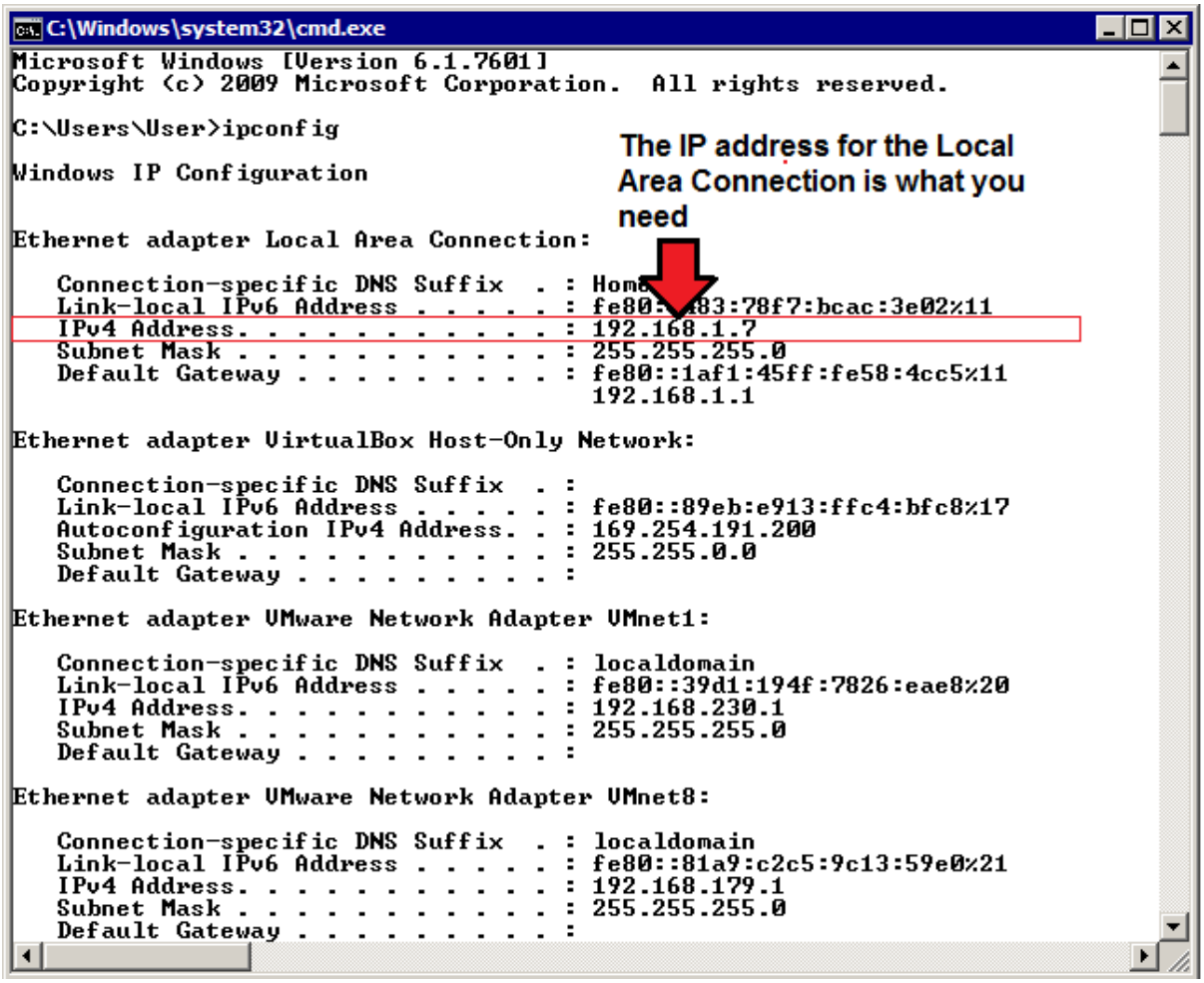
Note that the Linux machines are running python 2 and the Windows machines are running python 3, so there are some minor differences between the programs for the two operating systems.

On Windows:

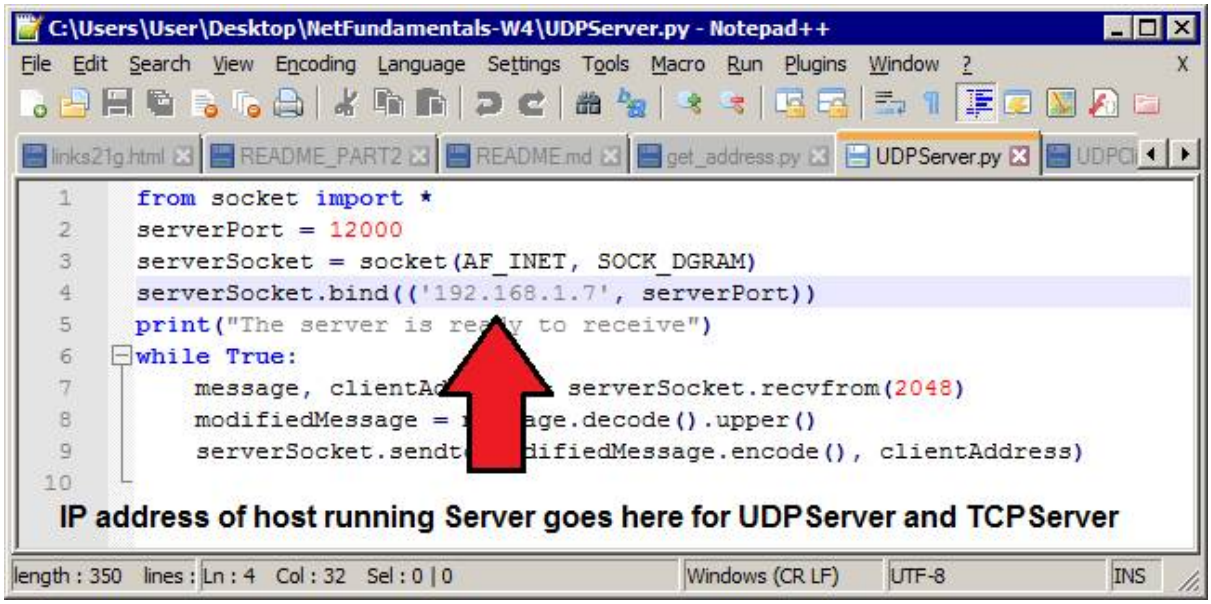
(1) Copy and paste the code from the end of this document into four separate files in an editor such as Notepad++ : TCPServer.py, UDPServer.py, TCPClient.py and UDPClinet.py

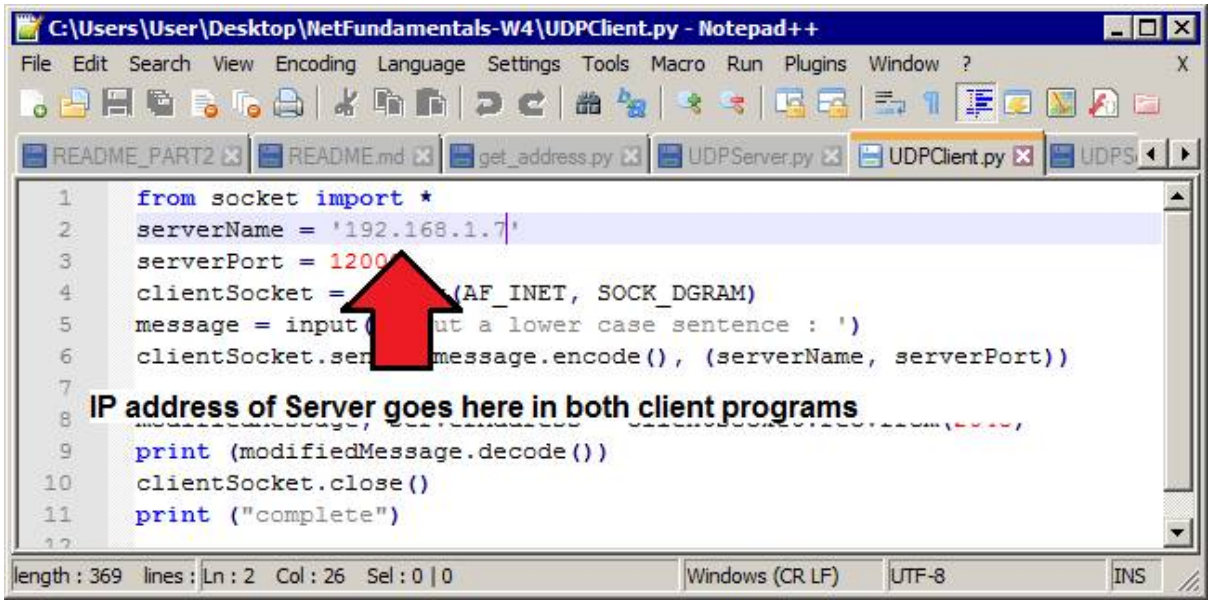
(2) Open a command line window by typing cmd into the search box on windows and pressing <ENTER>

(3) In the command window run the command ipconfig to work out what your IP address is. Note that there may be a lot of output which may scroll off the screen, so you may need to scroll back.



(4) Put the correct IP address into the right section for each of the four files.





(5) Create a directory called NetFundamentals-W4 on your windows desktop. Save each of the four files into this directory.

(6) Open a command line window and execute the following commands in the window:

```
cd Desktop

cd NetFundamentals-W4

python UDPServer.py
```

This will start the UDP server running. See the screenshot on the next page.

```
C:\Windows\system32\cmd.exe - python UDPServer.py
Microsoft Windows [Version 6.1.7601]
Copyright (c) 2009 Microsoft Corporation.  All rights reserved.

C:\Users\User>cd Desktop

C:\Users\User\Desktop>cd NetFundamentals-W4

C:\Users\User\Desktop\NetFundamentals-W4>python UDPServer.py
The server is ready to receive
```

(7) Open another command line window and navigate to the NetFundamentals-W4 directory

Enter the command

```
python UDPClient.py
```

The program will prompt you for a string of characters. Enter all lower case characters and press <ENTER>.

If everything has been properly configured you string should be returned in all upper case.

Here are some screenshots of a successful client session.

```
C:\Windows\system32\cmd.exe
Microsoft Windows [Version 6.1.7601]
Copyright (c) 2009 Microsoft Corporation.  All rights reserved.

C:\Users\User>cd Desktop

C:\Users\User\Desktop>cd NetFundamentals-W4

C:\Users\User\Desktop\NetFundamentals-W4>python UDPClient.py
Input a lower case sentence : test this string!
TEST THIS STRING!
complete

C:\Users\User\Desktop\NetFundamentals-W4>python UDPClient.py
Input a lower case sentence : hello Network Fundamentals World
HELLO NETWORK FUNDAMENTALS WORLD
complete

C:\Users\User\Desktop\NetFundamentals-W4>_
```

The procedure is the same for setting up and running the TCPServer and client.

On Linux:

Note : Unlike the windows programs, the linux code supplied here has been configured to run as a standalone scrypt.

(1) Logon to a linux machine in the lab.

(2) Open a terminal by going to Applications->System->Terminal

(3) Work out your IP address. This can be done by running the following command in nthe terminal:

```
/sbin/ifconfig eth0
```

(4) Create a directory called NetFundamentals-W4 or something similar.

(5) Using an editor such as gedit copy and paste the four files into separate windows.

(6) Enter the IP address from step (3) in the same places in the code as shown for windows.

(7) Open a terminal and using the cd command go to the directory containing the python programs.

(8) Set the permissions on the programs by running the command

```
chmod u+x *py
```

(9) Run UDPServer.py. In another terminal run UDPClient.py The procedure is identical to that for running with windows and the results should be the same.

(10) Open two more terminals, navigate to the correct directory and run TCPServer.py and TCPClient.py Again the results should be the same as for windows.

Extra Tasks

(1) Modify the Servers so they flag all client accesses and/or print the string they receive from the client.

(2) Find out the IP addresses of the machine next to you. Modify your server code so that it can access servers running on those machines. You could hard code in the server addresses or pass the data as part of the command line or have the program prompt the user for the IP address.

(3) By default the two servers use port 12000. Change the code so the port can be set from the command line. (Use a port number above 1024)

Windows UDPClient.py

```
from socket import *
serverName = '192.168.1.7'
serverPort = 12000
clientSocket = socket(AF_INET, SOCK_DGRAM)
message = input('Input a lower case sentence : ')
clientSocket.sendto(message.encode(), (serverName, serverPort))
modifiedMessage, serverAddress = clientSocket.recvfrom(2048)
print (modifiedMessage.decode())
clientSocket.close()
print ("UDP client completed - exiting")
```

Windows UDPServer.py

```
from socket import *
serverPort = 12000
serverSocket = socket(AF_INET, SOCK_DGRAM)
serverSocket.bind(('192.168.1.7', serverPort))
print("The server is ready to receive")
while True:
      message, clientAddress = serverSocket.recvfrom(2048)
      modifiedMessage = message.decode().upper()
      serverSocket.sendto(modifiedMessage.encode(), clientAddress)
```

WindowsTCPClient.py

```
from socket import *
serverName = '192.168.1.7'
serverPort = 12000
clientSocket = socket(AF_INET, SOCK_STREAM)
clientSocket.connect((serverName,serverPort))
sentence = input('Input a lower case sentence : ')
clientSocket.send(sentence.encode())
```

```
modifiedSentence = clientSocket.recv(1024)
print ('From Server : ',modifiedSentence.decode())
clientSocket.close()
print ("TCP client completed - exiting")
```

WindowsTCPServer.py

```
rom socket import *
serverPort = 12000
serverSocket = socket(AF_INET, SOCK_STREAM)
serverSocket.bind(('192.168.1.7', serverPort))
serverSocket.listen(1)
print("The server is ready to receive")
while True:
      connectionSocket, addr = serverSocket.accept()
      sentence = connectionSocket.recv(1024).decode()
      capitalizedSentence = sentence.upper()
      connectionSocket.send(capitalizedSentence.encode())
      connectionSocket.close()
```

Linux UDPClient.py

```
#!/usr/local/bin/python

from socket import *

serverName = '138.25.216.172'

serverPort = 12000

clientSocket = socket(AF_INET, SOCK_DGRAM)

message = raw_input('Input a lower case sentence : ')

clientSocket.sendto(message.encode(), (serverName, serverPort))

modifiedMessage, serverAddress = clientSocket.recvfrom(2048)

print (modifiedMessage.decode())

clientSocket.close()

print "UDP client completed - exiting"
```

Linux UDPServer.py

```
#!/usr/local/bin/python

from socket import *

serverPort = 12000

serverSocket = socket(AF_INET, SOCK_DGRAM)

serverSocket.bind(('138.25.216.172', serverPort))

print("The server is ready to receive")

while True:

      message, clientAddress = serverSocket.recvfrom(2048)

      modifiedMessage = message.decode().upper()

      serverSocket.sendto(modifiedMessage.encode(), clientAddress)
```

Linux TCPClient.py

```
#!/usr/local/bin/python

from socket import *

serverName = '138.25.216.172'

serverPort = 12000
```

```
clientSocket = socket(AF_INET, SOCK_DGRAM)
message = raw_input('Input a lower case sentence : ')
clientSocket.sendto(message.encode(), (serverName, serverPort))
modifiedMessage, serverAddress = clientSocket.recvfrom(2048)
print (modifiedMessage.decode())
clientSocket.close()
print "UDP client completed - exiting"
```

Linux TCPServer.py

```
#!/usr/local/bin/python
from socket import *
serverPort = 12000
serverSocket = socket(AF_INET, SOCK_STREAM)
serverSocket.bind(('138.25.216.172', serverPort))
serverSocket.listen(1)
print("The server is ready to receive")
while True:
      connectionSocket, addr = serverSocket.accept()
      sentence = connectionSocket.recv(1024).decode()
      capitalizedSentence = sentence.upper()
      connectionSocket.send(capitalizedSentence.encode())
      connectionSocket.close()
```