CSCE 560
Socket Programming Assignment 1
TCP Client and Server
Fall 17

Assigned: 18 Oct
Due: 6 Nov, 1400

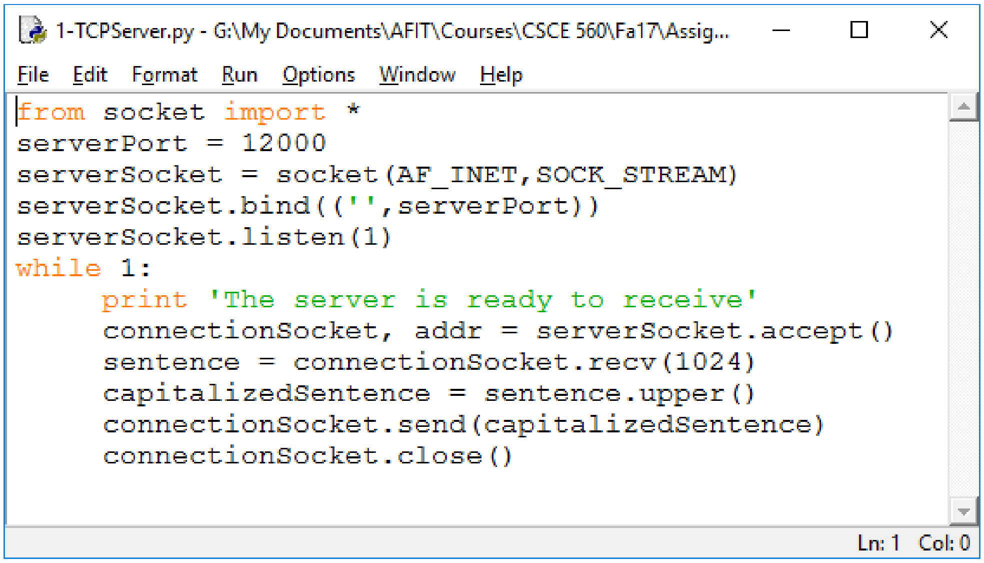
In this lab, you will learn the basics of socket programming for TCP connections in Python: how to create a socket, bind it to a specific address and port, as well as send and receive data. You will implement the TCP client and server found in the text; the code is on the L drive.

Install Python

Install Python on your computer using the instructions in the file `0-GettingStarted.pdf`.

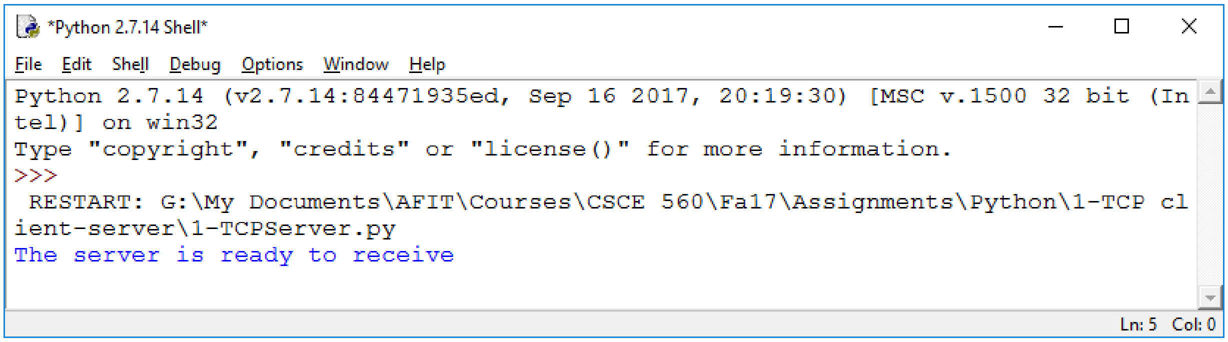
Running the Server

1. Determine the IP address (e.g., 10.1.1.123) of the host running the server by opening a command shell (Press the Windows key + r simultaneously → type cmd) on the server and type `ipconfig` to learn the IP address. If you are using the same computer for the client and server, you will want to use the loopback IP address of 127.0.0.1. This tells your computer to create connects with itself.
2. To open the server program, right click on the file (e.g., TCPServer.py) inside Windows Explorer and select "Edit with IDLE". You should see a window similar to the following:


1-TCPServer.py - G:\My Documents\AFIT\Courses\CSCE 560\Fa17\Assig...
File Edit Format Run Options Window Help
Ln: 1 Col: 0

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

3. To execute the program, press F5 or click **`Run`** → **`Run Module`**. You should see a window called Python Shell similar to the following:



```
Python 2.7.14 (v2.7.14:84471935ed, Sep 16 2017, 20:19:30) [MSC v.1500 32 bit (In
tel)] on win32
Type "copyright", "credits" or "license()" for more information.
>>> 
 RESTART: G:\My Documents\AFIT\Courses\CSCE 560\Fa17\Assignments\Python\1-TCP cl
ient-server\1-TCPServer.py
The server is ready to receive
```

4. Within a command shell, type **`netstat –nao`** to verify the server is listening on port 12000; IP address 0.0.0.0 is also your computer. Take a screenshot of the results of this command. As always, place a box or circle around the listening TCP connection on the screenshot as shown below.

Running the Client

1. Open the client program and update the IP address of your server (e.g., 127.0.0.1 or 10.1.1.123).
2. Run the client using the same steps used to start the server.
3. When prompted, enter a lower-case string of characters and press enter.
4. Observe the string returned from the server is the capitalized version.
5. For maximum enjoyment, you should run the client on a different computer (not the server). This is completely optional and not graded. You should know that Wireshark cannot see traffic on loopback interfaces on most operating systems; this is another reason to run your server and client on two different computers.

What to Hand in

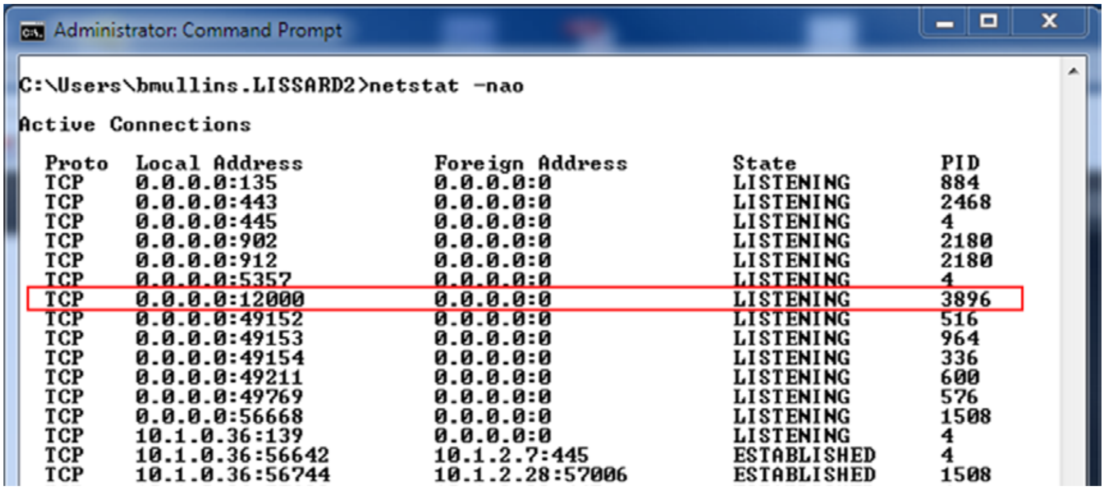
1. Provide a screenshot of the **`netstat`** results showing a listener at port 12000.
2. Provide screenshots of your server and client Python Shells after the server responds with the capitalized string.

Debugging

If you receive an error similar to the following, another process (probably an earlier version of your server program) is already listening on your port (i.e., 12000 in this case):

> error: [Errno 10048] Only one usage of each socket address (protocol/network address/port) is normally permitted

To determine if another process is listening on the port, open a command shell and once again type **`netstat –nao`**. You should see a list of connections similar to the following:

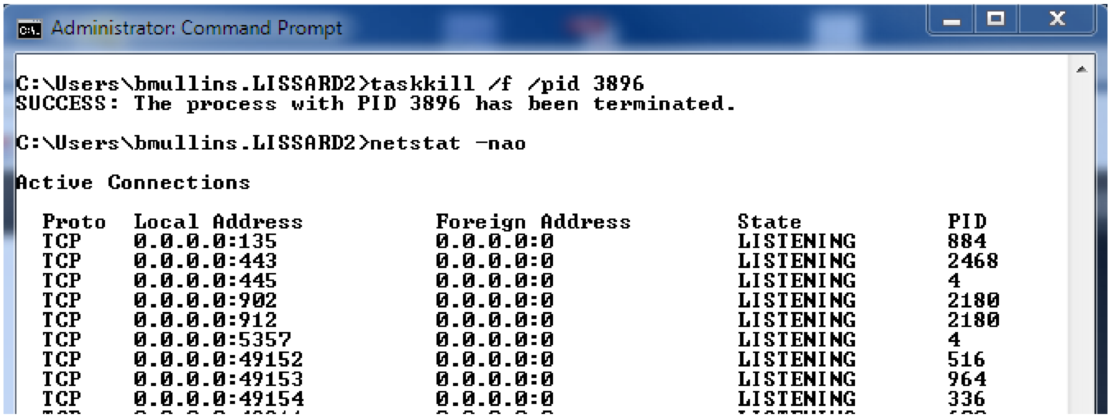


```
C:\Users\bmullins.LISSARD2>netstat -nao

Active Connections

  Proto  Local Address          Foreign Address        State           PID
  TCP    0.0.0.0:135            0.0.0.0:0              LISTENING       884
  TCP    0.0.0.0:443            0.0.0.0:0              LISTENING       2468
  TCP    0.0.0.0:445            0.0.0.0:0              LISTENING       4
  TCP    0.0.0.0:902            0.0.0.0:0              LISTENING       2180
  TCP    0.0.0.0:912            0.0.0.0:0              LISTENING       2180
  TCP    0.0.0.0:5357           0.0.0.0:0              LISTENING       4
  TCP    0.0.0.0:12000          0.0.0.0:0              LISTENING       3896
  TCP    0.0.0.0:49152          0.0.0.0:0              LISTENING       516
  TCP    0.0.0.0:49153          0.0.0.0:0              LISTENING       964
  TCP    0.0.0.0:49154          0.0.0.0:0              LISTENING       336
  TCP    0.0.0.0:49211          0.0.0.0:0              LISTENING       600
  TCP    0.0.0.0:49769          0.0.0.0:0              LISTENING       576
  TCP    0.0.0.0:56668          0.0.0.0:0              LISTENING       1508
  TCP    10.1.0.36:139          0.0.0.0:0              LISTENING       4
  TCP    10.1.0.36:56642        10.1.2.7:445           ESTABLISHED     4
  TCP    10.1.0.36:56744        10.1.2.28:57006        ESTABLISHED     1508
```

In this case, process 3896 is already listening on port 12000. You need to kill this process so your program can start a new listener on the port. Type **`taskkill /f /pid 3896`** to kill the process followed by **`netstat –nao`** to verify the port is now free.



```
C:\Users\bmullins.LISSARD2>taskkill /f /pid 3896
SUCCESS: The process with PID 3896 has been terminated.

C:\Users\bmullins.LISSARD2>netstat -nao

Active Connections

  Proto  Local Address          Foreign Address        State           PID
  TCP    0.0.0.0:135            0.0.0.0:0              LISTENING       884
  TCP    0.0.0.0:443            0.0.0.0:0              LISTENING       2468
  TCP    0.0.0.0:445            0.0.0.0:0              LISTENING       4
  TCP    0.0.0.0:902            0.0.0.0:0              LISTENING       2180
  TCP    0.0.0.0:912            0.0.0.0:0              LISTENING       2180
  TCP    0.0.0.0:5357           0.0.0.0:0              LISTENING       4
  TCP    0.0.0.0:49152          0.0.0.0:0              LISTENING       516
  TCP    0.0.0.0:49153          0.0.0.0:0              LISTENING       964
  TCP    0.0.0.0:49154          0.0.0.0:0              LISTENING       336
```