**Network Programming for Engineers (ECE 5650)**

**Lab 2**

**Team Members Names: Anika Tasnim, Li Lin**

**Source Code(s):**

**Source code for Client:**

#!/usr/bin/python

# -\*- coding: UTF-8 -\*-

# File Name：TCPEchoAppClient.py

# Created: 9/16/2020

# Author: Li Lin & Anika Tasnim

from socket import \*

serverName = 'localhost'    #'192.168.0.15'

serverPort = 12000

with socket(AF\_INET, SOCK\_STREAM) as clientSocket:

    #Setup connection with server

    clientSocket.connect((serverName,serverPort))

    isContinue = True

    while isContinue:

        #Ask user to input message

        message =input('Input lowercase sentence:')

        #Send message to server

        clientSocket.send(message.encode())

        #Receive message from server

        modifiedMessage = clientSocket.recv(1024)

        print('From Server the Sentence in Upper Case:', modifiedMessage.decode())

        messageNum = clientSocket.recv(1024)

        print('From Server number of words in the sentence:', messageNum.decode())

        print('\n')

        #User input confirmation

        while True:

            Yes\_No =input('Do you want to send more message? Y/N  ')

            if (Yes\_No == 'Y') or (Yes\_No == 'y'):

                print('\n')

                break

            elif (Yes\_No == 'N') or (Yes\_No == 'n'):

                isContinue = False

                break

            else:

                print('Please type Y or N only！\n')

print('Socket closed')

print('Client closed')

**Source code for Server:**

#!/usr/bin/python

# -\*- coding: UTF-8 -\*-

# File Name：TCPEchoAppServer.py

# Created: 9/16/2020

# Author: Li Lin & Anika Tasnim

from socket import \*

serverPort = 12000

with socket(AF\_INET, SOCK\_STREAM) as serverSocket:

    #bind with server address and port

    serverSocket.bind(('',serverPort))

    #Start to monitor

    serverSocket.listen(1)

    print('The server is ready to receive...')

    #wait for client's connection

    connectionSocket,addr = serverSocket.accept()

    with connectionSocket:

        while True:

            #Receive message from client

            message = connectionSocket.recv(1024)

            #Check if empty message

            if not message:

                break

            #seperating the words by using split functions

            words = message.split()

            #counting the number of words

            count = len(words)

            print('Received:' + message.decode() + ' ;Number of Words = ' + str(count) + '\n')

            #Change all letters to Upper case

            modifiedMessage = message.decode().upper()

            #Count the number of words from received message

            messageNum = str(count)

            #Send uppder case letters to Client

            connectionSocket.send(modifiedMessage.encode())

            #Send word number to Client

            connectionSocket.send(messageNum.encode())

    print('Connection closed')

print('Socket closed')

print('Server closed')

**Testing Procedure, including Description of Inputs**

1. Run the server firstly on the first terminal as a command line and then the client on the other terminal.
2. In client terminal, type a sentence to send it server
3. Check response from server
   1. Server should print the original sentence from Client and number of words
   2. Client should receive two messages from server: all upper case sentence and number of words
4. In client terminal, “Do you want to send more message? Y/N” should appear.
   1. Input “Y” or “y” to continue.
   2. After a step, “Input lowercase sentence:” should appear.
   3. Repeat Step2&3, to confirm if we always get correct response for different messages
5. In client terminal, after finish 4-c step, input none any letter except “Y, y, N, n” .
   1. “Please type Y or N only！” should be printed.
   2. “Do you want to send more message? Y/N” should be printed.
6. Input “N” or “n”, the connection should be closed. Both Client and Server app exit.

**Optional test on two different machines:**

1. In client code, change ‘localhost’ to ‘192.168.0.15’ which is my one laptop’s local IP address
2. Run client code in cmd terminal window on laptop with ‘192.168.0.12’ IP address
3. Run server code in cmd terminal window on laptop with ‘192.168.0.15’ IP address
4. Conduct step 2-6, should get similar result as above.

**Screenshots and Their Explanations:**

Basic task result:

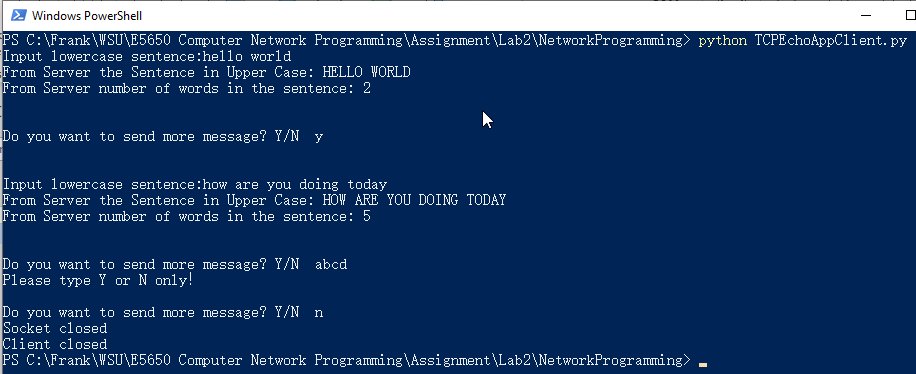
Server: Client:

## 

1. Client sent “hello world”, Server respond “HELLO WORLD” and number “2”, result is good.
2. Input ‘y’ to continue to send message.
3. Client sent “how are you doing today” , Server respond “HOW ARE YOU DOING TODAY” and number “5”, result is good.
4. Input ‘abcd’, message ‘Please type Y or N only!” message printed on client terminal
5. Input ‘n’, both client and server are closed.

**Optional task result:**

Client(192.168.0.12) Screen:



Server(192.168.0.15) Screen:

**A screenshot of a cell phone

Description automatically generated**

1. Client sent “hello world”, Server respond “HELLO WORLD” and number “2”, result is good.
2. Input ‘y’ to continue to send message.
3. Client sent “how are you doing today” , Server respond “HOW ARE YOU DOING TODAY” and number “5”, result is good.
4. Input ‘abcd’, message ‘Please type Y or N only!” message printed on client terminal
5. Input ‘n’, both client and server are closed.

**Completion Status and Self-Critique:**

For program **Client**:

* Does your program meet all requirements? If not, explain the problem.

Yes

* Does the program run correctly all the time? If not, explain the problem.

Yes

* Did you adequately test the program? If not, specify.

Yes

* Is the program well documented?

Yes

For program **Server**:

* Does your program meet all requirements? If not, explain the problem.

Yes

* Does the program run correctly all the time? If not, explain the problem.

Yes

* Did you adequately test the program? If not, specify.

Yes

* Is the program well documented?

Yes