**Network Design: Phase 1 Project**

**Pavan Kumar Revanuru** - **01578776**

**Design Document**

**Server3.py:**

import socket  *#Socket library*

host = 'localhost'

port = 12000 *# Port number assigned*

ss = socket.socket(socket.AF\_INET, socket.SOCK\_DGRAM)  *#UDP Socket (Datagram*)

ss.bind((host, port))  *# Binds socket to the localhost and port*

print " ready"  *#Prints when server is ready*

*# Receiving the data from the client, changing the characters to uppercase and Trasnmitting the data back to client.*

while 1:

message, clientAddress = ss.recvfrom(2048) *# Receives the data from the client*

modmessage = message.upper() *# Changes the data to uppercase*

ss.sendto(modmessage, clientAddress) *# Transfers the data to client.*

**Client3.py:**

import socket *# Socket library*

host = 'localhost' *# Creates the host address*

port = 12000 *# Assigns the port number*

cs = socket.socket(socket.AF\_INET, socket.SOCK\_DGRAM) *# Creates a UDP socket*

file = open('test.txt', 'r')  *# Opens the file “test.txt” with a read permission*

message = file.read(2048) *# Reads the contents of the file and saves in (buffer) “ message”*

cs.sendto(message, (host, port)) *# Sends the contents of “message” to the server*

modmessage, serverAddress = cs.recvfrom(2048) *# Receives the modified data from the server*

print modmessage *# Prints the modified data*

f = open('new.txt', 'w') *# Creates or Opens a file “new.txt” with write permission*

f.write(modmessage) *# Writes the content in the “modmessage” into the file – ‘new.txt’*

f.close() *# Closes the file function*

cs.close() *# Closes the socket*

**Communication:**

* Client opens the file ‘test.txt’ and reads the content present in it. (‘hello’ in this case)
* Client sends the data to server.
* Server receives and changes the data to Uppercase. (‘HELLO’)
* Server sends the modified data back to the client.
* Client creates or opens a file “new.txt” and writes the modified data into it. (‘HELLO’)
* Client prints the modified data on the terminal window. (‘HELLO’)

**Description of the code:**

**Import socket:** We are able to create the sockets with in our program by using this.

**ss = socket.socket(socket.AF\_INET, socket.SOCK\_DGRAM):** This line creates the server’s socket, called ss. The first parameter indicates the address family; in particular, AF\_INET indicates that the underlying network is using IPv4. The second parameter indicates that the socket is of type SOCK\_DGRAM, which means it is a UDP socket

**ss.bind((host, port)) :**The above line binds the port number to the server’s socket.

**message, clientAddress = ss.recvfrom(2048):** When a packet arrives at the server’s socket, the packet’s data is put into the variable message and the packet’s source address is put into the variable clientAddress. The variable clientAddress contains both the client’s IP address and the client’s port number.

**modmessage = message.upper()**: It takes the line sent by the client and uses the method upper() to capitalize it.

**ss.sendto(modmessage, clientAddress):** This line attaches the client’s address (IP address and port number) to the capitalizedmessage, and sends the resulting packet into the server’s socket.

**file = open('test.txt', 'r'):** This line opens the existing file in the read format.

**message = file.read(2048) :** This line reads the content present in the file “test.txt”

**cs.sendto(message, (host, port)):** The method sendto() attaches the destination address (host, port) to the message and sends the resulting packet into the process’s socket, cs.

**modmessage, serverAddress = cs.recvfrom(2048):** When a packet arrives from the Internet at the client’s socket,the packet’s data is put into the variable modmessage and the packet’s source address is put into the variable serverAddress. The variable serverAddress contains both the server’s IP address and the server’s portnumber.

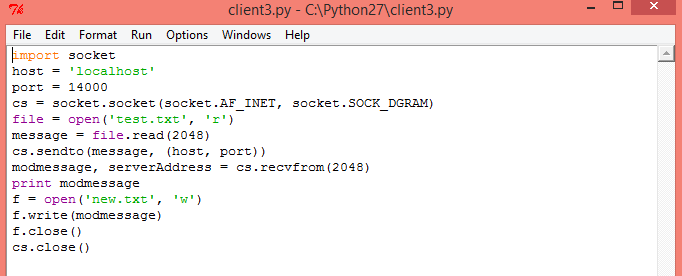
**f.write(modmessage):** The data in the modmessage is written into the file “new.txt”

**cs.close():** This line closes the socket. The process then terminates.

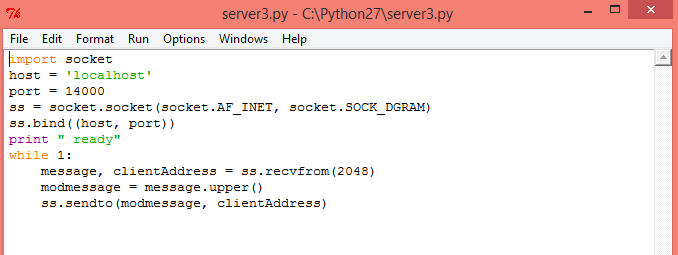
**Steps to execute:**

* Place all the source files in a single folder.
* Open IDLE and open client3.py
* Open IDLE and open server3.py
* Run both the files

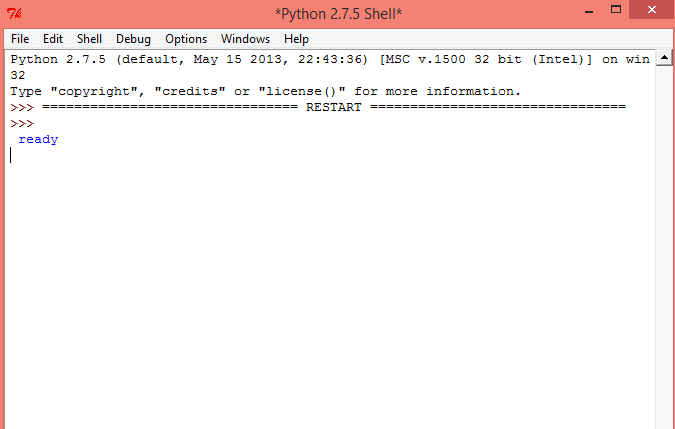
**Screenshots:**



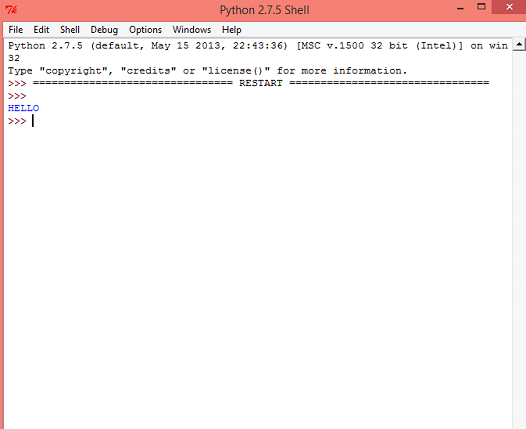
1. Client code

****

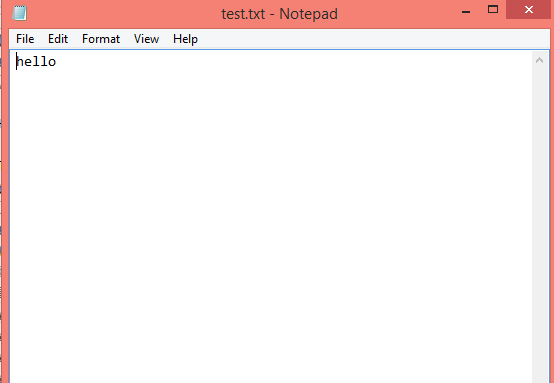
1. Server Code



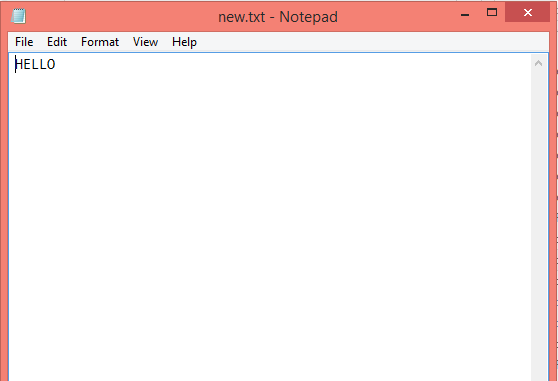
1. Server Terminal



1. Client Terminal



1. Test.txt Window



1. New.txt Window

**Reference: Computer Networking: A Top Down Approach, 6th Edition.**