LAB-5

Implement the socket program in TCP - socket programming on TCP - can be found in https://stackabuse.com/basic-socket-programming-in-python/. Again, do a client and server on two different VMs

Client Side Program

```
gurusaran@guru:~/Desktop/NS
File  Actions  Edit  View  Help
gurusaran@guru:~/Desktop/NS$ cat c1.py
# load additional Python modules
import socket
import time

# create TCP/IP socket
sock = socket.socket(socket.AF_INET, socket.SOCK_STREAM)

# retrieve local hostname
local_hostname = '192.168.0.8'

# get fully qualified hostname
local_fqdn = socket.getfqdn()

# get the according IP address
ip_address = socket.gethostbyname(local_hostname)

# bind the socket to the port 23456, and connect
server_address = (ip_address, 23456)
sock.connect(server_address)
print ("connecting to %s (%s) with %s" % (local_hostname, local_fqdn, ip_address))

# define example data to be sent to the server
temperature_data = ["15", "22", "21", "26", "25", "19"]
for entry in temperature_data:
    print ("data: %s" % entry)
    new_data = str("temperature: %s\n" % entry).encode("utf-8")
    sock.sendall(new_data)

    # wait for two seconds
    time.sleep(2)

# close connection
sock.close()
gurusaran@guru:~/Desktop/NS$
```

Server side Program

```
guru@ubuntu: ~/Desktop/Ns
guru@ubuntu:~/Desktop/Ns$ cat S.py
# load additional Python module
import socket

# create TCP/IP socket
sock = socket.socket(socket.AF_INET, socket.SOCK_STREAM)

# retrieve local hostname
local_hostname = ''

# get fully qualified hostname
local_fqdn = socket.getfqdn()

# get the according IP address
ip_address = socket.gethostbyname(local_hostname)

# output hostname, domain name and IP address
print ("working on %s (%s) with %s" % (local_hostname, local_fqdn, ip_address))

# bind the socket to the port 23456
server_address = (ip_address, 23456)
print ('starting up on %s port %s' % server_address)
sock.bind(server_address)

# listen for incoming connections (server mode) with one connection at a time
sock.listen(1)

while True:
    # wait for a connection
    print ('waiting for a connection')
    connection, client_address = sock.accept()

    try:
        # show who connected to us
        print ('connection from', client_address)

        # receive the data in small chunks and print it
        while True:
            data = connection.recv(64)
            if data:
                # output received data
                print ("Data: %s" % data)
            else:
                # no more data -- quit the loop
                print ("no more data.")
```

Client side input

```
gurusaran@guru:~/Desktop/NS$ cat c1.py
# load additional Python modules
import socket
import time

# create TCP/IP socket
sock = socket.socket(socket.AF_INET, socket.SOCK_STREAM)

# retrieve local hostname
local_hostname = '192.168.0.8'

# get fully qualified hostname
local_fqdn = socket.getfqdn()

# get the according IP address
ip_address = socket.gethostbyname(local_hostname)

# bind the socket to the port 23456, and connect
server_address = (ip_address, 23456)
sock.connect(server_address)
print ("connecting to %s (%s) with %s" % (local_hostname, local_fqdn, ip_address))

# define example data to be sent to the server
temperature_data = ["15", "22", "21", "26", "25", "19"]
for entry in temperature_data:
    print ("data: %s" % entry)
    new_data = str("temperature: %s\n" % entry).encode("utf-8")
    sock.sendall(new_data)

    # wait for two seconds
    time.sleep(2)

# close connection
sock.close()
gurusaran@guru:~/Desktop/NS$ python3 c1.py
connecting to 192.168.0.8 (guru) with 192.168.0.8
data: 15
data: 22
data: 21
data: 26
data: 25
data: 19
gurusaran@guru:~/Desktop/NS$
```

Server side output

```
guru@ubuntu: ~/Desktop/Ns
# bind the socket to the port 23456
server_address = (ip_address, 23456)
print ('starting up on %s port %s' % server_address)
sock.bind(server_address)

# listen for incoming connections (server mode) with one connection at a time
sock.listen(1)

while True:
    # wait for a connection
    print ('waiting for a connection')
    connection, client_address = sock.accept()

    try:
        # show who connected to us
        print ('connection from', client_address)

        # receive the data in small chunks and print it
        while True:
            data = connection.recv(64)
            if data:
                # output received data
                print ("Data: %s" % data)
            else:
                # no more data -- quit the loop
                print ("no more data.")
                break
    finally:
        # Clean up the connection
        connection.close()
guru@ubuntu:~/Desktop/Ns$ python3 S.py
working on  (ubuntu) with 0.0.0.0
starting up on 0.0.0.0 port 23456
waiting for a connection
connection from ('192.168.0.9', 7024)
Data: b'temperature: 15\n'
Data: b'temperature: 22\n'
Data: b'temperature: 21\n'
Data: b'temperature: 26\n'
Data: b'temperature: 25\n'
Data: b'temperature: 19\n'
no more data.
waiting for a connection
```