Echo Protocol Myles G-B

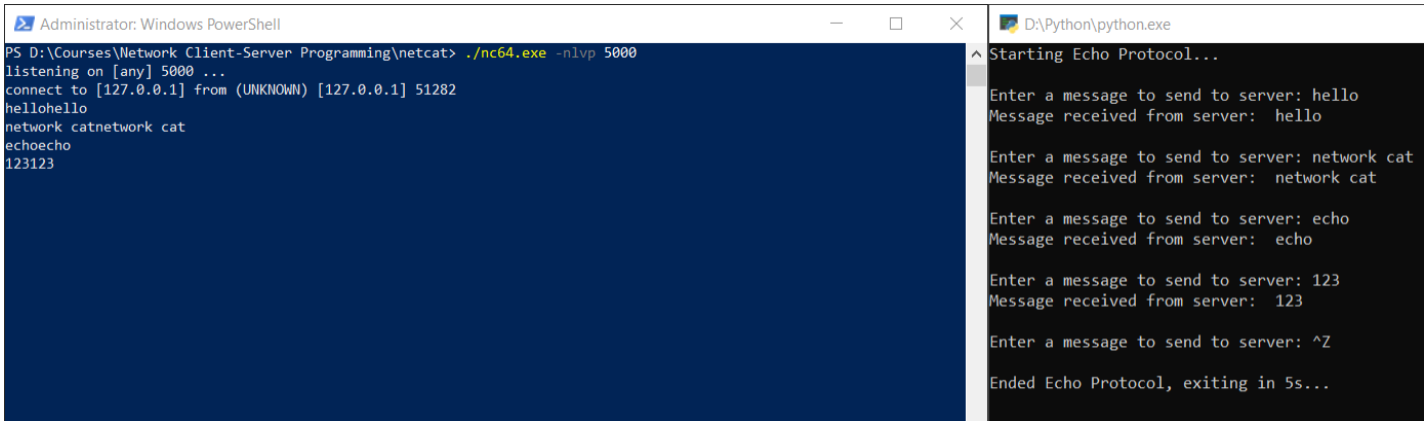
I wrote my echo protocol in python, and in order to implement it I only had to edit the baseTCPProtocolC() function and all the other boilerplate code is pretty much untouched. This is my protocol function:

```
def baseTCPProtocolC(csoc):
    print("Starting Echo Protocol...\n")

    while True:
        try:
            msg = input("Enter a message to send to server: ")
            csoc.sendall(msg.encode("utf-8"))
            data = loopRecv(csoc, len(msg) + 1)
            print("Message received from server: ", data.decode())
        except EOFError:
            break

    print("\nEnded Echo Protocol, exiting in 5s...")
    time.sleep(5)
```

When the function is called it goes into a while loop with a try-except in it, where the exception being checked is EOFError. Since I am on windows, ctrl+z is EOF, so if the user wants to stop sending messages to the server they can simply hit ctrl+z. At this point, the while loop will break and the user is informed that the program is exiting, and the socket is closed.

The function uses the length of the message the user typed plus one in order to know how much to loopRecv from the server, and once it gets the message from the server it decodes it with data.decode() so that it can be printed as a normal string and not a byte array.

Example Run



Source Code

```
import socket
import time

def loopRecv(csoc, size):
    data = bytearray(b" "*size)
    mv = memoryview(data)

    while size:
        rsize = csoc.recv_into(mv, size)
        mv = mv[rsize:]
        size -= rsize

    return data

def baseTCPProtocolC(csoc):
    print("Starting Echo Protocol...\n")

    while True:
        try:
            msg = input("Enter a message to send to server: ")
            csoc.sendall(msg.encode("utf-8"))
```

```
            data = loopRecv(csoc, len(msg) + 1)
            print("Message received from server: ", data.decode())
        except EOFError:
            break

    print("\nEnded Echo Protocol, exiting in 5s...")
    time.sleep(5)

if __name__ == "__main__":
    # create the socket
    #  defaults family=AF_INET, type=SOCK_STREAM, proto=0, filno=None
    commsoc = socket.socket()

    # connect to localhost:5000
    commsoc.connect(("localhost", 5000))

    # run the application protocol
    baseTCPProtocolC(commsoc)

    # close the comm socket
    commsoc.close()
```