Solution: Write a UDP Chat App!

WE'LL COVER THE FOLLOWING

Client

The client program uses a `while` loop to keep the conversation with the server alive. Furthermore, it uses `connect()` to ensure that only one server is connected to, and only replies from that server are received.

```
import argparse, socket

MAX_SIZE_BYTES = 65535 # Mazimum size of a UDP datagram

def client(port):
    s = socket.socket(socket.AF_INET, socket.SOCK_DGRAM)
    host = '127.0.0.1'
    while True:
        s.connect((host, port))
        message = input('Input message to send to server:' )
        data = message.encode('ascii')
        s.send(data)
        data = s.recv(MAX_SIZE_BYTES)
        text = data.decode('ascii')
        print('The server replied with {!r}'.format(text))
```

Server

```
import argparse, socket

MAX_SIZE_BYTES = 65535 # Mazimum size of a UDP datagram

def server(port):
    s = socket.socket(socket.AF_INET, socket.SOCK_DGRAM)
    hostname = '127.0.0.1'
    s.bind((hostname, port))
    print('Listening at {}'.format(s.getsockname()))
```

```
    while True:
        data, clientAddress = s.recvfrom(MAX_SIZE_BYTES)
        message = data.decode('ascii')

        print('The client at {} says {!r}'.format(clientAddress, message))
        msg_to_send = input('Input message to send to client:' )
        data = msg_to_send.encode('ascii')
        s.sendto(data, clientAddress)
```

Great! Let's look at how server and client programs can be written to run on TCP in Python3 in the next lesson!