What to do for each function
Project 1

init(udp_port1, udp_port2)

MUST DO:

-Store these two ports

-HINT: whenever you call sendto, send to the transmit port, whenever you call bind, bind to the receive port

MAY DO:

create sockets, bind sockets

socket() (__init__)

MUST DO:

nothing

SUGGESTED:

create sockets (Remember to use DGRAM sockets)

MAY DO:
bind socket to port (remember to do this only once for the server and once for the client)

bind(address)

MUST DO:

nothing

Suggested:

bind sockets

connect(address)

MUST DO:

Perform the Client side of the handshake

SUGGESTED:

bind the client port

listen(backlog)

MUST DO:

nothing

MAY DO:

record backlog to return a new port for accept (if this gives you any trouble at all use the easy way there is no penalty)

accept()

MUST DO:

perform the server side of the hand shake:

MAY DO:

bind the server to its port (remember to do only once per server)

HINT: you can return self as the client port to avoid the need for a new one.

send(buffer)

MUST DO:

Do go back N from the send side.

Include header as part of the message

SUGGESTED:

create ONE thread to receive messages, use count up timer to keep track of which threads have timed out.

MAY DO:

anything else, as long as you send all the data in packets and do god back N

recv(numBytes)

MUST DO:

Go back N from the receiver side

reassemble packets into one chunk (without their headers) to return.

HINT: remember to reject all packets that are not the expected one.

close()

MUST DO:

close the socket if there is nothing else left to receive. If you are waiting for something, do nothing.

Helpful links

-for people testing on macs:

https://stackoverflow.com/questions/22819214/udp-message-too-long

-threading with python tutorial:

https://www.python-course.eu/threads.php

-NOTE: if you read anything from a socket it takes the entire packet, so read one packet at a time.