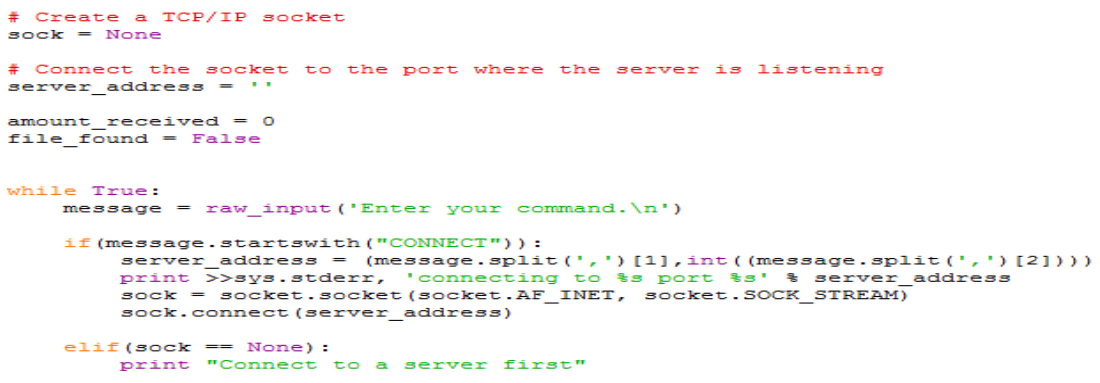
IOE Client Flow-Chart

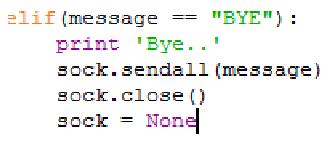
The Client has a variable “server_address” to store the server_address and a sock varible set to None. Both of these varibles are used for determineing if the connection has been made. “raw_input” prompts the user for input.

If sock is still “None” at this point, which means the user did not connect to a server yet, the client will response “connect to a server first.”



```
# Create a TCP/IP socket
sock = None

# Connect the socket to the port where the server is listening
server_address = ''

amount_received = 0
file_found = False


while True:
    message = raw_input('Enter your command.\n')

    if(message.startswith("CONNECT")):
        server_address = (message.split(',')[1],int((message.split(',')[2])))
        print >>sys.stderr, 'connecting to %s port %s' % server_address
        sock = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
        sock.connect(server_address)

    elif(sock == None):
        print "Connect to a server first"
```

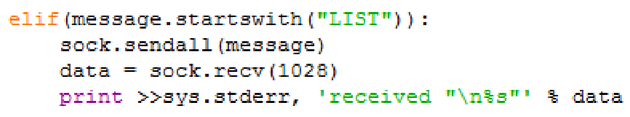
If the message starts with “BYE”, the client will send “BYE” to the sever using sock.sendall(message), and then it will close the socket and set “sock = None” again to indicate that no connection exist.



```
elif(message == "BYE"):
    print 'Bye..'
    sock.sendall(message)
    sock.close()
    sock = None
```

If the message starts with “LIST”,

The client will simply send the command “LIST” using sendall and receive the response from the server using sock.recv(1028), then print the response to the user.



```
elif(message.startswith("LIST")):
    sock.sendall(message)
    data = sock.recv(1028)
    print >>sys.stderr, 'received "\n%s"' % data
```

Jingnan Chen 1073196, Wei Zhang 0951321

IOE Client Flow-Chart

If the repsonse start with “Read”;

Call sock.recv() once to receive the file size from the server, because I know the next data sent from the server-side is the file size for the client to use for catching errors.

```
elif(message.startswith("READ")):
    if(len(message.split(',')) == 2):

        # Send data
        #print >>sys.stderr, 'Sending "%s"' % message
        sock.sendall(message)
        data = sock.recv(64)
        if(data.startswith("ERROR:")):
           print >>sys.stderr, '"%s"' % data
        else:
            filesize = int(data)
```

Then open a file using the open method with the filename given in “write byte” mode and set amount_received variable to 0. Then Using a while(amount_received < filesize) and a try and catch statement for suddent interrupted connection to receive the data by calling self.request.recv(),

```
    amount_received = 0
    f1 = open(message.split(',')[1], 'wb')
    while amount_received < filesize:
    #while True:
        try:
            data = sock.recv(64)
            f1.write(data)
            amount_received = len(data) + amount_received
            print str(data)
        except:

            f1.close()
            os.remove(f1.name)
            sock = None
            break
```

Seen above, write the data received to the file using the write() method. If anything goes wrong, statements in the except block is excuted, closing the file and deleting the broken file. It also set sock = None, so the client can make a new connection to the server.

If the file transfer is successful, then print file received and close the file.

```
if(amount_received == filesize):
     print >>sys.stderr, 'File Received.'
f1.close()
```

IOE Client Flow-Chart

↓

If the message starts with "WRITE",

First loop through all the files in the current directory to find if the file that the client is trying to write to the server exist, if not send back an error message.

```
elif(message.startswith("WRITE")):
    if(len(message.split(',')) == 2):
        file_found = False
        for files in os.listdir(os.getcwd()):
```

If the file is found, then send the message along with the size of the file, so that the server knows when the received file is completed and it can use the size to detect broken files.

```
for files in os.listdir(os.getcwd()):
    if(files == message.split(',')[1]):
        file_found = True
        sock.sendall(message + ','+ str(os.path.getsize(os.getcwd() + "\\" + files)))
```

Open the file in "read byte" mode and for each line in the file, send that line to the server using sock.sendall(line)

```
f1 = open(message.split(',')[1],'rb')
for line in f1:
    sock.sendall(line)
```

After finishing sending all the line of the file, close the file, and received the respone from the server, to indicate whether the file has been successfully received.

```
f1.close()
data = sock.recv(1028)
if(data != ""):
    print >>sys.stderr, '%s' % data
```

If the file was not found in the current directory, print back a message to the user, stating "file not found".

```
  if(file_found == False):
      print "\nFile is not in the current directory and include the extension"
```

Jingnan Chen 1073196, Wei Zhang 0951321