

**Gebze Technical University
Computer Engineering**

CSE 476 - 2020 Autumn

TERM PROJECT REPORT

**LEMYE CEREN GUMUS
151044071**

Course Assistant:Abdullah Salih BAYRAKTAR

1 INTRODUCTION

1.1 Problem Definition

Socket programming is a way of connecting two nodes on a network to communicate with each other. One socket(node) listens on a particular port at an IP, while other socket reaches out to the other to form a connection. Server forms the listener socket while client reaches out to the server.

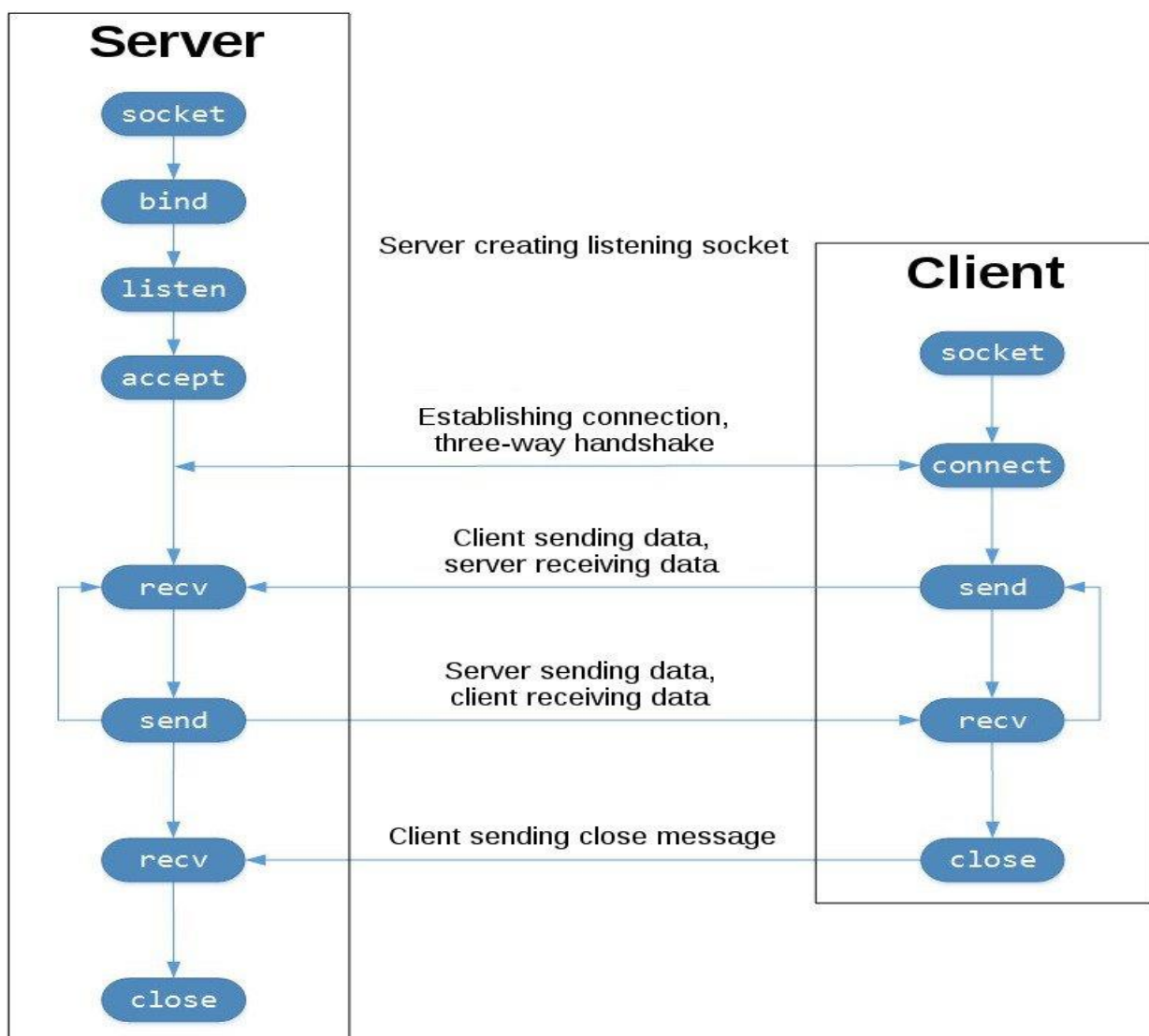
Socket programming is started by importing the socket library and making a simple socket.

```
import socket
serverSocket = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
```

Here we made a socket instance and passed it two parameters. Now we can connect to a server using this socket.

- AF_INET refers to the address family ipv4.
- The SOCK_STREAM means connection oriented TCP protocol.

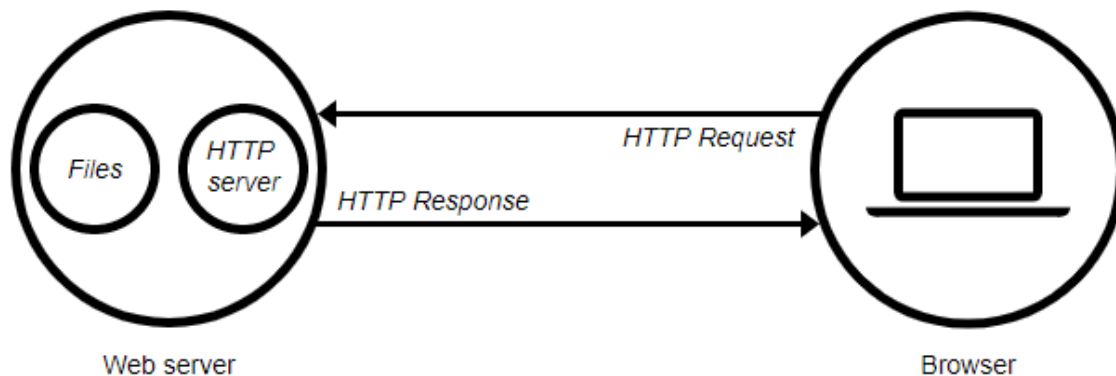
Server – Client Program:



It should be complete three Socket Programming using Python.

1. Web Server
2. UDP Pinger
3. Mail Client

Web Server : A web server is a computer that stores web server software and a website's component files. (for example, HTML documents, images, CSS stylesheets, and JavaScript files) A web server connects to the Internet and supports physical data interchange with other devices connected to the web. On the software side, a web server includes several parts that control how web users access hosted files. At a minimum, this is an *HTTP server*. An HTTP server is software that understands URLs (web addresses) and HTTP (the protocol your browser uses to view webpages). An HTTP server can be accessed through the domain names of the websites it stores, and it delivers the content of these hosted websites to the end user's device.



UDP Pinger : The client should send 10 pings to a UDP server. Because UDP is an unreliable protocol, a packet sent from the client to the server may be lost in the network, or vice versa. The client wait up to one second for a reply; if no reply is received within one second, the client program should assume that the packet was lost during transmission across the network. The client should also calculate the round trip time for each of the pings sent to the server.

Mail Client: An SMTP server is a computer or an app that is responsible for sending emails. It functions following the Simple Mail Transfer Protocol (SMTP). An SMTP server receives emails from the email client. Then it passes them on to another SMTP server and relays them to the incoming mail server. A mail server is the computerized equivalent of your friendly neighborhood mailman. Every email that is sent passes through a series of mail servers along its way to its intended recipient. Although it may seem like a message is sent instantly - zipping from one PC to another in the blink of an eye - the reality is that a complex series of transfers takes place. Without this series

of mail servers, you would only be able to send emails to people whose email address domains matched your own

1.2 System Requirements

For all program , it must be used Python Programming language

Web Server : Python standard library comes with a in-built webserver which can be invoked for simple web client server communication. The port number can be assigned programmatically and the web server is accessed through this port. Though it is not a full featured web server which can parse many kinds of file, it can parse simple static html files and serve them by responding them with required response codes.

```
port = 12345
```

Next bind to the port we have not typed any IP in the IP field instead we have inputted an empty string this makes the server listen to requests coming from other computers on the network

```
serverSocket.bind(("", port))
```

and put the socket into listening mode

```
serverSocket.listen(5)
```

Establish the connection

```
While(true):
```

```
    Try:
```

```
    Except:
```

```
        .
```

```
        .
```

```
        .
```

Establish connection with client.

```
c, addr = serverSocket.accept()
```

```
print ('Got connection from',addr)
```

Send one HTTP header line into socket

```
connectSocket.send("\nHTTP/1.1 200 OK\n\n".encode())
```

Send the content of the requested file to the connection socket

```
for i in range(0, len(out)):
```

```
    connectSocket.send(out[i].encode())
```

```
    connectSocket.send("\r\n".encode())
```

```
connectSocket.close()
```

Send HTTP response message for file not found

```
connectionSocket.send("\nHTTP/1.1 404 Not Found\n\n".encode())
```

UDP Pinger :

Server :

Assign IP address and port number to socket

```
serverSocket.bind(("", 12000))
```

while True:

Generate random number in the range of 0 to 10

```
rand = random.randint(0, 10)
```

Receive the client packet along with the address it is coming from

```
msg, addr = serverSock.recvfrom(1024)
```

Capitalize the message from the client

```
msg = msg.upper()
```

If rand is less is than 4, we consider the packet lost and do not respond

```
if rand < 4:    continue
```

Otherwise, the server responds

```
serverSock.sendto(msg, addr)
```

client :

```
msg = 'ping'
```

```
addr = ("localhost", 12000)
```

```
for i in range(1,11):
```

```
    start = time.time()
```

```
    clientSock.sendto(msg.encode(), addr)
```

```
    try:
```

```
        response, server = clientSock.recvfrom(1024)
```

```
        end = time.time()
```

```
        RTT = end - start
```

Mail Client:

Create socket called clientSocket and establish a TCP connection with mailserver

```
clientSocket = socket(AF_INET, SOCK_STREAM)
```

```
clientSocket.connect(mailserver) recv = clientSocket.recv(1024)
```

```
if recv[:3] != '220': print('220 reply not received from server.')
```

Send HELO command and print server response.

```
heloCommand = 'HELO Alice\r\n' clientSocket.send(heloCommand.encode())
```

```
recv1 = clientSocket.recv(1024) print(recv1)
```

```
if recv1[:3] != '250': print('250 reply not received from server.')
```

Info for username and password

```
username = "cerengumus0932@gmail.com"
```

the username for your server

```
password = "151044071"
```

the password for your server, changed here

```
base64_str = ("\x00"+username+"\x00"+password).encode() base64_str =
```

```
base64.b64encode(base64_str) authMsg
```

```
="message".encode()+base64_str+"\r\n".encode() clientSocket.send(authMsg)
```

```
recv_auth = clientSocket.recv(1024)
```

```
if recv1[:3] != '250': print('250 reply not received from server.')
```

command and print server response.

```
mailFrom = "MAIL FROM: <cerengumus0932@gmail.com> \r\n"
```

```
clientSocket.send(mailFrom.encode())
```

```
recv2 = clientSocket.recv(1024)
```

```
if recv1[:3] != '250':
```

```
# Send RCPT TO command and print server response.
```

```
rcptTo = "RCPT TO: <cerengumus0932@gmail.com> \r\n"
```

```
clientSocket.send(rcptTo.encode()) recv3 = clientSocket.recv(1024)
```

```
if recv1[:3] != '250': print('250 reply not received from server.')
```

command and print server response. data =

```
"DATA\r\n"clientSocket.send(data.encode())
```

```
recv4 = clientSocket.recv(1024)
```

```
if recv1[:3] != '250': # Send message data.
```

```
subject = "Subject: SMTP mail client testing \r\n\r\n"
```

```
clientSocket.send(subject.encode()) message = raw_input("Enter your message: \r\n")
```

```
clientSocket.send(message.encode()) clientSocket.send(endmsg.encode()) recv_msg
```

```
= clientSocket.recv(1024) print("Response after sending message
```

```
body:"+recv_msg.decode()) if recv1[:3] != '250': print('250 reply not received from server.')
```

Send QUIT command and get server response.

```
clientSocket.send("QUIT\r\n".encode()) message=clientSocket.recv(1024) print
```

```
(message) clientSocket.close()
```

2 METHOD

2.1 Use Case

Web server : *Python Web_server.py*

Browser : <http://127.0.0.1:6789/HelloWorld.html>

optional Exercises2: *Python client.py* <http://127.0.0.1:6789/HelloWorld.html>

UDP Pinger:

Python UDP_Pinger_server.py

Python client_UDP.py

Mail Client:

Python MailClient.py

2.2 Problem Solution Approach

Web Server :

- (i) Create a connection socket when contacted by a client (browser);
- (ii) Receive the HTTP request from this connection;
- (iii) Parse the request to determine the specific file being requested;
- (iv) Get the requested file from the server's file system;
- (v) Create an HTTP response message consisting of the requested file preceded by header lines;
- (vi) Send the response over the TCP connection to the requesting browser. If a browser requests a file that is not present in your server, your server should return a "404 Not Found" error message.
- (vii) Browsers running on different hosts. If you run your server on a host that already has a Web server running on it, then you should use a different port than port 80 for your Web server.

UDP Pinger :

- (i) Client will send a simple ping message to a server
- (ii) Receive a corresponding pong message back from the server
- (iii) Determine the delay between when the client sent the ping message and received the pong message.
- (iv) Ping program is to send 10 ping messages to the target server over UDP.
- (v) For each message, your client is to determine and print the RTT when the corresponding pong message is returned.
- (vi) The client cannot wait indefinitely for a reply to a ping message.

- (vii) It should have the client wait up to one second for a reply from the server; if no reply is received, the client should assume that the packet was lost and print a message accordingly.

Mail Client:

- (i) Create mail client that sends email to any recipient.
- (ii) Client will need to establish a TCP connection with a mail server (e.g., a Google mail server), dialogue with the mail server using the SMTP protocol, send an email message to a recipient (e.g., your friend) via the mail server
- (iii) Close the TCP connection with the mail server.

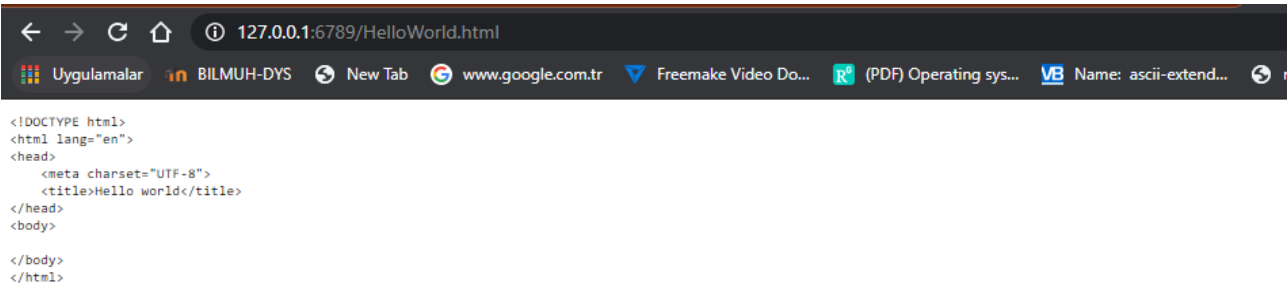
PROJECT CODE RUNNING RESULT:

Web Server:

Web_server.py

```
C:\Users\crngm\AppData\Local\Programs\Python\Python38-32\python.exe C:/Users/crngm/Desktop/151044071_Final/Web_server.py
socket binded to 6789
socket is listening
the web server is up on port: 6789
Ready to serve...
```

Browser :



```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <title>Hello world</title>
</head>
<body>

</body>
</html>
```


Optional Exercises – Web Server – Exercises1 :

Server.py

```
Terminal: Local x Local (2) x +
Microsoft Windows [Version 10.0.19041.685]
(c) 2020 Microsoft Corporation. All rights reserved.

C:\Users\crngm\Desktop\151044071_Final\optionalExercisesWebServer\Exercises1>PYTHON MultiThreadServer.py
Ready to serve...
█
```

Client.py

```
Terminal: Local x Local (2) x Local (3) x +
Microsoft Windows [Version 10.0.19041.685]
(c) 2020 Microsoft Corporation. All rights reserved.

C:\Users\crngm\Desktop\151044071_Final\optionalExercisesWebServer\Exercises1>python client.py
Ready to serve...

<!DOCTYPE html>
<html lang="">
<head>
  <meta charset="utf-8">
  <title>Hello World</title>
</head>
<style>
  body {background-color: cornsilk;}
  h1 {color:saddlebrown;font-family:calibri; opacity:0.4;text-align: center}
</style>
<body>
  <div>
    <h1>Hello World</h1>
  </div>
</body>
</html>
Ready to serve...
Ready to serve...
█
```

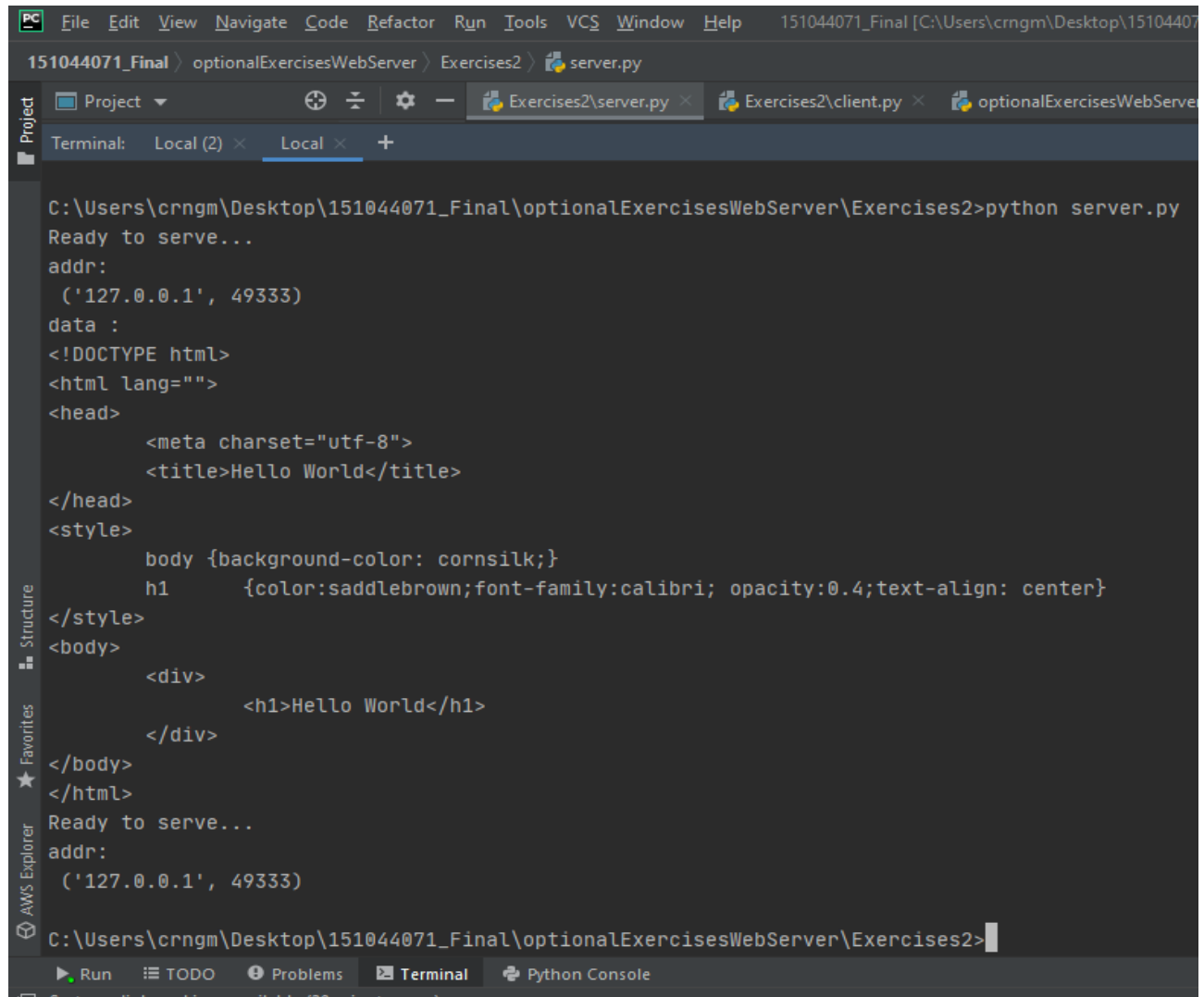
Browser:

```
← → ↺ 🏠 ⓘ 127.0.0.1/HelloWorld.html
Uygulamalar BILMUH-DYS New Tab www.google.com.tr Freemake Video Do... (PDF) Operating sys...

<!DOCTYPE html>
<html lang="">
<head>
  <meta charset="utf-8">
  <title>Hello World</title>
</head>
<style>
  body {background-color: cornsilk;}
  h1 {color:saddlebrown;font-family:calibri; opacity:0.4;text-align: center}
</style>
<body>
  <div>
    <h1>Hello World</h1>
  </div>
</body>
</html>
```

Optional Exercises – Web Server – Exercises2 :

Server.py



The screenshot shows an IDE window with the following components:

- Menu Bar:** File, Edit, View, Navigate, Code, Refactor, Run, Tools, VCS, Window, Help.
- Project Explorer:** 151044071_Final > optionalExercisesWebServer > Exercises2 > server.py.
- Terminal:** Local (2) x Local x +.
- Code Editor:** Contains the output of running `python server.py`. The output shows the server is ready to serve at address ('127.0.0.1', 49333) and displays the HTML content of the served file. The HTML includes a DOCTYPE declaration, a head section with a meta charset and title, a style section with CSS rules for the body and h1, and a body section with a div containing the h1 element.
- Bottom Bar:** Run, TODO, Problems, Terminal, Python Console.

```
C:\Users\crngm\Desktop\151044071_Final\optionalExercisesWebServer\Exercises2>python server.py
Ready to serve...
addr:
('127.0.0.1', 49333)
data :
<!DOCTYPE html>
<html lang="">
<head>
  <meta charset="utf-8">
  <title>Hello World</title>
</head>
<style>
  body {background-color: cornsilk;}
  h1    {color:saddlebrown;font-family:calibri; opacity:0.4;text-align: center}
</style>
<body>
  <div>
    <h1>Hello World</h1>
  </div>
</body>
</html>
Ready to serve...
addr:
('127.0.0.1', 49333)
C:\Users\crngm\Desktop\151044071_Final\optionalExercisesWebServer\Exercises2>
```

client.py

The screenshot displays a Windows 10 desktop environment. The primary focus is a Visual Studio Code (VS Code) editor window. The top of the window shows the file explorer with a project named '151044071_Final' and several open files, including 'optionalExercisesWebServer', 'Exercises2', and 'server.py'. The terminal window at the bottom of the editor displays the output of a Python script, which is an HTML document. The HTML content includes a meta charset declaration, a title 'Hello World', and a body with a single h1 element containing the text 'Hello World'. The taskbar at the bottom of the screen shows the Start button, a search bar, and several pinned application icons, including the VS Code icon, a file explorer, and a web browser. The system clock in the bottom right corner indicates the time is 7:23 on 7/23, with the system encoding set to UTF-8.

151044071_Final optionalExercisesWebServer Exercises2 server.py

Project Project Exercises2/server.py Exercises2/client.py optionalExercisesWebServer/client.py optionalExercisesWebServer

Terminal: Local (2) Local +

```
data : b'\n<!DOCTYPE html>\n<html lang="">\n<head>\n<meta charset="utf-8">\n<title>Hello World</title>\n</head>\n<style>\n\th1 \t{color:saddlebrown;font-family:calibri; opacity:0.4;text-align: center}\n</style>\n<body>\n<div>\n<h1>Hello World</h1>\n</div>\n</body>\n</html>\n'
```

C:\Users\crngm\Desktop\151044071_Final\optionalExercisesWebServer\Exercises2>python client.py 127.0.0.1 6791 HelloWorld.html

server host : 127.0.0.1

server port : 6791

filename : HelloWorld.html

host_port : 127.0.0.1:6791

data :

```
<!DOCTYPE html>
<html lang="">
<head>
  <meta charset="utf-8">
  <title>Hello World</title>
</head>
<style>
  body {background-color: cornsilk;}
  h1    {color:saddlebrown;font-family:calibri; opacity:0.4;text-align: center}
</style>
<body>
  <div>
    <h1>Hello World</h1>
  </div>
</body>
</html>
```

C:\Users\crngm\Desktop\151044071_Final\optionalExercisesWebServer\Exercises2>

Run TODO Problems Terminal Python Console

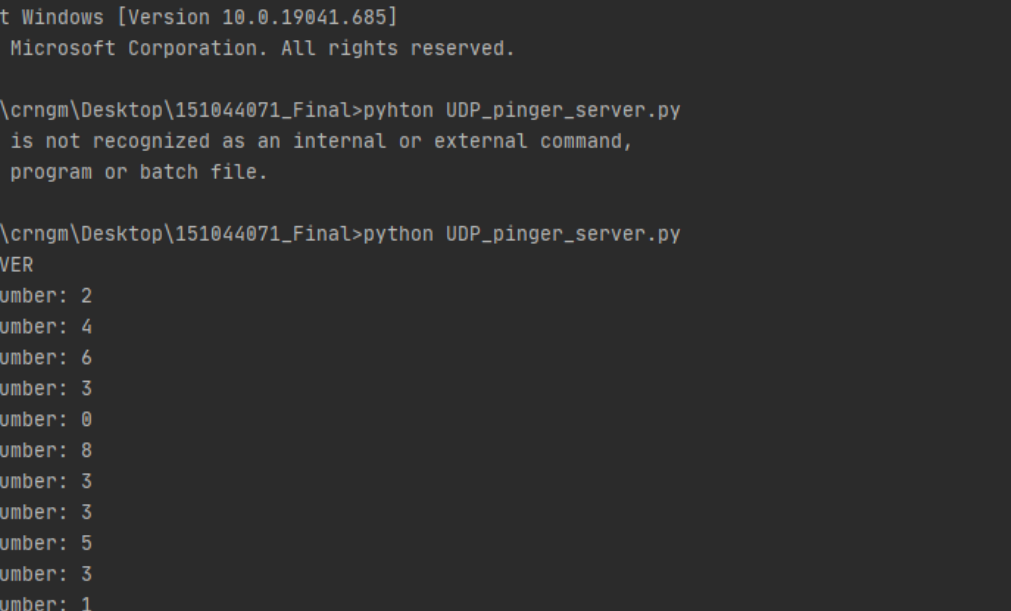
System clipboard is unavailable (moments ago)

7:23 CRLF UTF-

Type here to search

UDP PINGER:

UDP_pinger_server.py



```
151044071_Final C:\Users\crngm\Desktop\151044071_Final
Terminal: Local x Local (2) x +
Microsoft Windows [Version 10.0.19041.685]
(c) 2020 Microsoft Corporation. All rights reserved.

C:\Users\crngm\Desktop\151044071_Final>pyhton UDP_pinger_server.py
'pyhton' is not recognized as an internal or external command,
operable program or batch file.

C:\Users\crngm\Desktop\151044071_Final>python UDP_pinger_server.py
PING SERVER
random number: 2
random number: 4
random number: 6
random number: 3
random number: 0
random number: 8
random number: 3
random number: 3
random number: 5
random number: 3
random number: 1
```

client_UDP.py

```
151044071_Final C:\Users\crngm\Desktop\151044071_Final
Terminal: Local x Local (2) x +

C:\Users\crngm\Desktop\151044071_Final>PYTHON client_UDP.py
#1
Request timed out
#2
Response :b'PING'
RTT:0.001004sec
#3
Response :b'PING'
RTT:0.001000sec
#4
Request timed out
#5
Request timed out
#6
Response :b'PING'
RTT:0.002982sec
#7
Request timed out
#8
Request timed out
#9
Response :b'PING'
RTT:0.001114sec
#10
Request timed out
```

Optional Exercises –UDP PINGER– Exercises1 :

UDPPingServer.py:

```
151044071_Final optionalExercisesUDP Exercises1 UDPPingClient.py
Project
151044071_Final C:\Users\crngm\Desktop\151044071_Final
Terminal: Local x Local (2) x Local (3) x +

C:\Users\crngm\Desktop\151044071_Final\optionalExercisesUDP\Exercises1>PYTHON UDPPingServer.py
Recieve: b'PING 1 1609635359003'
Send: b'PING 1 1609635359003'
Recieve: b'PING 2 1609635359769'
Recieve: b'PING 2 1609635360773'
Send: b'PING 2 1609635360773'
Recieve: b'PING 3 1609635361132'
Send: b'PING 3 1609635361132'
Recieve: b'PING 4 1609635361511'
Recieve: b'PING 4 1609635362522'
Send: b'PING 4 1609635362522'
Recieve: b'PING 5 1609635363508'
Send: b'PING 5 1609635363508'
Recieve: b'PING 6 1609635363743'
Send: b'PING 6 1609635363743'
Recieve: b'PING 7 1609635364171'
Send: b'PING 7 1609635364171'
Recieve: b'PING 8 1609635364678'
Recieve: b'PING 8 1609635365694'
Recieve: b'PING 8 1609635366714'
Send: b'PING 8 1609635366714'
Recieve: b'PING 9 1609635367151'
Send: b'PING 9 1609635367151'
Recieve: b'PING 10 1609635367806'
Send: b'PING 10 1609635367806'
```

UDPPingClient.py:

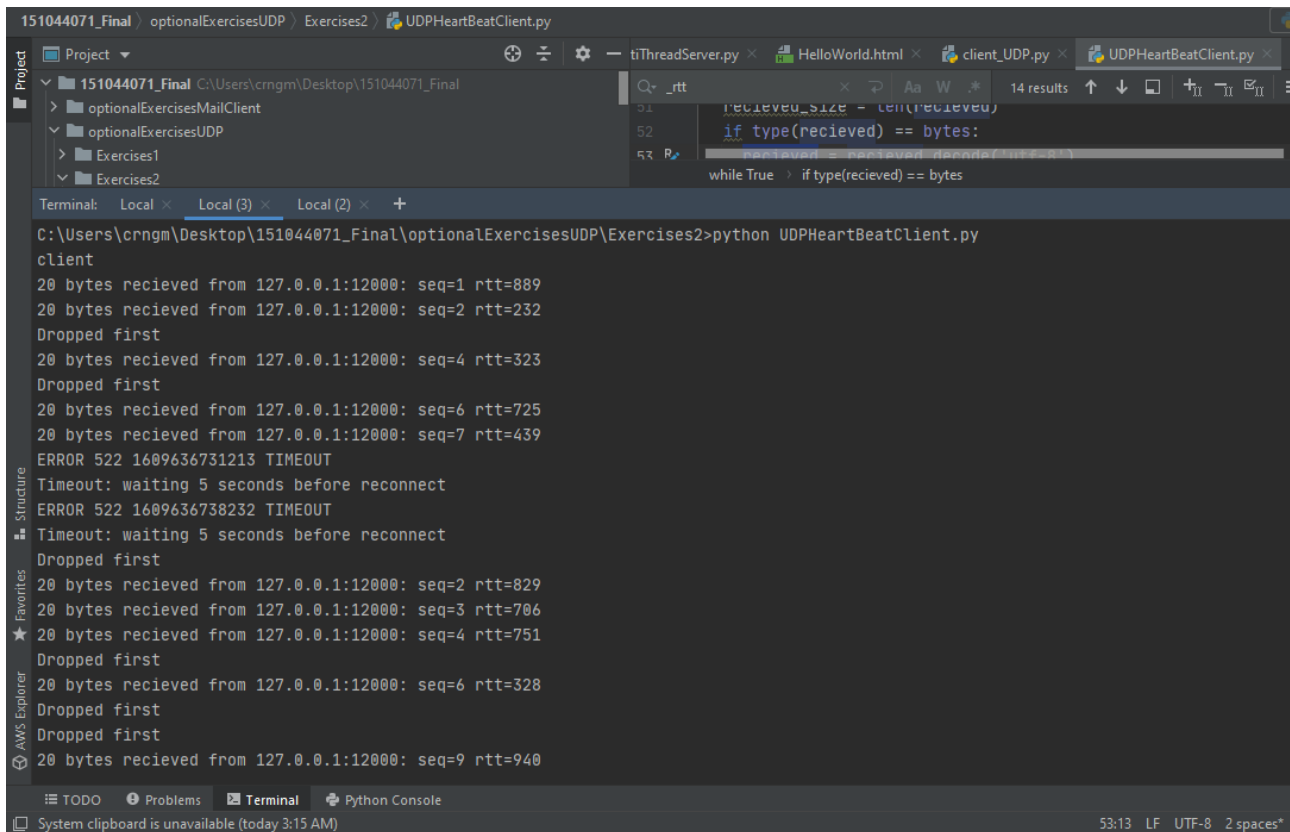
```
151044071_Final > optionalExercisesUDP > Exercises1 > UDPPingClient.py
Project
Terminal: Local x Local (2) x Local (3) x +
C:\Users\crngm\Desktop\151044071_Final\optionalExercisesUDP\Exercises1>python UDPPingClient.py
UDP PING CLIENT
message: PING 1 1609635359003
received: b'PING 1 1609635359003'
['PING', '1', '1609635359003']
20 bytes recieved from 127.0.0.1:12000: seq=1 rtt=766
message: PING 2 1609635359769
received: ERROR 522 1609635360770 TIMEOUT
['ERROR', '522', '1609635360770', 'TIMEOUT']
ERROR 522 1609635360770 TIMEOUT
message: PING 2 1609635360773
received: b'PING 2 1609635360773'
['PING', '2', '1609635360773']
20 bytes recieved from 127.0.0.1:12000: seq=2 rtt=358
message: PING 3 1609635361132
received: b'PING 3 1609635361132'
['PING', '3', '1609635361132']
20 bytes recieved from 127.0.0.1:12000: seq=3 rtt=378
message: PING 4 1609635361511
received: ERROR 522 1609635362520 TIMEOUT
['ERROR', '522', '1609635362520', 'TIMEOUT']
ERROR 522 1609635362520 TIMEOUT
message: PING 4 1609635362522
received: b'PING 4 1609635362522'
['PING', '4', '1609635362522']
20 bytes recieved from 127.0.0.1:12000: seq=4 rtt=986
```

Optional Exercises –UDP PINGER– Exercises2 :

UDPHeartBeatServer.py:

```
151044071_Final > optionalExercisesUDP > Exercises2 > UDPHeartBeatClient.py
Project
151044071_Final C:\Users\crngm\Desktop\151044071_Final
> optionalExercisesMailClient
> optionalExercisesUDP
> Exercises1
> Exercises2
Terminal: Local x Local (3) x Local (2) x +
C:\Users\crngm\Desktop\151044071_Final\optionalExercisesUDP\Exercises2>python UDPHeartBeatServer.py
ready to server...
1609636720541
message: b'PING 1 1609636720541'
Recieve: PING 1 1609636720541
Send: PING 1 1609636720541
1901
message: b'PING 2 1609636722442'
Client connect.
Recieve: PING 2 1609636722442
Send: PING 2 1609636722442
2251
message: b'PING 4 1609636724693'
Dropped Packet:2
Dropped Packet:3
Recieve: PING 4 1609636724693
Send: PING 4 1609636724693
2342
message: b'PING 6 1609636727035'
Dropped Packet:4
Dropped Packet:5
Recieve: PING 6 1609636727035
```

UDPHeartBeatClient.py:

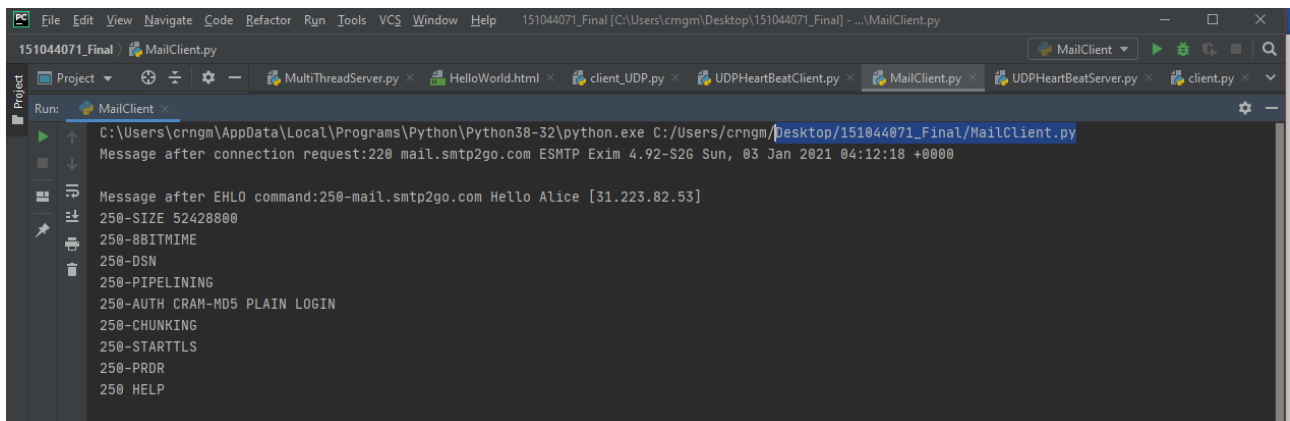


```
151044071_Final / optionalExercisesUDP / Exercises2 / UDPHeartBeatClient.py
Project
  151044071_Final C:\Users\crngm\Desktop\151044071_Final
    optionalExercisesMailClient
    optionalExercisesUDP
    Exercises1
    Exercises2
  Search: _rtt 14 results
  51 received_size = len(recieved)
  52 if type(recieved) == bytes:
  53     received = received.decode('utf-8')
  while True: if type(recieved) == bytes

Terminal: Local (3) Local (2)
C:\Users\crngm\Desktop\151044071_Final\optionalExercisesUDP\Exercises2>python UDPHeartBeatClient.py
client
20 bytes recieved from 127.0.0.1:12000: seq=1 rtt=889
20 bytes recieved from 127.0.0.1:12000: seq=2 rtt=232
Dropped first
20 bytes recieved from 127.0.0.1:12000: seq=4 rtt=323
Dropped first
20 bytes recieved from 127.0.0.1:12000: seq=6 rtt=725
20 bytes recieved from 127.0.0.1:12000: seq=7 rtt=439
ERROR 522 1609636731213 TIMEOUT
Timeout: waiting 5 seconds before reconnect
ERROR 522 1609636738232 TIMEOUT
Timeout: waiting 5 seconds before reconnect
Dropped first
20 bytes recieved from 127.0.0.1:12000: seq=2 rtt=829
20 bytes recieved from 127.0.0.1:12000: seq=3 rtt=706
★ 20 bytes recieved from 127.0.0.1:12000: seq=4 rtt=751
Dropped first
20 bytes recieved from 127.0.0.1:12000: seq=6 rtt=328
Dropped first
Dropped first
20 bytes recieved from 127.0.0.1:12000: seq=9 rtt=940

System clipboard is unavailable (today 3:15 AM) 53:13 LF UTF-8 2 spaces"
```

MAIL CLIENT:

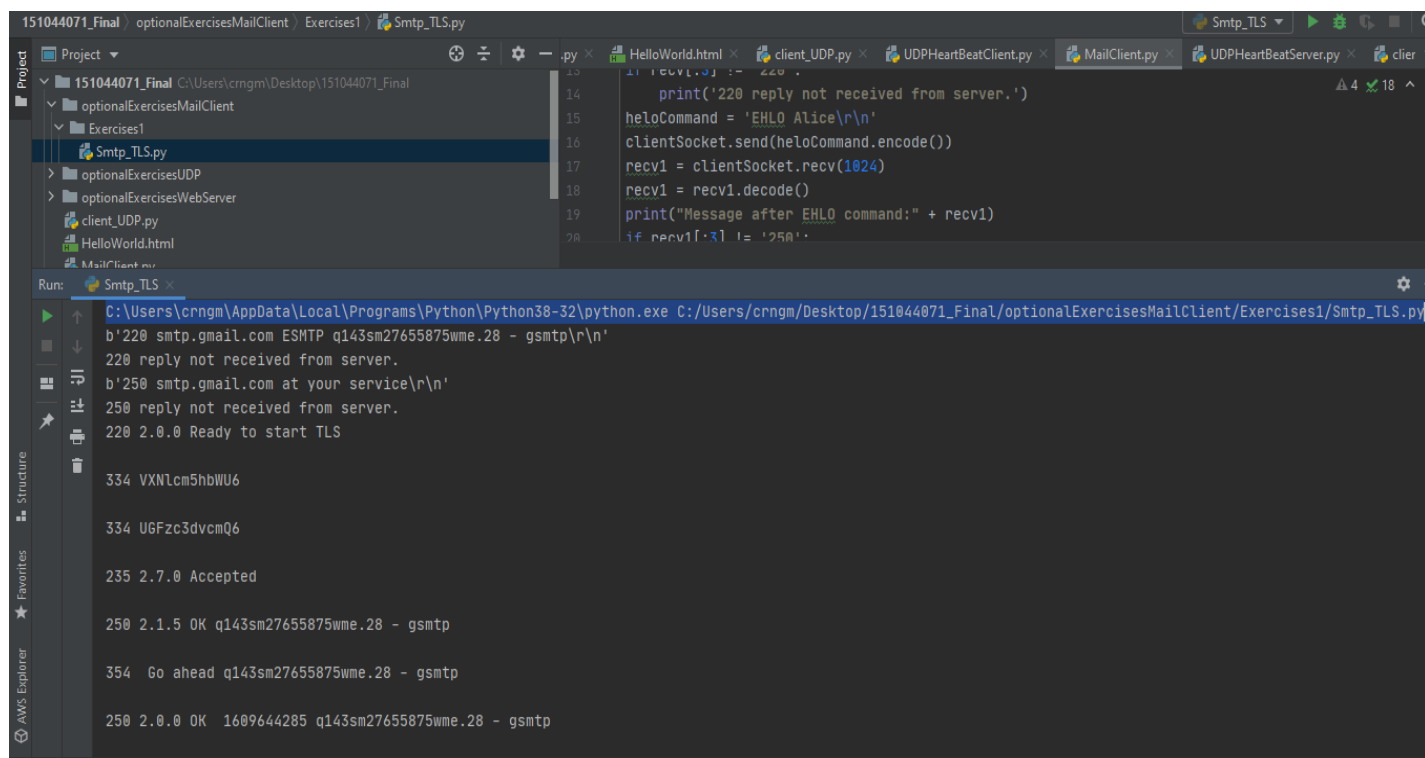


```
File Edit View Navigate Code Refactor Run Tools VCS Window Help 151044071_Final [C:\Users\crngm\Desktop\151044071_Final] - MailClient.py
151044071_Final MailClient.py
Run: MailClient
C:\Users\crngm\AppData\Local\Programs\Python\Python38-32\python.exe C:\Users\crngm\Desktop\151044071_Final\MailClient.py
Message after connection request:220 mail.smtp2go.com ESMTP Exim 4.92-S2G Sun, 03 Jan 2021 04:12:18 +0000

Message after EHLO command:250-mail.smtp2go.com Hello Alice [31.223.82.53]
250-SIZE 52428800
250-8BITMIME
250-DSN
250-PIPELINING
250-AUTH CRAM-MD5 PLAIN LOGIN
250-CHUNKING
250-STARTTLS
250-PRDR
250-HELP
```

Optional Exercises – MAIL CLIENT – Exercises :

Smtplib.py:



The screenshot shows a Python IDE with a project named '151044071_Final'. The file explorer on the left shows the project structure, including 'optionalExercisesMailClient' and 'Exercises1'. The 'Smtplib.py' file is selected. The code editor shows the following Python code:

```
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100
101
102
103
104
105
106
107
108
109
110
111
112
113
114
115
116
117
118
119
120
121
122
123
124
125
126
127
128
129
130
131
132
133
134
135
136
137
138
139
140
141
142
143
144
145
146
147
148
149
150
151
152
153
154
155
156
157
158
159
160
161
162
163
164
165
166
167
168
169
170
171
172
173
174
175
176
177
178
179
180
181
182
183
184
185
186
187
188
189
190
191
192
193
194
195
196
197
198
199
200
201
202
203
204
205
206
207
208
209
210
211
212
213
214
215
216
217
218
219
220
221
222
223
224
225
226
227
228
229
230
231
232
233
234
235
236
237
238
239
240
241
242
243
244
245
246
247
248
249
250
251
252
253
254
255
256
257
258
259
260
261
262
263
264
265
266
267
268
269
270
271
272
273
274
275
276
277
278
279
280
281
282
283
284
285
286
287
288
289
290
291
292
293
294
295
296
297
298
299
300
301
302
303
304
305
306
307
308
309
310
311
312
313
314
315
316
317
318
319
320
321
322
323
324
325
326
327
328
329
330
331
332
333
334
335
336
337
338
339
340
341
342
343
344
345
346
347
348
349
350
351
352
353
354
355
356
357
358
359
360
361
362
363
364
365
366
367
368
369
370
371
372
373
374
375
376
377
378
379
380
381
382
383
384
385
386
387
388
389
390
391
392
393
394
395
396
397
398
399
400
401
402
403
404
405
406
407
408
409
410
411
412
413
414
415
416
417
418
419
420
421
422
423
424
425
426
427
428
429
430
431
432
433
434
435
436
437
438
439
440
441
442
443
444
445
446
447
448
449
450
451
452
453
454
455
456
457
458
459
460
461
462
463
464
465
466
467
468
469
470
471
472
473
474
475
476
477
478
479
480
481
482
483
484
485
486
487
488
489
490
491
492
493
494
495
496
497
498
499
500
501
502
503
504
505
506
507
508
509
510
511
512
513
514
515
516
517
518
519
520
521
522
523
524
525
526
527
528
529
530
531
532
533
534
535
536
537
538
539
540
541
542
543
544
545
546
547
548
549
550
551
552
553
554
555
556
557
558
559
560
561
562
563
564
565
566
567
568
569
570
571
572
573
574
575
576
577
578
579
580
581
582
583
584
585
586
587
588
589
590
591
592
593
594
595
596
597
598
599
600
601
602
603
604
605
606
607
608
609
610
611
612
613
614
615
616
617
618
619
620
621
622
623
624
625
626
627
628
629
630
631
632
633
634
635
636
637
638
639
640
641
642
643
644
645
646
647
648
649
650
651
652
653
654
655
656
657
658
659
660
661
662
663
664
665
666
667
668
669
670
671
672
673
674
675
676
677
678
679
680
681
682
683
684
685
686
687
688
689
690
691
692
693
694
695
696
697
698
699
700
701
702
703
704
705
706
707
708
709
710
711
712
713
714
715
716
717
718
719
720
721
722
723
724
725
726
727
728
729
730
731
732
733
734
735
736
737
738
739
740
741
742
743
744
745
746
747
748
749
750
751
752
753
754
755
756
757
758
759
760
761
762
763
764
765
766
767
768
769
770
771
772
773
774
775
776
777
778
779
780
781
782
783
784
785
786
787
788
789
790
791
792
793
794
795
796
797
798
799
800
801
802
803
804
805
806
807
808
809
810
811
812
813
814
815
816
817
818
819
820
821
822
823
824
825
826
827
828
829
830
831
832
833
834
835
836
837
838
839
840
841
842
843
844
845
846
847
848
849
850
851
852
853
854
855
856
857
858
859
860
861
862
863
864
865
866
867
868
869
870
871
872
873
874
875
876
877
878
879
880
881
882
883
884
885
886
887
888
889
890
891
892
893
894
895
896
897
898
899
900
901
902
903
904
905
906
907
908
909
910
911
912
913
914
915
916
917
918
919
920
921
922
923
924
925
926
927
928
929
930
931
932
933
934
935
936
937
938
939
940
941
942
943
944
945
946
947
948
949
950
951
952
953
954
955
956
957
958
959
960
961
962
963
964
965
966
967
968
969
970
971
972
973
974
975
976
977
978
979
980
981
982
983
984
985
986
987
988
989
990
991
992
993
994
995
996
997
998
999
1000
```

The output window shows the following text:

```
C:\Users\crngm\AppData\Local\Programs\Python\Python38-32\python.exe C:/Users/crngm/Desktop/151044071_Final/optionalExercisesMailClient/Exercises1/Smtplib.py
b'220 smtp.gmail.com ESMTp q143sm27655875wme.28 - gsmtpl\r\n'
220 reply not received from server.
b'250 smtp.gmail.com at your service\r\n'
250 reply not received from server.
220 2.0.0 Ready to start TLS
334 VXNlcm5hbWU6
334 UGFzc3dvcmQ6
235 2.7.0 Accepted
250 2.1.5 OK q143sm27655875wme.28 - gsmtpl
354 Go ahead q143sm27655875wme.28 - gsmtpl
250 2.0.0 OK 1609644285 q143sm27655875wme.28 - gsmtpl
```

