17 Extend the client/server interaction to simulate a password dialogue. After receiving data from a client, the server returns access granted or access denied depending on whether the received data matches the password.

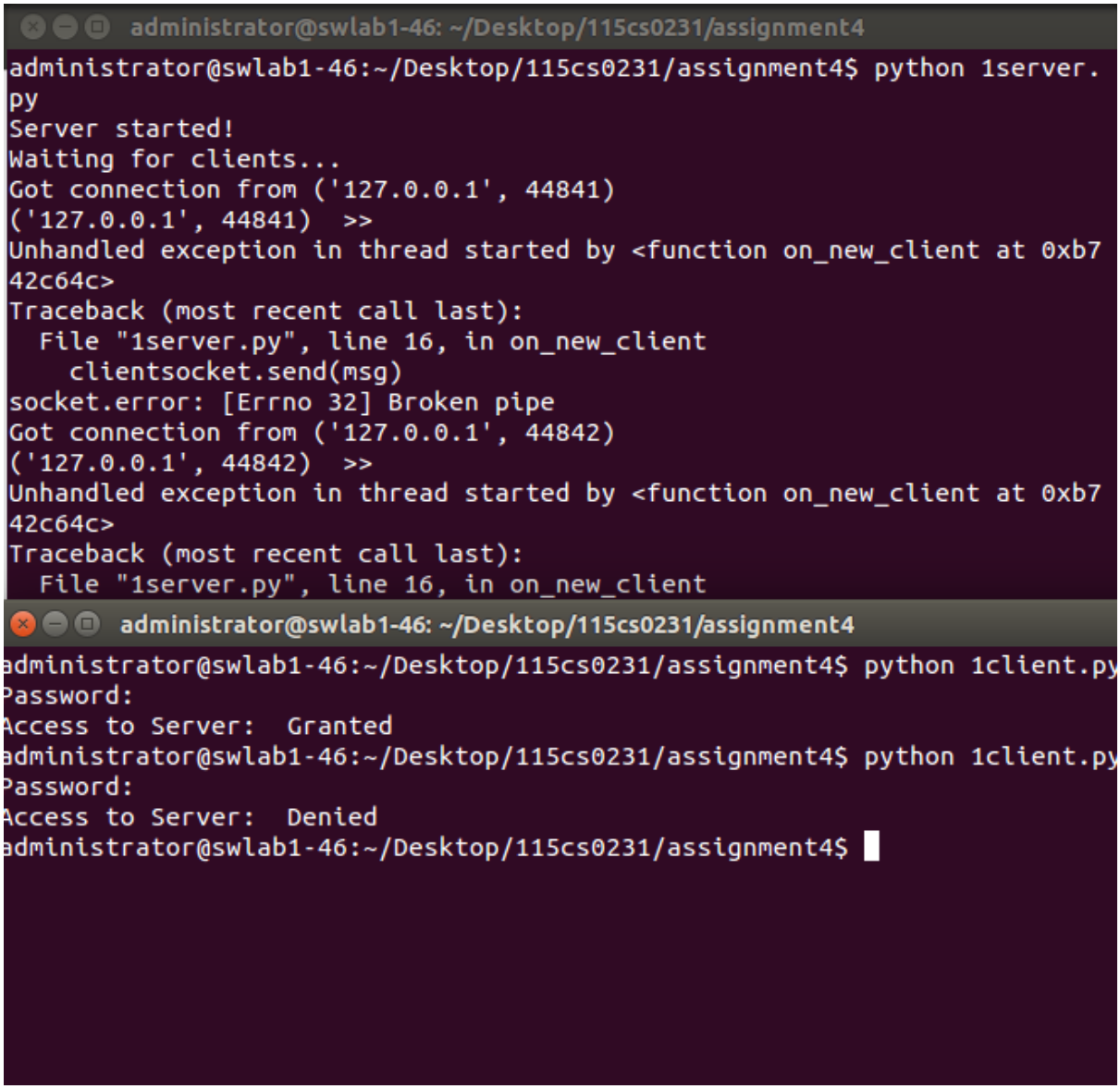
SERVER

```
import socket
import sys
import argparse

host = 'localhost'
data_payload = 2048

def echo_server(port):
    """ A simple echo server """
    # Create a UDP socket
    sock = socket.socket(socket.AF_INET, socket.SOCK_DGRAM)

    # Bind the socket to the port
    server_address = (host, port)
    print ("Starting up echo server on %s port %s" % server_address)

    sock.bind(server_address)

    while True:
        print ("Waiting to receive from client")
        data, address = sock.recvfrom(data_payload)

        #print ("received password bytes from %s" % (address))
        #print ("Data: %s" %data)

        if data == "PASSWORD":
            sent = sock.sendto("accept", address)
            print ("Client is granted access")
                                                    else:
                                        sent = sock.sendto("deny", address)
                                            print ("Client is denied access")

if __name__ == '__main__':
    parser = argparse.ArgumentParser(description='Socket Server Example')
    parser.add_argument('--port', action="store", dest="port", type=int, required=True)
    given_args = parser.parse_args()
    port = given_args.port
    echo_server(port)
```

CLIENT

```
import socket
import sys
import argparse

host = 'localhost'
data_payload = 2048

def echo_client(port):
    """ A simple echo client """
    # Create a UDP socket
    sock = socket.socket(socket.AF_INET, socket.SOCK_DGRAM)
```

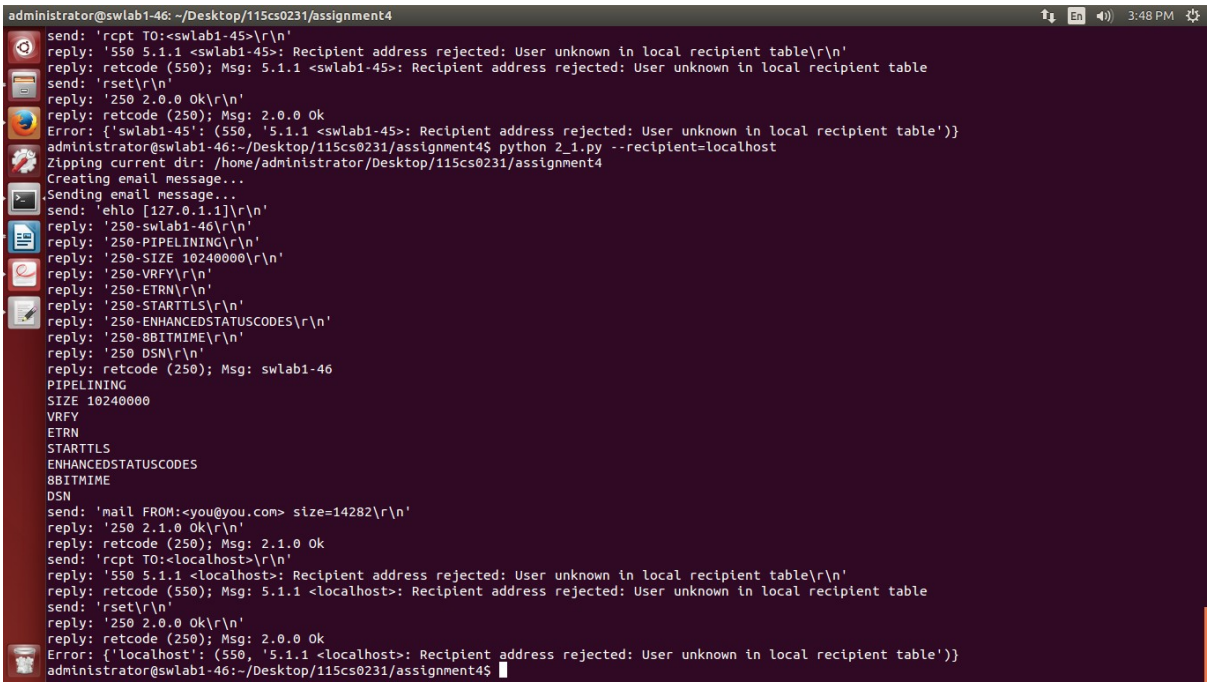
```
    server_address = (host, port)
    print ("Connecting to %s port %s" % server_address)
    message = 'This is the message.  It will be repeated.'

    try:

        # Send data
        message = "HELLO"
        print ("Sending password")
        sent = sock.sendto(message, server_address)

        # Receive response
        data, server = sock.recvfrom(data_payload)
                                            if data == "accept":
                                            print ("Access granted.")
                                                    else:
                                                                                print ("Access Denied.")

    finally:
        print ("Closing connection to the server")
        sock.close()

if __name__ == '__main__':
    parser = argparse.ArgumentParser(description='Socket Server Example')
    parser.add_argument('--port', action="store", dest="port", type=int, required=True)
    given_args = parser.parse_args()
    port = given_args.port
    echo_client(port)
```



18 Write a program that compress your working directory and email to a specific address?

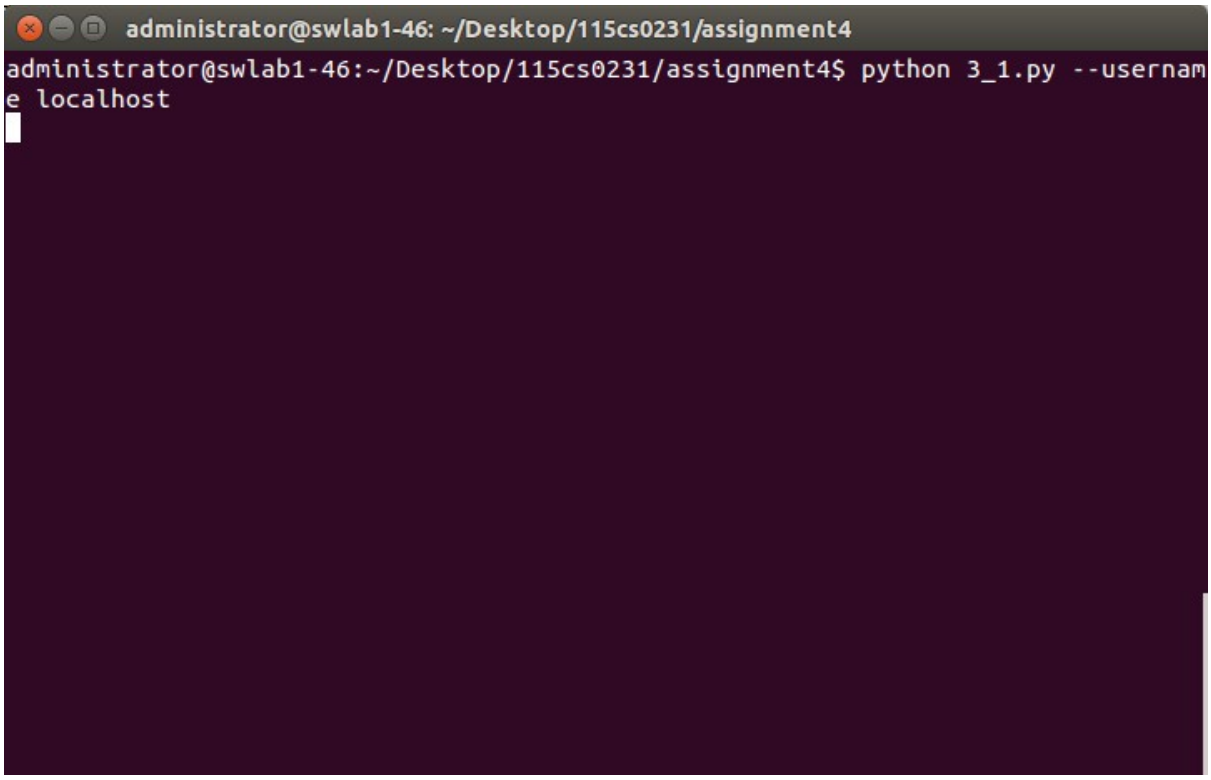
```
import os
import argparse
import smtplib
import zipfile
import tempfile
from email import encoders
from email.mime.base import MIMEBase
from email.mime.multipart import MIMEMultipart

def email_dir_zipped(sender, recipient):
    zf = tempfile.TemporaryFile(prefix='mail', suffix='.zip')
    zip = zipfile.ZipFile(zf, 'w')
```

```
    print ("Zipping current dir: %s" %os.getcwd())
    for file_name in os.listdir(os.getcwd()):
        zip.write(file_name)
    zip.close()
    zf.seek(0)

    # Create the message
    print ("Creating email message...")
    email_msg = MIMEMultipart()
    email_msg['Subject'] = 'File from path %s' %os.getcwd()
    email_msg['To'] = ', '.join(recipient)
    email_msg['From'] = sender
    email_msg.preamble = 'Testing email from Python.\n'
    msg = MIMEBase('application', 'zip')
    msg.set_payload(zf.read())
    encoders.encode_base64(msg)
    msg.add_header('Content-Disposition', 'attachment',
                   filename=os.getcwd()[-1] + '.zip')
    email_msg.attach(msg)
    email_msg = email_msg.as_string()

    # send the message
    print ("Sending email message...")
    try:
        smtp = smtplib.SMTP('localhost')
        smtp.set_debuglevel(1)
        smtp.sendmail(sender, recipient, email_msg)
    except Exception as e:
        print ("Error: %s" %str(e))
    finally:
        smtp.close()

if __name__ == '__main__':
    parser = argparse.ArgumentParser(description='Email Example')
    parser.add_argument('--sender', action="store", dest="sender",
default='you@you.com')
    parser.add_argument('--recipient', action="store", dest="recipient")
    given_args = parser.parse_args()
    email_dir_zipped(given_args.sender, given_args.recipient)
```



19 Write a python script to check email message from your Google account with Internet Message Access Protocol (IMAP).

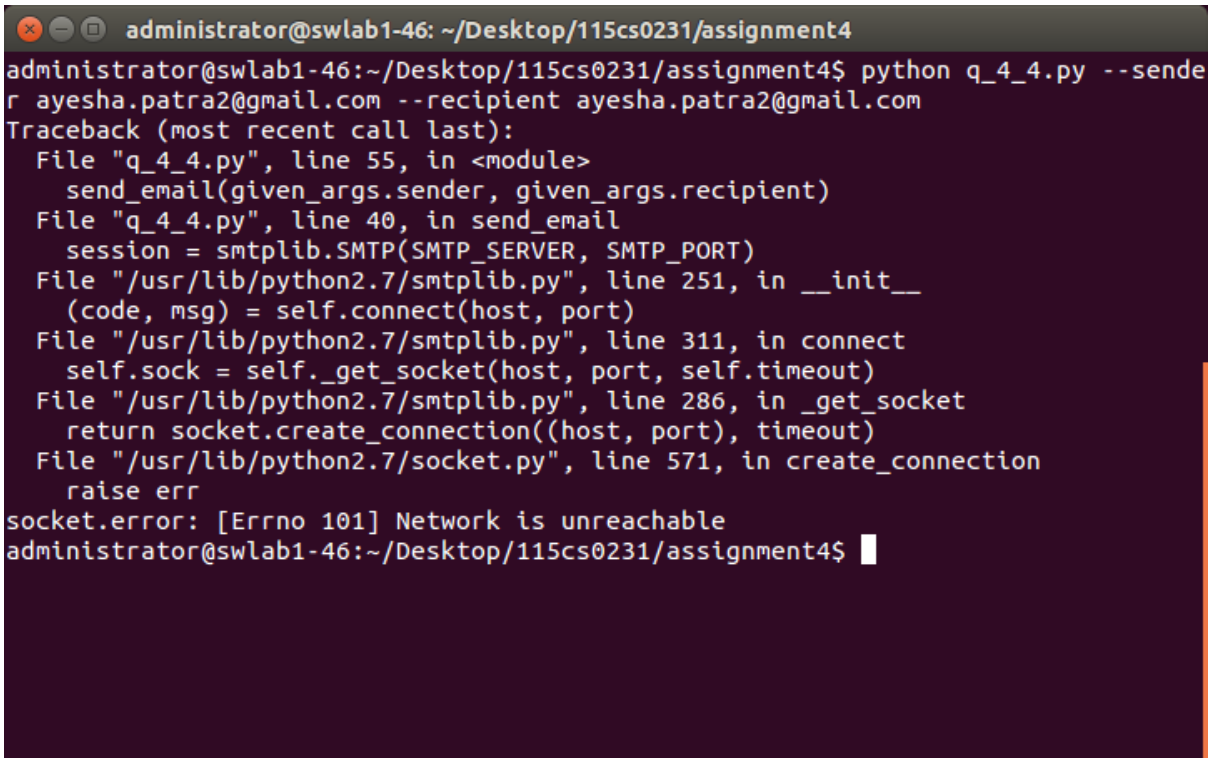
```
import argparse
import getpass
import imaplib
GOOGLE_IMAP_SERVER = 'imap.googlemail.com'
def check_email(username):
    mailbox = imaplib.IMAP4_SSL(GOOGLE_IMAP_SERVER, '993')
    password = getpass.getpass(prompt="Enter your Google password: ")
    mailbox.login(username, password)
    mailbox.select('Inbox')
    typ, data = mailbox.search(None, 'ALL')
    for num in data[0].split():
        typ, data = mailbox.fetch(num, '(RFC822)')
        print ('Message %s\n%s\n' % (num, data[0][1]))
        break
    mailbox.close()
    mailbox.logout()
if __name__ == '__main__':
    parser = argparse.ArgumentParser(description='Email Download Example')
    parser.add_argument('--username', action="store", dest="username",
default=getpass.getuser())
    given_args = parser.parse_args()
    username = given_args.username
    check_email(username)
```



20 Write a program to send an email to one or multiple users with an attachment via Gmail with Simple Mail Transfer Protocol (**SMTP**) server.

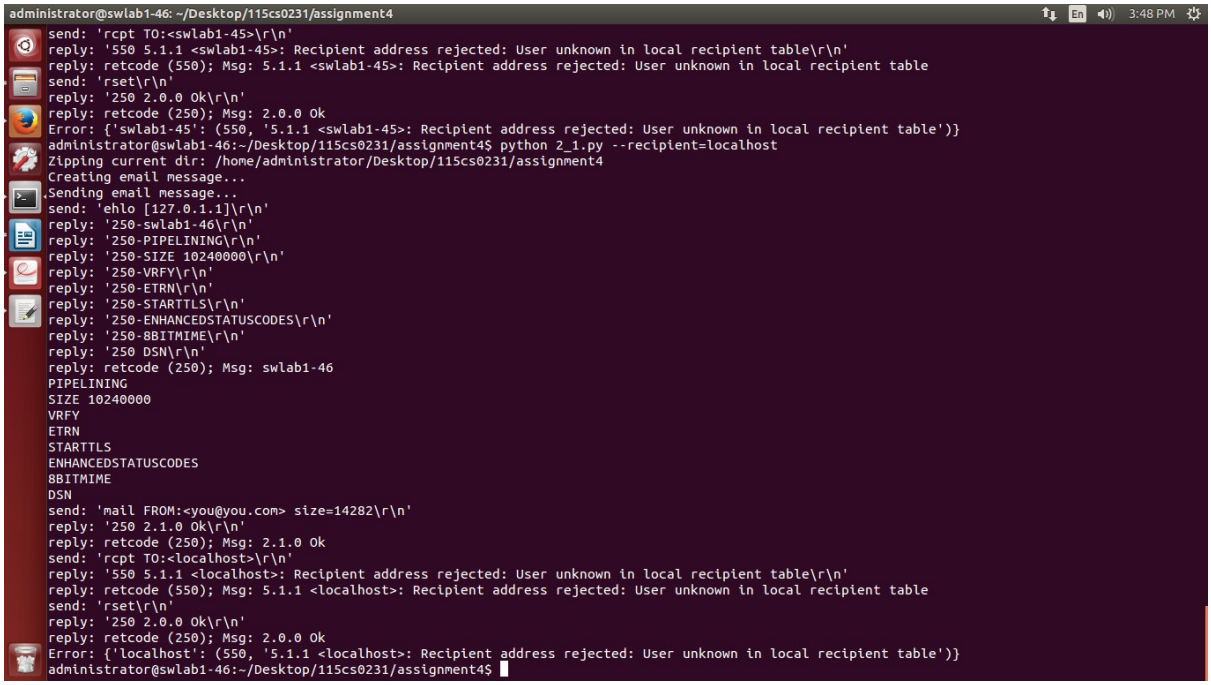
```
import argparse
import os
import getpass
import re
import sys
import smtplib

from email.mime.image import MIMEImage
from email.mime.multipart import MIMEMultipart
from email.mime.text import MIMEText

SMTP_SERVER = 'smtp.gmail.com'
SMTP_PORT = 587

def send_email(sender, recipient):
    """ Send email message """
    msg = MIMEMultipart()
    msg['Subject'] = 'Python Emaill Test'
    msg['To'] = recipient
    msg['From'] = sender
    subject = 'Python email Test'
    message = 'Images attached.'
    # attach imgae files
    files = os.listdir(os.getcwd())
```

```
    gifsearch = re.compile(".gif", re.IGNORECASE)
    files = filter(gifsearch.search, files)
    for filename in files:
        path = os.path.join(os.getcwd(), filename)
        if not os.path.isfile(path):
            continue
        img = MIMEImage(open(path, 'rb').read(), _subtype="gif")
        img.add_header('Content-Disposition', 'attachment', filename=filename)
        msg.attach(img)

    part = MIMEText('text', "plain")
    part.set_payload(message)
    msg.attach(part)

    # create smtp session
    session = smtplib.SMTP(SMTP_SERVER, SMTP_PORT)
    session.ehlo()
    session.starttls()
    session.ehlo
    password = getpass.getpass(prompt="Enter your Google password: ")
    session.login(sender, password)
    session.sendmail(sender, recipient, msg.as_string())
    print ("Email sent.")
    session.quit()

if __name__ == '__main__':
    parser = argparse.ArgumentParser(description='Email Sending Example')
    parser.add_argument('--sender', action="store", dest="sender")
    parser.add_argument('--recipient', action="store", dest="recipient")
    given_args = parser.parse_args()
    send_email(given_args.sender, given_args.recipient)
```



21 Write a program that establish secure connection to email server as Google or Yahoo through SMTP client secured with **Transport layer security** (TLS).

```
import os
import argparse
import smtplib
import zipfile
import tempfile
from email import encoders
from email.mime.base import MIMEBase
from email.mime.multipart import MIMEMultipart
def email_dir_zipped(sender, recipient):
    zf = tempfile.TemporaryFile(prefix='mail', suffix='.zip')
    zip = zipfile.ZipFile(zf, 'w')
    print ("Zipping current dir: %s" %os.getcwd())
    for file_name in os.listdir(os.getcwd()):
        zip.write(file_name)
    zip.close()
    zf.seek(0)
    print ("Creating email message...")
    email_msg = MIMEMultipart()
    email_msg['Subject'] = 'File from path %s' %os.getcwd()
    email_msg['To'] = ', '.join(recipient)
    email_msg['From'] = sender
    email_msg.preamble = 'Testing email from Python.\n'
    msg = MIMEBase('application', 'zip')
    msg.set_payload(zf.read())
    encoders.encode_base64(msg)
    msg.add_header('Content-Disposition', 'attachment',
                filename=os.getcwd()[-1] + '.zip')
    email_msg.attach(msg)
    email_msg = email_msg.as_string()
    print ("Sending email message...")
    try:
        smtp = smtplib.SMTP('localhost')
        smtp.set_debuglevel(1)
        smtp.sendmail(sender, recipient, email_msg)
    except Exception as e:
        print ("Error: %s" %str(e))
    finally:
        smtp.close()

if __name__ == '__main__':
    parser = argparse.ArgumentParser(description='Email Example')
    parser.add_argument('--sender', action="store", dest="sender",
default='you@you.com')
    parser.add_argument('--recipient', action="store", dest="recipient")
    given_args = parser.parse_args()
    email_dir_zipped(given_args.sender, given_args.recipient)
```


administrator@swlab1-46: ~/Desktop/115cs0231/assignment4
En
3:48 PM
send: 'rcpt TO:<swlab1-45>\r\n'
reply: '550 5.1.1 <swlab1-45>: Recipient address rejected: User unknown in local recipient table\r\n'
reply: retcode (550); Msg: 5.1.1 <swlab1-45>: Recipient address rejected: User unknown in local recipient table
send: 'rset\r\n'
reply: '250 2.0.0 Ok\r\n'
reply: retcode (250); Msg: 2.0.0 Ok
Error: {'swlab1-45': (550, '5.1.1 <swlab1-45>: Recipient address rejected: User unknown in local recipient table')}
administrator@swlab1-46:~/Desktop/115cs0231/assignment4$ python 2_1.py --recipient=localhost
Zipping current dir: /home/administrator/Desktop/115cs0231/assignment4
Creating email message...
Sending email message...
send: 'ehlo [127.0.1.1]\r\n'
reply: '250-swlab1-46\r\n'
reply: '250-PIPELINING\r\n'
reply: '250-SIZE 10240000\r\n'
reply: '250-VRFY\r\n'
reply: '250-ETRN\r\n'
reply: '250-STARTTLS\r\n'
reply: '250-ENHANCEDSTATUSCODES\r\n'
reply: '250-8BITMIME\r\n'
reply: '250 DSN\r\n'
reply: retcode (250); Msg: swlab1-46
PIPELINING
SIZE 10240000
VRFY
ETRN
STARTTLS
ENHANCEDSTATUSCODES
8BITMIME
DSN
send: 'mail FROM:<you@you.com> size=14282\r\n'
reply: '250 2.1.0 Ok\r\n'
reply: retcode (250); Msg: 2.1.0 Ok
send: 'rcpt TO:<localhost>\r\n'
reply: '550 5.1.1 <localhost>: Recipient address rejected: User unknown in local recipient table\r\n'
reply: retcode (550); Msg: 5.1.1 <localhost>: Recipient address rejected: User unknown in local recipient table
send: 'rset\r\n'
reply: '250 2.0.0 Ok\r\n'
reply: retcode (250); Msg: 2.0.0 Ok
Error: {'localhost': (550, '5.1.1 <localhost>: Recipient address rejected: User unknown in local recipient table')}
administrator@swlab1-46:~/Desktop/115cs0231/assignment4$