4.1 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.

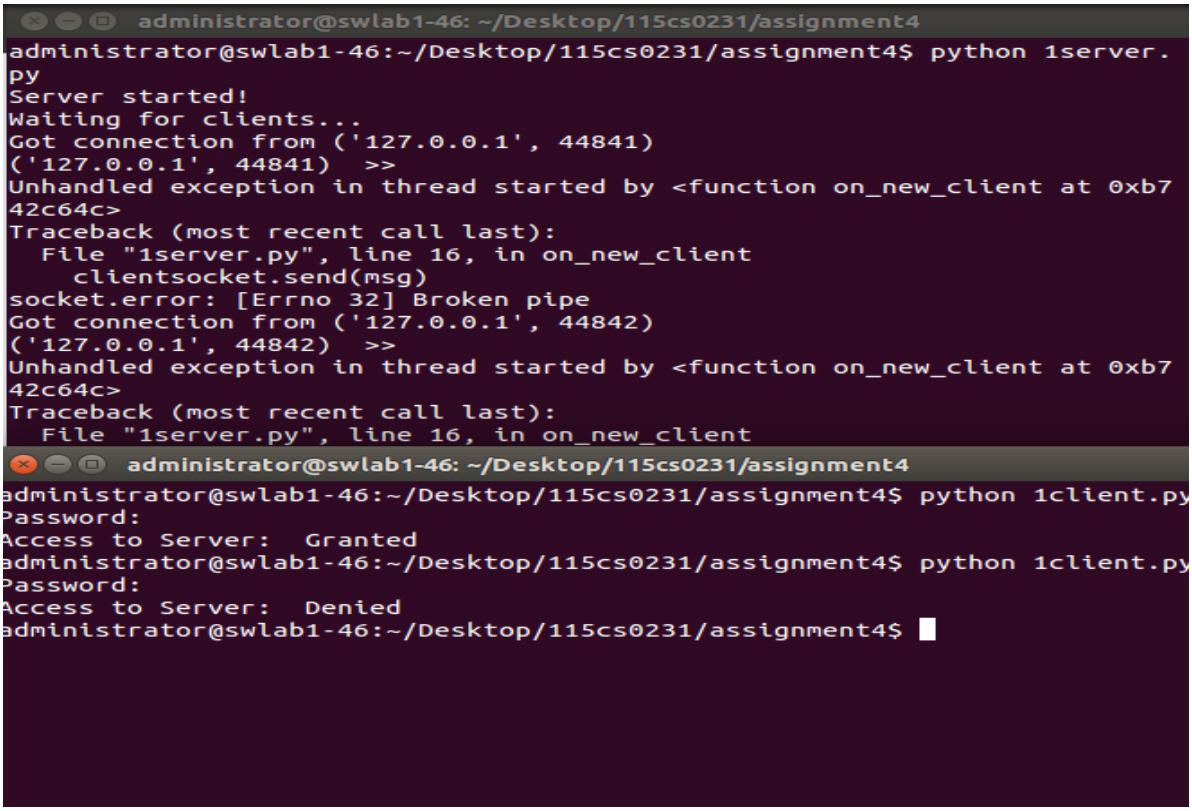
Server

```
import socket
import thread

def on_new_client(clientsocket,addr):
    f=1
    while (f==1):
        print addr, ' >> '
        while True:
            msg = clientsocket.recv(1024)
            #print msg
            if msg == 'speciallab1':
                msg = "Granted"
            else:
                msg="Denied"
                f=0
            clientsocket.send(msg)
    clientsocket.close()

s = socket.socket()
host = socket.gethostname()
port = 5000

print 'Server started!'
print 'Waiting for clients...'

s.bind((host, port))
s.listen(5)

while True:
   c, addr = s.accept()
   print 'Got connection from', addr
   thread.start_new_thread(on_new_client,(c,addr))
s.close()
```

Client

```
import socket
import getpass
s = socket.socket()
host = socket.gethostname()
port = 5000

msg = getpass.getpass('Password:')
```

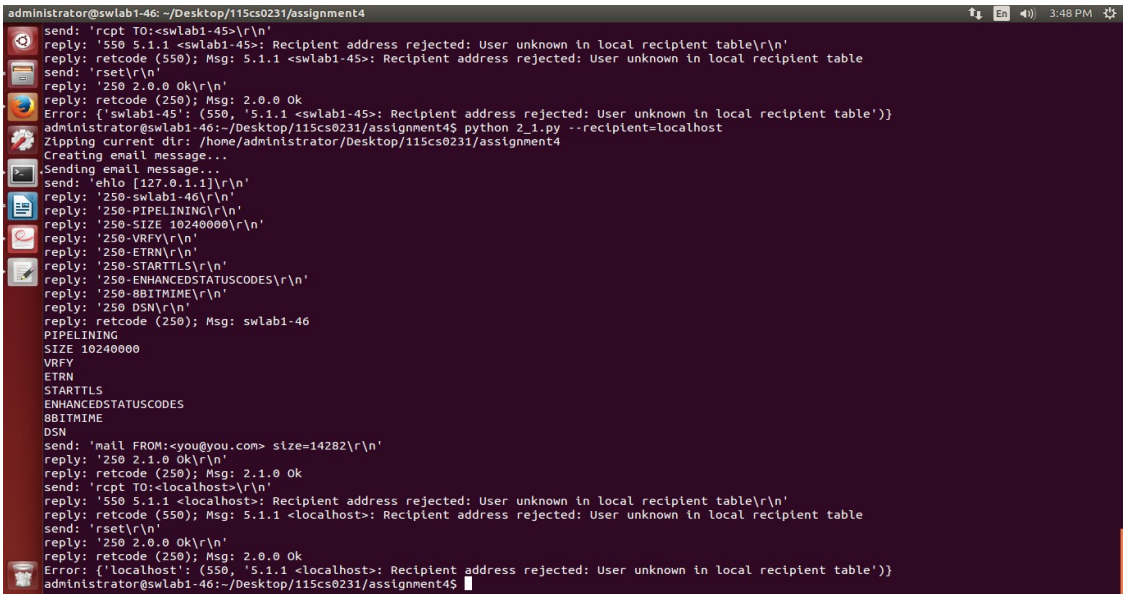
```
s.connect((host, port))
s.send(msg.encode())

print 'Access to Server: ',str(s.recv(1024).decode())
s.close()
```



4.2 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


sender='115cs0231@nitrkl.ac.in'
recipient='115cs0231@nitrkl.ac.in'
```

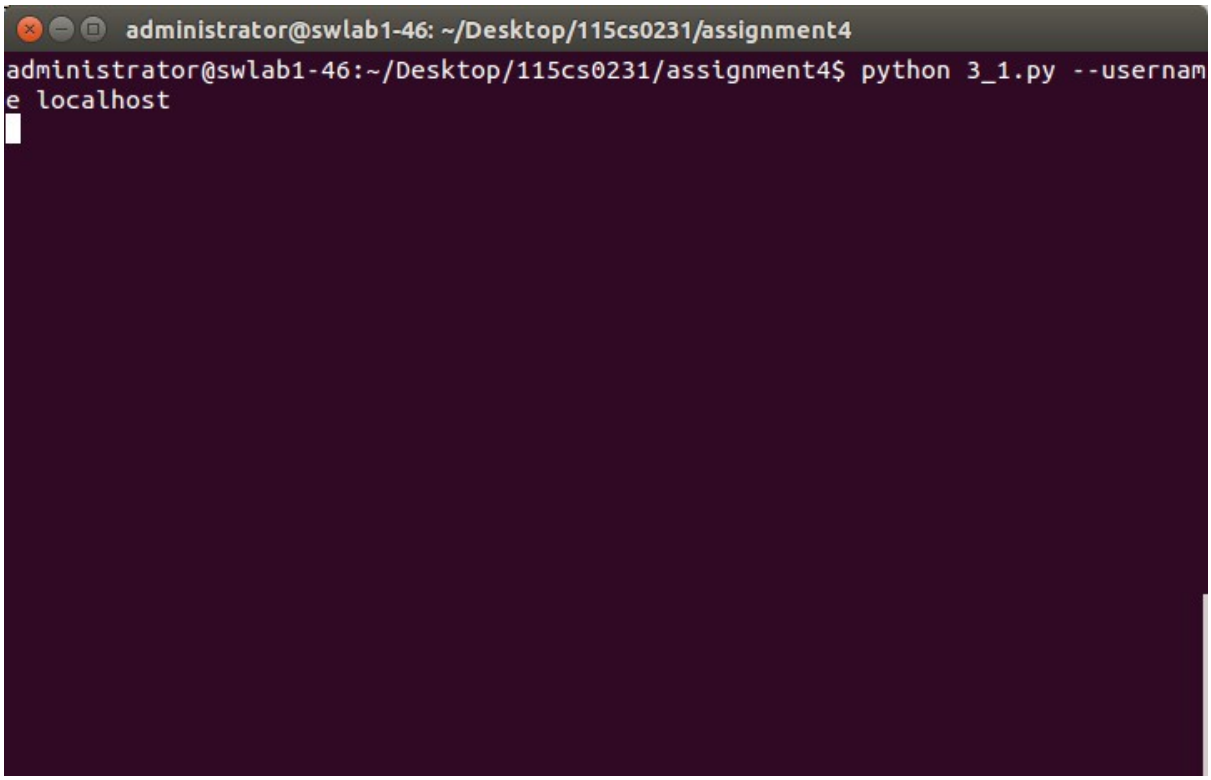
```
zf = tempfile.TemporaryFile(prefix='mail', suffix='.zip')
zip = zipfile.ZipFile(zf, 'w')
#print ("Zipping current dir: %s" %os.getcwd())
for file_name in os.listdir(os.getcwd()):
zip.write(file_name)
zip.close()
zf.seek(0)

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()
```



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

```
import imapclient
imapObj = imapclient.IMAPClient('imap.gmail.com', ssl=True)
imapObj.login(' ayesha199718@gmail.com ', ' MY_SECRET_PASSWORD ')
'ayesha199218@gmail.com '
imapObj.select_folder('INBOX', readonly=True)
UIDs = imapObj.search(['SINCE 05-Jul-2014'])

rawMessages = imapObj.fetch([40041], ['BODY[]', 'FLAGS'])
import pyzmail
message = pyzmail.PyzMessage.factory(rawMessages[40041]['BODY[]'])
message.get_subject()

message.get_addresses('from')

message.get_addresses('to')
```

```
message.get_addresses('cc')

message.get_addresses('bcc')
message.text_part != None
message.text_part.get_payload().decode(message.text_part.charset)
message.html_part != None
message.html_part.get_payload().decode(message.html_part.charset)
imapObj.logout()
```



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

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