Exercise 1: Send an Email

The objective of this exercise is to implement the sendmail.py command to send an email using the SMTP protocol, with or without authentication, in a secure or non-secure version.

To represent email messages in this project, we will use the EmailMessage type defined in the Python email.message module.

Here's an example:

```python
import email
import email.message

msg = email.message.EmailMessage()
msg['From'] = "toto@pouet.com"
msg['To'] = "tutu@pouet.com"
msg['Subject'] = "Test"
msg['Date'] = "Wed, 21 Jul 2021 14:00:00 +0200"
msg.set_payload("Hello World!")
# print email message as string
print(msg)
# convert email message into bytes
data = msg.as_bytes()
# convert message bytes into email
msg2 = email.message_from_bytes(data)
```

To complete this exercise, you have the following files:

- sendlib.py
- sendmail.py

To simplify this project, we will limit ourselves to the following SMTP commands: EHLO, NOOP, MAIL FROM, RCPT TO, DATA, QUIT. Regarding authentication, we will only use the AUTH PLAIN authentication method based on the Unix login/password of a user. For the secure version of SMTP, we will implement SMTPS, which uses regular SMTP in a secure socket with SSL/TLS. In particular, there is no requirement to implement the STARTTLS command, which is another way to secure the SMTP connection. Additionally, we will limit ourselves to a single recipient per email.

Here is the description of the arguments for the sendmail.py command, whose code is provided:

```
$ ./sendmail.py  -h
usage: sendmail.py [-h] [-H HOST] [-P PORT] [-S] [-A] [-l LOGIN] [-p PASSWORD]
                   [-f SENDER] [-t RECIPIENT] [-s SUBJECT] [-b BODY] [-v]
options:
  -h, --help                         help
  -H HOST, --host HOST               server host
  -P PORT, --port PORT               server port
  -S, --secure                       secure mode
  -A, --auth                         auth mode
  -l LOGIN, --login LOGIN            user login
  -p PASSWORD, --password PASSWORD   user password
  -f SENDER, --from SENDER           mail sender
  -t RECIPIENT, --to RECIPIENT       mail recipient
  -s SUBJECT, --subject SUBJECT      mail subject
  -b BODY, --body BODY               mail body
  -v, --verbose                      verbose
```

Here is a first example of using this command, which sends a test email using all default parameters. In this example, the email is sent without authentication or a secure connection. By default, the sendmail.py command will target the localhost:10025 server (LMA SMTP server).

~~~
$ ./sendmail.py
----------------------------------------
From: toto@pouet.com
To: tutu@pouet.com
Subject: Test
Date: Tue, 28 Nov 2023 09:37:14 +0100

Hello World!
----------------------------------------
[Success]
```
~~~

Here is a second example with the -A option implementing login authentication. Note that the email sender (toto@pouet.com in this example) must authenticate with the SMTP server using their login "toto" and password "toto".

~~~
$ ./sendmail.py -A -l toto -p toto
```
~~~

We will use the -S option to secure the connection over SSL/TLS. By default, the sendmail.py command will target the localhost:10465 server (LMA SMTPS server). Since the LMA server uses a self-signed certificate, you need to ignore its verification to avoid connection rejection. Add the following options to the SSL context:

```
context.check_hostname = False      # no hostname verification
context.verify_mode = ssl.CERT_NONE # no certificate verification
```

If everything works correctly, you should be able to use your command to send an email to the smtpauth.u-bordeaux.fr server in plaintext without authentication from the CREMI, or in a secure connection from outside with your academic credentials.

For example:

~~~
$ ./sendmail.py -H smtpauth.u-bordeaux.fr -P 465 -S -A -l auesnard -p xxxxxxxx \
                -f aurelien.esnard@u-bordeaux.fr -t aurelien.esnard@free.fr
```
~~~

Submission: To complete this exercise, you must implement the functions in the sendlib.py file that are used by sendmail.py. Then submit your code on Moodle. Do not modify the sendmail.py file.

Note: It is assumed that the commands and responses to these commands are lines of no more than MAXLINE=1024 characters.