1. You are encouraged to work in groups of two or three. If you cannot find partners, you may work alone. **2.** Please submit **one** copy of the assignment using `subversion`; if you are working with a partner, both names should appear on the assignment. **3.** Note that you will get a grade of zero if your program does not run.

In this assignment you are going to write a python tool that verifies `SSL` certificates. Your program should be invoked as follows:

`python sslverify` ⟨url⟩ ⟨port⟩

If the port is not given, you program should assume that it is the standard `SSL` port `443`. Once invoked, your program should retrieve the `SSL` certificate of the server, and print out information about it, as well as verify it against the VeriSign data base of root certificates, which can be found here: `http://www.verisign.com/support/roots.zip`

The certificate information should be the following: Subject, Not valid before, Not valid after, Issuer, Fingerprint, as well as the name and serial number of the issuer. For example, when invoked as follows

```
python sslverify easywebsoc.td.com 443
```

the output should be:

```
Subject:
  Country: CA
  St/Pr: Ontario
  Locality: Toronto
  Organization: Toronto Dominion Bank
Not valid before: May 13 00:00:00 2010 GMT
Not valid after: May 13 23:59:59 2011 GMT
Issuer: VeriSign
Fingerprint: 3F:37:76:4C:11:B6:3B:EF:37:12:50:0F:11:01:2D:2A:42:C7:EF:10
Certificate: VeriSign Class 3 Secure Server CA - G2
Status: OK
```

where the "Status: OK" means that the certificate is signed by VeriSign with the appropriate key ("VeriSign Class 3 Secure Server CA - G2").

You should use `openssl` to accomplish all this, as it has a lot of the certificate functionality built in; you can access `openssl` from a python program with an `import os`, and then, for example, `print os.system('openssl verify file.pem')`

This assignment involves little programming, and it is meant as an introduction to certificates, `SSL`, and PKI.

It is part of the assignment to do the required background research on how to check `SSL` certificates with python and `openssl`; keep in mind that a substantial component of any software project is to solve/eliminate the underlying technical difficulties. This often means hitting manuals and Google. To star you off, a good place to begin is `http://docs.python.org/library/ssl.html` . For the relevant `openssl` you may start at `http://www.madboa.com/geek/openssl/#cert-exam` .

For this assignment we want you to provide good documentation; write concisely in your report how you used the tools, the meaning of the different invocations of `openssl` and `SSL` functionality of Python. Make this part of "how you solved the problem." As usual, also write how was your program tested, any problems you encountered, and a sample run of the program (and an output dump of the sample run).