Term Assignment #1 announced on 20 Nov, due at 23:59 7 Dec for both classes.
Upload **your code (.py, .ipynb, or .java)** to the homework submission system.
Upload **a document (.pdf)** to the homework submission system.

Q1. Modify the example code of Lab #2 and make it as a HTTP server that supports the following functions.

- The server runs at **TCP port 80** on **127.0.0.1** IP address.
- Implement **non-persistent HTTP**.
- Implement a **multi-thread** HTTP server.
- The server can accept a HTTP **GET** request with request URL '**get.html**'. The server can reply a HTTP reply message with **200 OK** (and any necessary HTTP headers) and an html message containing a simple html "<html><body>good.html</body></html>".
- The server can accept a HTTP **GET** request with request URL '**redirect.html**'. The server can reply a HTTP reply message with **301 Moved Permanently** and any necessary HTTP headers (including **Location:**). The redirected page is good.html. get.html
- The server can accept a HTTP **GET** request with request URL '**notfound.html**'. The server has no such file so it replies a HTTP reply message with **404 Not Found** and any necessary HTTP headers.
- The server can accept a HTTP **HEAD** request with request URL '**head.html**'. The server can reply a HTTP reply message with **only** necessary HTTP headers.
- The server can accept a HTTP **POST** request with request URL '**post.html**'. The post message included in the HTTP body is 'id=*yourStudentID*'. The server can parse the POST content and reply a HTTP reply message with **200 OK** (and any necessary HTTP headers) and an html message containing a simple html "<html><body>*yourStudentID*</body></html>". (You may create a HTML form to simulate such POST action.)

Q2. The document should include the following items.

- Draw your HTTP server FSM.
- Please capture the HTTP packets by Wireshark. There should be 5 sets (GET + 200, GET + 301 + GET + 200, GET + 404, HEAD, POST) of packets. Please printscreen the packets in Wireshark and show their layer 7 contents (so that TA knows your implementation and packets are correct).

Note:

- TA will use a commercial browser (such as chrome) to connect to your server and see if the expected behavior occurs, i.e., showing the correct good.html, redirecting from redirected.html to good.html, and showing a 404 on browser.
- Your server should be runnable without any user input. TA will use a script to automatically execute your server codes. You should test your server codes in the same way. Otherwise, no points will be given.