SI 601 Winter 2016 Homework 2 (100 points)

Due at 5:30pm on Wednesday, Jan. 20, 2016

In this homework you will apply your knowledge of Python regular expressions to an important real-world problem: analyzing web access logs. A web access log records every HTTP request made to a web server, along with features such as the date, time, IP address of the computer making the HTTP request, and the request's resulting status code (e.g. 200 == success). Log analysis and filtering is an important method for identifying valid vs. suspicious requests, as you will see in this assignment.

First, download the attached si601_w16_hw2.zip and unzip it. You should see these files:

```
access_log.txt

valid_access_log_desired_output.txt

invalid_access_log_desired_output.txt

suspicious_ip_summary_desired_output.csv
```

The file `access_log.txt` came from a security researcher who set up a 'honeypot' server to attract hackers, in order to record the requests used in hacking attempts, to learn about their methods and origins. The original file was downloaded from http://log-sharing.dreamhosters.com/, and then the first 200 lines were extracted for use in this assignment. The access log was generated by a popular Web server program, Apache, and its format is explained here (http://httpd.apache.org/docs/2.2/logs.html), under the Access Log section. A couple of fields in the provided file are set to '-' because the data has been sanitized so that there won't be privacy issues.

You should browse through 'access_log.txt' to familiarize yourself with the format and think about how you might extract the data out of it.

Your goal is to create three output files:

(1) A text file named "valid_access_log_youruniquename.txt" that contains all of the valid log lines in access_log.txt. The notion of "valid" log lines is defined below.
(2) A text file named "invalid_access_log_youruniquename.txt" that contains all of the invalid log lines in access_log.txt. A log line in access_log.txt should go to either valid_access_log_youruniquename.txt or invalid_access_log_youruniquename.txt, and therefore, the total line number in these two output files must be 200.
(3) A summary file named "suspicious_ip_summary_youruniquename.csv" that contains the number of access attempts (same as the number of log lines) from each unique client IP address in the "invalid_access_log_youruniquename.txt" file. The IP addresses should be sorted in decreasing order of the attempts.

```
195.82.31.125 - - [09/Mar/2004:22:03:09 -0500] "GET
http://www.goldengate.hu/cgi-bin/top/topsites.cgi?an12 HTTP/1.0" 200 558
"http://Afrique" "Mozilla/4.0 (compatible; MSIE 5.5; Windows 98)"
```

Use the following rules to determine if a line should be counted as a valid line. A line in the log file represents a valid visit only if these conditions are true:

1. If the HTTP verb is GET, POST or HEAD (uppercase), then the following three conditions must be satisfied:
    a. The status code is 2xx, 3xx, or 5xx. See https://en.wikipedia.org/wiki/List_of_HTTP_status_codes for their meanings.
    b. AND the URL being accessed starts with `http://` or `https://` (case-insensitive, it can be HTTP, Http, etc).
    c. The value for each field in the URL query string (https://en.wikipedia.org/wiki/Query_string) do not exceed 80 characters long after URL decoding. To decode and parse the URL string, you should use the urlparse module (https://docs.python.org/2/library/urlparse.html).
2. If the HTTP verb is CONNECT, then it is followed by a hostname or IP address with status code of 2xx, 3xx, or 5xx.

As an example, we do not count lines like the following as valid:

```
68.48.142.117 - - [09/Mar/2004:22:41:42 -0500] "GET
/scripts/..%c1%9c../winnt/system32/cmd.exe?/c+dir HTTP/1.0" 200 566 "-" "-"
```

This is because /scripts/ obviously doesn't look like a web address since it doesn't start with `http://` or `https://`). This is a prime example of an attempt to get a file stored locally on that proxy server to exploit vulnerabilities.

Another example is on line 72 of access_log.txt. In the query string on that line, the supplied value string for the field 'l' is VERY long. This violates rule 1(c), so this line is not a valid line. A normal user wouldn't enter that much gibberish into a web form. This case is an example of the "buffer overflow" attack attempt (https://en.wikipedia.org/wiki/Buffer_overflow)

Note that the 80-character limit is on the VALUES in the query string, not the whole query string. For example, the following log line

```
24.168.72.174 - - [09/Mar/2004:22:11:38 -0500] "GET
http://sbc1.login.scd.yahoo.com/config/login?.redir_from=PROFILES?&.tries=1&.sr
c=jpg&.last=&promo=&.intl=us&.bypass=&.partner=&.chkP=Y&.done=http://jpager.yah
oo.com/jpager/pager2.shtml&login=exodus_510&passwd=matthew HTTP/1.0" 200 566 "-
" "-"
```

is valid because no value for any field is longer than 80 characters.

Useful debugging tools:

To compare your output and desired output files, you can use a handy "file compare" utility to see which lines are different. On Windows, try **http://winmerge.org/** For the Mac, some people recommend **http://www.sourcegear.com/diffmerge/**

Milestones:

1. (40 points) Write a function named is_valid(line) that will return True if line is valid, and False if line is invalid. This will involve regular expression matching.
2. (20 points) Write code that reads the input log file and process it by calling your is_valid function for each line, and outputs the two valid/invalid access log output files.
3. (20 points) Write a function named extract_ip(line) that will return the client IP address in a log line.
4. (20 points) Write code that calls your extract_ip(line) on each invalid log line, and then count the log lines seen for each IP address, and then output the results in a CSV file named suspicious_ip_summary_youruniquename.csv

What to submit:

A zip file named si601_w16_hw2_***youruniquename***.zip that contains:

1. Your python program file: si601_w16_hw2_***youruniquename***.py
2. Your output files:

    suspicious_ip_summary_***youruniquename***.csv

    invalid_access_log_***youruniquename***.txt

    valid_access_log_***youruniquename***.txt