Week 9 - Using Regular Expressions to find patterns in text

Regular Expression Tester: http://www.regexpal.com/
Python Regular Expression Documentation:
https://docs.python.org/2/library/re.html

Regular Expression patterns in text
www.regexpal.com
Try in the RegEx Pal Tester:
[wW]
[em]
[A-Z]
[a-z]
[A-Za-z]
[!] some non-alphabetic characters
[.]

Used to extract simple expressions from text

```
>>> import re
>>> line = '1.1.1.1 - - [21/Feb/2014:06:35:45 +0100] "GET
/robots.txt HTTP/1.1" 200 112 "-" "Mozilla/5.0 (compatible;
Googlebot/2.1; +http://www.google.com/bot.html)"'
```

First we extract the date by looking for the string inside of square brackets. To do this, we write a pattern that finds the left square bracket followed by any number of characters that are not a right square bracket and then the right square bracket. A set of parentheses captures the string inside the square brackets.

```
>>> pdate = re.compile("\[([^\]]+)\]")
>>> resultlist = pdate.findall(line)
>>> resultlist
['21/Feb/2014:06:35:45 +0100']
```

```
>>> datestring = resultlist[0]
>>> datestring
'21/Feb/2014:06:35:45 +0100'
```

Next, suppose we want to find what server contents are being requested by the GET requests. We can use a similar regular expression to find strings inside the double quotes, where the content of the first set of double quotes represents the request to the server. (Not all of these are GET requests, but they all are formatted with double quotes in this log.)

```
>>> pquotes = re.compile('\"([^\"]+)\"')
>>> quotecontents = pquotes.findall(line)
>>> quotecontents
['GET /robots.txt HTTP/1.1', '-', 'Mozilla/5.0 (compatible;
Googlebot/2.1; +http://www.google.com/bot.html)']
```

The request string is the first of these.

```
>>> requeststring = quotecontents[0]
>>> requeststring
'GET /robots.txt HTTP/1.1'
```

Now we get the actual server content request by selecting the part after the GET and before the HTTP version information.

```
>>> prequest = re.compile('GET ([\w/.]+) HTTP')
>>> request = prequest.findall(requeststring)
>>> request
['/robots.txt']
```

Now some lines don’t have GET requests.

```
>>> line2 = '7.7.7.7 - - [21/Feb/2014:08:51:34 +0100] "-" 400 0 "-
" "-"'
```

In this case, when we match the GET part, we get an empty string in return because no matches occurred.

```
>>> quotecontents = pquotes.findall(line2)
>>> quotecontents
['-', '-', '-']
>>> requeststring = quotecontents[0]
>>> request = prequest.findall(requeststring)
>>> request
[]
```

As one sidebar, note that we can convert the datestring to a Python datetime object if we want, using strptime.

```
>>> from datetime import datetime
>>> tt = datetime.strptime(datestring, "%d/%b/%Y:%H:%M:%S
%z")
>>> tt
datetime.datetime(2014, 2, 21, 6, 35, 45,
tzinfo=datetime.timezone(datetime.timedelta(0, 3600)))
```

As a second sidebar, let's look at an additional regular expression example. Suppose we want to extract more than one subtext pattern from a text at one time. We can try that with our server log line by making a pattern that looks for the date and the first quoted string at one time.

```
>>> pboth = re.compile('\[([^\]]+)\][^"]+\"([^\"]+)\"')
>>> result = pboth.findall(line)
>>> result
[('21/Feb/2014:06:35:45 +0100', 'GET /robots.txt HTTP/1.1')]
```

There is one result in the result list, and it is the pair of things that matched.