Programming for Data Analytics
Lab Topic 04-regex

Introduction.

This is not marked

The first activity on this sheet is a quick quiz, answers are at the end of the sheet.

Please download an accces.log.txt file (I changed the extension to .txt because of git.ignore), either from my repository
(in ***PFDA-courseware***/*code*/*topic04 cleaning data*/*data*)
here

- PFDA-courseware/code/topic04 cleaning data/data/access.log.txt at main · andrewbeattycourseware/PFDA-courseware

Or

From splunk

- https://docs.splunk.com/images/Tutorial/tutorialdata.zip

(you will need to unzip this file and find an access.log)

quiz:

1. Look at the following code, it searches through a file and prints out the lines that have some text that matches the regular expression in the variable regex.
   In this case it will print out the entire file quiz.txt

```python
# this is code for for the quiz
import re

regex = ".*"
filename = "./sample-files/quiz.txt"

with open(filename) as quizFile:
    for line in quizFile:
        searchResult = re.search(regex, line)
        if (searchResult):
            matchingLine = line
            # I set (variable) matchingLine: str line will already have a \n
            print (matchingLine, end="")

```

   Here is quiz.txt (I have left in the line numbers)

```
1   hello
2   Hello
3   Hello World
4   Helo John
5           Hello mary
6   Hellllllllllllo Anamaniacs
7   var = 123
8   change this #this will change
9   what [about] this.
10
```

   What lines will be printed out for the following regular expressions? Note that regular expressions in python are case sensitive:

   a. hello
   b. Hello
   c. ^Hello
   d. ^Hell*o
   e. ^Hell+o
   f. ^Hell?o
   g. ^hello [A-Z]
   h. ^Hello [A-Z]
   i. =
   j. #
   k. [
   l. ^$

Practice regex

2. Open up the access.log.txt (or access.log if you got it directly) file in vscode, I suggest that you copy the first few lines of it and put into another file to use (the sample access.log has a lot of lines), I called mine smallerAccess.log

In the search dialog (ctrl-f) select the regex icon (one on the right, I show this in the lecture)
And search for:

	Search for	You can use this regex	comments
a.	All the numbers	[0-9]	Or \d would do
b.	The first digits of the ip address at the start of each line	^[0-9]+\. Or ^[0-9]{1,3}	The dot needs to be escaped. {1,3} means one to three times
c.	The digits at the end of each line	\d+$	I choose to use \d this time we could have used [0-9]
d.	The dates and times We will use this below(3)	\[.*\]	[needs to be escaped .* is any character repeated 0 or more times. This search will return the [] as well
e.	The times	:[0-9:]{8}	I got lazy and am not checking that they are properly formatted and this would not include the first : [0-9]{2}:[0-9]{2}:[0-9]{2}
f.	The variable names in the urls	\w+=	\w is anything that can be in a word
g.	The variable values in the urls	=\w+	
h.	The last 2 triples of an Ip address. We will use this below (5)	\d{1,3}\.\d{1,3}[^.]	[^ means not one of these \. Is a dot So [^\.] means not a dot. It might have been easer to put in a space 😊 Ok this does not work…… It finds numbers in the document that have one digit followed by another number eg 1.5
i.	Find all the ip address	\d{1,3}\.\d{1,3}\.\d{1,3}\.\d{1,3}	I found the "not a dot" through up weird results so this just finds the ip and we can select the parts we want to keep in the replacement. See the code below

Some code

3. Try some of the regular expressions in following code to see what python returns
   I suggest that you use a sample file that only has a few lines, just to prove it works.

For this sample I have chosen *d. the dates and time*, this will return the date/time in this format
[15/Feb/2021:18:44:39]
If I did not want the [and] then I could use the string splitter to select from the 2 character to the second last character in the string.

```
# This is code will find some text in an access file
# Author: Andrew Beatty

import re

regex = "\[.*\]"
filename = "./sample-files/smallerAccess.log"

with open(filename) as inputFile:
    for line in inputFile:
        foundTextList = re.findall(regex, line)
        if (len(foundTextList)!= 0):
            #print(foundTextList)
            foundText = foundTextList[0]
            print(foundText)
            # if I did not want the [] at the beginning and end
            print(foundText[1:-1])
```

4. Try some other regular expressions

5. Write some code that will anonymise the sub domains of IP addresses by Xing out the last two triplets, the new lines should be stored in another file. We explored already (in 2 above) the appropriate regular expression, which is
   `"(\d{1,3}\.\d{1,3}\.)\d{1,3}\.\d{1,3}"`
   The brackets take the first two tripplets as a group that we include in the substitution with \1 (we need `\\1` because we want to tell python that we do want the \, so we escape that)

```
# This is code anonymise the sub domains of ip addresses
# Author: Andrew Beatty

import re

#regex = "\d{1,3}\.\d{1,3} " # this will find other numbers apart from ips
regex =  "(\d{1,3}\.\d{1,3}\.)\d{1,3}\.\d{1,3}"  # we make a group at the
beginning to keep
replacementText="\\1XXX.XXX " # note the space at the end to match above
filename = "./sample-files/smallerAccess.log"
outputFileName = "anonymisedIPs.txt"

with open(filename) as inputFile:
    with open(outputFileName, 'w') as outputFile:
        for line in inputFile:
            # for debugging
            #foundText = re.search(regex, line).group()
            #print(foundText)
            newLine = re.sub(regex, replacementText, line)
            outputFile.write(newLine)
```

Answers to quiz

a.	hello	Line 1
b.	Hello	Lines 2,3,5 (not 4 or 6)
c.	^Hello	Lines 2, 3 (not 5 because it does not start with Hello)
d.	^Hell*o	Lines 2,3,4,6 (* is 0 or more times)
e.	^Hell+o	Lines 2,3,6 (plus is 1 or more times)
f.	^Hell?o	Lines 2, 3,4 (? Is 1 or more times)
g.	^hello [A-Z]	No lines (there is not a hello with a capital after.)
h.	^Hello [A-Z]	Line 3
i.	=	Line 7
j.	#	Line 8
k.	[	Error [is special it should have been escaped \[
l.	^$	Line 10 it contains nothing