Lab - I

Regex

Re module

- The default offering for Python regular expressions is the ***re*** standard library module
- A regular expression (or RE) specifies a set of strings that matches it
- The functions in this module let you check if a particular string matches a given regular expression

re.search()

```
>>> sentence = 'This is a sample string'

# check if 'sentence' contains the given string argument
>>> 'is' in sentence
True
>>> 'xyz' in sentence
False

# need to load the re module before use
>>> import re

# check if 'sentence' contains the pattern described by RE argument
>>> bool(re.search(r'is', sentence))
True
>>> bool(re.search(r'xyz', sentence))
False
```

re.compile()

- function, which gives back a **re.Pattern** object.

```
>>> pet = re.compile(r'dog')
>>> type(pet)
<class 're.Pattern'>

>>> bool(pet.search('They bought a dog'))
True
>>> bool(pet.search('A cat crossed their path'))
False
```

```
#Q1.[3mins]
words = ['cat' , 'attempt' , 'tattle']
# Find all words that have two t's in them
# Without using the re module


#Q2.[2mins]
words = ['cat' , 'attempt' , 'tattle']
# Find all words that have two t's in them
# With re module



#Q3.[3mins]
sentence = "NLP is a very interesting subject"
# Find the word 'interesting' in the sentence
# without the use of re module



#Q4.[2mins]
sentence = "NLP is a very interesting subject"
word = re.compile(r'very')
# Find the word 'very' in the sentence


#Q5.[3mins]
# Use any() and all() built in function to check if
# A. "temp" is in any of the list of words"
# B. "at" is in all of the list of words"
```

bytes

- To work with bytes data type, the RE must be of bytes data as well.
- Similar to str RE, use raw format to construct a bytes RE.
- “Rb-raw byte”

```
>>> byte_data = b'This is a sample string'

# error message truncated for presentation purposes
>>> re.search(r'is', byte_data)
TypeError: cannot use a string pattern on a bytes-like object

>>> bool(re.search(rb'is', byte_data))
True
>>> bool(re.search(rb'xyz', byte_data))
False
```

Regex Module

- The third party regex module
- Standard re module.
- The regex module also offers advanced features

```
>>> import regex
>>> sentence = 'This is a sample string'

>>> bool(regex.search(r'is', sentence))
True
>>> bool(regex.search(r'xyz', sentence))
False
```

```
#Q6.[3mins]
# For the given input string, print all lines NOT containing the string 2
purchases = '''\
    apple 24
    mango 50
    guava 42
    onion 31
    water 10 '''
num = re.compile() # Add a solution here
# Use purchases.split('\n')
```