Quest2

August 27, 2018

1 Quest 2: Regex, Files, Urls

STUDENT_ID = "35460312"
QUEST_NAME = "QUEST2"
CODING_NAME = "MonsterPeach"

In [1]:
```
#Quest: Regex, Files, Urls

        import re, pytest, requests


        __STUDENT_ID__   = "35460312"        # replace with your 8 digit student id
        __QUEST_NAME__ = "QUEST2"            # QUEST NAME
        __CODING_NAME__ = "MonsterPeach"           # replace with your coding name - max 15 ch
```

In [2]:
```
def count_vowels(mystr):
    """return the number of vowels, upper and lowercase a,e,i,o,u in the string
    >>> count_vowels('aaacvemmikkOOzzuU')  -> 9
    """

    mystr = mystr.lower()
    vowel_list = re.findall(r'[aiueo]', mystr)
    return len(vowel_list)
```

In [3]:
```
count_vowels('aaacvemmikkOOzzuU')
```

Out[3]: 9

In [4]:
```
def is_valid_python_hex(mystr):
    """is string a valid hex: begins with 0x and contains only 0-9 and A-F (lower or u
     >>> is_valid_python_hex('0x1A2f') -> True
     >>> is_valid_python_hex('x1A2f')  -> False
     >>> is_valid_python_hex('0x1A2G') -> False
    """
    mystr = mystr.lower()
    if re.match(r'^[0x]{2}[a-f0-9]{4}$', mystr) is None:
        return False
    return True
```

```
In [5]: print(is_valid_python_hex('0x1A2f'))
        print(is_valid_python_hex('x1A2f'))
        print(is_valid_python_hex('0x1A2G'))

True
False
False
```

```
In [6]: def has_vowel(mystr):
            """   return True if a vowel upper or lowercase in string
            >>> has_vowel("zcxvsd")     -> False
            >>> has_vowel("vcbxvefjk")  -> True
            """
            mystr = mystr.lower()
            if re.search(r'[aiueo]', mystr) is None:
                return False
            return True
```

```
In [7]: print(has_vowel("zcxvsd"))
        print(has_vowel("vcbxvefjk"))

False
True
```

```
In [8]: def is_integer(mystr):
            """ returns True if integer with optional minus sign
            >>> is_integer("2345")     -> True
            >>> is_integer("-192345")  -> True
            >>> is_integer("234x5")    -> False
            """
            if re.search(r'^[\-]?[0-9]*$', mystr) is not None:
                return True
            return False
```

```
In [9]: print(is_integer("2345"))
        print(is_integer("-192345"))
        print(is_integer("234x5"))

True
True
False
```

```
In [10]: def get_extension(mystr):
             """ returns the extension for a filename or 'NONE' if no extension
             >>> get_extension('foo.zip')     -> 'zip'
             >>> get_extension('foo.doc.txt') -> 'txt'
```

```
            >>> get_extension('foozip')      -> 'NONE'
            """

            extension_type = re.search(r'[\.]{1}.\w*$', mystr)
            if extension_type is not None:
                return extension_type.group(0)[1:]
            return 'NONE'
```

```
In [11]: print(get_extension('foo.zip'))
         print(get_extension('foo.doc.txt'))
         print(get_extension('foozip'))
```

```
zip
txt
NONE
```

```
In [12]: def is_number(mystr):
             """ floating point number with optional - sign and optional decimal point
             >>> is_number('234')        -> True
             >>> is_number('-234')       -> True
             >>> is_number('234.')       -> True
             >>> is_number('234.999')    -> True
             >>> is_number('234.99.77')  -> False
             >>> is_number('234a.88')    -> False
             """

             if re.search(r'^[\-]?[0-9]*[\.]?[0-9]*[\.]?$', mystr) is not None:
                 return True
             return False
```

```
In [13]: print(is_number('234'))
         print(is_number('-234'))
         print(is_number('234.'))
         print(is_number('234.999'))
         print(is_number('234.99.77'))
         print(is_number('234a.88'))
```

```
True
True
True
True
False
False
```

```
In [14]: def convert_date_format(mystr):
             """ convert date format YYYY-MO-DAY  TO MO-DAY-YYYY. If not in date format
                 return "NONE" . Check only 4 digits for year and 2 digits for MO and DAY
```

```
            >>> convert_date_format('2018-03-04')  -> '03-04-2018'
            >>> convert_date_format('2018.03-04')  -> 'NONE'
            >>> convert_date_format('2018-03-054') -> 'NONE'
            """
            return_str = 'NONE'
            date_format = re.search(r'^\d{4}\-\d{2}\-\d{2}$',mystr)
            if date_format is not None:
                return_str = date_format.group(0)[5:7] + '-' + date_format.group(0)[8:10] + '-
            return return_str

In [15]: print(convert_date_format('2018-03-04'))
         print(convert_date_format('2018.03-04'))
         print(convert_date_format('2018-03-054'))

03-04-2018
NONE
NONE


In [16]: #File functions
         def readFileCountLines(filename):
             """use download file from Canvas: pytestFile1.txt - return number of lines
             >>> readFileCountLines('pytestFile1.txt')  -> 4
             """
             with open(filename, 'r') as f:
                 return_count = sum(1 for newline in f)
             f.close()
             return return_count

In [17]: readFileCountLines('pytestFile1.txt')

Out[17]: 4

In [18]: def readFileCountStringOccurrences(filename, stringval):
             """read file: pyTestFile1.txt  - return number of times  stringval appears
             >>> readFileCountStringOccurrences('pytestFile1.txt','rollo')  -> 3
             """
             word_count = 0
             with open(filename, 'r') as f:
                 for word in f.read().split():
                     if word.lower() == stringval:
                         word_count += 1
                 f.close()
                 return word_count

In [19]: readFileCountStringOccurrences('pytestFile1.txt','rollo')

Out[19]: 3
```

```
In [20]: def readFileSumDigitsGreaterThanNumber(filename, number):
             """e.g. file = "hello22world2100and18and 1000", number =999 -> 3100
              >>> readFileSumDigitsGreaterThanNumber('pytestFile1.txt',15)  -> 88
             """
             return_num = 0
             with open(filename, 'r') as f:
                 text = f.read().lower()
                 for num in text.split():
                         if num.isdigit() and int(num) > number:
                             return_num += int(num)
                 f.close()
                 return return_num

In [21]: readFileSumDigitsGreaterThanNumber('pytestFile1.txt',15)

Out[21]: 88

In [22]: def remove_all_but_alpha(mystr):
             """ remove all characters that are not alpha a-z A-Z
             >>> remove_all_but_alpha('hey-99-where8isthe**big_table**') -> 'heywhereisthebigt
             """
             mystr = re.sub(r'[^a-zA-Z]*','', mystr)
             return mystr

In [23]: remove_all_but_alpha('hey-99-where8isthe**big_table**')

Out[23]: 'heywhereisthebigtable'

In [24]: #URL functions
         def readurlCountStringOccurrences(urlname, stringval):
             """return number of times  stringval appears in text of url - ignore case
              >>> readurlCountStringOccurrences('http://s2.smu.edu/~coyle/testurls/foo.txt','r
             """
             word_count = 0
             uf = requests.get(urlname).text.split()

             for word in uf:
                 if word.lower() == stringval:
                     word_count += 1

             return word_count

In [25]: readurlCountStringOccurrences('http://s2.smu.edu/~coyle/testurls/foo.txt','rollo')

Out[25]: 3

In [26]: def readurlCountValidPhoneNumbers(urlname):
             """return count of valid phone number formats: no separator, dash separator, peri
             valid: 2126663333, 212-666-3333, 212.666.3333
```