**Class -** CSE 8392 Special Topics (Advanced Application Programming)  
**Quest** – Regix Files URLs Quest **Name** - Rishab Vaishya **SMU ID** – 47505527

**Code :**

*# Quest: Regex, Files, Urls***import** re, pytest  
**import** urllib.request  
  
\_\_STUDENT\_ID\_\_ = **"47505527"**\_\_QUEST\_NAME\_\_ = **"Quest2"  
  
  
def** count\_vowels(mystr, ):  
 **return** len(re.findall(**'[aeiouAEIOU]'**, mystr))  
  
  
**def** is\_valid\_python\_hex(mystr):  
 **return** re.fullmatch(**'0x[0-9a-fA-F]+'**, mystr) **is not None  
  
  
def** has\_vowel(mystr):  
 **return** len(re.findall(**'[aeiouAEIOU]'**, mystr))  
  
  
**def** is\_integer(mystr):  
 **return** re.fullmatch(**'-\*\d+'**, mystr) **is not None  
  
  
def** get\_extension(mystr):  
 extension = re.search(**'\.[\w]+$'**, mystr)  
 *# if extension found* **if** extension **is not None**:  
 *# return subString after the period (.)* **return** extension.group()[1:]  
 **return 'NONE'  
  
  
def** is\_number(mystr):  
 *# only 1 period is allowed with optional negative symbol at the front.* **return** re.fullmatch(**'-\*\d+\.?\d\*'**, mystr) **is not None  
  
  
def** convert\_date\_format(mystr):  
 date = re.fullmatch(**'\d{4}-\d{2}-\d{2}'**, mystr)  
 *# if date found in format yyyy-mm-dd* **if** date **is not None**:  
 *# return in format mm-dd-yyyy* **return** date.group()[5:7] + **'-'** + date.group()[8:10] + **'-'** + date.group()[:4]  
 **return 'NONE'***# File functions***def** readFileCountLines(filename):  
 file = open(filename, **'r'**)  
 *# counts next line charcters and adds one to it for the last line.* **return** (len(re.findall(**'[\n]'**, file.read())) + 1)  
  
  
**def** readFileCountStringOccurrences(filename, stringval):  
 file = open(filename, **'r'**)  
 **return** (len(re.findall(stringval, file.read())))  
  
  
**def** readFileSumDigitsGreaterThanNumber(filename, number):  
 sum = 0  
 file = open(filename, **'r'**)  
 *# finds all number in file* numbers = re.findall(**'\d+'**, file.read())  
 **for** currentNumber **in** numbers:  
 *# find all number greater than given number* **if** int(currentNumber) > number:  
 *# add to sum variable* sum += int(currentNumber)  
 **return** sum  
  
  
**def** remove\_all\_but\_alpha(mystr):  
 *# split by all non-alpha characters* alphas = re.split(**'[^a-zA-Z]'**, mystr)  
 *# join all alpha charcters to empty string* **return ''**.join(alphas)  
  
  
*# URL functions***def** readurlCountStringOccurrences(urlname, stringval):  
 response = urllib.request.urlopen(urlname)  
 *# add flags to ignore case* **return** len(re.findall(stringval, response.read().decode(**'utf-8'**), re.IGNORECASE))  
  
  
**def** readurlCountValidPhoneNumbers(urlname):  
  
 response = urllib.request.urlopen(urlname)  
 text = (response.read().decode(**'utf-8'**))  
  
 *# phone number in no seperated format* noSperator = re.findall(**'\d{10}'**, text)  
 *# phone number in dash seperated format* dashSperator = re.findall(**'\d{3}-\d{3}-\d{4}'**, text)  
 *# phone number in period seperated format* periodSperator = re.findall(**'\d{3}\.\d{3}\.\d{4}'**, text)  
  
 *# adds the length of all the 3 formats* **return** (len(noSperator) + len(dashSperator) + len(periodSperator))  
  
  
**if** \_\_name\_\_ == **'\_\_main\_\_'**:  
 print(**"Assertion ran without error!! "**)

**Output :**

