CIND830F20 Assignment 2

November 14, 2020

0.1 CIND830 - Python Programming for Data Science

0.1.1 Assignment 2 (10% of the final grade)

0.1.2 Due on November 09, 2020 11:59 PM

This is a Jupyter Notebook document that extends a simple formatting syntax for authoring HTML and PDF. Review this website for more details on using Juputer Notebook.

Use the JupyterHub server on the Google Cloud Platform, provided by your designated instructor, for this assignment. Complete the assignment by inserting your Python code wherever you see the string "#INSERT YOUR ANSWER HERE."

When you click the `File` button, from the top navigation bar, then select `Export Notebook As ...`, a document (PDF or HTML format) will be generated that includes both the assignment content and the output of any embedded Python code chunks.

Using these guidelines, submit **both** the IPYNB and the exported file (PDF or HTML). Failing to submit both files will be subject to mark deduction.

0.1.3 Question 1:

a) Define a function called `getEven` that takes a list of numbers as a parameter and returns a new list that contains only the even numbers. For example given [1, 2, 3, 4, 5, 7] the getEven function will return [2, 4]

[9]:
```
def getEven(n):
    lst = []
    for i in n:
        if i % 2 == 0:
            lst.append(i)
    return lst

getEven([1,2,3,4,5,7])
```

[9]: [2, 4]

b) Modify the `getEven` function you created in Q1.a to return a dictionary of the counts of the even and the odd numbers. For example given [1, 2, 3, 4, 5, 7] the function returns {'even': 2, 'odd': 4}.

[1]:
```
def new_getEven(n):
    oddCount = 0
    evenCount = 0
    for i in n:
        if i % 2 == 0:
            evenCount +=1
        else:
            oddCount+=1
    return {"even": evenCount, "odd":oddCount}
new_getEven([1,2,3,4,5,7])
```

[1]: {'even': 2, 'odd': 4}

c) Modify the function you created in Q1.b and add two more more entries to the returned dictionary: one to represent the sum of all the numbers and the other one to represent the ratio of the even to the odd numbers. For example, given [1, 2, 3, 4, 5, 7] the function returns {'even': 2, 'odd': 4, 'sum': 22, 'ratio': 0.5}.

[2]:
```
def new_getEven2(n):
    oddCount = 0
    evenCount = 0
    mySum =0
    for i in n:
        mySum += i
        if i % 2 == 0:
            evenCount +=1
        else:
            oddCount+=1
    return {"even": evenCount, "odd":oddCount,"sum":(mySum),"ratio": (evenCount/
 ↪oddCount)}
new_getEven2 ([1,2,3,4,5,7])
```

[2]: {'even': 2, 'odd': 4, 'sum': 22, 'ratio': 0.5}

d) Modify the function you created in Q1.c to traverse the values of the returned dictionary and return the dictionary keys as a tuple. For example, given [1, 2, 3, 4, 5, 7] the function returns ('even', 'odd', 'sum', 'ratio').

[3]:
```
def new_getEven3(n):
    oddCount = 0
    evenCount = 0
    for i in n:
```

```
        if i % 2 == 0:
            evenCount +=1
        else:
            oddCount+=1
    myDict= {"even": evenCount, "odd":oddCount,"sum":
 ↪(evenCount+oddCount),"ratio": (evenCount/oddCount)}
    return tuple((myDict.keys()))
new_getEven3 ([1,2,3,4,5,7])
```

[3]: ('even', 'odd', 'sum', 'ratio')

0.1.4 Question 2:

Let's define a base class called `Shape` with methods to calculate the area and perimeter of different shapes, as follows:

[24]:
```
class Shape:
    '''A class defining shapes'''
    def __repr__(self):
        return f'I am a {self.__class__.__name__}' #print the shape's name

    def area(self):
        pass

    def perimeter(self):
        pass
```

a) Define a subclass called `Rectangle` that extends the `Shape` class. The `Rectangle` class has two parameters: width and height. The area and perimeter of can be computed according to the following formulae:

$ area = width * height $

$ perimeter = 2 * width * height $

Hint: A subclass can implement a base class by overriding methods. The `Rectangle` subclass can be defined with the following header: `class Rectangle(Shape):`

For example, if the subclass `Rectangle` is instantiated with `Rectangle(3,4)`, the following will be returned:

```
I am a Rectangle

area = 12

perimeter = 14
```

[25]:
```
class Rectangle(Shape):
    ''' Defining a subclass of Shape, Rectangle '''
    def __init__(self, width, height):
        self.width =width
        self.height =height


    def area(self):
        return (print("area = ", (self.width * self.height)))
    def perimeter(self):
        return print("perimeter = ", 2*(self.width+self.height))
print(Rectangle(3,4))
(Rectangle(3,4).area())
(Rectangle(3,4).perimeter())
```

```
I am a Rectangle
area =  12
perimeter =  14
```

b) Define a subclass called `Square` that extends the `Rectangle` class.

Hint: The `Square` class should inherit all the attributes and behaviours of the `Rectangle` class. However, for any square, the width and height are the same.

[11]:
```
class Square(Rectangle):
    '''Defining a subclass of Rectangle, Square'''
    def __init__(self, side):
        Rectangle.__init__(self,side,side)
```

c) Define a subclass called `Circle` that extends the `Shape` class. The `Circle` class has only one parameter which is the `radius`, and the area and perimeter can be computed according to the following formulae:

$AreaOfCircle = R^2$

$PerimeterOfCircle = 2R$

Hint: You might consider using the `math.pi` constant that returns the value of pi: 3.141592653589793.

For example, if the subclass `Circle` is instantiated with Circle(radius = 5), the following will be returned:

```
I am a Circle

area = 78.54

perimeter = 31.42
```

[26]:
```
import math
class Circle(Shape):
    '''Defining a subclass of Shape, Circle'''
```

```
    def __init__(self, radius):
        self.radius = radius
    def area(self):
        a=math.pi*(self.radius**2)
        return print("area = ", round(a,2))

    def perimeter(self):
        p=math.pi*2*(self.radius)
        return print("perimeter =", round(p,2))
print(Circle(radius = 5))
Circle(radius = 5).area()
Circle(radius=5).perimeter()
```

```
I am a Circle
area =  78.54
perimeter = 31.42
```

0.1.5 Question 3:

Email spam, also called junk email, is undesirable message sent in bulk by email (spamming). Spam filtering is an example of a document classification task deciding whether an email is a spam or non-spam. Assume that `txtFile` variable includes the content of a recieved email that has been stored as `email.txt`, as follows:

[19]:
```
txtFile = open("email.txt", "w") #Opening a file for writing
txtFile.write("Subject: SAVE YOUR MUSIC AND REDEEM A SPOTIFY GIFT CARD\n\
We noticed you haven\'t used your Spotify account for accessing Spotify␣
 ↪services in quite a while. To protect your privacy, this account will be␣
 ↪deleted in 14 days, unless you sign in now!\n\
If you haven\'t experienced Spotify services recently, they\'re worth another␣
 ↪look.\n\
It just takes a few seconds to sign in to a spotify account.\n\
By clicking on any of the following links, you can get a gift card from Spotify␣
 ↪worth $10, $20 or $50!\n\
https://www.spotifytwo.ca\n\
https://www.spotifynow.com\n\
We hope to see you soon,\n\
Sincerely, \n\
Spotify account team")
txtFile.close()
```

a) Display the total number of lines and words in the email.txt file.

[19]:
```
opn = open("email.txt","r")
lcount=0
```

```
read = opn.read()
rsplit= read.split("\n")
for i in rsplit:
    if i:
        lcount+=1
print ("The total number of lines is " + str (lcount))
nwords= 0
with open("email.txt","r") as f:
    for line in f:
        words = line.split()
        nwords += len(words)
print("The total number of words is " + str(nwords))
opn.close()
```

```
The total number of lines is 10
The total number of words is 100
```

b) Display the subject of the received email existing in the `email.txt` file, then display the number of the upper and lower cases in the subject field.

[20]:
```
with open("email.txt","r") as f:
    subject = f.readline().replace("Subject: ","")
    print(subject)
upper=0
lower=0
for character in subject.split():
    if character.isupper():
        upper+=1
    elif character.islower():
        lower+=1
print ("The number of the upper cases is", upper)
print ("The number of the lower cases is", lower)
```

```
SAVE YOUR MUSIC AND REDEEM A SPOTIFY GIFT CARD

The number of the upper cases is 9
The number of the lower cases is 0
```

c) Does the email contain spam words such as 'gift', 'sign' or 'Buy'?

[21]:
```
word = ["gift","sign","Buy"]
with open("email.txt","r") as f:
    content = f.read()
    for i in word:
        if i in content:
            print (i, 'is found in the email.')
        else:
            print (i, 'is not found in the email.')
```

```
gift is found in the email.
sign is found in the email.
Buy is not found in the email.
```

d) Define a list that consists of 10 different email spam trigger words, then check whether any of those words exist in the body of the email or not. If the email contains a spam word, print it; otherwise, return a message that the email does not include any predefined spam words.

[22]:
```
spam = ["win","prize","money","lotto","millionaire","chance",␣
 ↪"https","brandnew", "now","final"]
with open("email.txt","r") as f:
    content = f.read()
    for i in spam:
        if i in content:
            print (i, 'is found in the email.')
        else:
            print (i, 'is not found in the email.')
```

```
win is found in the email.
prize is not found in the email.
money is not found in the email.
lotto is not found in the email.
millionaire is not found in the email.
chance is not found in the email.
https is found in the email.
brandnew is not found in the email.
now is found in the email.
final is not found in the email.
```

This is the end of assignment 2