2 F21_Assignment_2

November 18, 2021

0.1 CIND830 - Python Programming for Data Science

0.1.1 Assignment 2 (10% of the final grade)

0.1.2 Due on November 15, 2021 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 [30 pts]:

a) [**5 pts**] Define a function called `is_symmetrical` that takes a string value as a parameter and returns `True` if the given string is Symmetrical. A string is said to be symmetrical if the reverse of the string is the same as string. For example, "madam" is symmetrical, but "water" is not symmetrical. Note that strings are case sensitive.

```
[1]: def is_symmetrical(s):
         """
         is_symmetrical(str) -> boolean
         """
         if len(s)%2 == 0:
             a = int(len(s)/2)
             for j in range(a):
                 if s[j] != s[-(j+1)]:
                     return False
```

```
        return True
    else:
        b = int((len(s)-1)/2)
        for k in range(b):
            if s[k] != s[-(k+1)]:
                return False
        return True
```

b) [**5 pts**] Modify the `is_symmetrical` function you created in Q1.a to take any data type (int,str,etc.) as an input and return `True` if the input is symmetrical. For example, “12321” is symmetrical, but “12345” is not symmetrical.

[2]:
```
#already works for strings, lists, tuples etc but DOES NOT work for integers
#all we have to do is convert the data types of type int into strings
#then we can iterate over the individual digits

def is_symmetrical_1(s):
    """
    is_symmetrical_1(any data type) -> boolean
    """
    if type(s)== int:
        s = str(s)
    if len(s)%2 == 0:
        a = int(len(s)/2)
        for j in range(a):
            if s[j] != s[-(j+1)]:
                return False
        return True
    else:
        b = int((len(s)-1)/2)
        for k in range(b):
            if s[k] != s[-(k+1)]:
                return False
        return True
```

c) [**5 pts**] Define a function called `is_symmetrical_vec` that calls `is_symmetrical` function you updated in Q1.b to take a list of elements and return their results in a list. For example, given [“1441”, “Apple”, “radar”, “232”, “plane”] the function returns [`True`, `False`, `True`, `True`, `False`]

[3]:
```
def is_symmetrical_vec (l):
    a = []
    for i in range(len(l)):
        val = is_symmetrical_1(l[i])
        a.append(val)
    return a
```

d) [**7 pts**] Define a function called `is_symmetrical_dict` by modifying the function you created in Q1.c to return the results in a dictionary with their corresponding input elements.

For example, given [“1441”, “Apple”, “radar”, “232”, “plane”] the function returns {“1441”: `True`, “Apple”: `False`, “radar”: `True`, “232”: `True`, “plane”: `False`}

[4]:
```
def is_symmetrical_dict(l):
    d={}
    for i in range(len(l)):
        val = is_symmetrical_1(l[i])
        d[l[i]]=val
    return d
```

e) [8 pts] Define a function called `is_symmetrical_tuple` by modifying the function you created in Q1.d to traverse the values of the returned dictionary **in reverse order** and return the dictionary keys that has a `True` value as a tuple. For example, given [“1441”, “Apple”, “radar”, “232”, “plane”] the function returns (“232”, “radar”,“1441”).

[5]:
```
def is_symmetrical_tuple(l):
    t=[]
    for i in range(len(l)):
        val = is_symmetrical_1(l[-(i+1)])
        if val == True:
            t.append(l[-(i+1)])
    return tuple(t)
```

0.1.4 Question 2 [40 pts] :

The data file used in this question is downloaded from https://www.kaggle.com/arjunprasadsarkhel/2021-olympics-in-tokyo and converted to a csv file.

a) [15 pts] The Medals.csv is a delimited text file containing the number of medals of teams in the 2021 Olympics in Tokyo. The file contains the following columns separated by the comma character **,** * Rank * Team/NOC * Gold * Silver * Bronze * Total * Rank by Total

Write a code that asks the user to enter the file’s name: `Medals.csv`. If the program finds the file in the folder in which you stored it, it will read the file’s contents into a list of lists called `medalsList`. Each list inside the `medalsList` represents one row from the `Medals.csv` file. Ensure that your code reads and stores each row as a list of strings inside the `medalsList` data structure. For example the first row would be `['1', 'United States of America', '39', '41', '33', '113', '1']`

[6]:
```
from csv import reader
import os

user_input = input('Type in the file name and extension here:')
if os.path.exists(str(user_input)):
    with open(user_input, 'r') as a:
        b = reader(a)
        medalsList = list(b)
else:
    print('File not found')
```

```
Type in the file name and extension here: Medals.csv
```

b) [**15 pts**] Write a code that asks the user to enter a team name and medal type; then, it will display the total number of medals that the team has won. For example, if the user enters `Canada` as the team's name and `Gold` as the medal's type, the code should display `7`.

Make sure your code handles the following four issues: 1. The user is allowed to enter lowercase and uppercase letters; your code must be case insensitive (e.g. if the user enters `canADa`, it still needs to locate `Canada` correctly) 2. Your code should use the list `medalList` that you created in question Q2.a, rather than repeatedly reads the data from the `Medals.csv` file. 3. The user can enter either `Gold`, `Silver`, `Bronze` or `All` for the medal's type. For example, if the user enters `Canada` and `All`, the code should display `Gold: 7, Silver: 6, Bronze: 11, Total: 24`. 4. If the user enters a team's name that is not in the list, the code should notify the user. For example, if the user enters `Mozambique` as the team's name, the code would display: `Unfortunately, Mozambique did not participate in the 2021 Olympics!`

[7]:
```
def grammar_fix(s):
    for i in range(len(s)):
        if i == 0:
            new_s = s.upper()[0]
        else:
            new_s += s.lower()[i]
    return new_s
country = grammar_fix(input('enter country name:'))
Medal = grammar_fix(input('enter medal type:'))

def medal_dict(l):
    meds = ['Gold','Silver','Bronze','Total']
    md = {}
    for i in range(len(l)):
        md[meds[i]]=l[i]
    return md


for i in range(len(medalsList)):
    count = 0
    for j in range(len(medalsList[i])):
        if country in medalsList[i][j]:
            count += 1
            if Medal in ['Gold','Silver','Bronze']:
                a = medalsList[0].index(Medal)
                print(medalsList[i][a])
                break
            if Medal == 'All':
                b = medal_dict(medalsList[i][2:6])
                print(b)
                break
            else:
                print('invalid medal type.')
                break
    if count == 1:
        break
    if i == len(medalsList)-1:
        if count == 0:
            print('Unfortunately, ',country,' did not participate in the 2021␣
↪Olympics!')
```

```
enter country name: Namibia
enter medal type: alL
```

```
{'Gold': '0', 'Silver': '1', 'Bronze': '0', 'Total': '1'}
```

c) [**10 pts**] Write a code that will ask the first letter of the team name and display the total of the medals (grouped by the medal type) taken by the teams starting with the first letter as the selected one. E.g. if the user enters `f` (case insensitive) as the first letter of the team, then your code will scan the list, find the teams `France`, `Fiji`, and `Finland` then it will display the total medals `Gold: 11, Silver: 12, Bronze: 14` (the total Gold medals of the teams starting with `f`, the total Silver of the teams starting with `f`, etc.
Make sure your code handle the cases listed below:

- The user is allowed to enter lowercase and uppercase letter, your code must be case insensitive.
- Your code must **use** (do not read the file again) the list created in question **Q2.a** containing all the information read from the file
- If there is no team name starting with the selected letter then your code needs to display an error message like `No team name starts with your selected letter in the 2021 Olympics`

[8]:
```
first_letter = grammar_fix(input('enter the first letter of the countries:'))
index_list = []
medal_sums = [0,0,0]
count=0
for k in range(len(medalsList)):
    if first_letter == medalsList[k][1][0]:
        count +=1
        index_list.append(k)

for l in range(len(index_list)):
    for m in range(len(medal_sums)):
        a = index_list[l]
        medal_sums[m] += int(medalsList[a][m+2])

if count == 0:
    print('No team name starts with your selected letter in the 2021 Olympics')
else:
    print(medal_sums)
    total_medals = medal_dict(medal_sums)
    print(total_medals)
```

```
enter the first letter of the countries: f
```

```
[11, 12, 14]
{'Gold': 11, 'Silver': 12, 'Bronze': 14}
```

0.1.5 Question 3 [30 pts] :

a) Define three classes: **Vehicle**, **Car** and **Truck**. Vehicle will be the parent class of Car and Truck. - Randomly create five objects of Car and Truck using the for loop given below - The program should output the correct number of vehicles: Car and Truck objects. - **The count of objects should be calculated within the class definitions**. - Example output \ There are 5 Vehicle objects ['Car', 'Car', 'Truck', 'Truck', 'Car'] \ 3 of them are Cars \ 2 of them are Trucks \

[9]:
```python
# Define Vehicle class
class Vehicle():
    count = 0
    def __init__ (self,name):
        pass

# Define Car class
class Car(Vehicle):
    def __init__(self, name):
        Vehicle.__init__(Car,self)
        Car.count +=1

# Define Truck class
class Truck(Vehicle):
    def __init__(self, name):
        Vehicle.__init__(Truck,self)
        Truck.count+=1
# Run the following lines of code

from random import choice
list_of_vehicles = [choice(['Car', 'Truck']) for x in range(5)]
for vehicle in list_of_vehicles:
  if vehicle == 'Car':
    # create an object of Car class
```

```
        o = Car('Pick Car')
    else:
        # create an object of Truck class
        c = Truck('Pick Truck')

# print the list_of_vehicles
print(list_of_vehicles)
# print number of Vehicle objects
print(len(list_of_vehicles))
# print number of Car objects
print(o.count)
# print number of Truck objects
print(c.count)
```

```
['Truck', 'Car', 'Truck', 'Truck', 'Truck']
5
1
4
```

This is the end of assignment 2