Dev Nation

Data Science Lab-08

Grand Quiz

Instructions

1. **Should Attempt all questions**
2. **Google Search is allowed**
3. **Write comments to explain your code**
4. **In the case of Plagiarism. Zero would be marked and would be kicked out from the course**
5. **Discussion with other classmates is not allowed**
6. **Passing Marks 70**

Q1. Create a function **add_numbers** which takes 4 parameters

1. x (integer)
2. y (integer)
3. z (integer)
4. flag (Boolean)

z and flag are optional parameters
if function receives **z** then function should return the sum of **x y** and **z** else
return the sum of **x** and **y**

if flag is True also print the statement
"Flag is True"

Q2. Write a function **get_datetime**(int) that takes a param number_of_days and
if the number is **divisible by 2** returns (current_datetime + number_of_days) and
if number is **divisible by 3** returns (current_datetime - number_of_days)
else return current_datetime

Q3. Write a function **concat_lst** that takes two list as a parameter and combine two list together into single list and should not contains duplicate values.
You must use loop to solve this question

Q4. Write a function **string_case** using map that takes list of characters and return
List of tuples with containing lower and upper case of that character's list
Final Output must contain unique tuple element

Q5. Write a function **get_city_mpg_avg**, that groups the cars by number of cylinders, and finding the average cty mpg for each group.

Output should be in the list of tuples
Output = [(5,20),(4,18),(3,16)]

Note:

1. Use MPG.csv file
2. First index of tuple shows the number of cylinders and second index shows the average
3. Result should be sorted by first tuple index by using lambda functions

Q6. Write a function **get_hwy_mpg_avg** find the average hwy mpg for each class of vehicle
Hwy mpg means highway mileage

1. Use MPG.csv file
2. Hwy mpg is found as **hwy** column in a file
3. Vehicle class is found as **class** column in a file
4. Sort Hwy Mpg in ascending order

Output sample: [('pickup', 16.88),('suv', 18.13),('minivan', 22.36),('2seater', 24.80),('midsize', 27.29), ('subcompact', 28.14), ('compact', 28.30)]

Q7. Create a class Student with the following getters and setters

1. Constructor
    a. Create an Object with following parameters
        i. Name
        ii. Batch No
        iii. Address
        iv. Age
        v. Father Name
        vi. City
2. New
    a. Students with age more than 18 are not allowed in the school
    b. Students of Lahore Islamabad Faisalabad Karachi are allowed
3. Setters
    a. Name (str)
    b. Batch no (int)
    c. Address (str)
    d. Age (str)
    e. Father name (str)
    f. City (str)
4. Getters
    a. Name
    b. Batch no
    c. Address
    d. Age
    e. Father name
    f. City
5. Create a new class called DevnationStudents which is a child class of Student
    a. Write a constructor that
        i. That calls the Parent Class Constructor
        ii. Also save courses and marks as empty list in the child constructor
    b. Write setter functions
        i. Add courses
            1. Takes a list of courses
            2. Add only those courses in the list which are not already in the
        ii. Add courses_marks
            1. This takes the list of dictionaries
            2. If the particular exists in the courses lists then the particular dictionary should be added in the marks list
            3. Marks list should not contains duplicate entries of same courses
    c. Write getter functions
        i. Get courses
        ii. Get Marks

Q8. Write a function get_indices that receives the list of integers `nums` and an integer `target`, return *indices of the two numbers such that they add up to* `target`. You may assume that each input would have ***exactly* one solution**, and you may not use the *same* element twice.

You can return the answer in any order.

Example 1:

> **Input:** nums = [2,7,11,15], target = 9
>
> **Output:** [0,1]
>
> **Output:** Because nums[0] + nums[1] == 9, we return [0, 1].

Example 2:

> **Input:** nums = [3,2,4], target = 6
>
> **Output:** [1,2]

Q9. Write a function **backward_number**. That a receives a number and return the number in backwards
Input = 1234
Output = 4321

Q10. Create a function **remove_duplication** that takes a list and remove duplication from list.
You cannot use set function