Lecture 10 Coding Practices

Please open your PyCharm and proceed with the following tasks.

You may need to use online resources to complete this task. As a programmer, always know that google is your friend and stack overflow (https://stackoverflow.com/) is your best friend 😊

Classes and Objects

TRY IT YOURSELF

9-1. Restaurant: Make a class called `Restaurant`. The `__init__()` method for `Restaurant` should store two attributes: a `restaurant_name` and a `cuisine_type`. Make a method called `describe_restaurant()` that prints these two pieces of information, and a method called `open_restaurant()` that prints a message indicating that the restaurant is open.

Make an instance called `restaurant` from your class. Print the two attributes individually, and then call both methods.

9-2. Three Restaurants: Start with your class from Exercise 9-1. Create three different instances from the class, and call `describe_restaurant()` for each instance.

9-3. Users: Make a class called `User`. Create two attributes called `first_name` and `last_name`, and then create several other attributes that are typically stored in a user profile. Make a method called `describe_user()` that prints a summary of the user's information. Make another method called `greet_user()` that prints a personalized greeting to the user.

Create several instances representing different users, and call both methods for each user.

Write a method that updates the odometer in the following code and then instantiate an object and test your newly added method.

```
class Car:
    def __init__(self, make, model, year):
        """Initialize attributes to describe a car."""
        self.make = make
        self.model = model
        self.year = year
        self.odometer_reading = 0

    def get_descriptive_name(self):
        """Return a neatly formatted descriptive name."""
        long_name = f"{self.year} {self.make} {self.model}"
        return long_name.title()  # titleCased value

    def read_odometer(self):
        """Print a statement showing the car's mileage."""
        print(f"This car has {self.odometer_reading} miles on
it.")

my_new_car = Car('audi', 'a4', 2019)
print(my_new_car.get_descriptive_name())
my_new_car.read_odometer()
## Now let's assume that care has been used for 50 miles
my_new_car.odometer_reading = 50
my_new_car.read_odometer()
```

Remember that your method should be incrementing the odometer number , here is an example

```
def increment_odometer(self, miles):
    """Add the given amount to the odometer reading."""
    self.odometer_reading += miles
```


Source : Matthes, Eric. Python crash course: a hands-on, project-based introduction to programming. No Starch Press, 2019.

TRY IT YOURSELF

9-4. Number Served: Start with your program from Exercise 9-1. Add an attribute called `number_served` with a default value of 0. Create an instance called `restaurant` from this class. Print the number of customers the restaurant has served, and then change this value and print it again.

Add a method called `set_number_served()` that lets you set the number of customers that have been served. Call this method with a new number and print the value again.

Add a method called `increment_number_served()` that lets you increment the number of customers who've been served. Call this method with any number you like that could represent how many customers were served in, say, a day of business.

9-5. Login Attempts: Add an attribute called `login_attempts` to your `User` class from Exercise 9-3. Write a method called `increment_login_attempts()` that increments the value of `login_attempts` by 1. Write another method called `reset_login_attempts()` that resets the value of `login_attempts` to 0.

Make an instance of the `User` class and call `increment_login_attempts()` several times. Print the value of `login_attempts` to make sure it was incremented properly, and then call `reset_login_attempts()`. Print `login_attempts` again to make sure it was reset to 0.

Source : Matthes, Eric. Python crash course: a hands-on, project-based introduction to programming. No Starch Press, 2019.