Classes HW

1) **Restaurant:**
   1. Make a class called Restaurant.
   2. The __init__() method for Restaurant should store two attributes: restaurant_name and a cuisine_type.
   3. Make a method called describe_restaurant() that prints these two information.
   4. Make a method called open_restaurant() that prints a message indicating that the restaurant is open.
   5. Make an instance called restaurant from your class.
   6. Print the two attributes individually, and then call both methods.
2) **Three Restaurants:**
   1. Start with your class from Exercise (1).
   2. Create three different instances from the class, and call describe_restaurant() for each instance.
3) **Users:**
   1. Make a class called User.
   2. Create two attributes called first_name and last_name.
   3. create several other attributes that are typically stored in a user profile.
   4. Make a method called describe_user() that prints a summary of the user's information.
   5. Make another method called greet_user() that prints a personalized greeting to the user.
   6. Create several instances representing different users, and call both methods for each user.
4) **Number Served:**
   1. Start with your program from Exercise (1).
   2. Add an attribute called number_served with a default value of 0.
   3. Create an instance called restaurant from this class.
   4. Print the number of customers the restaurant has served, and then change this value and print it again.
   5. Add a method called set_number_served() that lets you set the number of customers that have been served.
   6. Call this method with a new number and print the value again.
   7. Add a method called increment_number_served() that lets you increment the number of customers who've been served.
   8. Call this method with any number you like that could represent how many customers were served in, say, a day of business.
5) **Login Attempts:**
   1. Add an attribute called login_attempts to your User class from Exercise (3).
   2. Write a method called increment_login_attempts() that increments the value of login_attempts by 1.
   3. Write another method called reset_login_attempts() that resets the value of login_attempts to 0.
   4. Make an instance of the User class and call increment_login_attempts() several times.
   5. Print the value of login_attempts to make sure it was incremented properly.
   6. Call reset_login_attempts().
   7. Print login_attempts again to make sure it was reset to 0.

6) **Ice Cream Stand:**
   1. An ice cream stand is a specific kind of restaurant.
   2. Write a class called IceCreamStand that inherits from the Restaurant class you wrote in Exercise (4).
   3. Add an attribute called flavors that stores a list of ice cream flavors.
   4. Write a method that displays these flavors.
   5. Create an instance of IceCreamStand, and call this method.
7) **Admin:**
   1. An administrator is a special kind of user.
   2. Write a class called Admin that inherits from the User class you wrote in (5).
   3. Add an attribute, privileges, that stores a list of strings:
      like "can add post", "can delete post", "can ban user", and so on.
   4. Write a method called show_privileges() that lists the administrator's set of privileges.
   5. Create an instance of Admin, and call your method.
8) **Privileges:**
   1. Write a separate Privileges class.
   2. The class should have one attribute, privileges, that stores a list of strings as described in (7).
   3. Move the show_privileges() method to this class.
   4. Make a Privileges instance as an attribute in the Admin class.
   5. Create a new instance of Admin and use your method to show its privileges.
9) **Imported Restaurant:**
   1. Using your latest Restaurant class, store it in a module.
   2. Make a separate file that imports Restaurant.
   3. Make a Restaurant instance, and call one of Restaurant's methods to show that the import statement is working properly.
10) **Imported Admin:**
   1. Start with your work from Exercise (8).
   2. Store the classes User, Privileges, and Admin in one module.
   3. Create a separate file, make an Admin instance, and call show_privileges() to show that everything is working correctly.
11) **Multiple Modules:**
   1. Store the User class in one module, and store the Privileges and Admin classes in a separate module.
   2. In a separate file, create an Admin instance and call show_privileges() to show that everything is still working correctly.