Wiki - Practice exercises for classes

Solve each of the practice exercises below. Each problem includes two CodeSkulptor links; one for a template that you should use as a starting point for your solution and our solution to the exercise.

1. Implement a Person class which has the fields first_name, last_name and birth_year. This class should include the methods:

    - __init__ which takes strings for the two name fields and an integer for the year of birth.
    - full_name returns the full name for a person as a string, which is the first name followed by a space, followed by the last name.
    - age which takes the current year as input and returns the age in years of the person. (Don't worry about days and months here, just return the difference of the two years.)
    - __str__ returns a string that includes the first name and last name of the person as well as their year of birth.

    Definition of Person class template
    Definition of Person class solution

2. Write a function average_age that takes a list of Person objects along with the current year and returns the average age of the people in the list. Remember that average_age should only use the methods defined in the Person class. (The body of average_age should not access the fields in a Person object directly.)

    Application of Person class template
    Application of Person class solution

3. Implement a Student class which has the fields full_name (string), password (string), and projects (list of strings). Note that we won't bother to separate the full name into parts in this example. This class should include the following methods:

    - __init__ which takes the student's full name and a password, both specified as strings, and creates a Student object. The list of projects should be empty to start.
    - get_name which returns the student's full name.
    - check_password which take a supplied password and returns a Boolean indicating whether the supplied password matches the student's created password.
    - get_projects which returns the list of the student's projects.
    - add_project(project_name) which adds the specified project to the student's list of projects. (Note that this method does not check whether the project already exists in the list.)

    Definition of Student class template
    Definition of Student class solution

4. Write a function assign that takes a list of Student objects, a student name, a password, and a project as parameters. This function should search the list of students for students whose name and password match the supplied information. When a match is found, the function checks the student's current list of projects for the supplied project. If the project does not already exist in the list, the function adds the project to the list. Remember to use only methods for the Student class to manipulate Student objects.

Application of Student class template
Application of Student class solution

5. Implement a Subimage class with fields image (SimpleGUI image), center (pair of numbers), size (pair of numbers), caption (string) and modified (Boolean). This class should include the following methods:

    - __init__ which takes an image, the center and size of a sub-image in that image, and a caption.
    - draw which takes the canvas as well as a target center and draws the sub-image (at its normal size) with the specified caption on the canvas. (Don't worry too much about positioning the caption robustly.)
    - get_caption which returns the caption for the image.
    - set_caption which changes the caption to the provided string. Note that each object's caption should only be allowed to be updated once. Use the the Boolean modified field to prevent subsequent updates to the caption.

    Definition of Subimage class template
    Definition of Subimage class solution

6. **Challenge:** Using the supplied image, create four Subimage objects corresponding to Joe, Scott, John and Stephen with their names as the initial captions. Add code that creates four buttons labelled "Joe", "Scott", "John", and "Stephen" which draw their sub-images on the canvas when clicked. Add a fifth button label "Awesome!" which adds the string " is awesome!" to the caption of the currently selected sub-image. Make sure that multiple clicks of this button avoid adding " is awesome!" repeatedly to the caption.

    Application of Subimage class template
    Application of Subimage class solution