CSC 4992
Python Programming
Winter Term 2018
Project
(75 points)
Due 04/23/2018 (11:55 P.M.)

The goal of this project is to give you more experience on the use of:

- classes
- inheritance
- turtle graphics
- files
- lists of lists

Project Problem 01: Employee (40 points)

Write an **Employee class** that keeps data attributes for the following pieces of information:

- Employee name
- Employee number

Next, write two classes named ProductionWorker and ShiftSupervisor that are subclasses of the Employee class.

ProductionWorker Class
The ProductionWorkerclass should keep data attributes for the following information:

- Shift number (an integer, such as 1, 2, or 3)
- Hourly pay rate

Shift attribute will hold an integer value representing the shift that the employee works. The day shift is shift 1 and the night shift is shift 2.

ShiftSupervisor Class
In a particular factory, a shift supervisor is a salaried employee who supervises a shift. In addition to a salary, the shift supervisor earns a yearly bonus when his or her shift meets production goals.
The ShiftSupervisorclass should keep a data attribute for the:

- Annual salary
- Annual production bonus that a shift supervisor has earned.

Write the appropriate accessor and mutator methods for each class.

Once you have written the classes, write a program that creates two objects, object of the ***ProductionWorker*** class and an object of ***ShiftSupervisor***r class, and prompts the user to enter data for each of the object's data attributes. Store the data in the object and then use the object's accessor methods to retrieve it and display it on the screen.

Deliverables
Turn in `Project1.py` containing all of your class and function definitions.

Project Problem 02: Draw House (35 points)

The Problem

This part of project will give you more experience with defining classes and using inheritance. In this part, you are going to use turtle graphics to define some elementary shapes that know how to draw themselves. You are going to **create at least 3 classes**. Using instances of these classes, you are going to draw a simple picture.

Your Task

1. You will define at least three classes: A Triangle class, a Rectangle class, and a Circle class.
2. Instances of each class respond to at least the following methods:
    a. __init : create an instance of the class from the following:
        i. For a Triangle: three pairs of coordinates, which indicate the three vertices (not a list, but three arguments, which you may give some default values for easy testing)
        ii. For a Rectangle: two pairs of coordinates—one giving the coordinates of the lower left corner and the other giving the coordinates of the top right corner (not a list, but two arguments).
        iii. For a Circle: a coordinate pair giving the center of the circle and a float giving the radius
        iv. For all, a string indicating a fill color (default value "", the empty string)
        v. For all, a string indicating the line color (default value "black")
    b. __str : conversion to a string; returns the string to be used for printing in the Python shell.
    c. draw: will take a turtle.Turtle object and will use it for drawing the shape;
3. All classes, methods and functions **require** a docstring for a general description of the object/method/function. (see docstring note below)
4. Triangle and Rectangle should both inherit from Polygon and reuse what they can from Polygon.
5. Once you have tested your class definitions, draw a simple picture using them. For instance, you could draw a house using rectangles, triangles, circles and lines. The only requirement is that it should be made up of multiple shapes.
6. *Deliverables:*
    a. Your class definitions (Project2.py file)
    b. In a word document, explain your class hierarchy and what benefits you have gotten from reuse.
    c. Include the picture generated by your code in the same word document.

Using turtle graphics: In order to use turtle graphics in Python you must first import the turtle module. You can then use the help function in idle to find out what methods this module includes and what they do. Just type import turtle in the idle command window, hit enter, and then type help(turtle) and scroll up through the list and information. Assuming the assignment pen = turtle.Turtle(), some useful features include,:

- pen.up(),pen.down(): Set the pen state to be up (not drawing) or down (drawing)
- pen.right(degrees), pen.left(degrees): Turn the direction that the pen is facing. The amount of turn is indicated in degrees.

- pen.forward(distance), pen.backward(distance): Move the pen forward or backward the amount of distance indicated. Depends on the direction the pen is facing. Draws a line if the pen is down, not if the pen is up.
- pen.goto(x,y): Move the pen to the specified point, drawing a line along the way if the pen is down, and not drawing if the pen is up.
- pen.pencolor(r,g,b), pen.pencolor(s): Set the color that the pen will hold for all drawing until the pen color is changed. In the first version, each argument is a floating point number between 0.0-1.0; the first is the amount of red, the second, the amount of green and the third the amount of blue. In the second version, the argument is a string indicating a color by name or by its hex code, e.g., "green", "red", "#66FFFF". Hex color codes are at: www.web-source.net/216_color_chart.htm
- pen.fillcolor(r, g, b), pen.fillcolor(s): Set the color for filling figures. Arguments are the same as for the pen color.
- pen.circle(radius): draw a circle of the indicated radius. The circle is drawn tangent to the direction of the pen in a clockwise direction (if radius is positive).
- pen.write(string): Write a string starting at the present pen point.
- To fill a figure, use the command pen.begin_fill() before you start drawing it. Draw the figure, then execute the command pen.end_fill(). The figure drawn between the two fill commands will be filled with the present color setting.
- pen.clear(): Clear (erase) everything written by the pen.

The picture below illustrates a house scene that could be drawn using your classes.