04 Prove Assignment: Writing Functions

Purpose

Prove that you can write functions with parameters and call those functions with arguments.

Problem Statement

Most modern computer operating systems have graphical user interfaces (GUIs) that are rich with colors, icons, images, menus, tabs, buttons, text fields, sliders, scroll bars, etc. However, many Python programs are designed to run in a console window, also called a terminal window, to read user input from a keyboard, and to print text as output to the console window. This means that many Python programs are not designed and written to take advantage of the rich features of graphical user interfaces.

Python includes two competing libraries of code named `tkinter` and `kivy` that enable a program to have a user interface. During the previous lesson's milestone, you wrote code to draw at least the sky, clouds, and ground of an outdoor scene. During this lesson, you will write code to finish drawing the scene.

Helpful Documentation

- The prepare content for the previous lesson explains how to write functions.
- The prepare content for this lesson lesson explains the properties of a good function.
- The prove milestone of the previous lesson describes this assignment in more detail and contains additional helpful documentation.

Assignment

During this lesson, you will write code that draws the remaining elements in your scene. The scene that your program draws must be outdoor, the sky must have clouds, and the scene must include repetitive elements such as blades of grass, trees, leaves on a tree, birds, flowers, insects, fish, pickets in a fence, dashed lines on a road, buildings, bales of hay, snowmen, snowflakes, or icicles. Be creative.

Each repetitive element must be drawn by a function that your program calls repeatedly, once for each repeated element. For example, your program may include a function named `draw_leaf` that your program repeatedly calls to draw each leaf at a different location.

As you write your program, be sure that it draws elements in the order of farthest to nearest. For example, your program should draw the sky first, then clouds, then the ground, then trees, and then insects in the trees.

Scene Gallery

All the scenes in this gallery were drawn by programs written by former CSE 111 students.

Testing Procedure

Verify that your program works correctly by following each step in this testing procedure:

1. Run your program and verify that it correctly opens a window and draws within that window a complete outdoor scene that contains the sky, clouds, the ground, and other elements.

Exceeding the Requirements

If you wish to exceed the requirements of this assignment, you could write code to add more items to your scene. The most efficient way to add more items to your scene is to write a function that draws something, such as `draw_pebble`, `draw_snowman`, or `draw_fish` and then call that function many times in your `draw_scene` function. Exceeding the requirements of this assignment is optional.

Ponder

After you finish this assignment, congratulate yourself because you wrote a Python program with many user-defined functions. As you wrote your program, what did you learn about organizing a program into functions? Did you learn that you can work more efficiently by writing a function once and calling it many times with different arguments?

Submission

To submit your program, return to I-Learn and do these two things:

1. Upload your program (the .py file) for feedback.
2. Add a submission comment that specifies the grading category that best describes your program along with a one or two sentence justification for your choice. The grading criteria are:
    - 1. Some attempt made
    - 2. Developing but significantly deficient
    - 3. Slightly deficient
    - 4. Meets requirements
    - 5. Exceeds requirements