Scene

The goal of this assignment is to practice using the graphics module, as well as simple control structures (`if/else` statements).

Deliverables: your solution file named `username1_username2.py` Please be sure to use this naming convention so that the grader can easily see from the file the 2 partners that worked on the project. Also don't forget to include a comment at the top of your .py file with both authors names.

Your task:

Use Zelle's graphics module to write a program that draws a scene. Remember you must have the file `graphics.py` in the same directory as your program and to import graphics at the top of your file. Your scene can be as simple or as complex as you like, as long as it meets the basic requirements. I do not expect any art skills whatsoever (I myself have basically none), but it's worth spending a little time playing around with the size, color, and position of even basic shapes in order to create some recognizable picture (e.g. a rectangle and 2 circles to form a car)…

The minimum requirements for your scene:

- uses at least 3 different shape objects from the graphics package (the 5 possible options are *line*, *circle*, *oval*, *rectangle*, and *polygon*).
- includes a stationary background of which some portion is sky (see the next part of the assignment for how the sky's color will be determined by the user running your program), the sky portion could be most of the scene, or could just be a small bit seen through a window of some sort.
- includes something animated - this should be something simple made of only a few shapes at most so that it doesn't slow your program down too much. Examples from the past include a puff of smoke moving up from a chimney, a simple V shape to represent a bird that flies across the sky, or a basic fish made up of just an oval and a triangle that swims across a lake.

In applications like CG animation, a single scene may be used as the general "set" for much of a story, though the time of day or current weather constantly change. It is useful, therefore, to be able to change the sky portion of a given scene only based on whatever current conditions are desired. For this project we will only worry about changing the background sky color based on the time of day that is input by the user. For simplicity assume that 12pm (noon) is the brightest time of any day, and the sky at 12pm should be pure white (rgb values 255, 255, 255). The darkest sky should occur at 12am (midnight) and be pure black (rgb values 0,0,0). The times in between should be varying shades of gray (any values where r, g, and b are all equal will yield a shade of gray).

You will ask the user to enter an integer 1-12 indicating the time of the day (only the hour, we are ignoring minutes), and then to enter 'am' or 'pm'. Your scene will be drawn with the appropriate sky color based on the user's input. Note that from 12am to 12pm the sky should get lighter each hour, while from 12pm to 12am the sky should get darker each hour. You should scale the lightness/darkness of the sky color more or less linearly, so at 6pm and 6am the sky should be approximately the rgb color (255/2, 255/2, 255/2), which should look like a gray halfway in between black and white.

You could write this program using something like 12 "if" statements, however there are much more efficient methods (remember your modulo division with the % operator), and for full credit you must do something more efficient. However, a few "if/else" statements are fine (even necessary) to deal with the edge cases of 12pm and 12am.

I can't really give you sample output for this HW without giving away the code you're supposed to figure out, but the idea is that when a user runs your program, the first thing it does is ask what hour of day to use, then to draw whatever your scene is with the appropriate sky color, and immediately start whatever small animation you choose to include.

Bonus (2pts):

For a 2 point bonus, you will automate your scene changing times in a simple time-lapse way, as well as use shades of blue instead of gray. Your background at the darkest hour should be a deep blue, something like (0,0,25) works, and at the lightest hour a light blue (225,225,255) looks good.

You should keep each color for somewhere around 1 second, it doesn't have to be exact, but it does have to stay long enough that a watcher can register each new color before it switches to the next one. This means the tricky part is coordinating the color change with your animation loop, as you will want your animation loop to run several times between each color change. Again think about using modulo, how could you make something happen once every 100 times a loop runs?

You will still begin the scene with the appropriate color at whatever time the user requests, and then continuously cycle through the colors for the hours that follow.