Lesson 13: Making a Night Sky with Tkinter

In this lesson, we'll learn more about how Tkinter draws shapes to the screen, and use this to draw an animated night sky with stars!

PART 1: Setting up Tkinter

Like before, we'll first need to import the `tkinter` and `random` modules.
We'll also import the `time` module in order to have greater control over the main loop.

Try typing the code below into a new program:

```
from tkinter import *          # Import the desired modules
import random
import time
myTk = Tk()      # Create a new Tk interface and store in myTk
canvas = Canvas(myTk, width=400, height=400, bg="black")
canvas.pack()   # Pack the canvas into the Tk window
```

This time, when we made our canvas we also set its background color to `"black"`.
We'll use a black background for the canvas to represent the night sky.
Afterwards, we stored the canvas object in the variable `canvas` before packing it.

PART 2: Adding a Color List

Next, we'll make a list of colors that our program will choose from when drawing stars.

Try adding the following code to your program:

```
canvas.pack()
starcolors = ["yellow", "gray30", "gray50", "gray70",    # This list
              "gray90", "gray95", "white"]  # can be on multiple lines
```

When we tell our program to draw stars later, we'll pick a random color from this list.

`Tkinter` lets us use many different versions of gray by adding a number to the name!
We can use any whole number between `gray1` and `gray99`.
`gray1` is almost black, `gray99` is almost white, and `gray50` is between them.

PART 3: Creating the Main Loop

Like in the previous program, we'll need to use a main loop when working with `Tkinter`.

Try adding the following code to the end of your program:

```
starcolors = ["yellow", "gray30", "gray50", "gray70",
              "gray90", "gray95", "white"]
while True:
    myTk.update()
myTk.destroy()
```

Just like before, we'll use `while True` to keep the `Tkinter` window running in a loop.

PART 4: Creating a Circle Function

Next, we'll add a function that draws circles, which we can use to create blinking stars!

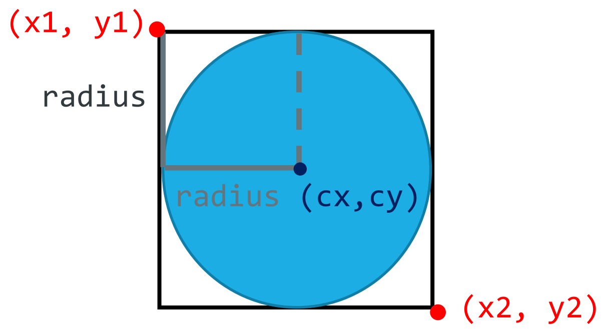
Unlike functions we've defined so far, this function will use more than one parameter.

The first two are `cx` and `cy`, which are the coordinates of the center of the circle.
The third parameter is `radius`, which is half of the distance across the circle.
The last parameter is `color`, which is whatever color we want that circle to be.

Try adding this code above the main loop:

```
starcolors = ["yellow", "gray30", "gray50", "gray70",
              "gray90", "gray95", "white"]
def drawcircle(cx, cy, radius, color): # Parameters are separated by a ,
    x1 = cx – radius   # cx and cy stand for center x and y
    x2 = cx + radius
    y1 = cy – radius
    y2 = cy + radius
    canvas.create_oval(x1, y1, x2, y2, fill=color)
while True:
    .  .  .
```

The `canvas.create_oval()` function asks for the coordinates of two corner points in a rectangle that surrounds the oval, as well as a fill color.



In order to draw a circle at the coordinates (`cx`, `cy`), we'll use `cx`, `cy`, and `radius` to find the corner points of the rectangle that surrounds our circle, (`x1`, `y1`) and (`x2`, `y2`).

After this, our `drawcircle()` function calls `create_oval()` using these coordinates.

PART 5: Adding Some Scenery

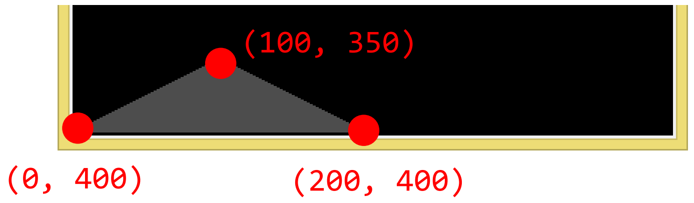
Before we draw any stars, lets add a mountain to our night sky scene!

Try adding the following code inside the main loop of your program:

```
while True:
    canvas.delete("all") # x1  y1   x2   y2   x3   y3
    canvas.create_polygon(0, 400, 100, 350, 200, 400,
                                  fill="gray30")
    myTk.update()
myTk.destroy()
```

First, we call `canvas.delete("all")`, which tells the canvas to delete all its objects. This will keep our program from slowing down by having too many objects being drawn.

After this, we call `canvas.create_polygon()`, which lets us make a shape by setting several points in the parameters. Here, we've used three points to create a triangle.



Now, you program should draw a mountain in the scene like this!

PART 6: Placing Stars in the Scene

Earlier in the lesson, we made a list of colors that we could use for drawing stars. Now, we'll use these colors with our `drawcircle()` function to draw blinking stars!

Try adding this code to the main loop:

```
while True:
    canvas.delete("all")
    canvas.create_polygon(0, 400, 100, 350, 200, 400,
                          fill="gray30")
    c = random.choice(starcolors)     # Copy these two lines
    drawcircle(200, 200, 4, c)        # several more times
    myTk.update()                     # with different locations
    time.sleep(0.05)
myTk.destroy()
```

Here, we're using `random.choice()` to pick a random color from the `starcolors` list. This color then gets stored in the variable `c`.

After this, we call `drawcircle()`, and pass the function two coordinates for the center of the circle, a radius, and a color. In this case, we'll draw a star at the screen's center.

Since we're finding a random color every update, `drawcircle()` will make the star's color different every repeat of the main loop!

After the update, we'll call `time.sleep()` to tell our program to wait `0.05` seconds (or a 20th of a second) before moving on to the next repeat of the main loop.

This will keep our program from changing the star colors too quickly.

Once you've tested your program, copy and paste the two lines marked above several times, and change the coordinates and size of each new star.

Make sure you set the variable c to a new random value before drawing each star, or all of the stars will blink with the same color.

Now, your program should create several blinking stars in the night sky!

PART 7: BONUS: Adding a Tree with Lights

Once your night sky scene has some stars, you can also add a tree to the scene!
To do this, we'll first add a new list of colors for the lights on the tree to choose from.

Try adding this code near the top of your program:

```
starcolors = ["yellow", "gray30", "gray50", "gray70",
              "gray90", "gray95", "white"]
lightcolors = ["red", "orange", "yellow",
               "blue", "violet", "white"]
def drawcircle(cx, cy, radius, color):
```

You can choose your own colors to make the lights blink by changing this list!

Next, we'll add the tree shape by creating another triangle in the scene.

Try adding the following code in your main loop:

```
while True:
    .  .  .
    c = random.choice(starcolors)
    drawcircle(200, 200, 4, c)
    canvas.create_polygon(350, 300, 325, 400, 375, 400,
                          fill="darkgreen")
    c = random.choice(lightcolors)   # Copy these two lines
    drawcircle(345, 350, 3, c)       # several more times
    myTk.update()
```

Like before, we used the `create_polygon()` function to add a triangle to the scene.

Last, try copying and pasting the marked lines above 2 or 3 times, and then try changing the coordinates so that the new lights are on the tree as well.

Now, your program should also draw a tree with lights on it!