Proximity Triggers Homework (Due 2/6)

In this assignment, you are going to get some extra practice with the `Vector3` class and with distance calculations. And also, Pokémon.

Project Setup

Create a new project (put your name in the title) and then:

1. Import the "Characters" Unity package. (`Assets -> Import Package -> Characters`)
2. Add the 3D Pokémon blender models to your project (download). They are 3D printable models from FLOWALISTIK, converted to Unity-compatible blender models.

Scene 1: Teleporting Pokémon Forest

1. Create a new scene `01_TeleportingPokemon` and add a ground plane. Bring in the RigidBodyFPSController controller and delete the unnecessary Main Camera.
2. Remove the default skybox and set up a more "cartoony" ambient lighting.
3. Add some Pokémon to your scene (that are positioned on the ground plane).
4. Create a script (`GlitchTeleport.cs`) and attach it to your Pokémon.
5. Write the script so that when the player gets within a certain distance of the Pokémon, it:
   - Teleports to a new location (on the ground plane)
   - Changes to a random color
   - Changes to a random scale
   - Changes to a random y-axis rotation. I.e. the Pokémon is "looking" in a random direction. Remember Quaternion.Euler?
6. Fill the world with teleporting Pokémon.

Scene 2: Lit Pokémon

1. Create a new, dark scene `02_LitPokemon` and add a ground plane. Bring in the RigidBodyFPSController controller and delete the unnecessary Main Camera.
2. Create a spotlight pointed at a Pokémon (or a gaggle of Pokémon).
3. Create a script (`LightTrigger.cs`) and attach it to the spotlight. Write the script so that the light only turns on when the player is nearby the spotlight.
   - Hint: you can use `light.intensity` to turn a light "on"/"off", or you can set `light.enabled` to `true` / `false` .
4. Extend the scene so that there is a path of proximity triggered lights that guide the player to the Pokémon & spotlight.
5. (Bonus) Make some aspect of the lights smoothly change as the player approaches (e.g. spot angle, range, intensity, color). E.g. the color could change from white to red, the closer the player gets to the light. You'll want `Mathf.Lerp` or `Color.Lerp` for this.

Scene 3: Staring Pokémon

1. Create a new scene `03_StaringPokemon` and add a ground plane. Bring in the RigidBodyFPSController controller and delete the unnecessary Main Camera.
2. Create a script (`Staring.cs`), create a Pokémon and attach the script to your Pokémon.
3. Write the script so that the Pokémon constantly turns to stare at the player.
   - Hint: It's a lot easier than it sounds. You'll just need the LookAt method of the Transform class. You can invoke the method in a script like this: `transform.LookAt(...)` . (You won't need the `worldUp` parameter.)
   - Use `transform.LookAt(...)` , but then make sure the Pokémon's rotation doesn't change around the x & z axes.
4. Now, fill your scene with many more Pokémon that all turn to look at the player. This will likely be unsettling...

Submitting the Assignment

Before the start of class on 2/6, direct message me on Slack:

1. A zip of your Unity project folder. Note: the project folder is the one that contains `Assets` , `Project Settings` , etc. If you share a zip of the `Assets` folder itself, I won't be able to see your project - it needs to be the folder that *contains* the `Assets` folder.