Lab2 - 755769

First some design choices:

- I am using Unity's new Input System, so the way to capture the mouse and button actions is kind of different. In the scripts there will be some On<Action> handlers that are called when a GameObject with a PlayerInput component with some defined actions is present.

- The player has a cylindrical model, which means that rotating the camera on the Y axis, will rotate the player to the left or the right instead of the camera. The camera is inside of the player transform so it will rotate too.

Scripts:

- **CameraMovement:** Locks the mouse to have a smoother cam control and rotates the camera in the x axis and the player in the y axis based on the mouse movements and camera sensitivity.
- **CheckObject:** checks if there are objects in front of the player and sets the interactable object to the object detected. The detection is limited to a certain distance and layer of objects.
- **OpenDoor:** checks if the player enters the detection area of the door and opens and closes it if it is inside or outside it respectively.
- **IInteractable:** Interface that defines functions to be implemented by interactable objects.
- **Chest:** implements interactable interface for chests. Defines chest states and the message and actions to be performed when interacting with it.
- **CollisionsCam:** Checks the collisions of the player with trigger colliders. Used to know the room that the player is in.
- **Destroy:** Implement IInteractable. Define the message and actions(destroy) when interacting with destructible objects.
- **Disappearing:** Script to control the transparency property for the Shader that makes objects invisible when you get close to them.
- **ItemSpawner:** Spawns objects once per second inside of the defined area. Has a boolean to control whether it is allowed to spawn that is set to true when the player is inside of the room. Toggle spawn by pressing q.
- **ManagerScript:** Manages the whole Museum experience. Defines the states that the player can be in and the actions to take in each state. Handles some of the input and all of the text logic.
- **PlayerMovement:** Controls the movement of the player.

- **SkullScript:** Controls the golden skull. Rotates it in function of the distance with the player and changes the position when it gets too close. It doesn't destroy and spawn a new one because that is not optimal.

Shaders:

- **CoolShader:** Sets the color of an object to the one received as parameter.
- **EmitterShader:** Takes a color, a texture and an emission parameter to set the object's Emission property.
- **GoldShader:** takes a glossiness and metallic parameter to set the object's properties of the same name. Playing with the sliders on the shader's interface and the right color we can get a golden appearance.
- **InvisShader:** takes a transparency parameter to set the alpha property of the object. In this case, the transparency is modified from scripts to produce real time modifications.
- **TextureSurface:** Takes a texture as input and uses it to set the albedo of the object.