Lebanese American University
School of Arts and Sciences
Department of Computer Science and Mathematics
CSC458 – Game Programming
Spring 2022

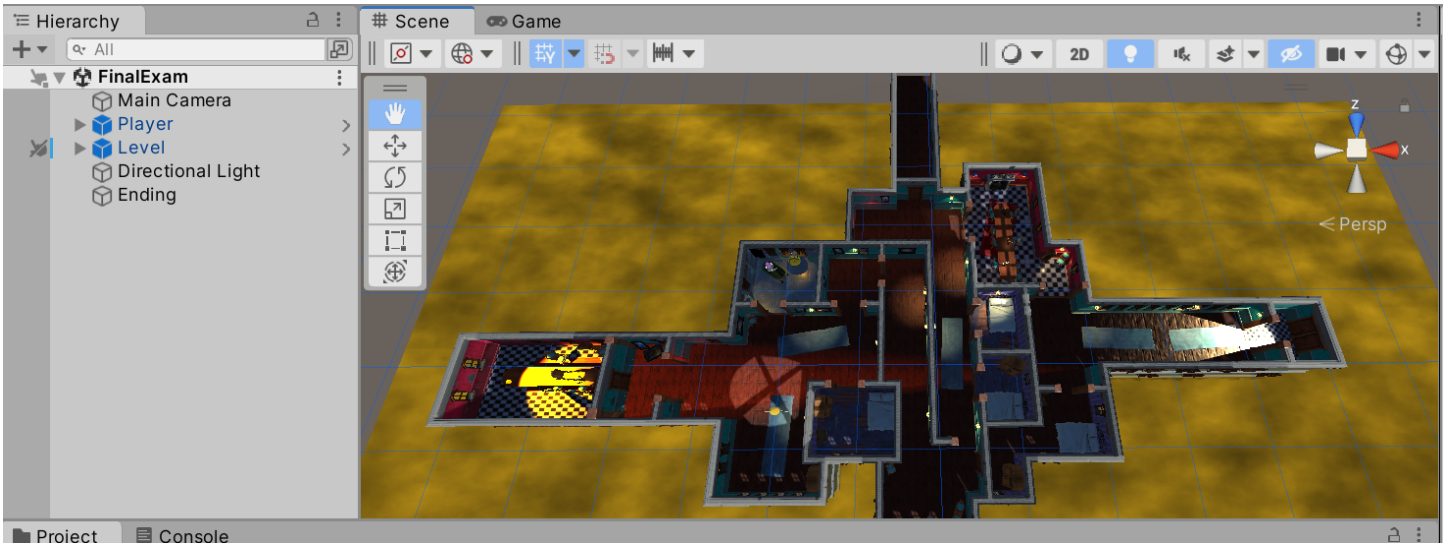
# Take-Home Final Exam

Sunday 8th of May 2022

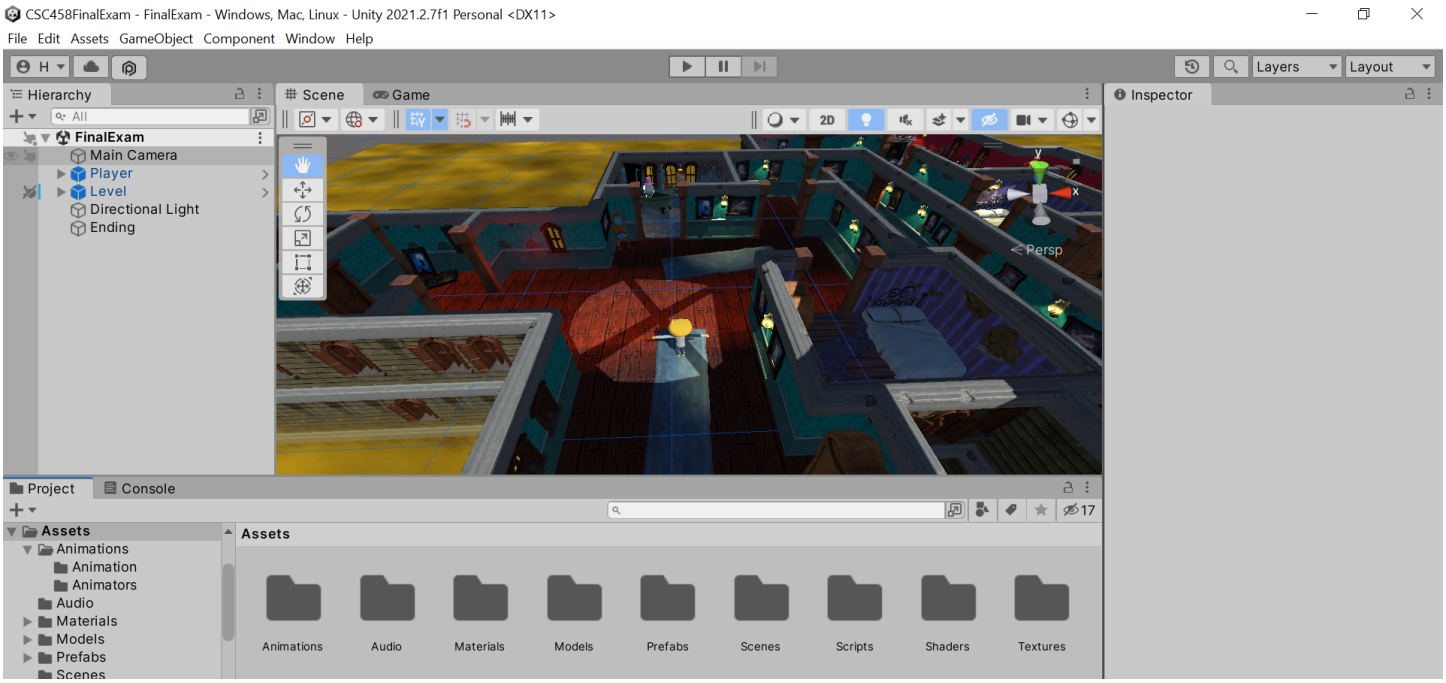
- The duration of this exam is **4 hours from 8:00 am till 12:00 pm.**
- You can submit the exam by **no later than 5:00 PM** (on same day Sunday 8th of May). Late submissions are not accepted. So, you must finish the exam on time (12:00 PM). The additional time is given to you just in case you need extra time due to electricity or Internet Issues. Please make sure that you work from the university and take advantage of the electricity and Internet provided in the university library.
- ***Cheating* or *any attempt* of cheating will be rewarded with a zero grade for all involved students. I do not care who copied from whom or what happened really. It does not matter the scope of cheating whether big or small, all cheating will be rewarded with a zero grade. I will put 0 for even a single line of code that is similar or that is copied.**
- ***Plagiarism of any kind and any degree (small or big) will be rewarded with a zero grade. You are not allowed to use any online material even if you cited it. There is no need. Everything is covered in lectures or projects.***
- Plan your time very wisely and do not spend too much time on any specific task. If you come across a task that you can't answer leave it and move on to the next task. You can always come back to it at the end.
- Code should be well commented and very tidy. Follow C# & Unity naming convention and programming style. It should be DRY. It should employ OOP design principles.
- Public questions are forbidden since they could be used as an attempt of cheating and pointing other students to certain solutions. If you have a clarifying question, drop me a private message on Discord or send me a private email to my LAU email and I will be happy to help. I am available all the time during the exam (8:00 am till 12:00 pm). I am also available afterwards but to a lesser degree.
- Good Luck!

Your exam is a stealth game that is missing some functionality. You will also add functionality as detailed in the document. You will be asked to write C# code and/or add some Unity functionality via the Editor to certain parts of this game.

You will also be provided with the starter project called CSC458FinalExam.zip. The Starter project can be found on BB. During the exam, make sure you save regularly, create backup folders (copy exam files to a folder midway). Please when you finish your final exam, delete the Library folder (as you did in your Projects) and submit the exam to BB. You can use cloud services such as Google Drive or OneDrive if the file is large although it should not be. **NB:** You are responsible for the accuracy of your submission to BB. Any corrupt files or missing files will be your responsibility. I repeat: it is your responsibility not mine to make sure that you have correctly submitted your final exam files.




Your main player has a movement script on it




**The scene of the Final Exam is in folder scenes: double click on Final Exam scene**

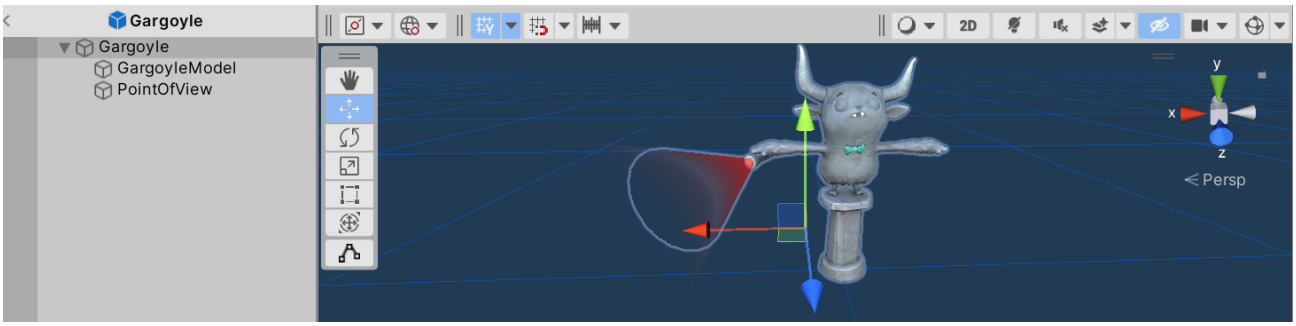

You must do the following task, each task has a certain weight of the overall grade:

**Task 1: Player Audio SFX, Cinemachine follow camera**

a) Add a component on the player that allows it to play audio SFX. We need to play the SFX audio which you can find it in the folder Audio. The SFX file is "SFXFootstepsLooping.wav". Disable Play on Awake and make loop checked. Change the PlayerMovement.cs script code to play the SFX when you walk only. Follow ### *** Task 1 - a *** ### in C# code to figure out where to include code.
b) Import the Cinemachine package and create a follow camera on the player (3rd person perspective), you can choose any algorithm you want (body and/or aim) as long as your virtual camera follows the player around in a 3rd person perspective (not top down). Keep the camera movement smooth and at specific nice offset to get the full grade of this question.

**Task 2: Triggers, UI, Scene Management**

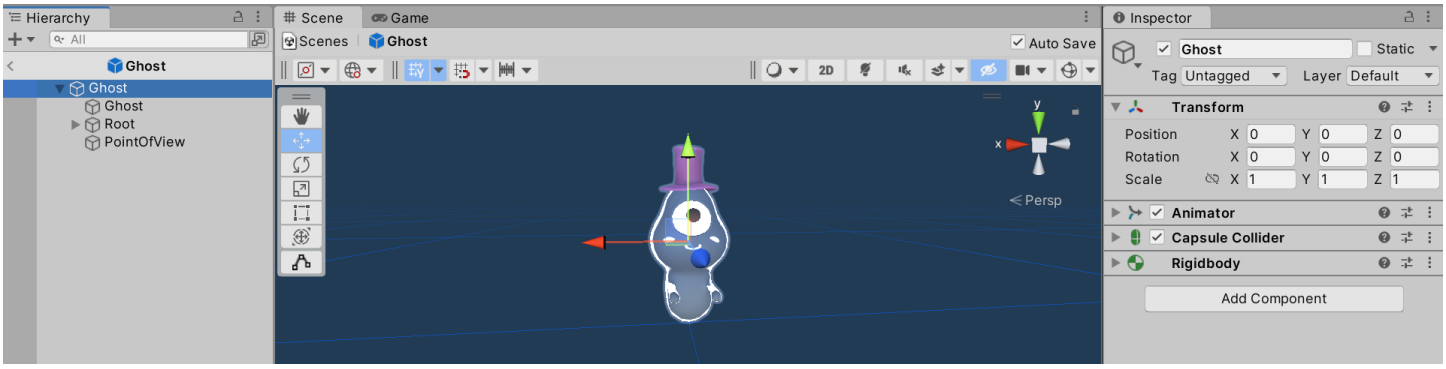
Your player has one type of the enemies: the evil Gargoyle (image below – prefab view). This enemy does not move but can detect the player if it is in the vicinity. As you can see from the image below, there is a PointOfView empty gameobject. The Gargoyle prefab can be found in folder Prefabs under the name Gargoyle.




a) In task 2 a, you must create a new C# script called scanning.cs which you should attach to the PointOfView empty gameobject. Add a component on PointOfView that allows you to create a trigger region. Think about what component can allow you to do this. The idea here is if the player step into this region, he will be busted. Make this invisible region expand a bit in front of the Gargoyle. Now write some C# code in scanning.cs to detect if the player has passed through the trigger region which you should have created graphically.
b) In this subtask, create a UI label on the screen (bottom left) which shows "Busted" for 1 second whenever the player enters the trigger region and then your scene will reload. Drag one instance of this Gargoyle prefab somewhere to test the effects of your C# code.
c) Drag this prefab and position several instances in the scene (scattered the way you like). You can place few of these Gargoyles near certain doors in corners precisely.

**Task 3: Ghosts navigation mesh capability, patrolling and raycasting**

The second type of enemies is Ghosts (as can be seen in the image below – prefab view). The Ghost prefab can be found in the folder Prefabs under the name: Ghost. Ghosts need to patrol certain waypoints. Here we have an empty game object called PointOfView which is at the eye level of the Ghost




a) In task 3 a, you must add a navigation mesh capability to this Ghost. Think about what component can make this Ghost an agent that can use the navigation mesh capability in Unity. Add this component at the parent level of the prefab i.e. with other components as Animator, Capsule collider etc… Do not forget to bake the navigation mesh in the environment. Speed of this agent should be set to 1.5, angular speed 120 and stopping distance 0.2. Enable also Auto braking.

b) Add a component on the Ghost prefab that allows it to play SFX when the ghost is passing by. We need to play the SFX audio which you can find it in the folder Audio. The SFX file is "SFXGhostMove.wav". Play on Awake and Loop should be activated.

c) Create a script called Patrolling.cs and attach it to parent gameobject in the Ghost prefab. In this script you need to serialize in the inspector an array or a list of waypoints that the game designer can drag to the inspector. These waypoints should hold location information of places the Ghost should go to. Think about what type of component that allow you to do that. So, the game designer would create as much waypoint locations in the scene as she wants for each Ghost instance in the scene. The Ghost must patrol through all these points in an efficient and smooth way infinitely.

d) Attach to PointOfView gameobject of the Ghost prefab a script called "RaycastingYouBuddy.cs". In which you use raycasting to generate a Ray going out of the PointOfView location (remember this is an empty object near the eye of the Ghost) and in a forward direction to a certain distance (which you specify). Fine tune the location of PointOfView so you can hit a player who suddenly appears in front of the ghost. Remember raycasting does not necessarily emanate from a camera but can be emanating from any game object as an origin. Hint: in this case PointOfView. If your ray hits the player, you need to show the same UI label "Busted" for a 1 second and then reload the scene. Make sure your code is DRY (Don't Repeat Yourself). You should have already created the functionality of the UI and reloading the scene previously.

## Task 4: Pausing and resuming the game

a) In Task 4 a, create an empty game object in your hierarchy. Name it either GameManager or GameController. Attach a script to it and name it GameManager.cs in which you write code that pause a game when the player presses the following keyboard keys: Escape or letter P, then resumes the play when they are pressed again.

## Task 5: Coroutines and Scriptable Objects

a) In Task 5 a, you must use Scriptable Object (SO) to store how many times you have been busted. Remember each time you are busted, a UI label appears for 1 second and then the scene is reloaded. Create a scriptable object template C# script + a scriptable object asset that stores the number of times in which your player is busted. Names of scripts are of your choice & you have the freedom to program this as you wish as long as it is a scriptable object. Now we need to use this scriptable object data container in a useful way. Create a UI label in the bottom right of the screen. This UI label should show

the number of times busted. Example: it could show "# busted: X" where X should be the number of times busted. Remember SO are dynamic i.e. only persistent between scenes but this is exactly what we want here.

b) Create a Point Light game object and set the intensity to 3 and put it in the bathroom where one of the Ghosts is taking a bath 😊.


Attach to this Point Light, a C# script called dimmingLightsGradually.cs. In this script write code to dim the lights gradually over time using a Coroutine. You can write this as you wish as long as it do the task accurately. Hint: you need to change the intensity gradually.

**Task 6: Game ending trigger**


a) Create approximately a trigger region in the circle shown in the image above. When the player reaches this region, a UI label in the center would be shown that says: "You won!!!". Once you reach this region you should quit (Application.Quit()). Try use also (optional) EditorApplication.isPlaying = false; to exit the playing mode in the editor itself. **Hint:** create an empty game object called ending and add a component that allows you to do that. Attach a script which you should create called endingScript.cs which should contain the functionality explained.

b) Just before you exit (in task 6 a), save the number of busted times either using playerprefs or serialization by saving to a file of your choosing. No other method of saving is accepted.

**Task 7: Global Post Processing**

a) In Task 7 a, create an empty game object called GlobalPostprocessing. On this game object you need to add a global post processing volume that would contain the effects: Color Grading, Bloom and Ambient Occlusion. Change these effects as you wish in a way that makes a difference. There are more steps to post processing than this, make sure you do all the steps necessary and make sure as well that I can see a clear visual effect in order to take the grade of this requirement.

**Task 8: Overall code tidiness, OOP Design principles, DRY code**

Code should be well commented and tidy. It should be DRY. It should employ OOP design principles. Follow C# & Unity naming convention and programming style as much as you can.

Good Luck!