Files worked on:

- Collisions
    - CollisionDetection.cs
    - CollisonHandling.cs
    - ICollisionHandler.cs
    - ObjectCollisionHandler.cs
        - IDoor collision w Player
    - PlayerCollisionHandler.cs
- LevelLoading
    - Room.cs
- Objects
    - Doors
        - All the files in here
- IGameObject.cs
    - This affected nearly every Interface
- LegendOfZeldaDungeon.cs
    - Did a decent amount in here, hard to keep track of what exactly. Feel free to DM if you have Qs
- LoZHelpers.cs
    - Added spawn locations

(below is just for reference)
Both reviews contain:

- Author of the code review
- Date of the code review
- Sprint number
- Name of the .cs file being reviewed
- Author of the .cs file being reviewed

Code Readability:

- Number of minutes taken to complete the review
- Specific comments on what is readable and what is not

Code Quality:

- Specific comments on code quality
- A hypothetical change to make to the game related to file being reviewed and how the current implementation could or could not easily support that change

KT's Review
Author: KT Goldstein
Date: 3/13/2021
Sprint #3
Files being reviewed: Listed above
Author: Simon

CollisionHandling.cs
Readability: Good call with the naming of the indexes; firstIndex and secondIndex is much easier to understand than using the typical “i” and “j” would be, which is important since HandleCollisions() is a very logic-heavy method.
Quality: I said this under IGameObject.cs too, but great use of the IGameObject interface design here. Keeps it a lot more simple and readable than it would be otherwise. Nice job decoupling the Collision Detection from the main game here.

CollisionDetection.cs
Readability: Simple and descriptive names. The ternary operator could confuse some people at first, but in my opinion it’s still better to sacrifice maybe a few seconds of extra comprehension to have it on one line.
Quality: Necessary math is made as simple as possible.

ICollisionHandler.cs
Readability: Descriptive method names, easy to read.
Quality: Good design, simple and minimal but covers all bases. It was a good idea to remove the Boundary and change that to an invisible object collision.

IGameObject.cs
Quality: Nice interface design, good design choice. I’m sure it took a ton of refactoring (and I’m sure that’s putting it lightly) but it makes all the other parts of the code really readable and generally less complicated (for instance, having only one list of IGameObjects in Collisionhandling.cs, the Collide() and GetCollisionHandler methods in CollisionHandling.cs as helper methods, etc).
Readability: I’m curious whether “HurtBoxLocation” refers to the hitbox or something else (actually I’ve been meaning to ask you about this), and if so it may possibly help with readability if you changed it to “HitBoxLocation” since that’s a more commonly-used term as far as I’m aware (unless there’s another reason for naming it this way).

LegendOfZeldaDungeon.cs
Quality: Good idea to put the dead projectiles in a List to have them all removed at once in a single update.
Readability: Good descriptive names overall, I think the only thing I’m confused about based on reading the code is the difference between PlayerProjectilesQueue and PlayerProjectiles (as well as the other versions of this in this class—is one static and one dynamic? As in, does the Queue hold one copy of every possible item in the game and then the PlayerProjectiles are the actual projectiles being drawn? Maybe a more specific name would help clarify the logical function of each list more easily, but this is just one small detail.

ObjectCollisionHandler.cs
Quality: Nice cohesion/separation of code responsibility, the CurrentObject is IDoor condition is kept really simple that way.

Readability: Since the code is simple, it's easy to read, too, since the method name is descriptive.

IDoor.cs + concrete implementations (e.g. OpenDoorUp.cs)
Quality: Good idea to have IDoor extend IObject so that it inherits all of IObject's new properties. Nice cohesion where the RoomUp/RoomDown/RoomLeft/RoomRight property is accessed to take care of the room change.
Readability: Simple logic and good variable names, easy to read.

Room.cs
Quality: Great idea having each of the cases with a locked door spawn both blocks at once; I was stumped on a solution to implement them spawning atop one another when I'd thought about it briefly. Good idea to have the Rooms keep track of their neighboring rooms.
Readability:

LoZHelpers.cs
Quality: Great idea so have the spawn locations in this file for easier access. Obviously some magic numbers are used but that's unavoidable; in the future the only way this can really be improved is by getting rid of them/replacing them with descriptive variable names, which will be part of our tasks soon anyways so it's not necessary to worry about now.
Readability: Descriptive names make it clear what's being described, easy to read.

Summary
Number of minutes taken to complete review: 110 min
Overall: Overall maintainable and high-quality, great readability. The IGameObject interface is an important and very helpful foundation for the rest of the project. Awesome job on this!

Code Review by Jaci Taylor
Code Author: Simon
March 14, 2021

CollisionDetection.cs:
- Nothing I would change here. Nice work keeping everything readable/understandable.

CollisionHandling.cs:
- Looks good to me. Code is readable and understandable (and short despite how much is being done in this file, which is good)

ICollisionHandler.cs:
- The only thing I noticed here was that the using statements at the top aren't being utilized, so they could be deleted.

ObjectCollisionHandler.cs (door interaction):

- I left this same comment on KT's, but I'll put it here as well. This is really nitpicky and is more of a personal preference, so it's totally up to you guys if you want to implement it. I would recommend changing line 44 to use a variable for the as statement for readability.

PlayerCollisionHandler.cs:

- Nothing necessary to change here. Same nitpicky choice as above though, I would change line 54 to utilize a variable for the 'as IDoor' statement for readability (definitely just my personal preference).

Room.cs:

- Magic numbers in Draw will need to be replaced (not sure if this is readability or code quality)
- Magic numbers in ProcessEntry lines 112-120 will need to be replaced
- Lines 8 and 9 are not being used in the code and could be deleted.

IDoor.cs:

- Looks fine to me

All other door files:

- Looks good to me. Files are all short which is good.

IGameObject.cs:

- Looks fine to me. This was a really good addition to the game

LegendOfZeldaDungeon.cs:

- Everything looks fine to me from what I could find that was new. Not sure if there are any remaining magic numbers or not but I couldn't see any. Everything is pretty readable too.

LoZHelpers.cs (spawn locations):

- Not sure if the numbers are actually magic numbers or not, but if they are then they'll need to be changed. Other than that it looks good.

Time taken: 25-30 minutes