Click “File -> Make a Copy”. Delete all red text when finished with essay.

Development and Iterative Process

1. Describe the 3-week development process that led to your recreation of a classic game (Do not simply reiterate the assignment specs).
2. To what extent did you utilize Project Management Software to keep your project on-track?
3. How did your game change as a result of playtesting and feedback?

1. At the beginning, my partner and I decided to begin on the first day of the project being released. We decided to both do our own templates so that we could both get a sense of how unity and the code worked. After that, we decided to split up the projects and decide on who will do what. We noticed that some tasks builds off of other tasks, therefore we decided to split it evenly in such a way that the tasks relate to one another. We did the same for alpha and gold too. In the beginning of each parts, we found ourselves more relaxed when it came to how we spent our time. But, towards the end of each part, we tend to spend all nights, usually 2-4 nights up until 3-4am trying to finish up the projects. Overall, I think our results was decent although we could have had better time management. In terms of developing the game, there were a few conflicts when it came to some decisions. Although it was a good idea to make a inheritance based enemy system, at the beginning, we almost decided to not do that, which could have cost us a huge amount of time. But, for the weapons, we did the exact opposite, and looking back

on it now, if we had made an inheritance system for weapons, we could have saved a decent amount of hours working on it.

2. We tried to used the project-management system, Jira, for all three parts of the project, but we had only logged our time whenever we finished a task. We tried to consistently use it, but found ourselves not needing to as we were mostly collaborating through discord. Posting on Jira did give us an idea of what we had to finish, and I always used it to look at a description of a task, but when it came to actually managing the project, we did not use it as much. Also, we did have some trouble using it, as we had never used Jira before.
3. The feedback was extremely helpful as it gave us a sense of direction towards what we needed to fix and complete. A lot of our bugs would not have been caught if it was not for the feedback, the feedback had drastically changed our end results. There were 2 other feedbacks that said the stalfos were walking through the walls and the player's movement was delayed when turning. For the stalfos, we had already knew about the walking through walls bug, and did “fix” it. But, in the end, after compiling it, the stalfos sometimes went through walls, which did not happen in the unity engine itself. Similar with the delayed player movement when turning, I had ignored this feedback as I thought it was a mistake. When using the unity engine, the player movement was not delayed one bit, and as a result, we did not change the grid based movement, but when we compiled the game, we felt the player movement delay when changing directions. But, at this time, it was already too late. We had to just submit it and let it go. But overall, the player feedback was extremely helpful and really impacted the game.

Custom-Level Creative Process

1. What was the initial idea for your custom level / mechanic?
2. How did you come up with this idea?
3. What were your goals / intended player impact with this idea?

1. Initially, we planned to implement some very simple, yet potentially deep mechanics, such as hidden doors that players wouldn't see right away, but would discover a passageway upon further interaction. We hoped that the player would be encouraged to explore areas surrounding stationary enemies in hopes of finding such passageways, but this mechanic was later scrapped as we felt like we'd need a more fleshed out mechanic, and players may still not notice the hidden passageways. We decided on implementing a custom grappling hook along with trap tiles. Traps are a feature in many platforming games, and in our Zelda iteration we wanted to provide consequences for playing too defensively. Now, these traps will collapse under the character if they stay on them, so players are incentivized to keep moving. Not only do these present new challenges, but both mechanics together present new ways to deal with enemies, such as by pulling them into a pit for an easy kill, or towards the player to obtain an item that couldn't be obtained otherwise. Aside from fighting enemies, we were able to develop puzzle minigames with these mechanics, where players would have to skillfully solve problems to obtain items such as keys, to proceed to the next room.
2. These ideas came to us in the spur of the moment. There were many ideas that we had, but we wanted something simple, yet with a lot of depth, and eventually narrowed it

down to these mechanics. Although we can say we came up with these ideas, it was probably subconsciously inspired by some games or shows that we had played or watched in the past.

3. With this mechanic, the impact we wanted for our players to have was a sense of fun and accomplishment. With the grappling hook, pit tiles, and falling tiles, we could make an abundant amount of different puzzles from an easy puzzle to a difficult one. We wanted it so that players can progress into harder puzzles as they move on. With players completing harder and harder puzzles, they would feel a sense of accomplishment when it came to solving something that they had spent a lot of time thinking about.

Future Improvement

1. What went right during this project?
2. What went wrong during this project?
3. What measures will you take to ensure an improved result for future projects?

1. One that went right during this project was most likely starting right when the project was released. If we had not started this project early, I do not believe that we would have met the deadline since we were constantly grinding (when given free time from other classes and work), yet managed to submit the project the night (roughly around 3-4am for all 3 parts) before the project deadline. Another thing that went right is our team management and laying out who will do what right at the beginning of each project. This gave us a sense of what tasks we needed to complete by the deadline and understanding the

approximate time we each needed to complete our tasks all together. Although our code could have been structured a lot more thoroughly, I think another thing that went right was the way we structured our code, since we didn't “need” to change much of what we had already done in milestone when entering alpha as there were a few people who had to rearrange most of what they had done.

2. One thing that went wrong in this project was that our outcome after gold was buggier than expected. Although our feedback for alpha said that player movement is delayed when changing directions and skeletons walk through walls, we did not really understand what it meant as our players and enemies did not walk through walls in the unity engine itself. Because of this, I had ignored it and focused more on the bugs that I could noticeably see and tasks that were incomplete, and this resulted in our player having delayed turns, and our enemies walking through walls sometimes. For our custom map, we trapped our enemy with a key in a “pit”, where if the enemy touches the pit and is not hooked to our grappling hook, he dies, making the first level of our custom map unbeatable. Another thing that went wrong is probably time allocation. There was a lot of time that we allocated into minor things, i.e. animations, sprites, panel UI, and minor unnoticeable “authentic” details, that probably wouldn’t have affected our grade and the game that much. We could have, instead, allocated more of our time into major buggy implementations, i.e the “enemy movement” and “player movement.” One more thing that went wrong is our code structure. Although this was our first time in developing a game, we could have structured our code a lot better, to the point where we could have saved a lot of time re-writing code. We could have made an inheritance structure for the enemies in the beginning, and also an inheritance structure for weapons as well, and

thoroughly design these polymorphic structures. One last thing that went wrong in this project was time management. Since on top of classes, I had to work 25-30 hrs a week, we spent roughly 3-4 nights up until 3-5 am each for each part (milestone, alpha, and gold). If we had managed our time a bit more wisely, I think we would not have had to spend much time this late at night.

3. One thing that we can improve on in future projects is the structure of our code. For future projects, I think planning out what objects we need in our game, and how we can structure our code and around them, at the beginning of a project is extremely important in the sense of saving time and creating a quality game. This will allow us to spot bugs much easier, and at the same time let team members understand your code faster. Another thing we can do to improve results for our future projects is time management. By planning out our weeks with our teammates, and letting each other know when we are free, and at the same time grasping the sense of time needed to complete the project, we can plan our days respective to when we are available. Lastly, what we can do to improve our project results is practice, and with practice comes experience. The more practice we have with not just unity, but game development as a whole, we could most likely drastically improve our project results and time spent on the project, as we wouldn't need to spend a lot of time trying to learn how to build the game, but rather just build it.