# Balancing Documentation

The balancing for the gamification was a three-part assignment. It was broken down into mini game balancing, incremental game balancing, and lastly user experience balancing.

## Mini game Balancing:

1. Coin and Experience Accumulation:

For coin and experience progression we decided to keep everything well balanced between each mini game. Initially the team before us had made each game balanced in such a manner that you would earn coins faster in some games than other games. We decided to change the balancing in such a manner that users would earn equal amount of coin and experience. The code for this was fixed in the incremental.cs script, and player.cs script. The coin and experience are a struct, so try not to change that. Initially this was made an object but had to be later changed since the classes weren't properly implemented. If you can figure out a better way to implement these two items as classes feel free to do so, but the current implementation works just fine.

## Incremental Game Balancing:

1. Coin and Experience Accumulation:

From all the coin and experience earned in each game our team decided we needed to apply these awards in the mini game. We made sure that all coin and experience accumulation was being earned in the mini games and then being stored correctly in the incremental game. This was done once again using the incremental.cs script and the player.cs script. The main issues we encountered during this balancing portion was not knowing how exactly the database stored this data. Make sure to check the player.cs script in the incremental folder to better understand this portion (saving player data) of the project.

## Game Play Balancing:

1. RPG: The RPG game needed to have the most amount of work done to it, but that was mainly handled by Nigel Haney. Please refer to his documentation for balancing information. The coin and experience accumulation are still the same implementation as mentioned above.
2. Conqueror: The conqueror game play was balanced by changing the a few things. First, our team slowed down the bullets being show by the enemies. Initially the game was too fasted paced for player and by slowing it down it made it easier for the player to progress through the game. The number of bullets being shot was also changed. Initially the game made it so that users were getting shot from multiple angels by multiple enemies and bullets. We changed the game in such a manner that users would have to face only a small number of enemies in earlier levels. Lastly, we added a timer to beginning of the game. This allowed the user to get set before beginning the game.
3. Sudoku: Nothing was changed in this game.
4. Sokoban: Refer to documentation for this game for details. Added more levels, timer, and reset button. Timer doesn't stop if user wants to reset the puzzle.

5. Daredevil: The rate at which the player falls was adjusted so the user has time to maneuver around the enemies and collect coins. The player was also moved higher up on the screen, so the user had more time to react to the objects flying their way. Please refer to Daredevil documentation for more information.

Files Changed:

Incremental Folder:

1. IncrementalManager.cs
2. PlayerInfo.cs
3. Update.cs

Changes to be made in coming semester:

1. Please look into balancing the gameplay of the Daredevil game and RPG game (Priority). Both these games are hard to test and very bulky so there are many bugs hidden in them regarding the balancing of the games. Gameplay balancing refers to the balancing the game in such a manner that users can play the game without getting bored because it is too easy or quit because the game is too hard to progress in.

Testing:

1. Main ways the balancing was tested was by doing user testing. I would allow users to play the game and write down or verbalize their thoughts on certain aspects of the game play. These written or verbal notes included things they liked and didn't like.
2. User testing was conducted on a bi-weekly basis or after every major feature implementation or game fix. Make sure you stay on top of user testing. After bug fixes make sure you play the game before allowing other to play it.