**Iteration Plan**

**High Level Objectives**

***Deployment:***

1. To deploy the game, you must go to <https://github.com/cncgen18/CSC440_RedSnapper> to download the zip file.
2. After getting it, you must also install gamemaker to open the files here: <https://www.yoyogames.com/get>
3. You will need to sign up for gamemaker, or if you have a steam account you can also grab the free version on steam for free.
4. Afterwards, open gamemaker and open the file debauchery.gmx. Then hit the “run the game” button. It should open the game, and you should be able to play around with it.

***Implementation:***

The four use cases we picked for this iteration is: move, jump, pause, and gameover. For each of these we had criteria of what we wanted to do. Unfortunately, some of those features will have to wait for the next iteration, as they tie into the other use cases.

**JUMP:**

For jump, we wanted to make sure of two things, that when you pressed the space bar, the player object would move up, and fall back down. The second was that if you pushed the directional keys (left or right), then the player object would move in that direction as well. While that was the main parts of the implementation we also realized that the jump had to be balanced and recognize collisions as well. If the jump wasn’t balanced, then the jump may just float and indefinite amount of time. As for the collisions we needed to make sure that when the player object jumped, it would land on the object walls. This was to make sure that it wouldn’t just go through the platform.

**MOVE:**

For move, we had to make sure that the player object was able to move in the direction corresponding with the key (left or right). If the directional key for right is stricken, then the player object would move right. Same with the left, you hit it, it moves left. Just like the jump, we had to make sure that collision happened. We didn’t want the player object to be able to move out of the room or obstacles in the game. In fact, it would be a game breaking bug.

**PAUSE:**

Pause was a bit easier. If you hit the esc key, then the game freezes. So it would take the most recent screen capture of the game, and darken the game. The game would also display “Paused,” in the middle of the screen. There are also other implementations that we would have added, but due to it connecting to other use cases we could not at the moment. We would like to add a menu that allows you to restart to the nearest checkpoint or go back to the title screen. Both are use cases we plan to do in the next iteration.

**GAMEOVER:**

The requirements for gameover are specifically if you fall off-screen, the screen darkens just like pause. There are also some other requirements that go with gameover, but those will be done later, as it connects to other use cases. The implementations are similar to pause, in which we would like to put a menu with options to: restart the level, or go to the title screen.

**List of work items with assignments**

1. ***Demo: Completed by – Alan***
   1. *Creation of objects*
      1. Player
         1. Create object for player
         2. Attach the main sprite to player
         3. Initialize variables for gravity, speed for jump, move, vertical, and horizontal.
         4. Assign keyboard checks for specific keys, and then change the x and y coordinates depending on type of action such as move and jump.
         5. Create the vertical and horizontal collision.
         6. Add global pause instance.
         7. Add an outside of room event for gameover.
      2. Room
         1. Attach the wall sprite to the object room.
      3. Pause
         1. Create a font for pause.
         2. Initialize a global variable for pause.
         3. Create a draw event when global variable = 1, have it change the tint and freeze the game.
         4. Create an event that activates pause when you press the escape key.
      4. Gameover
         1. Create a draw event exactly like the pause event.
   2. *Sprites*
      1. Gameover
         1. Create an empty sprite.
      2. Main (aka player sprite)
         1. Create a rudimentary sprite. Just needs to be a block at the moment.
      3. Wall
         1. Create a rudimentary sprite. Just needs to be a block at the moment.
   3. *Fonts*
      1. Gameover
         1. Create a font for the Gameover screen. This will be centered.
      2. Pause
         1. Create a font for the Pause screen. This will be centered.
   4. *Rooms*
      1. Test
         1. Add the player object to test room.
         2. Add the room object to test room.
      2. Gameover
         1. Add gameover object.
2. ***Domain Model: Completed by – Alan & Jeff***
3. ***System Sequence Diagrams: Completed by – Alan***
   1. *Fully Dressed Used Case*
      1. Jump
      2. Move
      3. Pause
      4. Gameover
   2. *System Sequence Diagrams*
      1. Jump + alt
      2. Move + alt
      3. Pause + alt
      4. Gameover + alt
4. ***Iteration Plan: Completed by – Alan***
   1. *Deployment Instructions*
   2. *Implementation*
   3. *Evaluation Criteria*
5. ***Design Document: Completed by – Jeff***
   1. *System Sequence Diagram*
      1. Jump
      2. Move
      3. Pause
      4. Gameover

**Evaluation criteria:**

1. ***Move (95%)*** – While we were able to successfully implement this, we are sure that there will be adjustments to the speed when we create the sprite and animation in future iterations. We were also able to successfully able to create collisions with the player object.
2. ***Jump (90%)*** – We were able to implement this with success, but I’m sure there will be a lot of balance for jump. We don’t want the player object to have a moon jump, but we also want to make it more realistic. This may take time to tweak around, but for now it has been successfully implemented.
3. ***Pause (80%)*** – We were able to pause the game, and have the word “Paused” in the middle. We were able to darken the screen as well. However, there is a bug that changes the coloring scheme a bit when you unpause it. Without a doubt it has to do with when the pause object is drawn. There are also future implementations we would like to add, which would be a pause menu. This ties directly to the title screen menu, and we are sure to implement it, when we do the title screen in the next implementation.
4. ***Gameover (80%)*** – While we were able to do a simple gameover screen. If you off-screen the game switches to the gameover screen. However, just like the pause screen, there are things we hope to implement later in the next iteration. The gameover menu would be a great addition, where it allows us to go to the title screen or restarts at a checkpoint. Both use cases have not been implemented yet.