1. Briefly describe the artifact. What is it? When was it created?

The artifact, like before, is a text based game in which the player that has the player collecting items while avoiding the boss enemy. Collecting all items before boss encounter wins the game, while encountering the boss before that loses the game. Player can choose to exit early, and has to make decisions in a turn-based gameplay loop.

1. Justify the inclusion of the artifact in your ePortfolio. Why did you select this item? What specific components of the artifact showcase your skills and abilities in software development? How was the artifact improved?

In this category, I had planned:

(1) to use different data structures to accomplish the intended functionality with

(2) improved map creation and

(3) streamlined navigation.

While in the course of the 1st enhancement, I inadvertently made the data structure switch from a dictionary-based game Map to a “list” or rather grid of linked node objects. This time I focused my attention on the **Item inventory**, which was messy and needed some reworking. It used to exist as an array in one place, an ArrayList in another, and as a transient set of strings in another sense. Now it has been **condensed into a single HashMap**, that is searchable by String, and used everywhere Item objects are created, accessed, or mutated.

The other big improvement was in the creation of the map. Initially, I had a 3x3 grid of manually initialized Room nodes. **Now, there is a 5x5 map, with automated creation that gets randomized each playthrough**. The player spawn location, boss spawn location, and item placements can be anywhere within the 5x5 grid. I also added an obstacle feature, that blocks a room off from players moving through, and slows down the boss moving through. There can be up to 3 of these obstacle rooms, depending on the RNG.

The existing navigation worked fairly well with the new map design, only needing a couple of minor adjustments.

I believe this improvement shows creativity, as I established new game mechanics, attention to detail as I had to be very intentional with the use of data structures, and persistence, as I had to work through several successive difficulties in implementation.

1. Did you meet the course outcomes you planned to meet with this enhancement in Module One? Do you have any updates to your outcome-coverage plans?

Yes, the 3rd outcome was addressed because I used algorithmic principles to design a solution while managing trade-offs between different approaches and data structures. The 4th outcome was also fulfilled because I had to use innovative techniques to implement computer solutions that deliver value.

1. Reflect on the process of enhancing and modifying the artifact. What did you learn as you were creating it and improving it? What challenges did you face?

I started off by identifying what I wanted to accomplish, and made a basic prioritized list of objectives, something like what developers might do in an AGILE Sprint, albeit without the numbers:

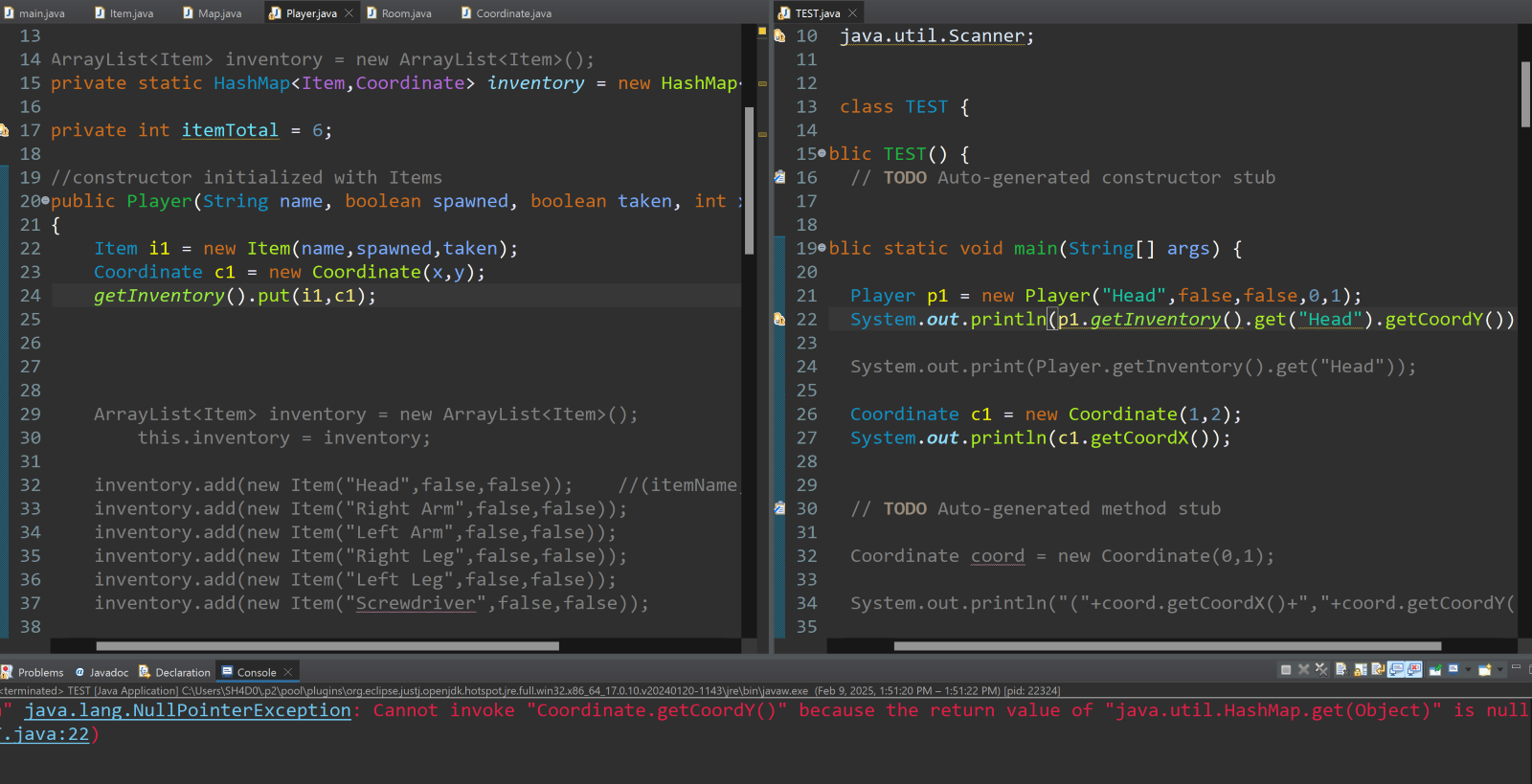
*Improvements planned for Algorithm and Data Structure category:*

* Inventory: multiple ArrayList convert to single Hashmap?
  + Hashmap: <Item, Coordinate>Class Coordinate
  + Hashmap <String, Item>
* Map: automated creation, randomized spawns
  + array [5][5]
  + obstacles?
* Boss(Map): navigation despite obstacles

*If there’s time:*

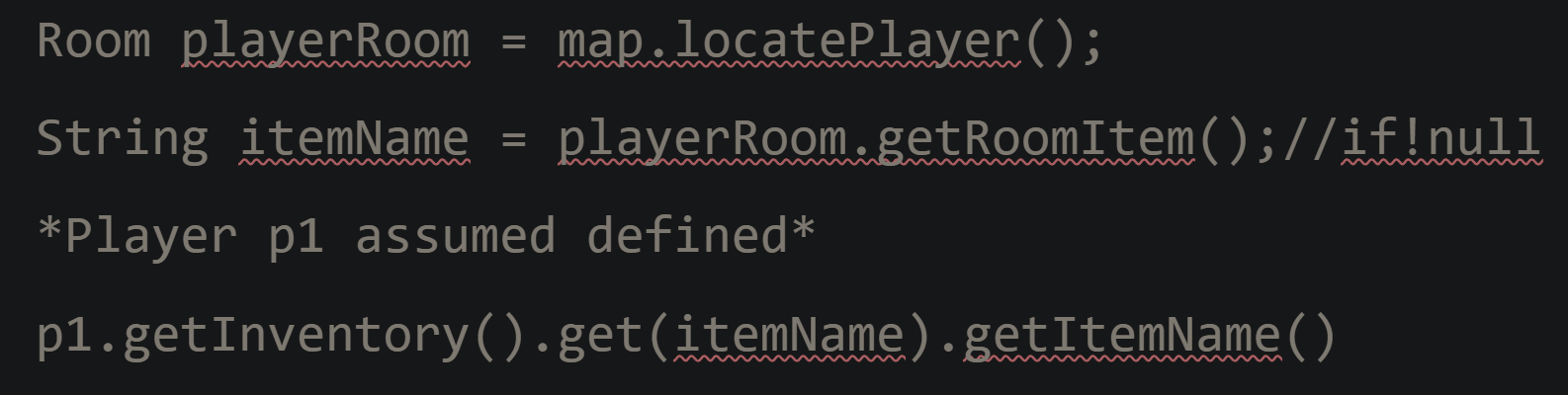
* Redo “randomizer”
  + Method that returns Coordinate obj
  + does random int nextInt(dmnX-1) and (rngY-1) and stores Coord
* Item: abilities
* Boss: create/destroy obstacles

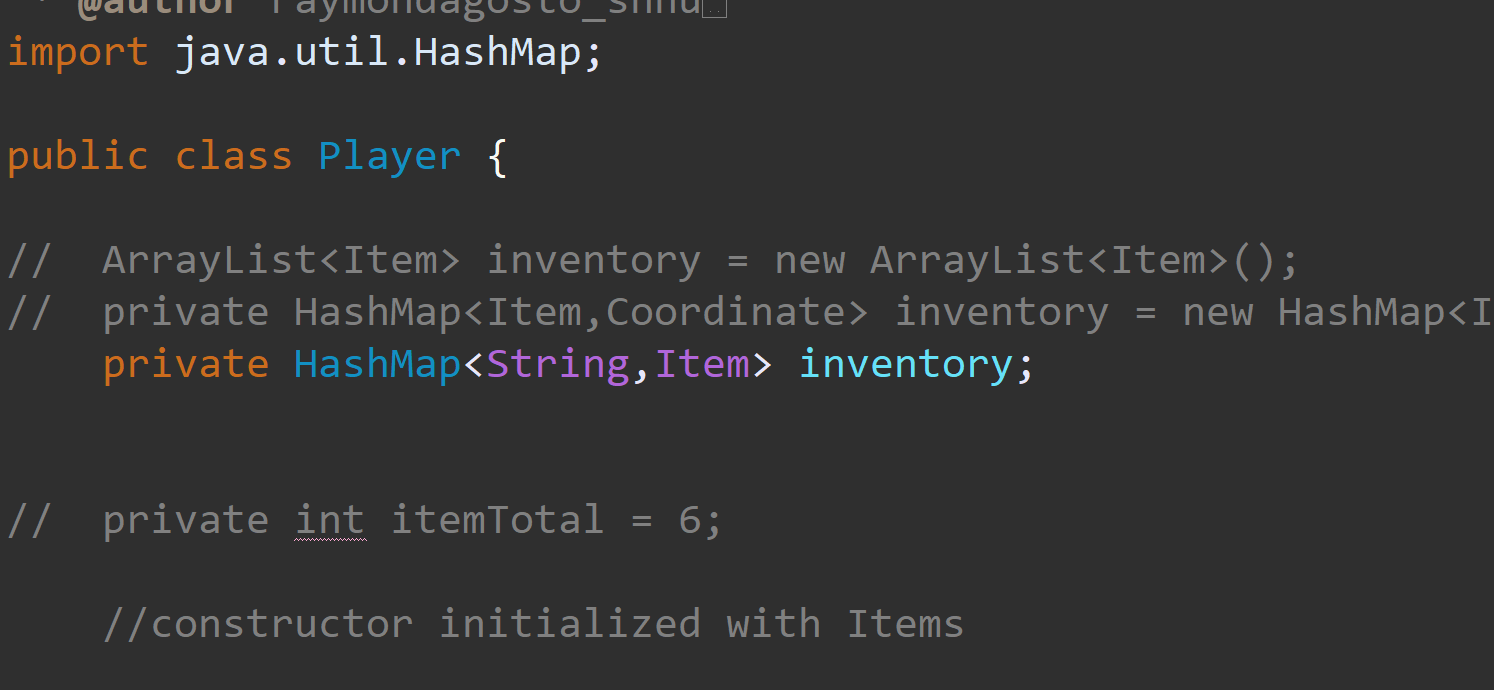
I tried to redo the inventory as a HashMap, but I was initially thinking of it as a data structure with 2 variables, rather than a key-value pair. I made it for Item class and Coordinate class (this class came back) objects, but those two are not initially related, though they are eventually. This confusion led to problems when trying to get one before the other was fully initialized.

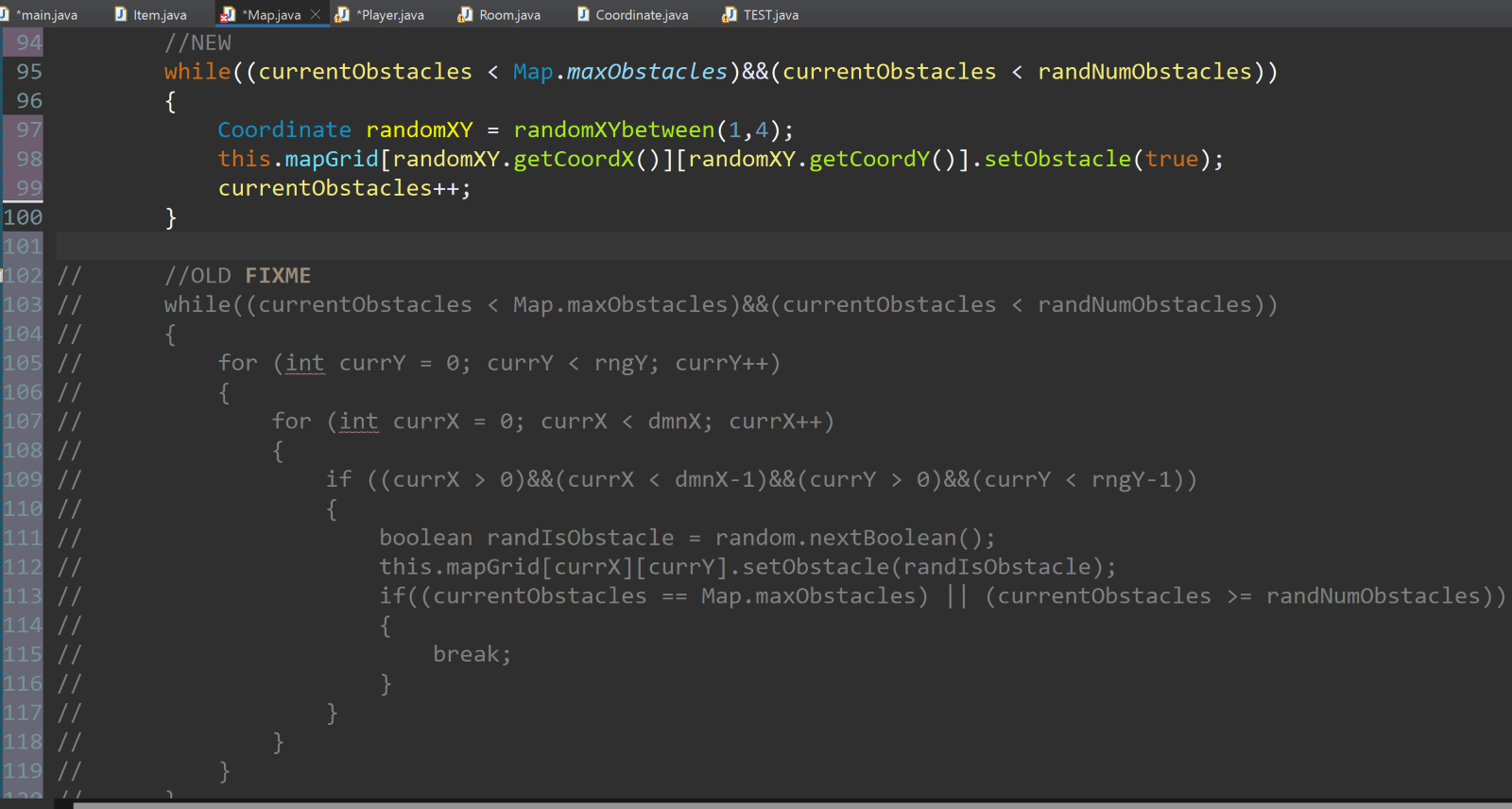
  
What I did not consider was in order to “Search” a HashMap <Item, Coordinate> you need to enter KEY of type Item... so if you called the item variable i2, HashMap.get(i2)... Searching such a HashMap with the Item.getItemName() returns null because the key is Item, not String....

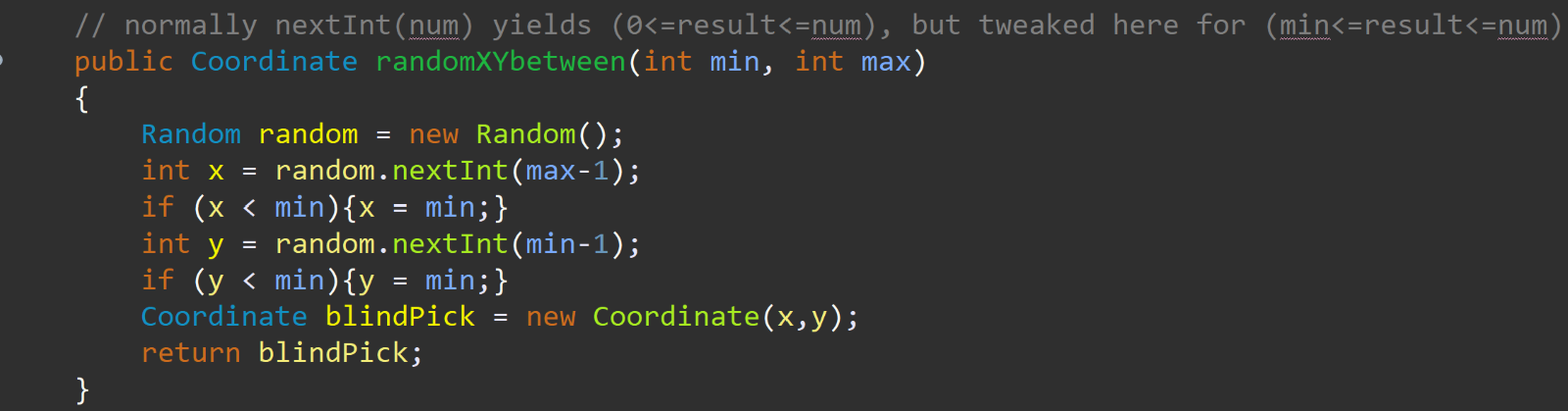
I then went with a new HashMap <String,Item> inventory that way the string can be used to search. I made a new attribute of type Coordinate for the Item class, so that by finding the item via String, you can still find that Coordinate necessary for linking an item to a Room on the map.

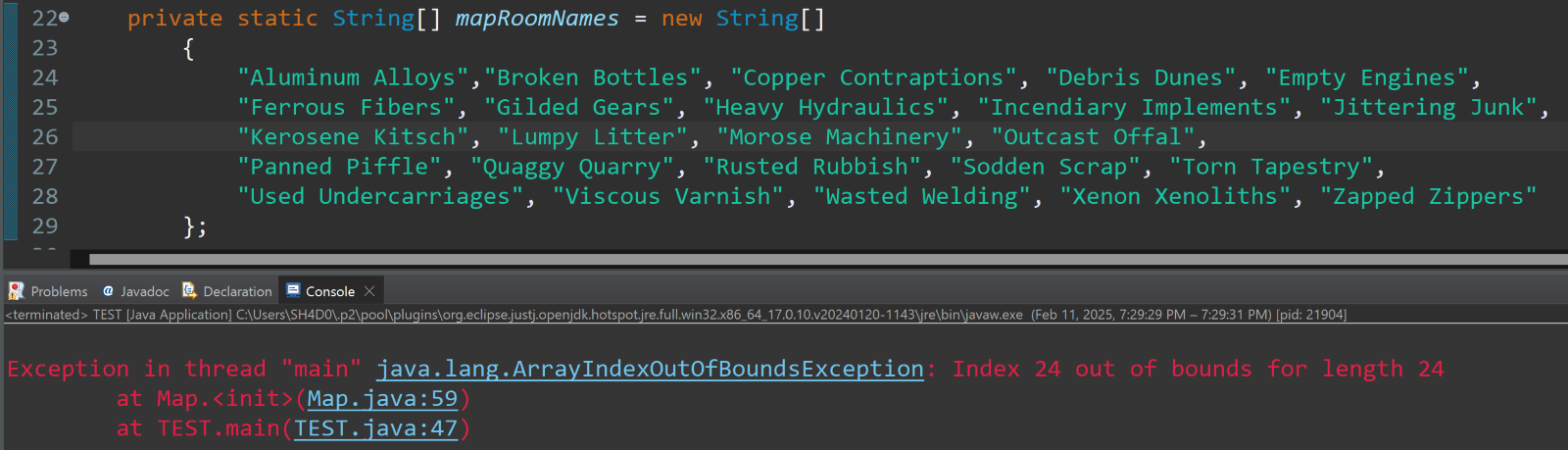
The pseudocode using existing methods looked like:

At this point I realized that there was not much reason for the Player class, as it now only contained the inventory HashMap<String,Item>, which was needlessly more difficult to reach being in a different class.

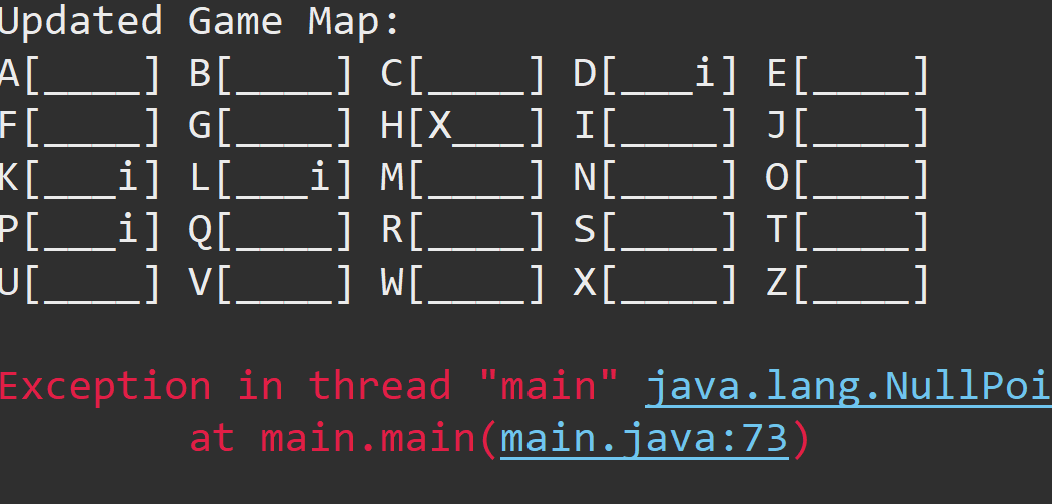


I gutted the Player class and moved inventory to the Map class. It was then that I started work on the automated random map creation. Initially, I was just looping through the entire map with a true-false “coin toss” whether or not each Room would be an obstacle, a player spawn, boss spawn, etc. However, with this there would be a heavy bias toward the initial corner of the map, as chances are everything would spawn before looping through the rest of the map. 

I then decided on a method that chooses a random set of xy coordinates and call that as many times as needed for the spawning/placement process.  
I then encountered a series of completely avoidable bugs. The automated creation was overrunning the bounds of the list of room names, which I was sure had 25 out of the 26 letters of the alphabet.

I was wrong, because I skipped Y and forgot N.

Player and Boss were not being placed at all..... so locatePlayer() returns null

  
Because I put while (spawned = false) instead of while(spawned == false)...

Items on the map were not as many as there should have been, because Items could be spawned in same place as others and overwrite the previous...

**These have since all been fixed.** I did encounter a rare bug involving the player and Boss moving at the same time, causing an exception to be thrown. I will need to recreate the bug and figure this out before the final version.

