CMPT 276 Assignment 3 - Code Review

David Wong, Peter Zhong (Group 18)

We began the refactoring process by identifying several code smells in our game. The most structural problem we noticed about our game was the lack of encapsulation, and how almost every variable was declared as public. Additionally, we noticed low cohesion and high coupling in files HitCheck.java and UI.java, where the law of demeter (single dot rule) is violated multiple times. We decided that our main priority would be to solve this by creating getters to avoid breaking the law of demeter, and privating fields which should be private. Fixing this would allow us to strive for the highest cohesion and lowest coupling possible, allowing for better modularity.

Other code smells which were more minor that we identified were the number of unused imports in almost all of our files. We removed most of these, and commented out ones which we may possibly need either for debugging or testing purposes. Following this, we improved our documentation by either removing or improving comments which had no purpose or were confusing to read. However, we did leave comments which are helpful for debugging purposes.

Finally, we identified a poor switch and if/else statement in HitCheck.java. There was an if statement which was repeated 4 times within a switch statement. We decided this could be improved by extracting the if statement outside of the switch statement for better code structure.

We first refactored HitCheck.java, removing unused imports, comments which were not necessary, and extracting the if/else statement in lines 45-90. This was done in commit `5a26a6ea.`

Additionally, we continued to refactor HitCheck by providing Getters in Entity.java, followed by implementing these Getters into HitCheck.java in order to satisfy the law of demeter, reduce coupling and increase code cohesion. These getters are located on lines 64-79 of Entity.java, and were vital to our entity collision function. This was completed in commits `5a26a6ea` and `fac9b796`

Following this, we refactored UI.java by first removing unused imports and then improving encapsulation. Due to the structure of our code, this unfortunately required changes in multiple different files. We first made the variables score and keyCount private, and then provided getters to each of these variables. We then had to provide an additional getter in Screen.java, in order to pass keyCount and score to player, which was then passed to gamepanel in order to be displayed in the UI (passed to UI.java). In hindsight, this could have been structured better. However due to time constraints and not wanting to recode our entire game, we decided that this would be a good ad-hoc solution for now in order to increase encapsulation and code safety. All of these changes were completed in commits `e69436d7` and `17ccbcfc`.

During this entire refactoring process we continuously tested our program to ensure that our changes did not prevent the program from running. Afterwards, we concluded our refactoring by removing variables which were not needed. The one place we did find this during our refactoring was in Player.java, where bonusItemCount is initialized by not used. The reason for this was because we initially wanted a counter for the bonus item, but instead decided to incorporate the bonus item into our regular score by having it award more points compared to a regular item. This was done in commit `6fc1d22e`.