

Assignment 3: Code Review

ObjectSetter.java

Poorly structured code:

This class had poor code structure due to long methods, bloated functions, code duplications and containing hard coded methods. This class had only two functions aside from the constructor: setObject() and setBonusReward(). The code in the setObject() was implemented to handle all components for creating objects, simply so we could plan where we wanted to place objects for map designing and play testing. Similarly, the setBonusReward() function was also implemented to handle all components of the bonus reward for similar reasons.

To fix this we restructured the entire class by breaking up the two methods and adding more functions. This helped make the methods more structured and helped with overall functionality. This made the code structure look much cleaner and reduced the number of lines in the class.

Long Methods:

Both methods setObject() and setBonusReward() were responsible for handling all objects in the game. The setObject() method would make all types of objects: rewards, punishments and the exit, and also set all of their locations all in this one method. The setBonusReward() method was also similar where it would by itself, create the bonus reward, obtain a random location for it to spawn on and then set that location to the object. Both methods, especially setObject() were unnecessarily long and could be broken up.

To fix this, we simply created helper functions for both methods. We created a maker function for each type of object: makeNewReward(), makeNewPunishment() and makeNewBonusReward(), which takes an index parameter to input into the object array and and two integer variables, x and y, for setting location (except for bonus). For setBonusReward() we made it a function that the game screen would call in order to remake bonus rewards. We also separated the part of the code that would find an appropriate location for the bonus reward to spawn at, called getBonusRewardYCoord().

Code duplication:

Both methods also contained duplicated code, which in this case was repeatedly setting the object locations for each object separately. For each object, the method would have these two lines of code: setting object worldX and setting object worldY. This was typed in every time an object would be created over and over again.

To fix this, we simply created a function called setObjLocation() which takes three parameters, an index, an x int value and a y int value. This way whenever an object was made using its make function, it would just pass it onto the setObjLocation() function with the object's index and x,y values to set its specific location.

Unnecessary if/else statements:

For the functionality of `getBonusRewardYCoord()`, it was coded without revision just so we could get bonus rewards to function as intended for planning map design and play testing. When refactoring the `ObjectSetter` class, we noticed that it contained unnecessary if/else statements that would do the same thing, resetting y.





To fix this, we simply revised how the function was supposed to work and realized that we could simplify the resetting conditions into a single if statement. We also got rid of an else statement that was redundant and wasted efficiency since failing its if statement was enough.

Lack of documentation:

As we were refactoring, we noticed that the class overall was lacking in comments that explained certain parts of the code. Now with the additions of new functions for refactoring it was evident that more comments and documentation was needed to explain the code.

We added more comments to old code and to new refactored code as well in order to improve the documentation of the class.

For all code smells and refactoring, please refer to these commits below:

	refactored objectsetter by making new fcns to make object setting more clean jra81 authored 1 day ago	2e75b774		
	refactored bonus reward fcn in object setter, got rid of useless else statements jra81 authored 22 hours ago	3822cae8	