Design rational - req 5

Toad class

This class acts as the vendor in the game. Toad class sells weapons (wrench) and magical items (Super Mushroom and Power Star) to players by taking in coins. It is also used to upgrade player's attributes. Toad class extends actor as it is an actor and it is associated with BuyAction as players are allowed to purchase items from it.

Description of attributes:

- SUPER_MUSHROOM_PRICE is a public static integer attribute with value of 400 that indicates the amount of coin to be deducted from player's wallet when player purchases the Super Mushroom
- POWER_STAR_PRICE is a public static integer attribute with value of 600 that indicates the amount of coin to be deducted from player's wallet when player purchases the Power Star
- WRENCH_PRICE is a public static integer attribute with value of 200 that indicates the amount of coin to be deducted from player's wallet when player purchases the Wrench

Description of method:

- tradeItem method that subtract a certain amount of coins from player's wallet and provide items to player if there is sufficient amount of coin in the player's wallet depending on option selected by player. If a is selected, subtract 600 coins from player's wallet; if b is selected, subtract 400 coins player's wallet; if c is selected, subtract 200 from player's wallet.

Why i choose to do it that way:

Since Toad is an actor, functions in the Actor class are required and hence need to extend the Actor class. I created this class in order to enable player to use coins to trade for weapons and magical items.

Advantages:

Excessive use of literals was also prevent by declaring SUPER_MUSHROOM_PRICE, POWER_STAR_PRICE and WRENCH_PRICE as private static attribute. This prevents confusion during coding process. Furthermore, if the value of HEALED_HIT_POINTS needs to be change, changes only need to be done at one place, which is at the line where that attribute is declared instead of going through entire code and changing their values . This minimise possibilities of producing errors too. Open Close Principle can also be implemented as this class extends the Actor class, by adding functionality to the Actor class without modifying its already available functionalities, in a way that does not change the way we use existing code in the Actor class. This enables the Actor class to support new functionalities as well as being added new methods easily.

Disadvantages:

N/A

Sapling class

Sapling is a subclass that extends the Actor class. In the game, sapling will spawn coins randomly and coins collected can be traded with the toad for Super Mushroom, Power Star and Wrench.

Description of attribute:

- SPAWNED_COINS is a public static integer attribute with value of 20 that indicates the amount of coin spawned by Sapling on its location in every turn.

Description of method:

- spawnCoin method is a method to add 20 coins into player’s wallet if it is collected by player

Why I choose to do it that way:

Since Sapling is an actor, functions in the Actor class are required and hence need to extend the Actor class. I created this class in order to enable player to obtain coins

Advantages:

Excessive use of literals was also prevent by declaring SPAWNED_COINS as private static attribute. This prevents confusion during coding process. Furthermore, if the value of SPAWNED_COINS needs to be change, changes only need to be done at one place, which is at the line where that attribute is declared instead of going through entire code and changing the value all of the “20” .

Disadvantages:

N/A

BuyAction class

BuyAction is an action that allows the player to use coin to purchase items from the toad. It is trigger by the Toad class when players use buy action to purchase items from the toad.

Description of method:

- purchaseItem method checks if actor’s wallet has enough money, if so subtract coins from player’s wallet. and provide player with the purchased item.

Why I choose to do it that way:

Since BuyAction is an action, functions in the Action class are required and hence need to extend the Action class. I created this class in order to enable player to purchase items from the toad.

Advantage:

This class is created using the Single Responsibility Principle where it does not need to take extra responsibility. In case of need to change responsibility, all pieces needed will be there. This makes the system easier to maintain and extend. Therefore, this class is only responsible for subtracting coin from player’s wallet whenever players purchase an item. This is because in the future there may be more items available for player’s to purchase. The Liskov Substitution Principle can also be fulfil as the initial meaning of the purchaseItem method behaviour from the Action class.

Disadvantage:

N/A