FIT2099 Object Oriented Design and Implementation
Assignment 1
Design Rationale

Group Name: LYYandTSM
Group Members: Lai Ying Ying(30526361)
Tan Sook Mun(30695759)
Lab: Monday,10am

Zombie

Zombie has an AttackBehavior class, and Zombie have many attack actions. The attack actions can be class into AttackBehaviour. AttackBehaviour will call AttackAction in getAction() so it will return the AttackAction that is available to the actor. The pickUpItemAction will allow Actor to pick up Item, which includes WeaponItem. The inheritances used enable all Zombie Actor to pick up Item, which prevents repeating code and adheres to the DRY (Don't Repeat Yourself) principle. Zombie must have 0 to 2 arms and 0 to 2 legs, otherwise throw an exception as this follows the FF (Fail Fast) principle. Zombie has a 30% chance of losing its limbs and this is managed in AttackAction to reduce dependencies from other classes (RED principle). If the target is a Zombie, execute() will call damaged_zombie() to decide whether or not the zombie will lose its limbs. Zombie has an enum of playerAttack. This is to ensure when calling AttackBehaviour, it only calls the attack behaviours that are eligible to Zombie. A bite attack action is added to Zombie's getIntrinsicWeapon method, so that the Zombie can either punch or bite if it does not have any other WeaponItem. The other features of Zombie such as movement speed, picking up item and groaning are added to playTurn method. GroanAction is inherited from Action and called by playTurn().

Player

Inherits from human because a player is human but with more of its own extra functionality The player has an AttackBehaviour. These classes allow the player to have many types of attack. One of which a player can craft weapons. A player has an arraylist of inventory that stores what the player has picked up eg: food, weapon and etc. Player is allowed to pick up items as it is inherited from Actor which is associated with the class PickUpItem. When the zombie drops its limb, it can be used as a simple club, therefore ZombieLeg and Zombie Arm are inherited from WeaponItem. Player is able to pick up ZombieLeg and ZombieArm, and either use them as a simple club or craft them into ZombieClub and ZombieMace respectively, which can cause more damage to the target. Player has an enum of playerAttack. This is to ensure when calling AttackBehaviour, it only calls the attack behaviours that are eligible to Player.

AttackAction

Both Player and Zombie have AttackBehaviour, therefore they are associated with AttackBehaviour which consists of AttackAction. AttackAction is a class consisting of

special Action for attacking other Actors. The damaged_zombie method is added to the AttackAction to determine if the Zombie should lose its limbs. The code to be reused as both zombie and player are able to attack other actors, this is in line with the DRY principles.

ZombieArm, ZombieLeg, ZombieClub & ZombieMace

ZombieArm, ZombieLeg, ZombieClub and ZombieMace classes are created and inherited from WeaponItem as players are able to use ZombieArm and ZombieLeg as it is or craft them into more powerful weapons(ZombieClub and ZombieMace). CraftAction inherits from Action and it is called in Player's playTurn and gives Player options to craft WeaponItem. Zombies can only pick up and use ZombieArm and ZombieLeg as a simple club.

Corpse

A Corpse is inherited from PortableItem. If a Zombie dies, execute() in AttackAction will create a new PortableItem, whereas when Human dies, a new Corpse() will be created. Only Corpse() are able to rise as a Zombie after 10 turns. The tick() is overridden and a counter is added. Inheriting from a class and overriding a method prevents repeated code (DRY principle). After 10 turns, if an actor is carrying the corpse, it will not rise as a Zombie till the actor drops the Corpse down.

Farmer

Inherits from a human but with more of its own extra functionality. Farmer extra functionality is creating food, fertilizing crops, harvesting crops.These extra functionality is added into FarmBehaviour. This FarmBehaviour is an inheritance of the behaviour interface. This FarmBehaviour checks if Farmer is standing next to dirt,crop or ripe crop and calls the FarmAction to sow, harvest, or fertilize the crop or dirt. When a farmer harvest a crop it becomes a Food because the Crop is inherited from Food. With these inheritance Food inherits the Item functionality and in a way is able to treat it as an Item. This allows the player to pick it up. This also reuses the code and following DRY. When a farmer stands over a patch of dirt it can sow a crop in it. The Dirt class can have crop on it. Thus it also connected to the Crop class. When the farmer harvest the crop, it will drop the food. Because Farmer is inherited from Human class which is also inherited from ZombieActor. ZombieActor is able to drop an Item with the connection to DropItemAction. This will allow the Farmer to drop the food after it is harvested. Before sowing a crop, the code will ensure the farmer is standing next to a dirt if not it will throw an exception. That is why Farmer can interact with the dirt class. Also, it is the same for harvesting, it ensures that it has already ripe or the crop is there before harvesting. The FarmBehaviour is only accessible to the Farmer and no one else for example Player or Zombie. This is following the FF(Fail Fast) principle.

Crop and Food

A Crop has CropCapability which is either Ripe or Unripe. tick() is overridden so a counter is added to count the number of turns. Food is inherited from Item and is portable. A ripe Crop will be replaced by Food and this is managed in the HarvestAction.