Design Rationale: Zombie Attacks
New Class Added: AttackHumanAction

Roles and responsibilities:
- Extends the AttackAction class because they share similar functionality, therefore duplication of code can be reduced.
- Random class has to be imported so that an instance of Random class can be created to generate the probability of hitting from the bite attack at 25%.
- Method nextBoolean() or nextDouble() from Random class can be implemented to achieve the probability of hitting.
- Represents an action carry out by the Zombie to attack the Human or Player at adjacent location of the player.
- Inherit AttackAction constructor which take in an Actor which is the target.
- Has attribute of type ZombieCapability to store the attackable team ALIVE so it only allows this action to be acted on the Human or Player.
- The execute() method will be override to let the zombie has 50% of probability of using bite instead of their normal attack. The bite attack is one of the intrinsic weapon of zombie, but it has a lower chance of hitting than the punch attack but higher damage. So, I will set the probability of hitting from the bite attack at 25% and the probabilities of the hitting from weapon or punch at 50%.
- Once the bite attack is not missed, heal() from Actor class should be called to restore 5 health points of zombie.

New Class Added: PickUpWeaponBehaviour
Roles and responsibilities:
- It implemets the Behaviour Class to override the method getAction() from Behaviour class.
- In the override getAction(), there will be a for loop that iterates every item in that specific location.( for ( Item item: map.locationOf(actor).getItems()) )
- Inside the for loop, portability and weaponization of the item will be checked.
- Method asWeapon() in Item class will be called to check the weaponization of the item.
- Method getPickUpAction() in Item class will be called to check the portability of the item.
- If the item can be weaponized and it is portable, then it returns PickUpItemAction().Otherwise, return null.

New Class Added: Word
Roles and responsibilities:
- It is a class that stores every words that can be shouted by zombie or human. For example, "Brainnn".
- It has an attribute with type String, **word**.
- It also has a constructor with a parameter String name.
- It has a method called getWord(), it returns the String **word**.

New Class Added: ShoutBehaviour
Roles and responsibilities:
-It is a class that returns ShoutAction that will undergo execute() to print a String.
-It implements Behaviour class to inherit the method getAction().
-It has an attribute with type ArrayList<Word>, it is named as **Words**.
-There is an override method called getAction(). The getAction() has 2 parameters
  which are Actor and Gamemap.
-In the getAction() a Word instance is created and **Words** adds the new instance. For
  example,
  ( Word word = new Word("Brainnn")    )
-There is an override method called getAction() that returns ShoutAction.

New Class Added: ShoutAction
Roles and responsibilities:
-It extends Action class because they share similar functionality, therefore
  duplication of code can be reduced, hence it improves reusability in java.
-There are override methods called Execute() to perform the shout action and
  MenuDescription() to display a descriptive string about what the actor is
  performing on the menu.
-It imports Random class from java.util.package. After importing Random class,
  Random instance can be created to perform its method to carry out the probability
  of shouting brain from each zombie in every turn.
-The probability of zombie shouting "brain" is set to 10%.

Class modified:Zombie
Modified part
● An override method called getIntrinsicWeapon().
● Add an override method called heal() from actor.
● Add pickUpWeaponBehavoiur into the attribute, Array of Behaviour type.
● Add ShoutBehaviour into the attribute, Array Behaviour type.
Roles and responsibilities:
-Bite is added as an intrinsic weapon of zombie.
-Zombie has 50% of probability using bite instead of punch. To achieve this
requirement, Random class has to be imported to Zombie Class.
-An instance of Random class can be created so that the method nextBoolean() can
  be used to generate the 50% probability of using bite for each zombie.

Class modified: AttackBehaviour
Modified part:
● An override method called getAction().
Roles and responsibility:
- In the getAction() , instead of returning a new AttackAction, it will first
determine the attackableteam, if attackableteam == ALIVE, it will return a new

AttackHumanAction. On the other hand, if attackableteaem == UNDEAD, it will return a new AttackZombieAction.


Class modified: AttackAction
Modified part:
- Become abstract class.
- A method called execute() is modified into an abstract method
- Add a new method called targetdDied()

Roles and responsibility:
-This class is modified into an abstract class that provide a base code for all the attack actions in the game, which are AttackZombieAction class and AttackHumanAction class where they share some similar functionality such as the execution part when the target dies.
- The execute method is modified into an abstract method since AttackZombieAction and AttackHumanAction has different execution.
- Has a new method called targetDied() that is responsible for the action when the target dies and this functionality is shared among AttackZombieAction and AttackHumanAction, therefore duplication of code can be reduced and it improves the reusability of code.