**Design Rationale**

For **Zombie attacks**, we created a ZombieAttackBehaviour which inherits AttackBehaviour and BiteAction class to inherit the AttackAction class. Since most of the methods and attributes are the same, we used the principle DRY (Don’t Repeat Yourself). Within the BiteAction class, we used named constants to set the probability, damage and health restored of each bite to be the same using the principle of DRY. Before beginning each turn a zombie will reuse methods (map.locationOf().getItems()) from Location class to obtain items on the spot it is standing on to pick up weapons.

For **Beating up the Zombies**, we added two attributes, armsNumber and legsNumber to the Zombie class. Since classes should be responsible for their own properties, it knows its own armsNumber and legsNumber at all times.

For **Crafting Weapons**, we created a crafting class. Crafting is added as a capability of the item zombieArm and zombieLeg using the addCapability() method of the item class. Crafting class has 2 attributes, club and mace which are both WeaponItem objects because the attributes and methods that club and mace are required to use are similar to objects of Weapon item class, using the principle of Don’t Repeat Yourself. The method createClub() takes a zombieArm item as a constructor to return a club while the method createMace() takes a zombieLeg item as a constructor to return a mace.