Design Rationale

REQ4:

Metal Pipe

In my implementation MetalPipe, it is both Item and Weapon, so it extends WeaponItem abstract class. If the MetalPipe is in the inventory of Actor, it can be used as weapon to attack other actor. The attack action is created by the MetalPipe instead of the owner. The reason for this is to achieve Open-closed principle, as we do not need to modify any existing code to accommodate the new action the actor can performs if the MetalPipe is in their inventory. The engine code will loop through all the Item in the actor's inventory and get the action the actor can perform using the item.

Consumable

To implement consumable item, I added an abstract class called Consumable which all consumables should implement. Because all consumables are items, Consumable abstract class extends Item abstract class, thus all concrete classes of Consumable abstract are consumable item. The reason Consumable is needed is because I want all the consumable items to be able to consume themselves by the given actor. This can be used to implement what happens when the actor consumes the consumable item. Each unique consumable item can now implement its own implementations. This abides by the single responsibility principle as each consumable item manages its own logic of what happens after being consumed.