Design rationale for Requirement 4:

Requirement 4 entails the creation of two classes: special scraps and a further implementation of Fruit.

For the special scraps, I designed the AttackItem class as a subclass of WeaponItem. Within this class, I incorporated an AttackAction into its allowableActions. When the Player picks up this AttackItem it can choose to attack with it. This design choice adheres to the Don't Repeat Yourself (DRY) principle, as creating separate classes for each special scrap would result in redundancy. Since other instances of WeaponItem, such as MetalPipe, can be picked up and utilized as weapons, consolidating these special scraps into a single class simplifies the design and encourages code reuse.

Regarding the consumption of Fruit, I implemented an interface called Consumable for all consumable items to implement. This interface is associated with ConsumeAction, which stores a Consumable class as a field. This approach adheres to the Open-Closed Principle, as it remains open for all consumable items to extend the interface while restricting modifications in the ConsumeAction class.

I implemented the consume method in the Fruit class since it implements the Consumable interface. Following that, I added the ConsumeAction to the allowableActions of Fruit. This strategy also aligns with the Single Responsibility Principle, as actions available to players are managed by the Action class. Consequently, during each actor's turn, the world examines the items in the inventory and incorporates the actions specified in the allowableActions of the item into the actor's available actions.