**Design Rationale**

Eggs and Corpses were initially going to be of type Dinosaur so that they could be easily made from dinosaur but after further discussion we realised they needed to be sold in the shop and left on the ground like items, therefore being of type item would be smarter.

Grass was originally going to be of type Dirt but again after more thought we realised that Ground included all the necessary methods for it and all we needed to do was change the Location setGround to Grass.  
All the classes in our design have been designed so that where possibly inheritance is used so that multiple classes can use the same code without repetitions. This follows good Object Oriented design practice.

|  |  |
| --- | --- |
| Actor | Used to implement the Dinosaurs, as well as the Player |
| Player | Needs to add an inventory to the currently implemented Player, also needs an int for money. |
| Dinosaur | Abstract class used to create the majority of each dinosaur. Variables needed include hunger, turn counter and a method to check the ground value at the actors location. By using the method in location for getGround. |
| Protoceratops | Extended from Dinosaur. If hungerCurrent > 20 then spawn an egg.  Protoceratops will seek out the nearest |
| BabyProtoceratops | Will also use the SeekBehaviour. After 30ish days delete and turn into Adult Protoceratops |
| Velociraptor | Velociraptors also have access to the SeekBehaviour class and a higher hungerMax. |
| Item | Item is already implemented but we will use it to create corpses, eggs, and DinosaurTags |
| Egg | Egg will be of type item, it will hold a variable for the type of egg it is, most likely an enum, as well as an integer for price and foodValue. foodValue will be used when a Velociraptor eats it so that the Velociraptor can gain a specified amount of hunger from it |
| DinosaurTag | A dinosaur tag is an item. When a player tries to use it will check the nearby squares for a dinosaur, then check if the dinosaur meets the requirements to be tagged. If so then the dinosaur will be sold/deleted and the player will gain $1000 |
| Corpse | A corpse will be created when a Protoceratops starves. It will be extended from item so that a velociraptor can eat it and get the health value from it |
| Food | Food is also extended from item. It will be an abstract class for Herbivore and Carnivore food. |
| HerbivoreFood | Herbivore food extends from food, therefore it is an item. This means it can be placed on the map and picked up by Protoceratops |
| CarnivoreFood | Similarly to HerbivoreFood but for the Velociraptors |
| Ground | This class is already created. |
| Dirt | Dirt is a subclass of ground. Each turn there needs to be a random chance to turn into Grass. For this reason Dirt can instantiate grass by calling setGround at the location. |
| Grass | Grass is also a subclass of Ground. It holds an integer foodValue so it can tell the Protoceratops to eat it. |
| Tree | Trees have a method that have a random chance to turn an adjacent location to a Tree, using getGround and setGround |
| Shop | We decided that instead of the shop holding an array list of objects it could simply hold Variables of sellPrice and buyPrice for each item. Because the shop can never get sold out, all the shop needs to know is sell and buy price. It also needs to hold methods to sell or buy from the player. |
| Behaviour | Already implemented |
| SeekBehaviour | Seek behaviour would follow a similar pattern to follow behaviour and would be given the closest food source using pre existing functions in followBehaviour. |