FIT2099 Assignment 2

Updated Design Rationale

By: Chong Chiu Gin (28842022) & Chin Wen Yuan (29975239)

1. **Trees and grass grow**

   - **Grass**
     Grass class extends Ground and has food points as attribute. The food points is used to increase the food level of Protoceratops dinosaurs when Grass is eaten by it.

   - **Dirt**
     Instead of adding a new method in Dirt class, tick method is override to enable Dirt to have a chance of 0.5% to grow into Grass at every turn.

   - **Tree**
     Just like Grass class, Tree class will have food points attribute for Protoceratops to increase food level after eating Tree. We override the tick method such that at each turn, a Dirt next to a Tree can have a 0.5% chance of growing into Tree. To control the growth of tress, we will implement that only Dirt can grow into Tree. This is to ensure that other classes that extends Ground like Grass, Wall and Floor that is neighbour to Tree won't be replaced into a Tree.

2. **Dinosaurs hunger and feeding**

   - **Dinosaurs**
     Dinosaurs is an abstract class because there will be no Dinosaurs objects instantiated from this class. Dinosaurs class contains attributes that stores food level of dinosaurs, age, the status of dinosaurs, the selling value of dinosaurs as well as behaviour of the dinosaur. These attributes are declared protected and can be accessed by the two classes that extends the Dinosaurs class which is Protoceratops and Velociraptors. The playTurn method is override and implemented in Dinosaurs class instead of individually implemented in both Protoceratops and Velociraptors class because we notice that the playTurn functionality is the same for both type of dinosaurs. This obeys the principle of "Do Not Repeat Yourself". In this playTurn method, an Action that should be taken by dinosaurs during that turn will be returned and to decide on this Action, we first look at the food level of the dinosaurs. If the food level is 0, a DieAction will be returned to indicate the death of the dinosaur. If food level is below a certain level, behaviour of dinosaur will change to HungryBehaviour. If food level is above a certain level, it will allow the dinosaurs a chance to return BreedAction. Otherwise, it will wander around.

- **Protoceratops**
  Each type of dinosaurs has their distinct attributes value which means Protoceratops and Velociraptors have different maximum food level, hunger level, selling value, display character on GameMap and so on. The Protoceraptors have two different constuctors, one for adult Protoceratops instantiated in Application class and one for baby Protoceratops newly hatched from Protoceratops eggs.
  Protoceratops class has override getAllowableActions method that allows Velociraptos to perform AttackAction on it and allows Player to perform TagAction if Player has dinosaur tag in their inventory and FeedAction if Player has herbivore food in their inventory on it.

- **Velociraptors**
  Just like the Protoceratops class, Velociraptors have their distinct attribute values too which is primarily why these two classes are created to differentiate the functionality of both type of dinosaurs. Velociraptors also has two differenct constructor, one for adult Velociraptors (this is only used for testing the functionality of a Velociraptor dinosaur in the Application class because Velociraptors do no appear in the beginning of the game and and only comes from buying its egg from the Shop) and one for baby Velociraptors hatched from eggs.
  Velociraptors class has override getAllowableActions method that allows Player to Player to perform TagAction if Player has dinosaur tag in their inventory and FeedAction if Player has herbivore food in their inventory on it.
  Velociraptors also has override getIntrinsicWeapon that returns an Intrinsic weapon that does 10 hit points damage to it's target.

- **HungryBehaviour**
  If food level gets too low, the dinosaurs will have HungryBehaviour. HungryBehaviour is a class that implements Behaviour class and it is used to decide on what is the next action the dinosaurs should take when it is hungry.
  Protoceratops with HungryBehaviour will check if current location has Grass, Tree or herbivore food first. If there is, EatAction will be returned. If no edible source on current location, it will find the closest edible source to it and return MoveActorAction towards the edible source.
  Velociraptors with HungryBehaviour will check if current location has eggs, corpse or carnivore food. If there is, EatAction is returned. If there's none, Velociraptors will check all location of 8 exits if there is a Protoceratops there. An AttackAction on the Protoceratops will be returned if there is Protoceratops in any of the location of all 8 exits of the location Velociraptor is on. If none are found, then it will find the closes edible food source or the closest Protoceraptor and return a MoveActorAction towards it.

- **EatAction**
  EatAction is an action that can be performed by both type of dinosaurs Protoceratops and Velociraptors. This class has three attributes which are item, ground and the name of the food. This class extends Action class and has two constructors because dinosaurs like Protoceratops can eat both objects that extends Item class and objects that extends Ground class. If the object being eaten is a Ground type, the Ground type of the location of the actor will be set to Dirt again to replace the Tree or Grass being eaten. The food level of the actor performing the EatAction will increase.
  If the object being eaten is an Item type, the item would be removed from the location and the food level of the actor will increase.

- **AttackAction**
  AttackAction can only be performed by Velociraptors onto target which are Protoceratops. Velociraptors will hurt Protoceratops which decrease Protoceratops' hit points. No changes are made to this class as it remains the same since base code is given.

- **FeedAction**
  Both getAllowableAction methods of Protoceratops and Velociraptors class will have FeedAction that extends the Action class when Player is standing next to the dinosaurs. FeedAction will have one target dinosaur. These two class has a dependency relation to dinosaur classes to ensure the principle of "Classes are responsible for their own properties" is met. FeedAction allows Player to hand feed herbivore food to Protoceratops or carnivore food to Velociraptors.
  The action can only be performed if Player has the food in their inventory and the target dinosaur is hungry. When executed, the food item is removed from Player's inventory and the food level of the target increases.

- **DieAction**
  DieAction extends Action class and is called when a dinosaur's food level reaches 0. When executed, it removes the dinosaur actor that died from the GameMap and adds a new corpse onto the GameMap.

3. Dinosaurs breeding

- **BreedAction**
  When the dinosaurs are sufficiently well-fed which means they have a sustaining food level, they are given a chance to breed by laying an egg. Random will be imported and called to decide whether the dinosaurs will breed as their next action. BreedAction that extends Action class will execute if it was decided that the dinosaurs will breed. The BreedAction class will create an Egg item in the execute() method. This Egg can be picked up by Player or left on the Ground.

- **Egg**
  Egg class extends PortableDinoItem and is designed because dinosaur eggs have extra functionality like having an age attribute to indicate the age of the egg. Egg class will override both tick methods in the Item class, one tick method is for eggs on the ground whereas the other tick method is for eggs in Player's inventory. A method called hatch() is called after 15 turns creates a new instance of dinosaur and removes the Egg item from the GameMap or from the inventory. These dinosaurs will be baby dinosaurs for 30 turns before turning into adult. These baby dinosaurs cannot be perform BreedAction.

4. Buying and Selling items in Shop

- **PortableDinoItem**
  Aside from Egg class, items like corpse, carnivore food, herbivore food and dinosaur tag are made and designed as instances of PortableDinoItem rather than a concrete class. This is to avoid redundancy classes that don't have particular functions or operations to implement since they are mostly being acted upon. This also follows the principle of reduce dependency because this design ensures there are lesser classes extending to PortableDinoItem. PortableDinoItem has attributes like their buy value, sell value, whether the item is edible, food points as well as the type dinosaurs the item is catered for.

- **DinoType**
  This is an enum class that determined the type of the item belongs to Velociraptors or Protoceratops because there are Protoceratops eggs and Velociraptor eggs as well as Protoceratop corpse and Velocirapor corpse.

- **BuyAction**
  This class that extends Action class has Item attribute. When executed, it adds the item into Player's inventory and decrease Player's money.

- **SellAction**
  This class that extends Action class has Item attribute. When executed, it removes the item from Player's inventory and increase Player's money.

- **Shop**
  Shop class is created and extends Ground class because Shop has to be displayed on GameMap as 'S' character in the building on the top left. The allowableAction method needs to be override in the Shop class so it can allow Player to perform BuyAction and SellAction whenever Player reaches the building and is standing next to 'S' character of the Shop on GameMap. Shop has a dependency relation with both BuyAction and SellAction class instead of Player to fullfill the principle of classes should be responsible for their own properties.

5. Selling Dinosaurs

- **TagAction**
  Both getAllowableAction methods of Protoceratops and Velociraptors class will have TagAction that extends the Action class when Player is standing next to the dinosaurs. TagAction will have one target dinosaur. These two class has a dependency relation to Dinosaurs class to ensure the principle of "Classes are responsible for their own properties" is met. TagAction allows player to tag the dinosaurs that is to be sold.
  The action can only be performed if Player has dinosaur tag in their inventory and the target dinosaur is an adult dinosaur, is well fed as well as has more than 50 hit points.
  When executed, the dinosaur tag is removed from inventory and the target dinosaur is removed from GameMap. The Player earns money.

- **Player**
  Money attribute is added into Player's class because Player needs money in order to perform buy and sell action with Shop. Additional methods are added to this class to implement the process of earning and spending money as well as retrieving items from inventory.