Exits are not intuitive:
Found it difficult to understand how to use exits to search for items/actors and what exits even were. We had to call exits on the current exit's location to search and recursively go from there. Meaning we had to code searching functionality outside the exit class .
Solution: have a method in the exit class that takes in an integer representing the radius we want to search around an actor and return any locations that have what the actor is looking for.

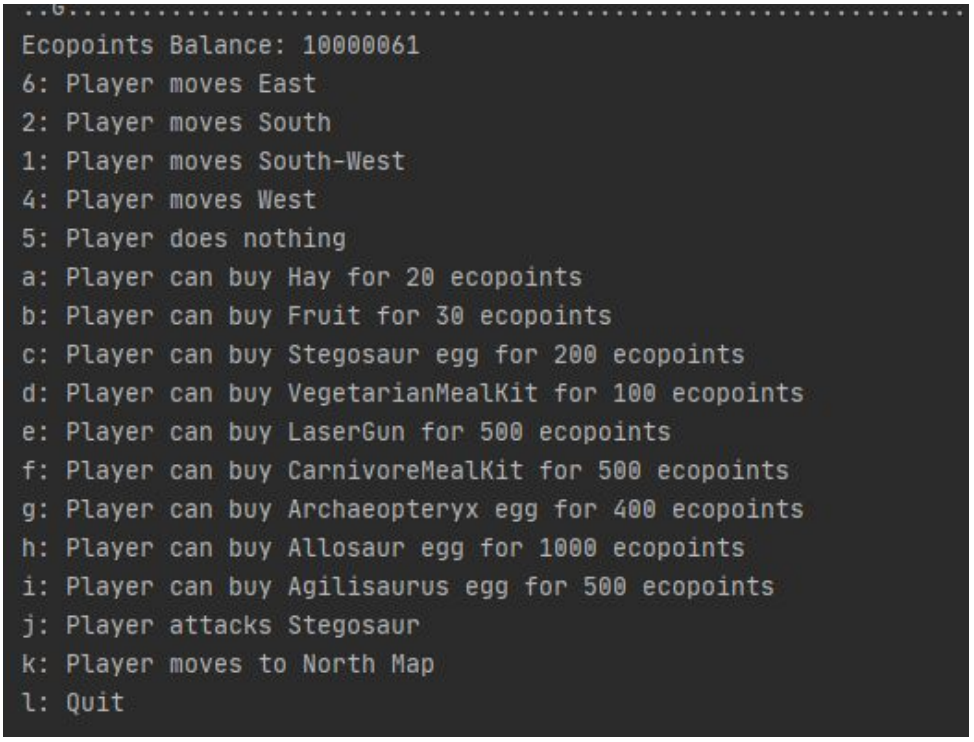
Location:
Any class that has to do with existing at a location is handled nicely by the location class.Having a ground object stored within a location allows the us to easily check whether an actor can enter that location. Location also contains a list of items which lets items such as food be stored there, making it easy to place dinosaur corpses for example when a dinosaur dies.

Age of the map:
Something that I really appreciate and its simplicity is how everything is processed with age (turn). Although there is dependency of classes such as with tick methods, they are really good at dealing with its own class, instead of affecting other classes, so it just deals with itself. Especially tick methods and allowableactions which ensure that they only deal with their own classes.

CAPABILITY is good:
Capabilities allow programmers to check if one actor can interact with another, Capabilities allow us to check what foods dinosaurs can eat, what terrains they can travel.
Similar checking can be done with strings but poses the risk of errors from misspelling, while enums will throw an error straight away if a constant does not exist.
Using enums, which is good because if a enum is misspelt, it will produce an error unlike using strings

Menu:
The overall Menu functionality has been very intuitive but it does have limitations in terms of how much can be displayed through 0-9 numbers and a-z letters. Which can be an issue if we have a lot of decisions for the user to decide. But that most likely won't be the case since. One issue I have with the current menu there isn't an ability to create a sub-menu. I think having sub

menus provided by the engine would not only make it more visually appealing to the user but it will also makes it less overwhelming. For instance



There should be a way to split it so if they want to purchase an item they click on a purchase item and then it shows a breakdown of all purchasable items. One way to do this is once “open items to buy “ option is selected we create another action in which makes a new menu that is shown to the user in terms of potentially items to buy.

DISPLAY:
The current display only is able to read characters for like menu options, but one of our new features we had to implement for Assignment 3 was that if the user decided to play the challenge mode then they would have to select the amount of ecopoints they needed to achieve. The current engine does not support this. As such we needed to have a displayModified to readInt method for the future engine code there should be a method in Display class that does this for us rather than extending the class to make it happen.