Project 6: Happy Go Farming

Team Members:
Tinie Doan
Wyett Considine

State Summary

- Wyett: Item Hierarchy, Factory, and Unit Tests
    - A significant part of the game is resource management. I created and tested the item system. There are four types of items and all have different relationships.
        - Seeds: Seeds can be gained from the market or by harvesting plants. After a certain number of days planted, they produce a Plant.
        - Plant: Plants can be grown and harvested for Yields after a certain number of days while planted.
        - Produce: Produce can be sold at the market for money, or traded for resources.
        - Yields: The Yield system is a Strategy pattern connecting the Plant's harvest function to the appropriate seeds and produce.
        - Tools: Tools can enhance player actions such as watering, harvesting or fundraising.
        - The game items so far include seeds, plant and produce forms of the following: Carrots, Potatoes, Tomatoes, Berries, Onions, Lavender, and Lettuce.
    - Patterns:
        - Factory Pattern: Many of the different types of Items transform into other items. For example, seeds grow into plants, and plants can be harvested into yields. The Factory pattern is very important to consolidate and encapsulate the constructor logic. It also facilitates the storing of information and generalizes the creation of objects, which helps the entire codebase. To implement this, i used a
        - Strategy Pattern: The Yield system of harvesting plants needs to be very flexible in accommodating the different types of items plants can produce. This system capitalizes on delegating yield behavior to Seed and Produce objects instead of storing all of the logic in the Plant objects, or excessively subclassing.

- Tinie: actionType hierarchy, Friend class, Strategy, and Unit Tests
    - I implemented a Friend class, which are the “characters” that the player will interact with.
    - GardenPlot class, which will be a plot of plants that the player will need to tend to on top of tending to the plants themselves.
    - I also implemented the actions that the player can do> this includes:
        - actionType: This is a Strategy pattern that connect the Friend’s actions to a specific type
        - HarvestingPlants: The player will harvest the plant of their choosing
        - WaterPlants: The Player will water plant of their choosing
        - PlantPlants: The player will plant the plant of their choosing
        - PullWeeds: The player will pull the weeds of their choosing
        - Fundraise: The player will try to raise funds
    - Patterns:
        - Strategy Pattern: The actionType class allows for us to be able to send one initiated actionType to either the character or the player so that we don’t need to have large amounts of code that try to decipher which action needs to be performed based on something like a string.
    - Changes/challenges:
        - Performing the actions of actionType will no longer be void, they will need to return a type so that the game can store the changes made to the items after being watered, harvested, planted, etc.

Plan For Next Iteration:

- Approximately half of the work still remains.
- Now that we have completed most of the underlying item structure and classes, the next step is to finish the implementations of:
    - the Event system
    - development the cohesion of the simulation
    - Player + Game system
    - Garden Plot
    - still up for debate: decide whether we will be using a command pattern for user interaction
    - store highscores in a csv file

Class Diagram

- Note: the implemented classes are the colored ones