Gabriel Barratt
Darian Payan

Work done

Gabriel Barratt: Spent time with Darian splitting up the initial work. I started with the graphs and project design deliverables for project 5 while he set up the different Java files we were going to use. We both spent time working on each java file and attempting to implement the required patterns. I focused on creating the factory patterns and we both worked on decorators as well as the strategy pattern. All the work that has been contributed to the github has been done in a call with each other and we live shared our Visual Studio Code in order to work on the same files at the same time.

Darian Payan: Initialized files and set up some skeleton code for our project. From there I spent some time developing the User Interface file for where our game would interact with the user. I also worked on making our Bee variants. With Gabe, we worked on assigning attributes to our Bees and how they will be used in productions and expansions of the hive.

Changes and Encountered Issues

- We have added extra decorators in order to keep things more organized and fluid throughout our project
    - Combat type
    - Job type
    - Upgrade type
- We have a move set but we are still deciding if we really need to use it or not
- How bee's attack and get hp have been discussed and is still changing
    - Still figuring out how to balance things
    - Decided to have bees roll for resources and damage
- Since it's a resource game, we are still changing and fixing how different mechanics work overall
    - Upgrading buildings/possibly upgrading bee's
    - How to regen hp or the possible addition of a hospital
    - The cost of items and resources
        - Dealing with larger or smaller values for currency
            - Bees bring in a large amount of nectar, 100 nectar = 1 honey
            - Bees bring in a small amount of nectar, 10 nectar = 1 honey
            - (same outcome just balancing)
- Bee's now have more specific jobs and abilities instead of one purpose
- We are no longer using Unity and have committed to using the terminal

Patterns

- Command - We use Command Pattern in our UserCommands.java. User decisions from UserInterface will be passed into here, where the actual decisions of the game are called and operated.
- Factory - This design pattern is used in our creation of bees. For our game, when a nursery is called to produce more bees, factory allows the initialization of many bees.
- Decorator - In our game we will have different types of bees with varying capabilities. We use a decorator as a way to extend the Bee type with each ‘class’ of bee.
- Strategy - Depending on the class of the bee, they will have different values that are specific for their role in the hive. All bees however will still provide *some* value in regards to combat during events and collection of nectar. Strategy allows us here to be able to set specific values for their effectiveness in all roles based on their own implementations.

Plan for Next Iteration

For our game, we are looking to next implement a good portion of the engine in order to start setting up the flow of the game and testing out some of our current work. We would have to set up our facilities (nursery, hospital,supply factories) that take in the values from production and do the proper conversions. From here we expect to be able to finish up the engine and complete the expected functionality to the game, with random events being a main thing.

A stretch goal we have that we might be able to add depending on how fast progress goes is to add a save game feature, where we would be able to export all the necessary information about the game to a csv. From the csv it would also become possible to load a game when starting up the program again.

Class Diagram