Project Description:

BelowGround Burgeria

As the owner of the brand new burger chain, the player will cook burgers and fries to customers’ order to earn money, make upgrades to their restaurant, and ultimately keep the customers happy.

Similar Projects

This project is inspired by the Papa’s game series, which are an assortment of restaurant style games that are focused on time management and accuracy. The most similar game will be Papa’s burgeria, although my version will be far simpler. Firstly, my version will only have 3 cookable objects: patties, fries, and buns. There will also only be 3 toppings: lettuce, tomato, and cheese. In Papa’s, each customer creates their own custom order, but because of the limited ingredients in my version there will be preset burger types. For now, I’ve drawn a normal BelowGround burger and the deluxe version, but I can create new versions like the veggie burger or meat lover burger. My version will be similar to Papas in that each cookable item has an optimal “cook time”, but my version does not have a quality assessment system. Instead, if the player pulls it off too early they cannot use it, and if you wait too long it will burn and it will need to be thrown away as well.

Structural Plan

I will create an ingredient class with three subclasses, buns, fries, and patties. Each ingredient will have a cook status, a ‘being held’ status, and a location status. While the ingredient is cooking, the cook status will change at certain intervals, and the visual will also change.

There will also be a class ‘station’ where the ingredients will be cooked. The ingredient drawn in the station will change depending on the cook status of the ingredient.

I will do most of my drawings myself and keep them in the same folder as the other files to have easy access to them. If the drawings prove to be too time consuming, I can use cmu graphics to represent my ingredients.

Each of my classes will have their own functions, and since no ingredient has any special capabilities there should not need to be any subclass functions.

There will also be a customer class where the orders are (limitedly) randomly generated and a patience timer is assigned

Algorithmic Plan

Tricky Parts:

* Adding each of the ingredients to each status
  + Each station will consist of multiple slots, which are interacted with individually
  + When a slot is hovered over, check if ingredient type is correct and if not cooked
  + When cooking, add cooking value to the ingredient
* Creating a cook timer visual for each ingredient
  + A tall rectangle beside the ingredient will display the cooking progress. The bar will fill based on timeCooked/cookDuration \* barHeight, and if the timeCooked exceeds cookDuration then the ingredient will burn.
* Visualizing the parts of the burger that are already placed on a plate
  + Each plate will have an array of items, from bottom to top. Buns must be the bottom item, but the order of the ingredients and toppings do not matter. Each plate will keep track of the sum of the thickness of the ingredients so that the next ingredient can be placed on top.
  + In order to check whether a burger is valid, compare the sorted ingredients in the plate to the sorted correct burger.
* Clearing the customers after they receive their orders
  + Each customer will request a number of orders, and each time they receive a burger their request for that type of burger will go down by 1. To do this, I will use a dictionary where the key (burger type) points to the number requested. When all keys are 0, the customer pays and leaves.
* Increasing the difficulty in an interesting way
  + Introducing catering customers, who order a lot. This will force the player to decide whether to take up the challenge of the catering, losing other customers, in an attempt to make more money off the cater.
* Balancing the economy
  + This will come through playtesting and time, but will be streamlined by not making any magic numbers and having an easy way to control the variables

Timeline Plan

By Nov 23 I want to be able to comfortably import the drawings I create and use them in the python environment. They should have transparent backgrounds so that they don’t interfere with visuals behind them.

By Nov 24, I want to have good progress on the art style for the ingredients, as well as an ingredient - station pairing working. The basic framework for the classes should be complete. I want to have my menu designed here so that future references to burger types are solidified.

TP1: By Nov 27, I want to have a basic cooking functionality, including the art for the ingredients and stations. I hope to be able to stack ingredients on a plate and create a trash can for if things need to be tossed out. At this point, a rudimentary customer system is good for testing.

TP2: By Dec 1, I want to polish the rest of the game’s functionality, including the cook timer bar and customer sliding in animation. Also, the money system should be implemented, alongside upgrades at the end of each day. The upgrades will improve cook times and restaurant environment leading to more tips on average.

TP3: All bugs have been fixed and game balance has been tested. Remainder of deliverables are completed.

Version Control Plan:

I will mainly be using Github as my storage for code. After each work session, I will push the new code version to Github. This way, if any bugs arrive during the session that I cannot fix, I can revert to an older version.

(Photo attached in zip file)

No other libraries will be used.

**Updated Design**

I am going to transition my game away from a static, first person cooking game into a top down view of a kitchen where the player has to pilot the chef to complete orders. The chef must run around cutting ingredients, cooking them, and sending them to the window.

Similar games

This game will be heavily inspired by the game Overcooked. I also found a good simplified version on Scratch at <https://scratch.mit.edu/projects/651462531/>

Structural Plan

In essence, the game happens on a grid. However, the player is not locked to the grid, meaning they can move in between tiles to create smoother movement. There will be a floating icon above the chef to display what is being held, which will eliminate the need to draw the chef holding each item. The rest of the structure is similar to before, except now I will create a counter class where items can be placed and is a ‘wall’ to the chef in terms of the grid.

Anticipated Tricky parts:

* I want to add a path finding algorithm that leads the chef to the desired work station.
  + Using recursion and a BST searching algorithm to light up a path