Name: Erin Brooks

Class: Object Oriented Development

Assignment: Final Project

Professor: Michael J Nowak

Date: 3/2/2020

For this project I created an application that can be used to run a bakery including turning on the necessary appliances. The application prints out the menu and takes an input of what number item on the menu a customer wants to order. Then the application prints out the steps for creating the item including turning on and off the necessary appliances. If a customer wishes to order a custom cake the system helps facilitate the creation of the cake before printing out its instructions. To accomplish this task, I used four design patters: decorator, factory, command, and iterator.

The decorator pattern was used for the creation of the cakes, there description and cost. First the cake flavor would be chosen and up to two decorations can be wrapped around the cake. The code can allow for more than two but to simplify this code I limited it to two. The decorator pattern is located in the following classes: BirthdayDecorations, Cake, ChocolateButtercream, ChocolateCake, Decoration, FootballDecorations, VanillaButtercream, YellowCake. The decorator pattern is used in CounterWorker class and is used to create the cakes.

The factory patter was used to put together the instructions for creating the robot. The different decorations and cakes used by the decorator pattern. These call the ingredient factory to print the instructions for that specific part. The files specifically dedicated to the factory pattern are: Batter, BirthdayPack, ChocolateBatter, ChocolateIcing, Decorations, FootballPackage, Icing, IngrediantFactory, VanillaIcing, and YellowCake.

The command pattern was used to create and activate the devices that would be used in the cake. This pattern is used by the iterator pattern when creating the instructions for baking the cake. It also turns the necessary devices on and off. This pattern can be found in the mixer and oven classes.

The iterator pattern is used to create and print the menu for the bakery as well as to take orders and call the decoration class. It can be found in the following classes: BakeryIterator, BakeryMenu, Menu, MenuItem, MyIterator, and CounterWorker.

The first difficulty I had was combining the decorator pattern and the factory pattern. I had used both to similar purposes in my previous assignments and had difficulty eliminating redundancies. In the end I decided the decorator pattern would handle the description and the cost of the cakes then call on the factory pattern to create the instructions on the cake’s creation. After that it was a matter of integrating the other two patterns. I quickly settled on having the iterator call the mixer and oven since that would be the easiest way to turn the ‘machines’ off and on at the correct times. The iterator pattern gave me some difficulty at first. The menu and iterator itself worked just fine but I wanted the customer to be able to order custom cakes. Common sense kicked in when I realized the person at the counter in a bakery would handle that and that the counter work could serve the same function here. So, I made it so that the counterworker can take in a customer’s order and call the decorator to create it.

This experience has taught me that should I ever need to combine design patterns I must decided what patern will be called by what other pattern and to also choose carefully to avoid redundencies. I believe I could have completely eliminated the factory pattern and made up for the loss with additions to the decorator pattern.