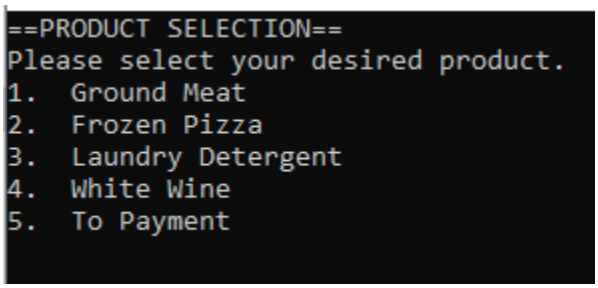
RetailStoreAutomatedCashier

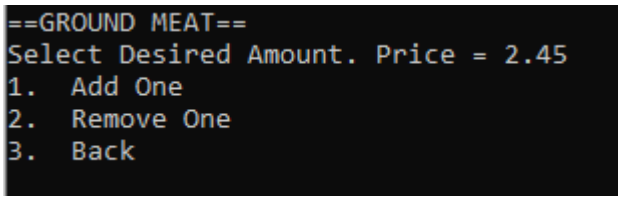
The program presented for this assignment is an automated retail store cashier. This software is a command-line interface designed to be used for hours during which staff may not be present at a retail store. Customers may use the interface to add products from a list to their cart. Afterwards, the customer may pay in cash, and the machine will return change. Finally, a mobile robot inside the building can collect the requested products, and return them to be dispensed for the customer.

Instructions of Operation

In order to run the program, please open the main.py file. Upon doing so, the following screen is presented:

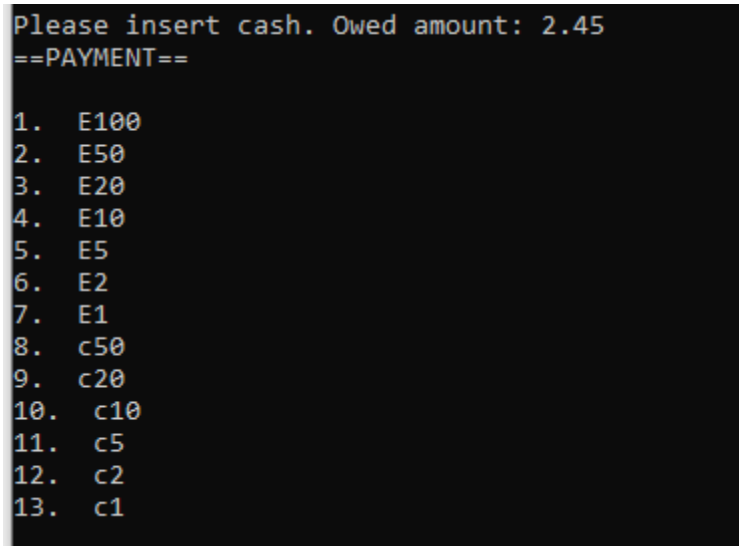


All interfacing uses the user's input of a number corresponding to the menu item in the list. From this screen, products may be chosen. Once a product is selected, a seperate menu for said product will start:

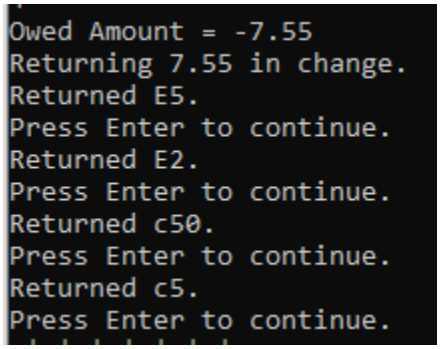


From here, the user may add as many of the item as they want, by incrementing or decrementing the quantity of the product one at a time. The total quantity will be shown upon doing so. Upon selecting "back", the user may select other items to add to their cart, until they have the amount desired of each product.

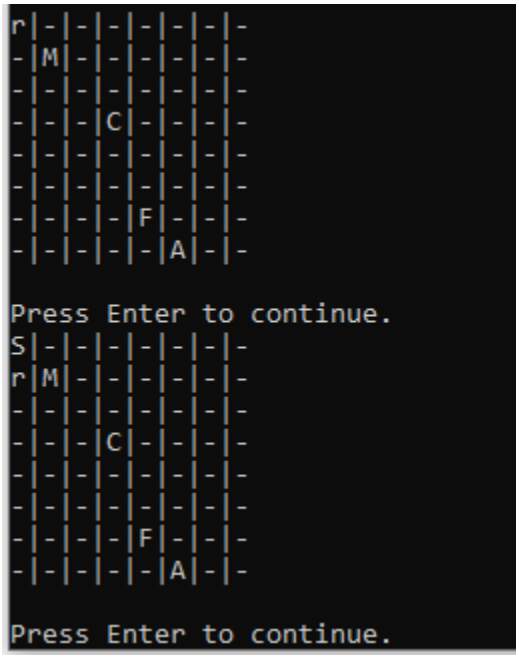
After selecting the "To Payment" option, the products' total value is calculated and presented to the user, who may now select which units of cash to insert into the machine using the interface:



The user must insert cash until the amount owed is 0 euros or less, upon which the machine will return change back in currency (starting from highest to lowest units). Press Enter after each returned coin or bill to advance:



Finally, the user is presented with the randomized grid layout of the store. Here they can input Enter to see the movements of the robot step by step as it collects requested items from each relevant aisle and returns to the starting location:



Please consult the following legend for the denomination of each character:

1. **S** - Starting location (The robot starts here and must return to here in order to deliver the collected items)
2. **r** - Robot
3. **M** - Meat Aisle (Product - Ground Meat)
4. **F** - Freezer Aisle (Product - Frozen Pizza)
5. **C** - Cleaning Aisle (Product - Laundry Detergent)
6. **A** - Alcoholic Beverages Aisle (Product - White Wine)

The robot will only move one square at a time in any cardinal direction, and will take the shortest possible path.

Once it collects items from each relevant aisle and it returns, the number of steps it took is displayed. The products are delivered to the customer and the program closes.

tests.py contains testing data.

Implementation Specifics

Implementation for this project did not vary much from the given design document. The main use of OOP principles is within the menu system. This system is built as an experimental Entity Component System, where each item on a menu can be given a component, via passing a function reference and the arguments required.

More class-based objects were created than expected. Cash units, menus and menu items, product types and aisles all are objects instantiated from classes.

Outside of the menu system, the implementation of the final phase, during which the robot finds the optimal path to collect the requested products, was the most involved. This implementation did not utilize OOP principles, outside of accessing the existing Aisle and Product classes, as it would not benefit from instanting class objects.

The pathfinding algorithm was a simple self-made design, making use of permutations to calculate the shortest path from all possible combinations.

Limitations

The Entity Component Menu System proposed in the design document was used in the making of this program. It was found to be less efficient than expected, as passing functions by reference with less control of their execution caused the need for multiple workarounds, and led to less readable and iterative code. It is unclear if this specific approach of treating functions as components could have been improved on to be more beneficial.