**Shopping Center Project**

Kyle Kaminski & Kiima Ballantyne

* The most frequent operation that is expected to be performed during the entire execution of the program is addition. When the store is first opening, we add customers to the shopping array list. When they add items to their carts, we add those items to an arraylist for their carts. When they want to check out, we add them to the DEQueues of the register lines.
* Our choices for these operations optimize the frequent operation of addition because it is a lot easier to create and manage elements to put into an arraylist than to create a lot of variables.
* The ADTs we plan to use include Arraylists and DEQueues. The reason we choose these are because they are a lot easier to manage. By using arraylists, we can freely add an unorganized amount of items to arraylists for shoppers in the store and the amount of items in their carts. This way, we can add and remove items as we please. By using DEQueues for our checkout lines, we can manage the specific FIFO order for the amount of customers on a line.
* The instances of ADT(s) we plan to use are the creation of an arraylist for a new customer’s shopping cart, because this is the only type of ADT that will be instantiated during runtime of the program.

**There will be 5 classes that will be used to efficiently and easily run this project**

1. **Driver class**
2. **Shopping Center class**
3. **Customer class**
4. **Items class**
5. **Checkout Lines class**

**Driver Class**

* This class will have the 10 menu options that are necessary to run the program.
* Before printing the menu to the user, they are prompted to enter all of the available items that the store carries, and the respective item quantity.
  + When the user is done entering items, they input the word “Done”. This will end the loop and redirect them to the menu system.
* Option 0 will close the shopping center (terminate the program). Everything will reinitialize when the program is rerun.
* Option 1 will add a customer to thearraylist of customer’s shopping in the shopping center and accepts their name. All names will be unique.
* Option 2 will ask the user which customer took an item, and which specific item they took and decreases the stock for that item by 1.
* Option 3 will ask the user which customer wants to remove an item, and which specific item they want to remove.
* Option 4 takes the least recently added customer (oldest) and will automatically push the customer into the most efficient checkout Line.
* Option 5 will look throughout ALL of the three checkout lines, and will find the customer at the front of the line, with the MOST amount of time spent in the store. This customer will be the first one to be checked out (whether they were on an express or regular checkout line does not matter; for example, if Regular Line 1’s first customer has been shopping for 20 minutes, Regular Line 2’s for 10 minutes, and Express Line’s for 15, then Regular Line 1’s customer will be first).
  + The user will then be prompted to input whether the customer wants to leave the shopping center, or if they want to go back to shopping.
  + If the customer wants to leave the store, then they are removed from their respective checkout line array list.
  + If the customer wants to go back to shopping, then they are removed from their respective checkout line array list, and added back to the array list of customers shopping in the store.
    - The customer’s amount of items in their shopping cart is reset
    - The customer’s time spent in the store is reset
* Option 6 will print the array lists for:
  + The customers who are currently still shopping
  + The items that are in each customer’s respective shopping cart
  + The amount of time each individual customer has spent in the store
* Option 7 will print the array lists for:
  + The amount of customers currently waiting in Regular checkout line 1
  + The amount of customers currently waiting in Regular checkout line 2
  + The amount of customers currently waiting in Express checkout line
* Option 8 will print the array list of each individual items that need to be reshelved (items that were added from option 3).
  + The specific item will be at “restocking level” when the item is less than or equal to the restock item warning (we will declare this as an inventory of 3 for that item). If an item is at “restocking level”, then the user is given a message that they will need to reorder more of that item (option 9).
  + When the item has an amount of restock item amount (will be declared as 0, meaning no more of the item is in stock or inventory), the customer cannot take any more of that item.
* Option 9 will prompt the user to to input the specific item they want to reorder, and the amount of that item they want.
  + When reordering successfully, then the size of that item’s arraylist is updated with the new size (original stock + reordered amount).
  + When reordering unsuccessfully (an item that we do not carry was inputted), the item is denied a reorder.
  + When too much of an item we *do* carry is reordered, the user is notified that the reorder will overstock that item.
    - They will be prompted if they want to still reorder that item.
      * If the user does want to reorder that item, then the size of that item’s arraylist is updated with the new size (original stock + reordered amount)
      * If the user does not want to reorder that amount of the item, then they will be denied a reorder of that item.

**Shopping Center Class**

* This class will be where we initialize the array lists of customers in the store
* This class will store the array list of customers shopping in the store
* This class will have a method to print all of the customer’s still shopping in the store

**Customer Class**

* If a name that is entered is the same as another customer’s, then the user is denied the option to use that exact name, and is prompted to enter a different name.
* Upon successful customer creation,
  + The customer’s name is added to the array list of customers currently shopping in the Shopping Center
  + A new instance of an arraylist of the customer’s shopping cart will be created during runtime (no size limit for the customers’ shopping cart)
  + A new integer variable for the amount of time the customer has been shopping will also be initialized
* Any items the customer takes will be added to their specific cart’s arraylist
  + shopping cart
    - It will also add that item to the array list of items that need to be returned to the shelves. (will NOT automatically be added back to the shelf)
  + It will also increase the minutes for each specific customer in the store by 1.It will also decrease the stock of that specific item by 1. (Handled in the Items class)
  + It will also increase the minutes for each specific customer in the store by 1.
* We will have a method that can print a specific customer’s shopping cart of items
  + This method will also print the amount of time they have been shopping.
* We can add or remove any item that is available in the store from the customer’s

**Item Class**

* This class will be where the items of the shopping center will be initialized
* This class will remove items from an items inventory amount to add to a customer’s shopping cart
* From the menu, the user can also reorder more of a specific item for the shopping center’s inventory. That process will be handled here.
* This class will handle the creation of more items if need be.
* This class will handle the addition of an item’s inventory when more stock is ordered.

**Checkout Lanes Class**

* This class will be where the initialization of the 3 checkout lanes will be.
* There will be a fixed amount of checkout lanes every time the program is executed (2 regular and 1 express)
* If the customer has less than or equal to 4 items, they will be removed from the array list of customers shopping, and added to the DEQ list of the express checkout line.
  + If the express line is twice as long as regular line 1 and regular line 2, then the customer will be added to the smallest regular line.
* If the customer has more than 4 items, they will be removed from the array list of customers shopping, and added to the DEQ list of the smallest regular checkout line, whether that is regular line 1 or regular line 2.
* If the customer with the most amount of shopping time is done shopping **AND** their cart is empty, they will be prompted to leave or return shopping.
  + If the customer wants to leave the store, then they are removed from the arraylist of customers shopping in the store (they will not be added to the arraylists of checkout lines)
    - If the customer wants to go back to shopping, then they are removed and then re-added from the arraylist of customers shopping in the store (they are re-initialized under the same name).
* This class will have a method that will print out the DEQueue lists (from front to back) of all the customers in Regular Line 1, Regular Line 2, and the Express Line