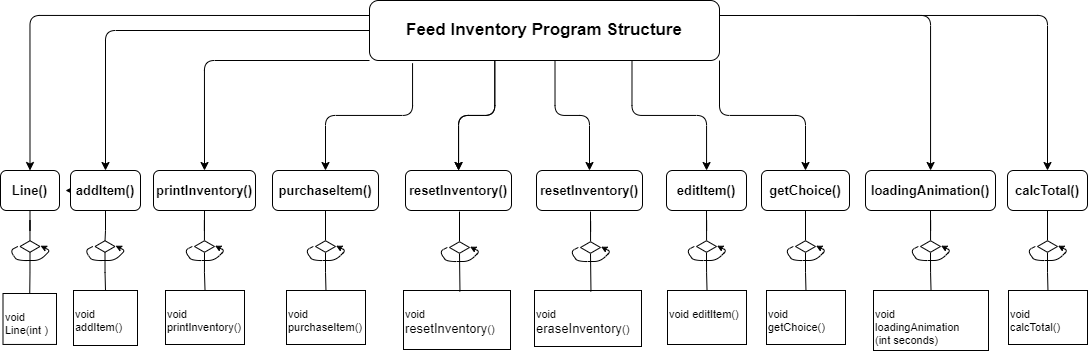
**CSC 250 – Program 6 Design Document**

**John Akujobi**

**Structure Chart**



**Function Design**

Note: I made all the functions so they do not need to take in values and have their own declared objects used within

1. **addItem()** - Add a new item to the inventory.
   * void addItem();
   * Asks the user for the name of the item, quantity in bags, and price per bag. Then it creates a new object with the input data and appends it to the end of the "feed.dat" file.
2. **printInventory()** - Print the information for all items in table form.
   * void printInventory();
   * Reads all the structures in the "feed.dat" file and displays the information for all items in a well-organized table format.
3. **purchaseItem()** - Buy’s an item by name and quantity.
   * void purchaseItem();
   * Asks the user for the name of the item and the quantity to purchase. Then it searches for the record that matches the name, updates the quantity, and writes the updated inventory back over the original one.
4. **calcTotal()** - Function to calculate and display the total value of the items on hand.
   * void calcTotal();
   * Reads all the structures in the "feed.dat" file and calculates the total value of the items on hand. It then displays the result to the user.
5. **getChoice()** - Function to get the operation choice from the user.
   * int getChoice();
   * Displays the operations menu to the user and prompts them to input their choice. Uses the enum type then returns the user's choice as an integer.
6. loadingAnimation() - Displays a loading animation while the program loads.
   * void loadingAnimation();
   * Displays a loading message and a moving message to show the user that the program is loading. It uses sleep() function to create a delay between each message.
7. editItem() -Edit an item's quantity or price.
   * void editItem();
   * Asks the user for the name of the item to edit, then displays the current information for that item. The user is then prompted to enter the new quantity or price, and the information for that item is updated in the "feed.dat" file.
8. resetInventory() - Function to reset all the quantities and prices of the items in the inventory.
   * void resetInventory();
   * Displays a confirmation message to the user before resetting all the quantities and prices of the items in the "feed.dat" file to their default values. This function should not change the names of the items.
9. eraseInventory() - Function to completely erase all the data in the "feed.dat" file.
   * void eraseInventory();
   * Completely erases all the data in the "feed.dat" file. Asks the user for confirmation first
10. Line() - Function to print a line of dashes
    * void Line(int num);
    * Prints a line of dashes to the console.
    * Takes an integer num as input, which determines the number of dashes to print, and returns nothing.

**Time Estimate**

|  |  |  |
| --- | --- | --- |
| **Function Name** | **Estimated Time** | **Actual Time** |
| Program Design | 60 mins | 45 mins |
| (list each function name and the time required to code each) |  |  |
| addItem() | 20 mins | 15 mins |
| printInventory() | 20 mins | 15 mins |
| purchaseItem() | 20 mins | 15 mins |
| getchoice() | 20 mins | 15 mins |
| loadingAnimation() | 0 mins | 5 mins |
| editItem() | 0 mins | 15 mins |
| eraseInventory() | 0 mins | 10 mins |
| resetInventory | 0 mins | 10 mins |
| Line() | 1 mins (copy from last program) | 1 mins |
| main() | 40 mins | 5 mins |
| Program Test | 30 | 20 mins |
| **Total Time** | 210 mins | 140 mins |