Elisha Parslow, Justice Martinez, Elizabeth Young

FINAL PROJECT Requirements Specification –FINALIZED

When the program opens,:

1. Pantry1 of the pantryCategory() class is created
2. The program goes to the menu function where all of the text files are created and open to be written into and assigned to pointers using the <stdio.h> library
3. all of the contents of the text files will be read into a vector to later be used to display the contents and clear the contents of the pantry.
4. Switch-Case: Main menu options – all options leading to a function in the class pantryInventory()

1) add inventory 2) remove inventory 3) list all complied inventory 4) clear all inventory 5) close program

Display sorted list of all items inventoried:

Use the listPantry() class function to call the vectorOfItems() function with the specific parameters to trigger option three in the switch case. Option three will tell the user the total number of contents in the pantry and then display them by reading out the main vector of all of the items. (The main vector reads in all of the text file items at the beginning of the program) The user will then be returned to t the main menu.

Clear all inventory?

If Y/y – clear all of the text files using the fopen(“filename”, w+) to make a new blank text file of the same name where the w+ allows the user to later write into the new text file. Erase the contents of the main vector containing all of the items

If N/n – return user to main menu

This is where the user leaves the program. The program is exited in the switch case inside the menu() function to avoid when the additem() function from looping through the menu.

removeInventory (pantryCategory\* p, vector<string> v):

1. Program goes to removeInventory() class function and asks user to select the category they wish to remove an item from:

- Beverage

- Bread

- Breakfast

- Dessert

- Fruit

- Meat

- Snack

-Vegetables

2. In each specific case, polymorphism is used because we don’t know which category the user will select. They all use inheritance of the pantryCategory() class as well and the case uses a pointer of the new category type to go to the removeItem() function.

3. The removeItem() function passes all of the required parameters and goes to the vectorOfItems() function case 2.

4. In case 2, a loop is created for the size of the main vector and the vector is searched for the string of the item that is being requested to be removed. Once found, v.erease() is implemented of the main vector and total count is decreased.

5. A temporary vector is created for holding the contents of the text file that is being pointed to, based on the parameters that are passed when the user selects their category.

6. The program then clears the file of the pointed to text file (as done in the clearPantry() function) and recreates a new text file that reads the updated contents from the temporary vector into it.

7. The file is closed so the changes are saved and the user is taken back to the menu where that file and all of the others are reopened. The user can continue to use the program or quit.

addInventory (pantryCategory\* p, vector<string> v):

1. Program goes to addInventory() class function and asks user to select the category they wish to add an item to in a switch case and the quantity to add:

- Beverage

- Bread

- Breakfast

- Dessert

- Fruit

- Meat

- Snack

-Vegetables

2. In each specific case, polymorphism is used because we don’t know which category the user will select. They all use inheritance of the pantryCategory() class as well and the case uses a pointer of the new category type to go to the addNewItem() function.

3. The addNewItem() function reads all of the chars of the string item the designated text file of the pointer. Total count is incremented to later be displayed in the listPantry() function. Then the function calls vectorOfItems() function case 1 to push back the added item into the main vector through a loop of the quantity of that item.

4. The user is then returned to the main menu where they can continue to use the pantry program or quit.

Class pantryCategory{

private:

static int totalCount; //used to keep track of every item in the pantry

public:

pantryCategory(); // constructor

void vectorOfItems(); // used to manipulate the vector containing strings of everything in the pantry

void addNewItem(); // function used to add a new item to the pantry and vector

void removeItme(); // used to remove an item from the pantry and vector

void setTotalCount(); // initializes initial count of items in the pantry, used when we reopen program

int getTotalCount(); // used to return the total count of items in the pantry

virtual string getItem() = 0 // creates virtual class for use of polymorphism

};

We created 8 separate sub-classes for each different category of food we had, we utilized polymorphism through this main class, each sub-class containing the name of the item in that category.

Problem Definition:

            The program is expected to keep an inventory of items the user inputs in text files. The user has four options, to add or remove items, view the items or delete the entire pantry. In the File Input/Output, each food category will have its own text file to organize the storage. The inventory will be counted by the number of strings that item is written in the text document. In the view option, the program will go through the vector containing all of the items and it will display everything currently in the vector listed out by what was first added to the vector. We see each item listed out from oldest in the pantry to newest. We don’t want to assume the user will remember if they put a capitalized item or not into the program so there will be a check to make all user input lower-case by *tolower*.