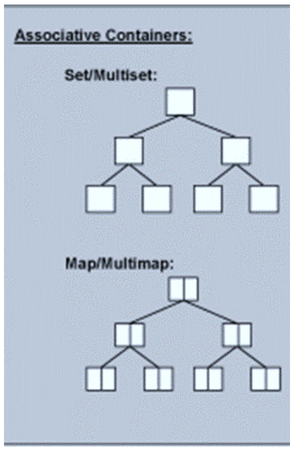
CPSC 131, Data Structures - Fall 2019
Inventory: Associative Containers Homework

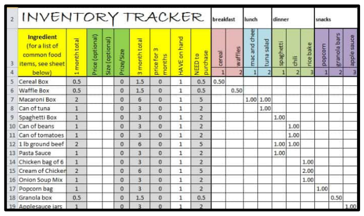
Learning Goals:



- Familiarization and practice with key/value association data structure usage and binary search tree concepts
- Familiarization and practice using the same key across associative containers to join the contents into a single view
- Familiarization and practice using the STL's map container interface
- Reinforce the similarities and differences between sequence and associative containers
- Reinforce reading persistent data from disk files and storing in memory resident data structures
- Reinforce modern C++ object-oriented programming techniques

Description:

This Grocery Store Inventory assignment builds on the Grocery Item, Shopping Cart, and Grocery Item Database from previous assignments by adding the maintenance of the grocery store's inventory at the checkout counter. Here you are given a collection of customers (e.g., Woodstock, Lucy, Carlie) pushing their shopping cart, each already filled with groceries. Each customer goes through the checkout line where the items are scanned and a receipt is given. As items are scanned, the store's inventory is updated by reducing the number of items on hand for that item. At the end of the day after all customers have been processed, you take inventory and reorder items that you're getting low on.



2	INVENTORY TRACKER									breakfast		lunch		dinner			snacks		
3	Ingredient (for a list of common food items, see sheet below)	1 month total	Price (optional)	Size (optional)	Prize/Size	3 month total	price for 3 months	HAVE on hand	NEED to purchase	cereal	waffles	mac and chee	tuna salad	spaghetti	chili	rice bake	popcorn	granola bars	apple sauce
4										1	2	1	2	1	2	3	1	2	3
5	Cereal Box	0.5			0	1.5	0	1	0.5	0.50									
6	Waffle Box	0.5			0	1.5	0	1	0.5		0.50								
7	Macaroni Box	2			0	6	0	1	5			1.00	1.00						
8	Can of tuna	1			0	3	0	1	2				1.00						
9	Spaghetti Box	1			0	3	0	1	2					1.00					
10	Can of beans	1			0	3	0	1	2						1.00				
11	Can of tomatoes	1			0	3	0	1	2						1.00				
12	1 lb ground beef	2			0	6	0	1	5					1.00	1.00				
13	Pasta Sauce	1			0	3	0	1	2					1.00					
14	Chicken bag of 6	1			0	3	0	1	2							1.00			
15	Cream of Chicken	2			0	6	0	1	5							2.00			
16	Onion Soup Mix	1			0	3	0	1	2							1.00			
17	Popcorn bag	1			0	3	0	1	2								1.00		
18	Granola box	0.5			0	1.5	0	1	0.5									0.50	
19	Applesauce jars	1			0	3	0	1	2										1.00

You are provided with starter code that together with work from previous assignments form your point of departure to complete this assignment.

1. **main.cpp** – This file is provided, requires no modifications, and will be overwritten during the grading process. Function main() orchestrates the flow of execution and tests your intermediate results along the way. The main flow is summarized as:

    a. Create a collection of customers and their shopping carts already filled with groceries. The collection, implemented using the STL's map class, is defined as a binary search tree with the customer's name as the key and a shopping cart as the value. As a convenience, an alias called ShoppingCarts has been created in file functions.hpp.

b. Load the store's inventory. The inventory, defined as a binary search tree and implemented using the STL's map class, is an association between UPC (the key) and the quantity on hand (the value). As a convenience, an alias called Inventory_DB has been created in file functions.hpp. You must write this function in file functions.cpp.

c. Process customer's shopping carts by scanning all the items in the basket, printing a receipt with an amount due, and deducting the items bought from the store's inventory. You must write this function in file functions.cpp.

d. Reorder items. For all the items sold today, if the quantity on hand has fallen below a re-order threshold, reorder more of those items. You must write this function in file functions.cpp.

2. **GroceryItem.hpp/GroceryItem.cpp** – Reuse the GroceryItem class from your previous assignments. Unless you find an error or omission in your code, these files should require no modification.

3. **GroceryItemDatabase.hpp/GroceryItemDatabase.cpp** – Reuse the GroceryItemDatabase class from your previous assignments, with modifications. The previous assignment used the sequential container std::vector for the memory resident database. For this assignment you are to modify this to use the associative container std::map, a binary search tree. File GroceryItemDatabase.hpp is provided, requires no modifications, and will be overwritten during the grading process. You must provide and modify file GroceryItemDatabase.cpp.

4. **functons.hpp/functions.cpp** – This pair of files provide the declarations and definitions for the following functions, which you must write. File functons.hpp is provided, requires no modifications, and will be overwritten during the grading process. You are to provide file functons.cpp and implement the functions declared in functons.hpp. Specifically:

a. loadInventory – This function takes no arguments and returns the store's inventory database. Open a text file called "GroceryStoreInventory.dat" containing pairs of UPC and quantity, each pair on a separate line. Populate the store's inventory database with the contents of the file. For example, the contents of "GroceryStoreInventory.dat" may look like:

```
00051600080015   36
00019600923015   34
00688267141676   36
00657622604842   25
```

b. processCustomerShoppingCarts – This function takes a collection of shopping carts and the store's inventory database as parameters and returns nothing. For each customer print out a receipt containing a full description (from the GroceryItemDtabase) of items in their shopping cart along with the total amount due. As items are being scanned, decrement the quantity on hand for that item in the store's inventory database. The logic to scan items and print a receipt should be carried forward from your last assignment.

c. reorderItems – This function takes a collection of shopping carts and the store's inventory database as parameters and returns nothing. For each grocery item in each shopping cart, check the store's inventory for the quantity on hand. If there are less than 15 items, print a message indicating the current number on hand then order 20 more by adding 20 to the current number on hand.

Reminders:

- The C++ using directive `using namespace std;` is never allowed in any header or source file in any deliverable products. Being new to C++, you may have used this is the past. If you haven't done so already, it's now time to shed this crutch and fully decorate your identifiers.
- Remember file names are case sensitive.
- Use Build.sh to compile and link your program – it employs the correct compile options.
- You may redirect standard input from a text file, and you must redirect standard output to a text file named output.txt. See How to build and execute your programs.

Deliverable Artifacts:

Provided files	Files to deliver	Comments
main.cpp GroceryItemDatabase.hpp functions.hpp	1. main.cpp 2. GroceryItemDatabase.hpp 3. functions.hpp	You should not modify these files. The grading process will overwrite whatever you deliver with the ones provided with this assignment. It is important that you deliver complete solutions, so don't omit these files in your delivery.
	4. GroceryItem.hpp 5. GroceryItem.cpp	Provided from the previous assignment.
	6. GroceryItemDatabase.cpp	Start with your solution from the previous assignment. Modify the file as described above.
	7. functions.cpp	Create this file and populate it as described above.
	8. output.txt	Capture your program's output to this text file and include it in your delivery. Failure to deliver this file indicates you could not get your program to execute.
Grocery_UPC_Database_Sample.dat GroceryStoreInventory.dat		Text files with a grocery store's item and inventory databases. Do not modify these files. They're big and unchanged, so don't include it in your delivery.
sample_output.txt		Sample output of a working program. Use for reference, your output format may be different. But the contents should match. Do not include this file with your delivery.