Zachary Balda

CS230 Final Project Documentation

Last Updated: 12/8/15

**Python Shopping Navigator**

**Overview and Summary**

The purpose of this program is to help anyone find their way through a store (grocery store, sporting goods store, clothing store, etc.). Ideally, each store will have a database containing information about every item held at that store. Included will be information about the item name, price, section, and shelf. To help the users navigate, there will be an image diagram that maps out the sections of the store.

Through a GUI, the user will be able to search for any item they need and add their items to a grocery list. A ‘Find Items’ button will then allow them to see every section they will need to go to, and what items they will need while at those sections.

**UI Design**

**Main Screen**



**What Each Button Does**

Search – Searches the CSV File and returns results

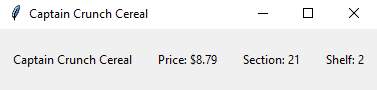
Locate Item – Opens a window containing information about the selected item

Add to List – Adds the selected item to the shopping list

View Map – Opens an image file of the store map

Find Items – Opens a window showing every isle you will need to go to and what items to get

**Located Item**



**Map**



**Organized List**



**Data Design**

Product information for the Store Database will be stored in a CSV file.

Store Database Sample:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Name** | **Key Term 1** | **Key Term 2** | **Key Term 3** | **Price** | **Section** | **Shelf** |
| Lemons | LEMONS | LEMON |  | 3.89 | 17 | 1 |
| Cheese Slices | CHEESE SLICES | SLICED CHEESE | CHEESE | 2.29 | 6 | 3 |
| Coffee Mix | COFFEE MIX | COFFEE |  | 5.99 | 12 | 4 |

When an item is searched for, an array for each column in the CSV file is created. Index position [1] in the itemNames array, for example, holds the text ‘Cheese Slices’ (the top row is excluded, meaning the Name, Key Term 1, etc. titles are not considered). Index position [1] in the itemPrices array holds 2.29, which is the price for Cheese Slices.

The search algorithm (see Search Algorithm in the algorithm section for more info) then creates arrays for the search results. These search results arrays include item names, item prices, item sections, and item shelfs. A radio button is then created for each search result and stored in an array to be accessed and removed later.

The ‘add to list’ button takes the selected item search results and adds them to shopping list arrays. These shopping list arrays hold all the same types of item information as the search results. Shopping list data is later used for the ‘Find Items’ algorithm.

Item information as well as GUI objects are stored in arrays.

Included Files

* CSV file for store item data
* PNG file of store map

**Algorithm**

The Initializer adds all GUI components (except for search results and shopping list items), creates arrays for the search results, search results GUI objects, shopping list items, and shopping list item GUI objects.

**Search Algorithm**

The search algorithm takes the text entered into the search entry, capitalizes it, and finds any matches in the Key Terms columns of the Main Database. To do this it starts at Key Terms 1, and looks for a match. If a match is found, that item is added to the search results and it moves to the next row. If there is no match, it moves to Key Terms 2, and does the same thing. If there are any matches in the Key Term, the item is added. If there are no matches, it is not.

**Add to List Button**

Each radio button that is created from the search results holds an integer value. The first (top) radio button holds the integer value 0, the second holds the integer value 1, and so on. The add to list button takes the integer value of the selected radio button, and uses it to find the index positions of the search result arrays. Search results item names, prices, sections, and shelfs at the index position of the selected radio button are then added to the shopping list arrays.

**Shopping List**

Shopping list item info is added to the end of the shopping list. These GUI objects are stored in arrays. The remove (“x”) button is given an integer value corresponding to its position in the array it is stored in. When it is click, it de-grids and deleted itself. Each button is then assigned a new number corresponding to its new index position. Additionally, each item is re-gridded to the correct position

**Find Items Button**

The find items button shows every section to go to, and what items to get while at those sections. It does this by searching through the shopping list and creating a list of all the ‘included’ sections. While cycling through this list, it does the following: 1. Finds the section at the index position it is looking at in the array 2. Creates a title for that value 3. Finds all items in the shopping list that share that section value 4. Grid’s those items 5. Repeats for each section in the ‘included sections’ array

**Sources:**

<http://effbot.org/tkinterbook/pack.htm#patterns>

* Useful .pack() info

<http://stackoverflow.com/questions/34054619/how-to-delete-or-de-grid-tkinter-gui-objects-stored-in-a-list-python?noredirect=1#comment55865799_34054619>

* How to de-grid tkinter objects stored in an array

<https://www.youtube.com/watch?v=K_oXb04izZM>

* Storing CSV Data in arrays

<http://stackoverflow.com/questions/11068872/pil-imagetk-equivalent-in-python-3-x>

* Opening a new window with an image

<http://stackoverflow.com/questions/34136434/making-buttons-that-locate-themselves-in-a-list>

* Buttons that delete themselves