WeGitIt APCS1 pd1

Roster: Lynne Wang, Sophia Xia, Thet Htay Zaw 2018-01-04

Final Project Proposal
Tea Database

Our project idea is to make a tea database. The user will choose between what symptoms they have and what taste they would prefer, and the terminal will return a suggested tea.

Rather than using classes to represent the teas, we plan to use a **2D arrays** to organize the data.

- There will be a 2D Array that organizes teas by what symptoms they can remedy
    - The first layer (Array[s]) will be the symptom
    - The second layer (Array[s][t]) will be the specific type of tea
    - Overall, accessing the teas that remedy a symptom will look like Array[Symptom][Teas]
- There will also be a similar 2D Array that organizes teas by their flavors
- A **search method (either linear or binary)** will go through the 2D arrays and record all teas that fulfill the criteria that the user wants
    - The recorded teas should be kept in a separate array (Not yet fully decided)
    - As more symptoms are looking to be remedied, the recorded teas array should get shorter
        - This array organization may implement **SuperArray (or its methods)**
    - The same recorded tea list will also get thinned out when the user chooses what flavors
    - In the case that there is no combination of symptoms and flavors, the terminal will return a “No possible combination” message
        - On the other hand, in the case that more than one tea fits the criteria, the terminal will suggest multiple teas
- The **Comparable** wrapper class can be used when we search and sort the arrays

- We will use compareTo to put the Strings of symptoms in alphabetical order (String unicodes are compared)
- This allows us to use binary search to look for the symptom / flavor that matches the user input (when compareTo returns 0, that means we have found a match)

The user will be able to input symptoms and flavors into the terminal through the **Scanner class (and Keyboard.java)**

- However, rather than being able to freely input flavors and symptoms, the user will be given a list to choose from
    - This is to combat any future case-sensitive input issues we may have
    - Invalid inputs should prompt the user to try again

One thing we are considering is the use of **ArrayList** instead of Arrays. This will require us to learn about (the possibility) of a 2D ArrayList and see how it compares to regular 2D Arrays. If it provides an advantage or more efficiency, we may opt to use a 2D ArrayList, but this is only being considered as of now.

Teas covered in the database:
Start out with 4 and slowly add more as we develop the versions

- Chamomile: stress, sleep, skin care, stomach
- Peppermint: fever, stress, digestion, nausea
- Ginger: nausea, stress, digestion
- Green: headaches, sleep, stress, nausea, aging
- Lemon, Oolong, White, Black, etc.

<u>To Do List</u>

1. Write a basic main method for Woo.java (driver)
2. Generate a list of teas; research the symptoms they alleviate and their flavor profile.

3. Write a method for each tea that simply returns information about the particular tea (in the form of a String). Information should include what symptoms the tea alleviates, the tea's flavor profile, and other pertinent details.
4. Generate sorted (according to alphabetical order) 2D arrays based on the information and write a method to search through the first layer of the array.
5. Write a method to compare different rows of the array (for multiple symptoms / flavors) and return tea that fulfills the most requirements.
6. Add user input to Woo.java using Scanner
7. Additional methods if desired.
8. Debug along the way, but final debugging for small issues.
9. Make code robust
10. Stretch

<u>Rough Timeline</u>

Complete points 1-4 from the priority list by 1-7-18

Complete points 5 and 6 by 1-14-18

Complete points 7 to 9 by 1-20-18

Tackle/complete 10 if finished early

Our project uses the OOP meaningfully by creating a database organized through a 2D array data structure. The 2D array consists of objects that each have their own unique set of features and groups them together based on those features.

The project uses the concepts of **2D arrays**, **searching**, and **sorting** using the most efficient runtime (**Big Oh**). The 2D arrays are being used to store and organize our data. They will be sorted according to alphabetical order to facilitate **binary search** for a faster runtime. We will be using the **Comparable** class for the **compareTo** method to facilitate alphabetical order. **Scanner** functionality will also be covered in order to allow the user to interact with the interface (also includes **exception handling**).