Food Fitter

Tawsi Studio

By: Wyatt, Sophia, Illya, Alexie, Thomas

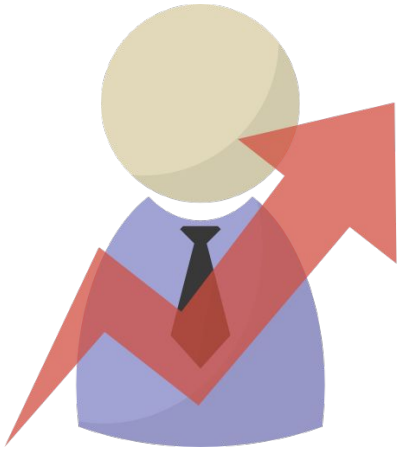
Objectives

- To provide a “virtual meal assistant” service.
- To plan out ideal meals for the user based on:
  - Dietary preferences
  - Nutritional preferences
- To allow users to save their meal plans for future reference
- To create an intuitive, responsive, mobile application
- To utilize searching, graph, and sorting algorithms to produce results in an efficient manner

Scope

Our project uses the information on different foods and food ingredients from our dataset to create meals: groups of said foods & ingredients, that match nutrient goals set by the user as closely as possible.

Motivations



- Personal experience with other nutrition applications
- Difficulty finding nutritional values for homemade food
- Current applications attempt to attain intake for the more common nutrients such as calories, fibres and protein, as opposed to taking into account more specific nutrients and vitamins, of which our implementation can handle 152.

Datasets Used

The dataset used was one published by Health Canada which contain statistics of nutritional values of many different foods and meals, with a lot of supporting information spread over multiple files. This included information such as food groups (with a total of 22 different ones), food refuse (which we didn't use), and food yield (which, sadly, was missing too much information to be used).

Functional Requirements

- Name-indexed food retrieval
  - Utilization of a prefix tree to index items from the dataset in a “natural” manner of language processing
- Option to save favourable meal plans
- Saved data deletion
- Future Meal Planning
- Back-end analysis using constraints and algorithms
  - Minimize error of selected foods and nutrition objectives using a Knapsack algorithm
  - Item sorting using Timsort

Non-Functional Requirements

- Security
  - Asynchronous local storage for meal plans
  - Asynchronous local storage for user preferences
- Capacity
  - Application doesn’t utilize more than 32 MB of RAM
- Performance
  - Back-end request time should not exceed 50 seconds
- Maintainability
  - Ability to handle changes to the data set (additions, deletions)
- Usability
  - Accessibility through font selection, contrast, and keystrokes

Algorithmic Challenges

Prefix Tree (Graph)

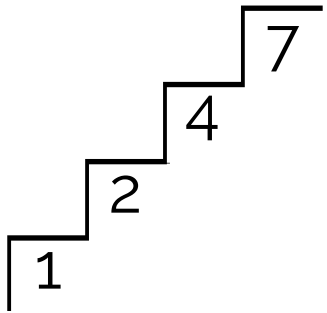
- Word library is represented by a tree of characters
- Supports find and insert operations
- Find words by traversing tree

TimSort (Sorting)

- Combination of Merge Sort **O(n lg n)** and Insertion Sort **$O(n^2)$**
- When number of items is small Insertion sort is used

Knapsack (Search)

- Fills buckets with food selections
- Each bucket represents a certain # calories
- Buckets are filled using Dynamic programming (Davis' Staircase)



Input/Output

Input

- Collection of prefered foods
- Calorie goal
- Nutritional value goals
  - Protein, carbs, etc

Output

- Collections of foods that prioritize nutrition and calorie goals and preferences

Verification & Validation

- Black box testing
- Trying sample queries, and testing if the outputs generated are within a reasonable range of the targeted nutritional goals
- JUnit 4