SFWRENG 2XB3: Software Engineering Practice and Experience: Binding Theory to Practice

Requirements Specifications Document

ezHealth

Rehan Theiveehathasan - 001416031
Edwin Do - 400079331
Timothy Chen - 400129081
Brian Kibazohi - 400077789
Ridhwan Chowdhury - 400075426

14 April 2019

TABLE OF CONTENTS

Introduction

1.1 Domain

Using the user’s weekly grocery budget, ezHealth will generate the most varied and healthy grocery list that the user can buy from a database of wholesale foods. Using the data of the weekly wholesale market price of food, we can help the user stay healthy and save both time and money. The goal of this project is to reduce the amount of time the users need to think about what to buy for groceries and help them buy healthy and balanced meals. The objective of the project is to help those who wish to start a healthy diet have an easier transition by generating a budget-friendly grocery list for the week.

Products such as the Cozi Recipe Box & Dinner Planner, SmartyPig, and YAZIO all share something similar to my expected product. Cozi offers features that allow the users to find recipes and create shopping lists which is very similar to how ezHealth is going to create a weekly shopping list (Person,2). The main difference is that ezHealth is more designed towards helping the user leading a healthy diet while trying to balance their budget. When compared to SmartyPig, we see that SmartyPig offers the budget management features but does not incorporate the aspects of healthy eating (Lockert et al., 5). YAZIO is one of the most similar products to ezHealth. It offers features to help the user maintain a healthy diet by allowing the user to track the food they eat and their weight (Weindling et al., 1). What separates ezHealth from YAZIO is that ezHealth does not track every meal, instead it tracks what the user buys and goes into the fridge every week. The goal of ezHealth is to incorporate different features from Cozi, SmartyPig and YAZIO to create a product that plans a weekly grocery plan using the given budget that is healthy and nutritious.

Specification API

2.1 Read.java

ID: R-1	private static void writeFile(String output, String data)
Parameters	String output, name of the file to write to String data, The data being written into file
Return	void
Description	Writes to a text file and outputs it.

ID: R-2	private static List<String[]> readin(String fileAddress, boolean ignoreHeader)
Parameters	String fileAddress, path to file (csv or text)
Return	List<String[]>
Description	Reads data file

ID: R-3	private static String[][] conv(List<String[]> arr)
Parameters	List<String[]> arr
Return	String[][]
Description	Returns a temporary array for accessing and manipulation

ID: R-4	private static void sortbyColumn(String arr[][], int col)
Parameters	String arr[][], array of type String Int col, column entry
Return	List<String> of user’s groceries
Description	Sorts in descending order based on comparison of columns

ID: R-5	private static List<String[]> filter(String fileAddress, boolean ignoreHeader)
Parameters	String fileAddress, path to file (csv or text)

Return	List<String[]>
Description	Filters through text or csv file to output grocery list

ID: R-6	private static float price(String p, String u)
Parameters	String p, price of grocery item String u, unit price of grocery item
Return	float
Description	Parses through database and out price and unit price

2.2 Bag.java

ID: B-1	public Bag()
Parameters	N/A
Return	N/A
Description	Initializes an empty bag

ID: B-2	public boolean isEmpty()
Parameters	N/A
Return	Returns boolean value (True or False)
Description	Returns true if Bag is empty

ID: B-3	public int size()
Parameters	N/A
Return	Integer
Description	Returns number of items in bag

ID: B-4	public void add(Item item)
Parameters	Item, the item to add to this bag
Return	N/A

Description	Adds the item to this bag

2.3 DirectedEdge.java

ID: D-1	public DirectedEdge(int v, int w, double weight)
Parameters	v, the tail vertex w, the head vertex weight, the weight of the directed edge
Return	N/A
Description	Constructs a directed edge from vertex v to vertex w with a given weight.

ID: D-2	public int from()
Parameters	N/A
Return	the tail vertex of the directed edge
Description	Returns the tail vertex of the directed edge.

ID: D-3	public int to()
Parameters	N/A
Return	Returns the head vertex of the directed edge.
Description	Returns the head vertex of the directed edge.

ID: D-4	public double weight()
Parameters	N/A
Return	Returns the weight of the directed edge.
Description	Returns the weight of the directed edge.

ID: D-5	public String toString()
Parameters	N/A

Return	Returns a string representation of the directed edge.
Description	Returns a string representation of the directed edge.

2.4 Food.java

ID: F-1	public Food(String name,double a_price)
Parameters	name: a string that corresponds to the name of the food A_price: the average wholesale price of that food obtained from data
Return	N/A
Description	Creates new Food object with parameters of its name and average price

ID: F-2	public String getName()
Parameters	N/A
Return	Returns a string of the Food object's name
Description	Returns the name of a food object

ID: F-3	public double getPrice()
Parameters	N/A
Return	Returns the double value that corresponds to the food object's average price
Description	Returnsthe average price of a food object

ID: F-4	public String toString()
Parameters	N/A
Return	Returns both the name and the average price of the Food object in string format
Description	Converts both the name and average price into string

ID: F-5	public int compareTo(Food food)
Parameters	Food: the food object being compared to the current one
Return	0: if the average price between the two are equal 1:if the average price of the current Food is greater -1: if the average price of the current Food is less
Description	Gives a comparison result of the average prices between two Food objects

2.5 EdgeWeightedDigraph.java

ID: E-1	public EdgeWeightedDigraph(int V)
Parameters	V: Number of vertices to construct the edge-weighted digraph with.
Return	N/A
Description	Creates a V sized edge-weighted digraph.

ID: E-2	public int V()
Parameters	N/A
Return	Returns number of vertices in this edge-weighted digraph.
Description	Returns number of vertices in this edge-weighted digraph.

ID: E-3	public int E()
Parameters	N/A
Return	Returns the number of edges in this edge-weighted digraph.
Description	Returns the number of edges in this edge-weighted digraph.

ID: E-4	public void addEdge(DirectedEdge e)
Parameters	E: DirectedEdge object
Return	N/A
Description	Adds a directed edge object to graph.

ID: E-5	public String toString()
Parameters	N/A
Return	Returns a string representation of this edge-weighted digraph.
Description	Returns a string representation of this edge-weighted digraph.

ID: E-6	public void simpleLongestPath(double budget, List<Food> food)
Parameters	Budget: Budget constraint for simple longest path algorithm Food: List of food objects
Return	N/A
Description	Finds the longest path in the graph and prints the food items in it to a file called traces.txt.

Requirements

4.1 Non Functional Requirements

- Reliability - The project to be implemented has been considered to be reliable by the team due to the following explained factors. The team expects the program to be able to run 24 hours a day as long as the user has access to a machine capable of running the program. Today, the privacy of data is a serious issue in this fast growing world and to handle this, the user's input data will remain in his own machine ensuring integrity and security of the product.
- The accuracy of results - Taking into account our project's designed API and specification our program is simple using one dataset to provide the user with the required output. The datasets are fairly discrete hence retrieving the required information would not be difficult. Hence we are expecting accurate results as a team.
- Performance - Best practices will be used by designing and also optimizing the algorithms from relevant sources and implementing them where required in the project. Bad runtime algorithms will be avoided.
- Human-computer interface issues - The project's main focus is on the implementation of the algorithms and not the user interface. Regardless of this, the team has decided to implement a simple java interface when the time is available for demo purposes.
- Portability issues - The product is expected to portable in most systems because of Java being the programming language that will be used.
- Safety - If this application fails, the user will not be affected physically in any way possible, hence it is safe.

4.2 Functional Requirements

For the ezHealth application to perform the main task of creating the most varied and largest healthy wholesale grocery food shopping list given a budget constraint it must perform the following tasks:

- Take in user input of a budget
- Parse wholesale food database into uniquely named food objects, with their respective common unit's (mass/quantity) associated average price
- Sort the food objects by price from least to greatest
- Create an edge weighted digraph
- Add directed edges between the food objects in the graph such that: there are no cycles, and directed edges connect only from an object with a lower price to an object of higher price.
- Perform a simple longest path algorithm to find the longest path, and thus finding the largest possible shopping list for the user given their budget
- Save their shopping list in a text file that they can personally edit if they so choose.

4.3 Requirements on the development and maintenance process

- Quality control assured through system tests via eclipse JUnit testing for each function
- Priorities of required functions in order of most important/critical to least: all functions found in Read.java, all functions found in Food.java, E-6, then everything else.. Please refer to System Requirements Specifications above to reference ID values.
- Tests functions are expected to be created on the fly after each module/method is completed.
- Consistent code styles will be followed to improve readability and clarity of code.