Swinburne University of Technology

Faculty of Information & Communication Technologies

HIT3037/ HIT7037 Programming in Java

Assignment 2, 2011 Semester 2

Due: 8:30am Tuesday, October 25, 2011 (Week 11)

NOTE

1. This assignment can be done by ***one person*** *or* ***a pair.*** It is worth 15% of the total subject score.
2. You are required to electronically submit a zip file that includes a user manual, detailed class diagram, source code (.java files only - excluding .class and project files) for your program, and a subdirectory of html generated by javadoc. The submission system is located at https://esp.ict.swin.edu.au/. Use your SIMS username and password to login and submit your assignment there.
3. You may be required to attend an oral assessment.

1. System Functional Requirements

Your task is to develop a GUI program that allows the user to read and search cake receipts from html files *stored* in the user specified directory. The user specified directory is the first command line argument, that is, args[0], to the program. If args[0] is not provided or if no html files are present in the user specified directory, then the program should print an error message to the command prompt, and stop the execution.

In this assignment, you are given 10 html files and 10 images (one html and one image per cake) copied from the following links.

http://www.taste.com.au/recipes/17173/almond+cake+with+strawberries+caramel+sauce
http://www.taste.com.au/recipes/9524/blueberry+cake
http://www.taste.com.au/recipes/2953/banana+date+ice+cream+cake
http://www.taste.com.au/recipes/8958/birthday+cup+cakes+with+cream+cheese+frosting
http://www.taste.com.au/recipes/21604/carrot+pineapple+daisy+cake
http://www.taste.com.au/recipes/12032/chocoholics+delight+cake
http://www.taste.com.au/recipes/24622/caramel+apple+upside+down+cake
http://www.taste.com.au/recipes/2959/fruitcake+stars
http://www.taste.com.au/recipes/19089/strawberry+cheesecake+roulade
http://www.taste.com.au/recipes/25581/spelt+and+honey+banana+bread+with+dates+and+walnuts

Use the Notepad to open the given html files to understand the structure of these files.

Your program needs to process each html file to obtain the following *key data*.

- name of the cake
- file name (.jpg file) of the cake's image,
- preparation time,
- cooking time,
- ingredients, and
- method (steps used to make the cake).

If an html file excludes *cake's name*, *ingredients* or *method*, your program should report an error message about this html file to the command prompt, but keep processing other html files. If no error message was reported for an html file, your program should create a CakeRecipe object based on the data of this file. In other words, a CakeRecipe object is allowed to have an undefined image's file name, preparation time or cooking time. When the image's file name is given in the html file, the system should check whether the image file is accessible in the user specified directory. If the file is inaccessible, your program should report an error message and change the image's file name to null.

After processing all the html files, if none of the input html files can be used to create CakeRecipe objects, your program should print an error message to the command prompt, and stop the program execution. On the other hand, if some html file(s) can be used to create CakeRecipe object(s), your program should create a JFrame and make it visible. Your program should ask each CakeRecipe object to print its key data to a file (with .txt as its file extension). When printing the key data to the file, your program should ensure that each line is no more than 60 characters long. Furthermore, your program should not split any single word into multiple lines.

Figure 1 shows a sample file that the program could generate for a CakeRecipe object. The name of the text file is associated with the cake's name, and has .txt as the file extension.

Using your GUI program, the user can make *queries* in the constructed CakeRecipe objects. For each query, the user can enter up to five *key words*. Moreover, the user can choose to find the words in a certain scope. For example, he/she can find words only in "Cake's name" or "method", or anywhere (the default).

Once the system find the CakeRecipe objects that contain any of the words in a query, it will do the following calculation. For each of these found objects, the system calculates how many of these key words has been found in the object. The system uses this figure as a measure (calling "matching-degree" here) to indicate how relevant the object is for the query.

Your program will sort the found CakeRecipe objects based on one of the following criteria:

- the matching-degree in descending order or ascending order
- the preparation time in descending order or ascending order
- the cooking time in descending order or ascending order
- the sum of preparation and cooking time in descending order or ascending order

If the data for preparation time or cooking time of a certain recipe is missing (not shown in the html file), you have to assume that the corresponding cake needs longer time to prepare and cook than other cakes whose recipe explicitly describes the preparation and cooking time.

After sorting these CakeRecipe objects, your program should give a simple report on these objects by displaying the cake's name and image (if .png file is present). If the user is interested in any recipe, he/she will be able to open the entire recipe (containing ONLY Cake's name, image, preparation time, cooking time, ingredients and method) in another JFrame. On the other hand, if the user is not interested, he/she will also be able to do another query.

2. System design

1. Your system should be developed using JFrame, JList, JComboBox, JTextfield, JTextArea, JLabel, JButton and some kind of dialog components. You are free to use more GUI components other than the ones listed here. These components should be displayed using Layout Managers. The java file with the main() function should be named "**Assign2.java**".
2. Your program should use the Generic Collection classes (Vector, ArrayList, HashMap or Stack) to store the ingredients of a recipe.
3. Your system should have a main window that displays a welcome message and your student ID (no name should be given here). From this main window, the user runs different functions of the system.
4. Your program should inform the user of the outcome of a function under execution. You cannot let the user guess the progress of execution.
5. Code for all GUI classes must be hand-written, that is, not generated by any IDE.
6. Your system should have good GUI and event handling design, so the user finds the system user-friendly (easy to learn/navigate the system).
7. Your system should prevent the user from making mistakes (for example, wrong data type) that may throw exceptions and/or cause the system to enter an abnormal state. For such mistakes, your system should return some meaningful warning messages, so the user knows what they have done incorrectly.
8. Your program must avoid or catch all kinds of exceptions, so its execution won't be terminated or it won't enter any abnormal state. After catching an exception (regardless of whether it is a user-defined exception or system-based exception, such as IOException), your system should output some meaningful error message related to this exception, so the user knows what went wrong.

3. User manual

Create a user manual to explain how the user should use the system. You can include some system snapshots in the user manual. This manual should not be long (10 pages at maximum). This user manual will not be given any marks because it is used by the tutor to mark your assignment.

4. Detailed Class Diagram

Create a detailed class diagram showing relationships among classes as well as variables, methods and associated visibility modifiers of each class for your Assign2 program.

5. Other Requirements

1. Your software must separate the GUI classes from non-GUI classes (where the core functions are implemented) as much as possible. One of your non-GUI classes should be called "**CakeRecipe**".
2. Each class has meaningful methods and instance variables associated with it, so its design can be reused easily in other similar applications.
3. Each method must have a single purpose and should be kept small.
4. When a method is only invoked internally within its class, it should be declared as private.
5. Each class cannot have public variables, but it can have public constants.
6. Good coding style standards must be followed:
   - Indentation is consistent.
   - No *magic* (unexplained) *numbers* will be used, except for counters like "for (int i = 0; ...)".
   - Identifiers are meaningful (not too abbreviated).
   - Every Java file has a Javadoc class header comment showing its main purpose, version and author (including the student ID).
   - Every method has a Javadoc method header comment indicating its purpose, as well as the purpose of its individual parameters and return value. Important pre-/post-conditions of the method (if any) are also described in its method header comment.
   - Line/block comments are placed in front of any block of code that does something not obvious (Do not overdo line/block comments; e.g. do not comment normal language usage. Assume that these comments are aimed at proficient Java programmers).

```
**********
Name of the cake
Spelt and honey banana bread with dates and walnuts
**********
Name of the cake's image
25581.jpg
**********
Preparation time
20 minutes
**********
Cooking time
40 minutes
**********
Ingredients
1 1/2 cups (225g) whole spelt flour
1 1/2 cups (225g) white spelt flour
3/4 cup (155g) brown sugar
3 tsp baking powder
1 tsp bicarbonate of soda
2 tsp ground cinnamon
1 cup (100g) walnuts, coarsely chopped
1 cup (150g) dried pitted dates, coarsely chopped
2 cups mashed banana
3/4 cup (185ml) vegetable oil
3 eggs, lightly whisked
1/3 cup (80ml) honey
**********
Method
1. Preheat oven to 180°C. Grease and line the base and sides of two 8 x 20cm loaf
pans.
2. Combine the spelt flours, sugar, baking powder, bicarbonate of soda and cinnamon
in a large bowl. Add the walnuts and dates and stir to combine.
3. Combine the banana, oil, eggs and honey in a small bowl. Add to the flour mixture
and stir with a wooden spoon until just combined. Pour into the prepared pans and
smooth the surface. Bake in preheated oven for 35-40 minutes or until a skewer
inserted in the centre comes out clean. Remove from oven.
4. Transfer the loaves to a wire rack and set aside to cool. Use a large serrated
knife to cut into slices. Serve the banana bread warm, at room temperature or
toasted with butter, if desired.
```

Figure 1. Sample output file

HIT3037/ 7037 Assignment 2 Feedback

Student ID/ name:	Lab class (time/day)

Items	Marks
Non-GUI functionality and computation correctness (25 marks)	+
GUI/Usability design and event handling (40 marks)	+
Exception handling and avoidance (15 marks)	+
Class diagram, OO design and implementation (14 marks)	+
Coding styles and comments (6 marks)	+
	+
Total marks	+

Late penalty (10% penalty per working day) ***Formula***: Number of days late × 10% × Total marks	-

FINAL MARKS (= Total marks – Late penalty)	

COMMENTS:

Special cases: Any of the following cases will override the marks given above

(a) Detection of plagiarism (YES or NO); (b) Set-up of an oral assessment (YES or NO)