COOKIE COOKBOOK

Ultimate C# Masterclass Assignment

Overview

This application lets the user create and save cookie recipes. The user can select the ingredients that will be included in the recipe from a list. The recipe is then saved to a text file along with recipes that have been created before. The text file might be either in a *.txt or a *.json format, depending on the setting defined in a program.	Console App

Glossary

- **Ingredient** - represents a single ingredient that can be included in the recipe, for example, **Wheat Flour** or **Sugar**. Each ingredient has an **ID**, a **name,** and a string with **instructions on preparing** it. See "**Ingredients**" for more info.
- **Recipe** - a collection of **Ingredients** (for example, we could have a simple recipe with three ingredients: **Wheat Flour**, **Butter**, and **Sugar**).

Main application workflow

[Only if some recipes are already saved] When the application starts and some recipes are already saved, it prints all existing recipes. See "**Printing existing recipes.**"
Then the application prints:
Create a new cookie recipe! Available ingredients are:
...and then it prints all available ingredients. See "**Printing available ingredients**."
Then, the user can select ingredients from the list of ingredients. See "**Selecting ingredients for a new recipe**."

If the user has **not** selected any ingredient:	If the user selected **at least one** ingredient:
"No ingredients have been selected. Recipe will not be saved." is printed.	"Recipe added:" is printed, and the newly-added recipe is printed. See "**Printing single recipe**."
No recipe is saved.	A new recipe is saved in the text file. See "**Storing recipes in a text file.**"

Then, the application prints:
Press any key to exit.

Ingredients

Ingredients are the components of a recipe. For example, a single **Recipe** can consist of **Wheat Flour**, **Sugar**, **Butter** and **Cinnamon**. In this case, a Recipe is composed of 4 Ingredients.

Each ingredient has an **ID**, a **name,** and **preparation instructions**. For example, we could have an Ingredient named "**Butter**", with id **3**, and the instruction of preparing "**Melt on low heat. Add to other ingredients**."

What ingredients are available and how exactly they are represented in the program is at the developer's discretion. The developer may choose to use base classes, interfaces, or any other mechanisms they want.

Below is the list of **example** ingredients:

ID	Name	Instruction of preparing
1	Wheat flour	Sieve. Add to other ingredients.
2	Coconut flour	Sieve. Add to other ingredients.
3	Butter	Melt on low heat. Add to other ingredients.
4	Chocolate	Melt in a water bath. Add to other ingredients.
5	Sugar	Add to other ingredients.
6	Cardamom	Take half a teaspoon. Add to other ingredients.
7	Cinnamon	Take half a teaspoon. Add to other ingredients.
8	Cocoa powder	Add to other ingredients.

Printing existing recipes

Scenario	Result
Some recipes are already saved in the text file.	**All recipes** should be loaded from the file (See "**Storing recipes in a text file.**") "Existing recipes are:" is printed, and then, after a line break, **all recipes** are printed on the screen. Every recipe is separated from others with "***** {N} *****" string, where {N} is 1 for the first recipe, 2 for the second, and so on. Every single recipe shall be printed according to the "**Printing single recipe.**"
Example:	Existing recipes are: ***** 1 ***** Wheat flour. Sieve. Add to other ingredients. Coconut flour. Sieve. Add to other ingredients. Butter. Melt on low heat. Add to other ingredients. ***** 2 ***** Butter. Melt on low heat. Add to other ingredients. Chocolate. Melt in a water bath. Add to other ingredients. Sugar. Add to other ingredients.
No recipes are yet saved in the text file.	Nothing is printed to the console.

Printing single recipe

Scenario	Result
The recipe contains at least one ingredient (it is validated when a recipe is created, so no other scenario is possible).	Each ingredient from a recipe is printed to the console in a separate line, along with its preparation instructions.
Example:	Wheat flour. Sieve. Add to other ingredients. Coconut flour. Sieve. Add to other ingredients. Butter. Melt on low heat. Add to other ingredients.

Storing recipes in a text file

The program should support both *.**txt** and *.**json** file formats. Which one will be used is, for simplicity, controlled by a const variable in the program (optionally, you can make the user choose it when the app is run for the first time. Next time, it can be deduced by checking if an existing file is in the *.txt or *.json format).

For example, this is how it could be "configured" in the program with a custom enum:

```
const string FileName = "recipes";

//We can imagine that the user chooses this
//format when te app is first run,
//but let's skip it to not overcomplicate the code.
const FileFormat FileFormat = FileFormat.Json;
```

Now, the recipes will be stored in a JSON format. To change the format from *.json to *.txt, the developer must simply change the const FileFormat value and run the app again.

Since each **recipe is just a collection of ingredients**, we can save each recipe as a text containing all the **IDs** of its ingredients.

As an example, let's consider the following ingredients:
- **Wheat Flour** has ID **1**
- **Sugar** has ID **2**
- **Cocoa powde**r has ID **3**

*.txt files
Each line represents one recipe.
This could be the example content of the *.txt file:

```
1,2,3
1,2
```

This file contains two recipes:
- First with ingredients with IDs 1,2,3, so Wheat Flour, Sugar, and Cocoa
- Second with ingredients with IDs 1,2 so Wheat Flour and Sugar

*.json files
JSON contains a single array of strings. Each string contains all IDs of ingredients separated by a coma.

This could be the example content of the JSON file (the same recipes as in the example for

the *.txt file):

```
["1,2,3" , "1,2""]
```

Scenario	Result
No recipes have been added yet.	A new text file is created, and a recipe is saved in it.
A new recipe is added.	A recipe is added at the end of the existing file.

Printing available ingredients

Scenario	Result
Ingredients are printed to the console	Each ingredient's name is printed to the console in a separate line. The line should start with the ingredient's ID. The preparation instructions should not be shown.
Example:	1. Wheat flour 2. Coconut flour 3. Butter 4. Chocolate 5. Sugar 6. Cardamom 7. Cinnamon 8. Cocoa powder

Selecting ingredients for a new recipe

After the list of all available ingredients is shown to the console (See **"Printing available ingredients"**), the user can select ingredients in a loop.

In the loop, the following steps are executed:
- "Add an ingredient by its ID or type anything else if finished." is printed to the console.
- Then, the user selects the ingredient by entering its ID.
  - If a valid ID is entered, the ingredient with this ID is added to the recipe, and the loop is executed again.
  - If a number is entered, but it does not match any ingredient, we execute the loop again.
  - If anything else than a valid number is entered, we consider the recipe finished. The loop stops. We continue to the next step described in the **"Main application workflow"**, which is printing and saving the recipe.

At this stage, the recipe is not validated in any way. **The only validation that happens before the recipe is saved is checking if there is at least one ingredient in the list of ingredients (See "Main application workflow").** So even a recipe with single ingredient **Butter** should be saved as a valid recipe. Ingredients can also be repeated, so a recipe with **Wheat Flour**, **Wheat Flour,** and **Sugar** is also valid.

Scenario	Result
The user selects a number matching an existing ingredient.	The selected ingredient is added to the list of ingredients that will be included in the new recipe.
The user selects a number, but it doesn't match any of the existing ingredients.	No ingredient is added to the list of ingredients in this iteration. The loop executes again.
The user does not select a valid number.	The loop stops and we consider the recipe finished.