**Description**

In a world brimming with culinary diversity, exploring and organizing recipes can be a delightful journey. The Recipe Management System is designed to simplify this culinary adventure by offering a user-friendly platform for discovering, organizing, and saving your favorite recipes.

Whether you're an aspiring chef, a seasoned home cook, or simply curious about the world of gastronomy, the system that you are going to build provides a convenient way to navigate through a vast collection of recipes.

Follow the instructions below to get started! After you are finished, make sure that you meet all the requirements (below instructions).

**Instructions**

 Create a function that returns all authors of a given recipe list. Use the **.forEach** method. The function takes recipes, such as cakeRecipes, as arguments and returns a list of unique authors. This means that each author can only be found once in the returned list.

 Log this list to the console.

 Create a function that logs the name of each recipe. As input, it takes in a list of recipes with the same format as cakeRecipes. Use **object destructuring** in this function. If there are no recipes found in the input, then log that there are no recipes found.

 Now, create a function that returns all recipes of a given author. Use the **.filter method**. The function takes recipes and author (string) as arguments and returns all recipes from the given author.

 Log the names of these recipes to the console using the print function you just created.

 Create a function that returns a list of recipes that contain a given ingredient. The function takes a list of recipes as input and an ingredient as a string. Use the **.filter()** method to filter the recipes and the **.some()** method to check if the ingredient list contains the given ingredient (input).

 Log the names of the found recipes that contain a specific ingredient using your print function. For example, when you give the cakeRecipes list as input and “140g caster sugar” it should return: Christmas pud cupcakes, Baby buttermilk pancakes with sticky bananas & Brazils, Fruit-filled clementine cake.

 Now, create a function that takes a list of recipes and a name (string) as input. The function returns a single recipe that matches the given name. Use the **.find()** and **.includes()** method.

 Log the recipe into the console.

 Finally, create a function that returns all ingredients of a given recipe list into a single array. You can use this function, for example, when you want to create a grocery list. Use the **.reduce()** method to flatten the recipe array.

* Also log the output of the above function into the console. As input, we recommend using a shorter array than cakeRecipes. For example, you can use the list of recipes of a given author from one of the previous made functions.

**Completing the Recipe Management System**

In the starter code, you can find a displayMenu function that displays all the available options that our management system has. It also shows a while loop that runs until the user has selected to exit the system (by selecting option 0).  
  
To complete the system we will complete the switch statement and use all the previously built functions to make it one coherent system. Please complete the following functionalities:

**Show All Authors**

* When this option is selected, the program should display a list of all unique authors found in the recipe data.

**Show Recipe Names by Author**

* If you choose this option, you will be prompted to enter the name of an author.
* The program will then display a list of recipe names authored by the specified author.

**Show Recipe Names by Ingredient**

* This option allows you to search for recipes containing a specific ingredient.
* You will be prompted to enter the name of the ingredient.
* The program will display a list of recipe names that include the entered ingredient.

**Get Recipe by Name**

* If you select this option, you can search for a recipe by its name.
* Enter the name of the recipe you're looking for.
* The program will display the details of the found recipe.
* You'll also have the option to save the ingredients of the recipe. Save them in a global variable, so that you can save the ingredients of multiple recipes and also use it for the next step.

**Get All Ingredients of Saved Recipes**

* Choose this option to view a list of ingredients from the saved recipes.
* The program will display the ingredient list from all the saved recipes.

**Tips**

* Don’t forget to take regular breaks
* You can use code examples from previous exercises and projects
* Make sure to use good names for your variables and functions that describes it’s purpose.
* Properly decomposing problems into functions is important for this assignment. Follow the process that you learned for that. Be sure to test the smaller functions you use for parts of the problem to check if they’re working correctly.

**Requirements**

 All functions are pure functions except the printing function.

 **Function to Get Unique Authors**

* Create a function that returns all authors of a given recipe list.
* Utilize the `.forEach` method for iteration.
* **Input**: `recipes` (an array of recipes with the same format as `cakeRecipes`)
* **Output**: A list of unique authors (each author appears only once in the returned list).
* Users can choose this option from the menu to view authors' names.

 **Function to Log Recipe Names**

* Create a function that logs the name of each recipe.
* **Input**: `recipes` (a list of recipes with the same format as `cakeRecipes`).
* Utilize object destructuring within the function to access recipe names.
* No specific output is required; it should log the names to the console.
* Users can select this option from the menu and triggers this function.

 **Function to Get Recipes by Author**

* Create a function that returns all recipes of a given author.
* Utilize the `.filter` method for filtering recipes by author.
* **Input**: `recipes` (an array of recipes with the same format as `cakeRecipes`) and `author` (a string representing the author's name).
* **Output**: An array of all recipes authored by the given author.
* This option is available in the menu and triggers this function.

 **Function to Get Recipes by Ingredient**

* Create a function that returns a list of recipes that contain a given ingredient.
* Utilize the `.filter()` method to filter recipes.
* Use the `.some()` method to check if the ingredient list of each recipe contains the given ingredient.
* **Input**: `recipes` (an array of recips with the same format as `cakeRecipes`) and `ingredient` (a string representing the ingredient to search for).
* **Output**: An array of recipes that contain the specified ingredient.
* This feature is accessible from the menu and triggers this function.

 **Function to Get Recipe by Name**

* Create a function that returns a single recipe that matches a given name.
* Utilize the `.find()` method to find a recipe by name.
* Use `.includes()` method to check for name match.
* **Input**: `recipes` (an array of recipes with the same format as `cakeRecipes`) and `name` (a string representing the recipe name to search for).
* **Output**: The recipe object that matches the given name, or `null` if no match is found.
* After running this function from the menu, give the user the option to save the found recipe so that it can be used to print an ingredient list (see next requirement).

**Function to Get All Ingredients**

* Create a function that returns all ingredients of a given recipe list in a single array.
* Utilize the `.reduce()` method to flatten the recipe array.
* **Input**: `recipes` (an array of recipes with the same format as `cakeRecipes`).
* **Output**: An array containing all the ingredients from the recipe list.
* Users can select this option from the menu to view the ingredient list from the recipes that are saved in the previous function.