Problem Description:

In this project, you are required to build a system that helps users plan their meals in a conscious manner by randomly generating for them a set of recipes per session. The system tracks users that want to ensure that their recipes comply with their health needs. To perform this tracking, the system creates user profiles by storing personal and health-related information in a file. This allows users to load their information from the file for later usage. For a user, the system should generate 4 to 6 random recipes per session from a file that contains a collection of recipes from allrecipes.com. Based on the user health requirements, the system should calculate and display statistics about the calories from each session and the overall user progress in terms of keeping up with their caloric intake goals. The system should allow users to view and assess their meal choices based on the nutritional value of their selected recipes.

The program should be menu driven to allow the users to choose one of the following tasks:
1. Create or load a user profile
2. Edit or delete a user profile
3. View user profile
4. Generate recipe recommendations for the session
5. View user meals and generate health information
6. Exit the program

Here are the details of each choice that the user will be able to perform in the program:

1. Create or load a user profile
When the user chooses 1, the program should perform the following tasks:
a. Check if the file 'userInformation.txt' exists, if it does not exist (i.e., you are running the program for the first time ever), create a new user profile by asking the user to enter the following fields to the userInformation.txt file in append mode, like the following sample input:

Join Date (string)	Name (string)	Height (float)	Weight (float)	Year of birth (integer)	Age (integer)	BMI (float)	BMI range (string)	BMR (float)	Activity (integer)	TEE (float)	Allergies (list of strings)
17/10/2020, 05:38:26	Maryam	1.58	61	1992	28	24.44	healthy	1399.82	2	1924.75	almond, oats

- From the year of birth, calculate the user age. Asking the user to enter their date of birth and calculating the exact age in years is a bonus.
- Using the height and weight information, calculate the user's BMI using the following equation:

$$BMI = \frac{Weight\ (in\ KG)}{Height^2\ (in\ meters)}$$

- Based on the calculated BMI value, find out the user's BMI range by using the following table:

BMI value	Range
< 18.5	Underweight
$18.5 \leq$ BMI < 24.9	Normal
$25.0 \leq$ BMI ≤ 29.9	Overweight
≥ 30.0	Obese

- The gender should be a string that corresponds to either female or male only.
- Using the user height, weight, and gender, you can calculate the Basal Metabolic Rate (BMR) using the following equation:
  - **For females:** BMR = 655.1 + (9.563 x weight) + (1.85 x height x 100) - (4.676 x age).
  - **For males:** BMR = 66.47 + (13.75 x weight) + (5.003 x height x 100) - (6.755 x age).
- The user should choose the number that corresponds to their activity level based on the following table:

Number	Activity level	Value
1	Little/No exercise	1.2
2	Light exercise	1.375
3	Moderate exercise (3-5 days/week)	1.55
4	Very active (6-7 days/week)	1.725
5	Extra active (very active & physical job)	1.9

- Based on the user's chosen activity level, you can use the activity level value to calculate the Total Energy Expenditure (TEE) with the following equation: TEE = BMR x activity level value.
- Read the file Ingredients.csv, which contains a list of all the ingredients from all the recipes in this project and show this list to the user as follows: Ingredient number \t Ingredient name.
- Ask the user to enter the number of the ingredient that they are allergic to. The user should keep entering the number of the ingredient or enter -1 to stop entering ingredients. If the user is not allergic to anything, they can just enter -1 and the field Food allergies will be empty (empty allergies are allowed, just use '-' to refer to empty allergies).
- Once you compute all the user information, store it in the userInformationFile.txt in append mode using the following order (with \t as a separator):
  Joindate\tName\tHeight\tWeight\tYearofBirth(or DOB for
  bonus)\tAge\tGender\tBMI\tBMIRange\tActivity\tBMR\tTEE\tAllergies\tEditDate
- Get the current date and time as a variable, called Joindate and convert it to string, then write it to the userInformation.txt file.
- For now, keep the field EditDate empty, you will fill it later.

b. If the file 'userInformation.txt' exists, ask the user if they created a profile before. If the user answers no (n or N), show the user all the steps in (a). However, if the user answers with yes (Y or y) (i.e., they did create a profile before and the information exists in the file userInformation.txt), perform the following:
  - Ask the user to enter the name that they used to register their profile.

- Open the file userInformation.txt in read mode, and search for the user information using the name that the user provided.
- If the user makes a spelling mistake, keep asking the user to enter their name until it is found in the file.
- If the user name is found successfully in the userInformation.txt file, load it in the program (i.e., return the information to the user).

2. Edit or delete a user profile

When the user chooses 2, the first thing that it should do is to check whether the user information is loaded in the program (i.e., check if the user information is passed to the function that generates recipe recommendations). If the user information is passed to the function (i.e., the user chose option 1 before choosing option 2), the program should show the user the following menu:

return val by call

load_profile()

    Hello (user name)

    You can perform one of the following operations:
1) Delete your profile
2) Edit your profile

a. If the user chooses 1, perform the following subtasks to delete a user profile:
1- Search for the user profile in the file userInformation.txt using the user name in read mode; once you find the user profile (i.e., the line that contains all the user information), pass it to a function that deletes the user information.
2- The function should create a temporary file called temp.txt in write mode and search the file userInformation.txt in read mode for the user that you want to delete (the user returned by the previous step). Write all the content of the file userInformation.txt in the file temp.txt so long as the user is not there, in other words, do not write (delete) the user. The end of this process should yield a file called temp.txt that does not include the user.
3- Use the os module to delete the file userInformation.txt, then use it to rename the file temp.txt as userInformation.txt.
4- Print to the user a message stating that the record was deleted successfully.
5- If the file userInformation.txt contains just one user profile and you chose to delete this user, delete the file.
6- Make sure you also delete or clear the contents of the variable userInformation when you return to the main menu after successfully deleting the user profile.

b. If the user chooses 2, perform the following subtasks to edit a user profile:

Show the user the following menu, which displays the records that they can edit:

Hello (user name)

These are the fields that you can edit in your profile:

1) Name
2) Year of birth (or Date of Birth for bonus)
3) Gender
4) Height (m)
5) Weight (Kg)
6) Activity level
7) Food allergies

1- If the user enters the number 1, perform the following:
1) Ask the user to enter their new user name.

2) Create a new record of user information that replaces the old name with the new name but keeps all the other fields the same (use string concatenation to do this).

3) Add the edit date to the end of the new user information record, this date corresponds to the date and time in which the user edited their profile.

4) Then, call a function that takes the old user information and the new user information.

5) The function should read the contents of the file userInformation.txt and write it to the file temp.txt, at the same time, search for the old user information, if found, replace it with the new user information and write it to the file temp.txt.

6) Just like what you did in deleting a file, delete the file userInformation.txt, then rename the file temp.txt to userInformation.txt.

7) Make sure you also delete or clear the contents of the variable userInformation when you return to the main menu after successfully deleting the user profile.

2- If the user selects the number 2, perform the following:

1) Ask the user to enter their new year or date of birth, using this information, re-calculate the user age, the BMR, and the TEE.

2) Create a new record of user information that replaces the old year of birth or date of birth with the new information, replace the age with the new age, replace the BMR with a new BMR calculated from the new age, and replace the TEE with a new one from the new BMR. the other fields the same (use string concatenation to do this).

3) Repeat steps 3) to 7) from (a).

3- If the user selects the number 3, perform the following:

1) Ask the user to enter their new gender. Using this information, re-calculate the BMR and TEE.

2) Create a new record of user information that replaces the old gender with the new gender, replace the BMR with a new BMR calculated from the new gender, and replace the TEE with a new one from the new BMR. the other fields the same (use string concatenation to do this).

3) Repeat steps 3) to 7) from (a).

4- If the user selects the number 4, perform the following:

1) Ask the user to enter the new height. Using this information, re-calculate the BMI, the BMI-range, the BMR, and the TEE.

2) Create a new record of user information that replaces the old height with the new height, replace the BMI with the new BMI calculated from the new height, replace the BMI range with a new range calculated from a new BMI, replace the BMR with a new BMR calculated from the new height, and replace the TEE with a new one from the new BMR. the other fields the same (use string concatenation to do this).

3) Repeat steps 3) to 7) from (a).

5- If the user selects the number 5, perform the following:

4) Ask the user to enter the new weight. Using this information, re-calculate the BMI, the BMI-range, the BMR, and the TEE.

5) Create a new record of user information that replaces the old weight with the new weight, replace the BMI with the new BMI calculated from the new weight, replace the BMI range with a new range calculated from a new BMI, replace the BMR with a new BMR calculated from the new weight, and replace the TEE with a new one from the new BMR. the other fields the same (use string concatenation to do this).

6) Repeat steps 3) to 7) from (a).

6- If the user selects the number 6, perform the following:

1) Ask the user to enter the new activity level, just like the way you did when the user creates anew profile. Using this information, re-calculate the TEE.

2) Create a new record of user information that replaces the old activity level with the new one, and replace the TEE with the new TEE but keeps all the other fields the same (use string concatenation to do this).

3) Repeat steps 3) to 7) from (a).

7- If the user selects the number 7, perform the following:

4) Ask the user to enter the new list of allergies, just like the way you did when the user creates anew profile.

5) Create a new record of user information that replaces the old allergies with the new allergies but keeps all the other fields the same (use string concatenation to do this).

6) Repeat steps 3) to 7) from (a).

3. View user profile

When the user chooses 3, the first thing that it should do is to check whether the user information is loaded in the program (i.e., check if the user information is passed to the function that generates recipe recommendations). If the user information is not passed to the function (i.e., the user did not chose option 1 before choosing option 3), call the function that allows the user to either load or create a user profile (option 1 from the main menu). Otherwise, perform the following tasks:

- Open the file userInformation.txt in read mode and retrieve (i.e., read) and print the following user information:
    - User name.
    - Join date
    - Edit date (if available, don't show it if it doesn't exist)
    - Year of birth or date of birth as bonus.
    - Age.
    - Height (m).
    - Weight (Kg).
    - Gender.
    - Body Mass Index (BMI).
    - BMI range.
    - Basal Metabolic Rate (BMR).
    - Total Energy Expenditure (TEE).

4. Generate recipe recommendations for the session

When the user chooses 4, the first thing that it should do is to check whether the user information is loaded in the program (i.e., check if the user information is passed to the function that generates recipe recommendations). If the user information is passed to the function (i.e., the user chose option 1 before choosing option 2), the program should show the user the following menu:

Hello (user name)

You can perform the following tasks:

1) Generate recipes randomly
2) Generate recipes randomly based on your caloric needs
3) Generate recipes randomly based on food allergies
4) Generate recipes randomly based on caloric needs and food allergies

a. If the user chooses 1, perform the following subtasks to generate recipes randomly:

1- Ask the user how many recipes they would like to generate. A user cannot generate less than 4 recipes or more than 6 recipes per session. Check the entered number and make sure that it is no less than 4 and no more than 6.

2- Based on the entered number of recipes, open the file recipes.csv in read mode and randomly read x number of recipes. E.g., if the user wants to generate 5 meals, you should read 5 random recipes from the file. To do this, use the random module to randomly generate a line number, and read this line number from the file.

3- When you get a random recipe, show the user the recipe title and ask the user if they are interested in seeing the recipe. If the user answers yes (Y or y), show the full recipe by printing the following information:
   o The recipe name.
   o The total calories.
   o The serving size.
   o The prep time.
   o The cook time.
   o The total time.
   o Bonus: The recipe image.
   o The ingredients.
   o The method.

4- If the user does not like the randomly generated recipe and answer the previous question with no (N or n), keep generating random recipes until they answer with yes. A recipe is only counted if the user answers with yes, otherwise, it does not count (i.e., don't change the counter that keeps track of the number of generated recipes).

5- When the user generates all the recipes for the session, save the recipe information to a file as follows:
   a. Create a file called username-recipes.txt, where username is the user's name.
   b. In this file, store the session time (i.e., session date) for the user, the randomly generated recipe names, the total calories from all the recipes, and the average number from all the generated recipes.

b. If the user chooses 2, perform the following subtasks to generate recipes randomly based on caloric needs:
   1- Repeat step 1 from (a).
   2- Using the information from the file userInformation.txt, retrieve the user TEE and divide this value by the number of recipes that the user wants to generate. The number you obtain here (call it caloric needs) should be used as a condition to randomly generate recipes.
   3- Open the file recipes.csv in read mode and only read recipes (i.e., lines) where the number of calories per serving is less than or equal the caloric needs of the user.
   4- Keep these recipes in a list and randomly read x number of recipes. E.g., if the user wants to generate 5 meals, you should read 5 random recipes from the list. To do this, use the random module to randomly generate a line number from the list, and read this line number from the file.
   5- Repeat steps (3-5) from (a).

c. If the user chooses 3, perform the following subtasks to generate recipes randomly based on food allergies:
   1- Using the information from the file userInformation.txt, check if the user provided a list of allergies from certain meal ingredients. If the field allergies is empty (i.e., it

contains -), show the user a message that they cannot generate meals based on food allergies, and show them the menu again that allows them to generate meals.

2- If the user provided some food allergies, repeat step 1 from (a).

3- Then, retrieve the user allergies and to use it as a condition to randomly generate recipes. If a user has allergies, the program should avoid generating recipes with ingredients that contain such allergies.

4- Open the file recipes.csv in read mode and only read recipes (i.e., lines) where the ingredients <u>do not</u> include food allergies provided by the user.

5- Repeat steps (4-5) from (b).

d. If the user chooses 4, perform the following subtasks to generate recipes randomly based on their caloric needs and food allergies:

1- Repeat all the steps from (b) and (c) to filter recipes based on caloric needs and allergies from the file recipes.csv. If the user has no allergies, just show them the menu again and tell them that it is not possible to filter based on allergies if they do not have any allergies.

e. As a last note here, after the user successfully generates and saves recipes, if they go back to the main menu and choose 4 (i.e., they want to generate more recipes), show a message telling the user that they generated their recipes for the current session. In other words, a user can only generate recipes once per one program run, they can generate recipes again if they exit and run the program again. This is what is meant by a session. To do this, return the user choice for the number of meals that they wanted to generate, and use it in your main function to check for this condition after generating meals.

5. View user meals and generate health information

When the user chooses 5, the first thing that it should do is to check whether the user information is loaded in the program (i.e., check if the user information is passed to the function that generates recipe recommendations). If the user information is not passed to the function (i.e., the user did not chose option 1 before choosing option 4), call the function that allows the user to either load or create a user profile (option 1 from the main menu). Otherwise, perform the following tasks:

- First, check the file username-recipes.txt exists, if the file does not exist, ask the user to generate recipes to view their health information, and call the function that allows the user to do so (i.e., call the function that shows choice 2 from the main menu).

- If the file username-recipes.txt exists, open it in read mode and retrieve (i.e., read) all the information that is stored in this file.

- Ask the user to enter the number of servings per meal that they would like to perform the health analysis for (integer value).

- Write a function that opens the file recipes.csv in read mode and uses the randomly generated recipe names of the user to search for their details from the file recipes.csv. The function should return the details of all the randomly generated recipes by the user.

- Pass the information of each recipe to a function that prints all the recipe details like step 3 from (a).

- Since the user provided the number of servings per recipe, the calories per meal retrieved from the file recipes.csv should be updated as follows: meal calories = meal calories x serving size.

- For each printed meal, check if the calories per serving size cause a caloric surplus or caloric deficit. A caloric surplus happens when the calories per serving for a meal is greater than TEE of the user when divided on the number of meals in a session. Print this status for every meal in the generated report.

- Add all the new values of calories to a list for later usage.

- After printing the details of every recipe that the user generated from every session, show the total and average number of calories per session; such information can be found in the file username-recipes.txt. Remember to update the values by multiplying them with the serving size that the user provides.
- Check if the user experienced an overall trend of calorie surplus or calorie deficit based on the meals that they generated. To do this, check the following condition:
    - Compare the total calories for all the meals per session (i.e., the sum of the calories in the list of meal calories) with the user's TEE. If the total calories are more than TEE, then the user is experiencing a calorie surplus.
    - Otherwise, the user is experiencing a calorie deficit.
- Print the overall trend of the caloric status of the user (surplus or deficit) in each session.
- Use matplotlib to plot the list that contains the calories of each recipe from all the sessions using a pie chart plot.

6. **Exit the program**
When the user chooses 6, the program should end. Otherwise, your program should be user friendly and always loop back to the main menu after completing a choice.

Note:
- For any menu driven in your code: If the user entered any other choice (out of range), keep asking the user to enter a valid menu choice.

Evaluation criteria
The following evaluation criteria items must be met in your submitted project, which include:
   1- **Correctness of application implementation and results**
      Correct implementation of all the requirements using appropriate Python programming techniques. Testing should be done to ensure that the results produced by the program match the expected results.
   2- **Program Structure and Efficiency**
      - Use of functions and re-use of functions whenever possible.
      - Use of arrays.
      - Use of loops.
      - Use of files.
      - Handling user input validations: e.g. menu choices, values entered …etc.
      - Code quality:
          - Using meaningful variable names.
          - In-program comments, where necessary.
          - Proper indentation and spaces.
          - Informative input prompts.
   3- **Team Cooperation and Coordination**
      Evident and clear role of each member (use the attached form).
   4- **Documentation**
      - Cover Page with necessary information: QU logo, Students' names and registration ID, Department name, Term and Project title.

- Short report that includes a brief description of your application and a list (better to be in a tabular structure) of the functions used with a short description of each function.
- Evidence of correct execution and results using snapshots of program execution (sample runs).
- Code listing.

5- Discussion and demo of the project

Students should demo their program, explain how it works and answer questions about the implementation. Each team member must contribute to the discussion and demo.

The grade for the project is **15 points** distributed as follows:

Criterion	Weight
Deductions: • Plagiarism detected, Compilation error (-100%) • Late submission 1 day (-25%), 2 days (-50%) of whatever student earns • Submission format and following variable naming conventions (-10%) • Using global variables (-10%) • Improper use of functions (void parameters) (-15%) • **Not using 1-D Array and Files (-50%)** • **Failure to demonstrate project correctly (-50%)**	-100%
Implement each function correctly: 1. Create or load user profile (10%) 2. Edit or delete user profile (10%) 3. View user profile (10%) 4. Generate recipe recommendations for the session (10%) 5. Based on user caloric needs (5%) 6. Based on food allergies (5%) 7. Based on both caloric needs and food allergies (5%) 8. View user meals and generate health information (10%)	65%
Main function integrates all the other functions correctly saving to file and validation.	10%
Comments are placed correctly in the code and describe the code sufficiently and user input validation.	10%
Presentation and Documentation.	15%

Guidelines and submission instructions

- Each team will have 2 members. You should give me the names of the team members, and a "nickname" of the team in this week by email.
- Team members are required to meet regularly online for discussion and workload distribution.
- Submission for the 1st phase pf project is due **November 5th, 2020 by 11:59 PM**
- Submission for the 2nd phase pf project is due **November 22nd, 2020 by 11:59 PM**
- The role of each team member must be clearly and briefly described using the attached cooperative project evaluation form CooperativeProjectEvaluationForm.pdf. Each team member should submit a copy of this form signed digitally by both members of that team.
- **Note: Your project will not be graded if these forms are not submitted.**

- You are required to Submit following items:
  - Turn in: 1) your programs (as .py files **not** .docx); and 2) your report (refer to the report template on blackboard) along with a sample run of your application electronically on blackboard.
  - For the electronic submission, submit a zipped folder that includes all the requested items. Adopt the following naming convention for your folder:
  - **GENG106-Project1-[*nickname of the team*]-B52**
  - Each team should submit only one electronic submission, but two cooperative evaluation forms.
- **It is the right of the instructor to use any way of testing the student in the discussion and demo session, and according to that in some cases (100%) graded project may be down to (-100%) graded project.**

A sample output of the program can be shown below:

```
+--------------------------------------------------------+
|+------------------------------------------------------+|
|| Welcome to Recipeosity, your personal meal planner   ||
|+------------------------------------------------------+|
+--------------------------------------------------------+
In this system, you can perform one of the following tasks
1. Create or load a user profile
2. Edit or delete a user profile
3. View user profile
4. Generate recipe recommendations for the session
5. View user meals and generate health information
6. Exit the program

To perform any of the previous functions, enter its number:
```

Figure 1 Sample snapshot from the main menu

```
Hello Maryam
You can perform one of the following operations:
1) Delete your profile
2) Edit your user profile

Which action would you like to perform? 2
Hello Maryam
These are the fields that you can edit in your profile:
1) Name
2) Date of Birth
3) Gender
4) Height (m)
5) Weight (Kg)
6) Activity level
7) Food allergies
```

Figure 2 Sample snapshot from the edit profile menu

```
How many recipes would you like to generate?:5
Randomly generated Recipe # 1
============================
Hearty Oatmeal Loaf Recipe
100.0 calories

Are you interested in viewing this recipe? (Y/N):Y

~Recipe Name~
Hearty Oatmeal Loaf Recipe

~Total Calories~
100.0

~Total Servings~
12
```



```
~Ingredients~
- apple juice
- milk
- brown sugar
- margarine
- salt
- flour
- oat
- yeast

~Directions~
1-Place ingredients into the pan of the bread machine in the order suggested by the manufacturer
2-Select White Bread or Basic cycle, and Start

Do you want to save this recipe? (Y/N):
```

Figure 3 Sample snapshot from generating meal based on calories and food allergies

```
+--------------------+
|+----------------+|
|| Maryam's Profile ||
|+----------------+|
+--------------------+
Join Date: 17/10/2020, 04:11:50
Date of Birth: 12/5/1992
Age: 28
Gender:Female
Height (m): 1.58
Weight (Kg): 61.0
Body Mass Index (BMI): 24.44
BMI range: healthy
Basal Metabolic Rate (BMR): 1399.82
Activity Level: Light exercise
Total Energy Expendature (TEE): 1924.75
Food allergies:almond,oat
```

Figure 4 Sample snapshot from printing user profile


```
Overall, Maryam shows that there is a calorie surplus in the 6 meal calories from record # 1
The total calories in the meals is  2948.0 calories
The average calories of the meals is  491.32 calories
```

Calorie distribution of the meals from session #1(total is 2948.0 calories)
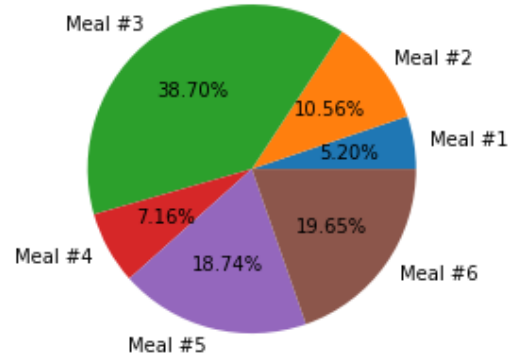
```
Done viewing meal records of Maryam
```

Figure 5 Sample snapshot from showing health statistics with serving size = 4