3. WRITTEN RESPONSES

3 a.

3.a.i.

My program aims to provide information on how to recreate cuisines worldwide in your kitchen. and cures the problem of not being able to decide what to eat.

3.a.ii.

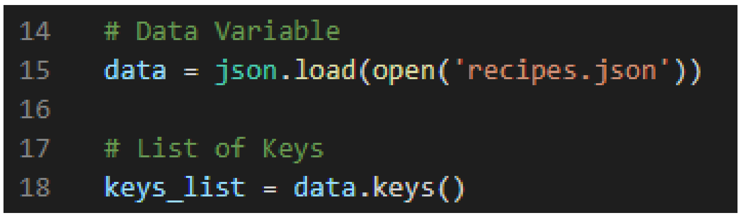
The video illustrates both procedures described above, 1 - Find Information About Food, and 2 - Find a random food. The video initially illustrates a test case for the first procedure; you enter the food of your choice and find out information about the food based on the choices. After, I exit the first procedure into the main menu and show the functionality for the second procedure. I answered the given questions based on my food preferences in the second procedure. Lastly, I show how you can use both procedures in harmony as after you're recommended food; you can find the recipe to replicate it.

3.a.iii.

For procedure 1, after inputting what you want to know about the food you selected, it outputs that information. In the video, I entered 'Baked Shrimp Scampi,' and when I typed '2' (the menu displays '2 -- Ingredients', so I know what '2' is), it gave me the ingredients for the food. For procedure two, after inputting my answers to the preferences, it outputted a food recommendation. I then demonstrate the unique feature that lets you misspell the name of the food, which then outputs the most similar food name and asks you if that was the food you meant to type.

3 b.

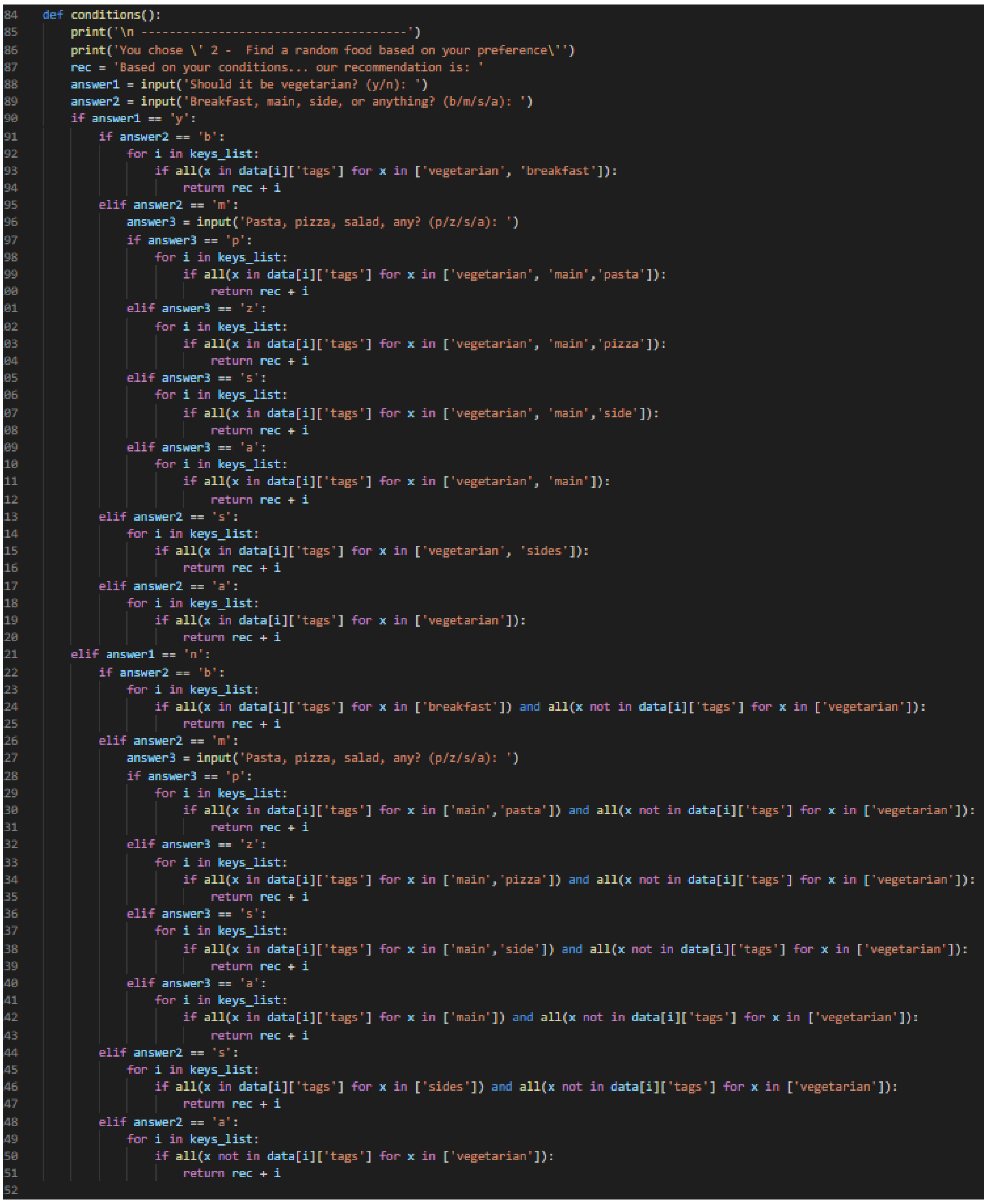
3.b.i.



```
# Data Variable
data = json.load(open('recipes.json'))

# List of Keys
keys_list = data.keys()
```

3.b.ii.



```python
def conditions():
    print('\n -------------------------------------')
    print('You chose \' 2 -  Find a random food based on your preference\'')
    rec = 'Based on your conditions... our recommendation is: '
    answer1 = input('Should it be vegetarian? (y/n): ')
    answer2 = input('Breakfast, main, side, or anything? (b/m/s/a): ')
    if answer1 == 'y':
        if answer2 == 'b':
            for i in keys_list:
                if all(x in data[i]['tags'] for x in ['vegetarian', 'breakfast']):
                    return rec + i
        elif answer2 == 'm':
            answer3 = input('Pasta, pizza, salad, any? (p/z/s/a): ')
            if answer3 == 'p':
                for i in keys_list:
                    if all(x in data[i]['tags'] for x in ['vegetarian', 'main','pasta']):
                        return rec + i
            elif answer3 == 'z':
                for i in keys_list:
                    if all(x in data[i]['tags'] for x in ['vegetarian', 'main','pizza']):
                        return rec + i
            elif answer3 == 's':
                for i in keys_list:
                    if all(x in data[i]['tags'] for x in ['vegetarian', 'main','side']):
                        return rec + i
            elif answer3 == 'a':
                for i in keys_list:
                    if all(x in data[i]['tags'] for x in ['vegetarian', 'main']):
                        return rec + i
        elif answer2 == 's':
            for i in keys_list:
                if all(x in data[i]['tags'] for x in ['vegetarian', 'sides']):
                    return rec + i
        elif answer2 == 'a':
            for i in keys_list:
                if all(x in data[i]['tags'] for x in ['vegetarian']):
                    return rec + i
    elif answer1 == 'n':
        if answer2 == 'b':
            for i in keys_list:
                if all(x in data[i]['tags'] for x in ['breakfast']) and all(x not in data[i]['tags'] for x in ['vegetarian']):
                    return rec + i
        elif answer2 == 'm':
            answer3 = input('Pasta, pizza, salad, any? (p/z/s/a): ')
            if answer3 == 'p':
                for i in keys_list:
                    if all(x in data[i]['tags'] for x in ['main','pasta']) and all(x not in data[i]['tags'] for x in ['vegetarian']):
                        return rec + i
            elif answer3 == 'z':
                for i in keys_list:
                    if all(x in data[i]['tags'] for x in ['main','pizza']) and all(x not in data[i]['tags'] for x in ['vegetarian']):
                        return rec + i
            elif answer3 == 's':
                for i in keys_list:
                    if all(x in data[i]['tags'] for x in ['main','side']) and all(x not in data[i]['tags'] for x in ['vegetarian']):
                        return rec + i
            elif answer3 == 'a':
                for i in keys_list:
                    if all(x in data[i]['tags'] for x in ['main']) and all(x not in data[i]['tags'] for x in ['vegetarian']):
                        return rec + i
        elif answer2 == 's':
            for i in keys_list:
                if all(x in data[i]['tags'] for x in ['sides']) and all(x not in data[i]['tags'] for x in ['vegetarian']):
                    return rec + i
        elif answer2 == 'a':
            for i in keys_list:
                if all(x not in data[i]['tags'] for x in ['vegetarian']):
                    return rec + i
```

3.b.iii.

The name of the list being used is **keys_list**.

3.b.iv.

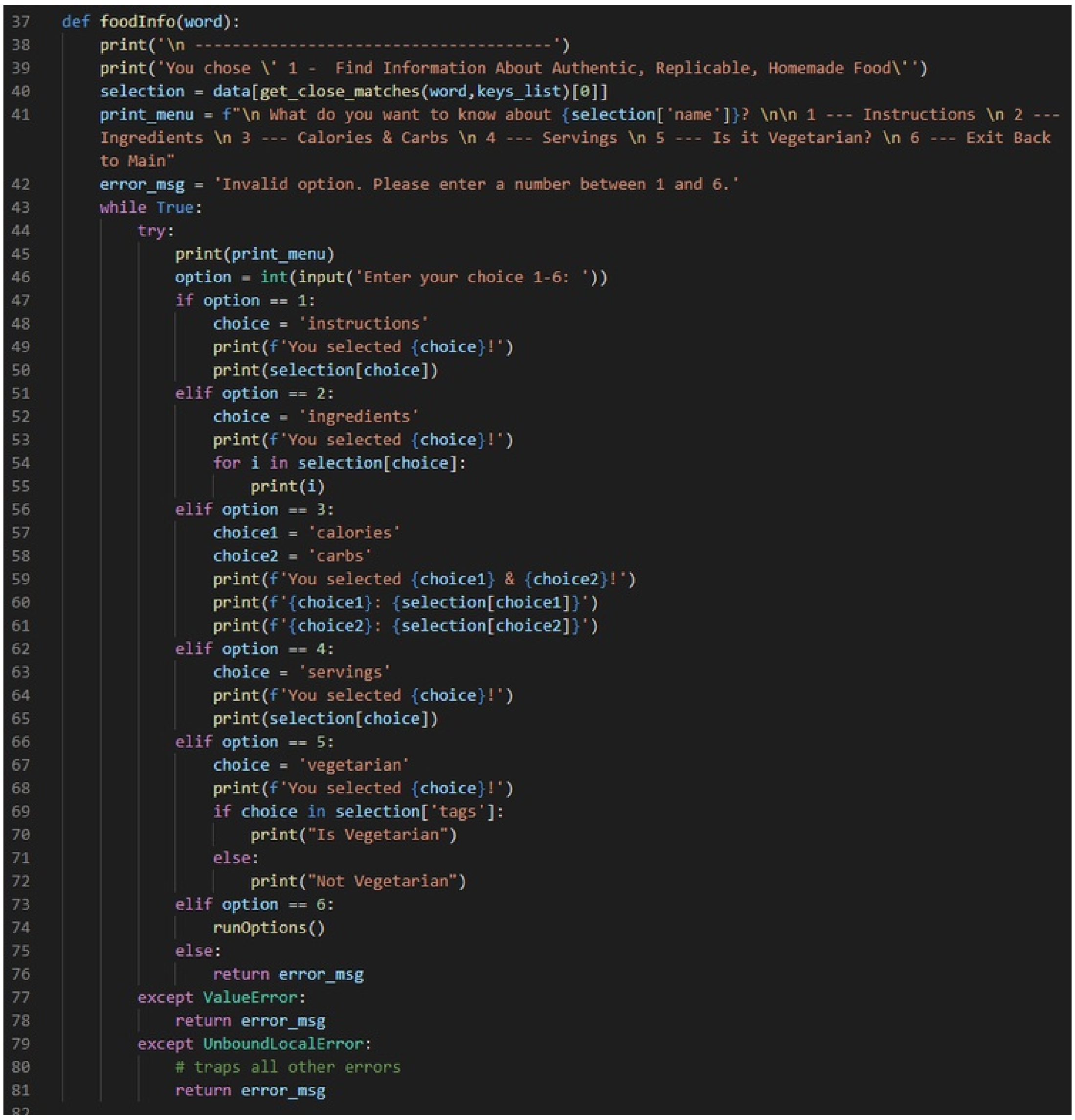
In my program, **keys_list**
 is a list that holds all the names of the food items from 'data.' 'data' is a variable that contains the collection of all the food items, which all have their own attributes like ingredients and instructions to prepare, calories, and dietary type in a nested dictionary.

3.b.v.

I used a JSON file to contain all the information about my food, as each food had many nested elements. Each food name is the key to its dictionary in recipes.json. Getting hold of these keys was done by calling the keys to function on this collection type which returned a list.
keys_list
 is used to loop through all the food names and use it as an index for 'data' and find out whether certain information exists within the food's dictionary.

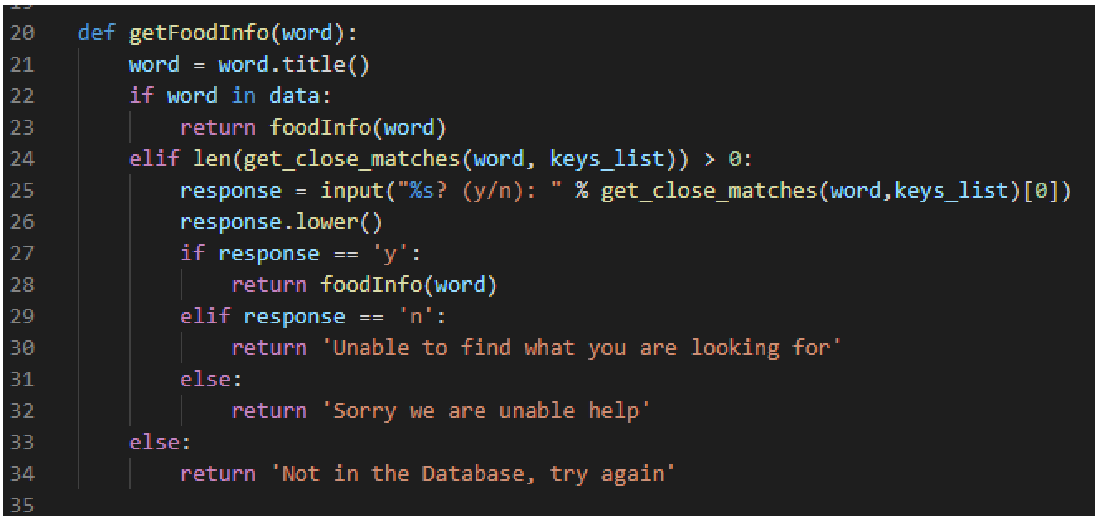
3 c.

3.c.i.



```python
def foodInfo(word):
    print('\n -------------------------------------')
    print('You chose \' 1 -  Find Information About Authentic, Replicable, Homemade Food\'')
    selection = data[get_close_matches(word,keys_list)[0]]
    print_menu = f"\n What do you want to know about {selection['name']}? \n\n 1 --- Instructions \n 2 --- Ingredients \n 3 --- Calories & Carbs \n 4 --- Servings \n 5 --- Is it Vegetarian? \n 6 --- Exit Back to Main"
    error_msg = 'Invalid option. Please enter a number between 1 and 6.'
    while True:
        try:
            print(print_menu)
            option = int(input('Enter your choice 1-6: '))
            if option == 1:
                choice = 'instructions'
                print(f'You selected {choice}!')
                print(selection[choice])
            elif option == 2:
                choice = 'ingredients'
                print(f'You selected {choice}!')
                for i in selection[choice]:
                    print(i)
            elif option == 3:
                choice1 = 'calories'
                choice2 = 'carbs'
                print(f'You selected {choice1} & {choice2}!')
                print(f'{choice1}: {selection[choice1]}')
                print(f'{choice2}: {selection[choice2]}')
            elif option == 4:
                choice = 'servings'
                print(f'You selected {choice}!')
                print(selection[choice])
            elif option == 5:
                choice = 'vegetarian'
                print(f'You selected {choice}!')
                if choice in selection['tags']:
                    print("Is Vegetarian")
                else:
                    print("Not Vegetarian")
            elif option == 6:
                runOptions()
            else:
                return error_msg
        except ValueError:
            return error_msg
        except UnboundLocalError:
            # traps all other errors
            return error_msg
```

3.c.ii.



```python
def getFoodInfo(word):
    word = word.title()
    if word in data:
        return foodInfo(word)
    elif len(get_close_matches(word, keys_list)) > 0:
        response = input("%s? (y/n): " % get_close_matches(word,keys_list)[0])
        response.lower()
        if response == 'y':
            return foodInfo(word)
        elif response == 'n':
            return 'Unable to find what you are looking for'
        else:
            return 'Sorry we are unable help'
    else:
        return 'Not in the Database, try again'
```

3.c.iii.

This procedure is one of the program's main functions, serving as the entry to finding information about the food the user is searching for. The foodInfo procedure prints a list of options: instructions, ingredients, calories & carbs, servings, and is it vegetarian. Each option has a corresponding number, and when the user picks a number in the prompt, it returns information on that option. FoodInfo gets called after getFoodInfo validates if the food item is in the database.

3.c.iv.

- Prompt the user to input a value as seen on the printed menu.
- Capture the entered value and check whether it's in the list of options.
    - If not: return error message and prompt the user to input a valid option.
    - If there is a match, print corresponding information to that match and send the user back to the start of the menu, prompt, selection loop so they can find other information about the food.
- Keep going through the loop until user inputs exit value as seen on the printed menu.

3 d.

3.d.i.

First call:

I wrote a method that checks if the inputted food item is present in the database, recipes.json, or not and calls the foodInfo procedure accordingly. The test cases are based on the conditions of what food item is selected and what option about the food item is chosen.
The first call occurs when user inputs "Greek Salad," a valid food item, calling foodInfo("Greek Salad"), and selects "Is it Vegetarian."

Second call:

When a user inputs a not exact but a close match, "Lintel Sald", the program prompts with a close matched food item found, "Lentil salad? (y/n)". The second call occurs when the user inputs 'y' calling foodInfo("Lentil salad") and selects "Is it Vegetarian".

3 d.ii.

Condition(s) tested by first call:

For the food item, "Greek Salad", the method checks if 'vegetarian' tag exists within the "Greek Salad" dictionary.

Condition(s) tested by second call:

For the food item, "Lentil salad", the method checks the same condition, if 'vegetarian' tag exists within the "Lentil salad" dictionary.

3.d.iii.

Results of the first call:

The tag 'vegetarian' does not exist within the "Greek Salad" dictionary. The condition is false; therefore the program broadcasts 'Not Vegetarian'.

Results of the second call:

The tag 'vegetarian' exists within the "Lentil Salad" dictionary. The condition is true; therefore the program broadcasts 'Is Vegetarian'.