RECIPE ORGANIZER

NAME: LIYA PRINCE

ROLL NO: 50

BTECH COMPUTER SCIENCE

DATE:17-07-2024

INTRODUCTION

- Project Overview: Recipe organizer

The "Recipe Organizer" project in C provides a foundational structure for managing and interacting with recipes. It demonstrates basic file handling, input/output operations, and efficient data structure management. Developing a system to organize and manage personal recipes involves creating a comprehensive and user-friendly platform that stores and categorizes recipes, tracks ingredients, and facilitates meal planning. This system would include a robust database for storing detailed recipe information, such as ingredients, instructions, cooking times, and serving sizes, along with a user interface for easy input and editing. It would also feature an inventory management component to track ingredient quantities, generate automated shopping lists, and adjust for different serving sizes. Meal planning functionality would be integrated with a calendar to schedule recipes and manage nutritional information, while advanced search and categorization capabilities would allow users to quickly find and organize recipes based on various criteria.

- Problem Statement:

Develop a system to organize and manage personal recipes, including ingredients, instructions and meal planning. This system should store recipes with detailed information, manage ingredient inventories, and facilitate meal planning. This program aims to streamline recipe management, enhance meal preparation efficiency and promote better organization.

- Objectives:

The objectives of developing a recipe organizer in C programming are as follows:

1. Recipe Storage: Create a system to store and retrieve detailed recipe information, including ingredients, instructions, cooking times, and serving sizes.

2. Ingredient Management: Develop functionality to manage ingredient inventories, allowing users to track the quantities of ingredients they have on hand and update them as needed.

3. Meal Planning: Implement a meal planning feature that enables users to schedule recipes for specific days, facilitating organized meal preparation.

4. User Interface: Design a user-friendly interface that allows for easy input, editing, and retrieval of recipe data, ensuring a smooth user experience.

5. Performance Optimization: Optimize the program for efficiency and performance, ensuring quick access to data and smooth operation even with a large number of recipes and ingredients.

STYSTEM REQUIREMENTS

- Hardware Requirements

  -Any basic computer or laptop with a compatible operating system like Windows, macOS, Linux, etc.
  -Minimum 1 GB RAM for better performance.
  -At least 10 MB of free disk space to store the program and any related files.

- Software Requirements:

  *Operating System:
  -Compatible with Windows, macOS, or Linux distributions.
  * C Compiler:
  1.GCC: Available on Linux and Windows
  2.TURBO C/C++: An older but widely used one for beginners on Windows.
  3.Code: Blocks: A user-friendly IDE that supports various compilers.
  4.Dev-C++: Another simple IDE for Windows users.

DESIGN AND DEVELOPMENT

- Description of the program logic:

1. Data Structures and Global Variables:

   - Recipe Structure: Defines a recipe with fields for the recipe name, ingredients, and instructions.
   - recipes Array*: Holds up to MAX_RECIPES recipes.

2. addRecipe() Function:
   - Checks if the recipe list has reached its maximum capacity. If so, it prints an error message and exits.
   - Prompts the user to input the recipe name, ingredients, and instructions.

3. viewRecipe() Function:
   - Prompts the user to enter the name of the recipe they want to view.
   - Searches through the recipes array for a recipe with a matching name.

4. main() Function:
   - Implements a continuous loop displaying a menu with options to add a recipe, list recipes, view a specific recipe, or exit the program.
   - Reads user input to determine which option was selected.
   - Calls the corresponding function based on user choice and handles invalid input by printing an error message.
   - Uses getchar() to consume any leftover newline character from the input buffer after reading the choice.

This program provides a basic recipe management system with essential functionalities for adding, listing, and viewing recipes. It demonstrates fundamental C programming concepts such as structures, arrays, string handling, and control flow.

- Pseudocode

```
BEGIN
   DEFINE MAX_RECIPES AS 100
   DEFINE MAX_NAME_LENGTH AS 50
   DEFINE MAX_INGREDIENTS AS 20
   DEFINE MAX_INSTRUCTION_LENGTH AS 500
 STRUCT Recipe
      STRING name [MAX_NAME_LENGTH]
      STRING ingredients [MAX_INGREDIENTS]
      STRING instructions [MAX_INSTRUCTIONS]
   END STRUCT
   FUNCTION addRecipe
      IF recipe_count >= MAX_RECIPES THEN
         PRINT "Recipe list is full!"
         RETURN
 CREATE newRecipe AS Recipe
 PRINT "Enter recipe name:"
      READ newRecipe.name
      REMOVE newline from newRecipe.name

      PRINT "Enter ingredients:"
      READ newRecipe.ingredients
      REMOVE newline from newRecipe.ingredients

      PRINT "Enter instructions:"
      READ newRecipe.instructions
      REMOVE newline from newRecipe.instructions

      PRINT "Recipe added successfully!"
   END FUNCTION

   FUNCTION listRecipes
      IF recipe_count IS 0 THEN
         PRINT "No recipes available."
         RETURN
 FUNCTION viewRecipe
      DEFINE searchName AS STRING OF MAX_NAME_LENGTH
```

```
PRINT "Enter recipe name to view:"
    READ searchName

  FUNCTION main
    DEFINE choice AS INTEGER

    WHILE TRUE DO
      PRINT "Recipe Manager"
      PRINT "1. Add Recipe"
      PRINT "2. List Recipes"
      PRINT "3. View Recipe"
      PRINT "4. Exit"
      PRINT "Enter your choice:"
      READ choice

      SWITCH choice
        CASE 1:
          CALL addRecipe()
          BREAK
        CASE 2:
          CALL listRecipes()
          BREAK
        CASE 3:
          CALL viewRecipe()
          BREAK
        CASE 4:
          EXIT PROGRAM
        DEFAULT:
          PRINT "Invalid choice. Please try again."
      END SWITCH
    END WHILE

    RETURN 0
  END FUNCTION
END
```

TESTING AND RESULTS

- Test Cases

Test Case 1: Adding a Recipe
Input:
Select option 1 (Add Recipe)

Expected Output:

```
EXPLORER
> C
> OUTLINE
> TIMELINE

PROBLEMS   OUTPUT   DEBUG CONSOLE   TERMINAL   PORTS

PS C:\Users\mayap\Desktop\c> cd "c:\Users\mayap\Desktop\c\" ; if ($?) { gcc recipeorganizer.c -o recipeorganizer } ; if ($?) { .\recipeorganizer }

1. Add a new recipe
2. View all recipes
3. View meal plan
4. Exit
Enter your choice: 1
Enter the name of the recipe: pancakes
Enter the number of ingredients: 3
Enter the ingredients (one per line):
flour
milk
eggs
Enter the instructions for your recipe:
Mix the ingredients and cook on a griddle.
Recipe added successfully!
```

Test Case 2: Listing Recipes

Input:

Select option 2 (List Recipes)

Expected Output:

```
1. Add a new recipe
2. View all recipes
3. View meal plan
4. Exit
Enter your choice: 2
Recipe List:
1. pancakes
```

Test Case 3: Viewing a Recipe

Input:

Select option 3 (View Recipe)

Expected Output:

```
1. Add a new recipe
2. View all recipes
3. View meal plan
4. Exit
Enter your choice: 3
Recipe List:
1. pancakes

Enter the number of the recipe you want to prepare: 1
Preparing meal with recipe: pancakes
Ingredients needed:
- flour
- milk
- eggs
Instructions:
Mix the ingredients and cook on a griddle.
```

Test Case 4: Exit

Input:

Select option 4 (Exit)

Expected Output:

```
1. Add a new recipe
2. View all recipes
3. View meal plan
4. Exit
Enter your choice: 4
Exiting....
PS C:\Users\mayap\Desktop\c>
```

CONCLUSION

- ## Summary:

The recipe organizer program is a straightforward C application designed to manage personal recipes. It allows users to add new recipes, list all existing recipes, and view details of a specific recipe. The program utilizes a Recipe structure to store key information such as the recipe's name, ingredients, and instructions. Users interact with the program through a text-based menu, where they can select options to input recipe data, display a list of recipes, or search for and review a particular recipe by name. The program handles up to 100 recipes, ensuring that it provides feedback when the recipe list is full or if a recipe is not found. It is built to offer an intuitive way to keep track of recipes, demonstrating basic operations and control flow in C programming.

- ## Future Enhancements:

Future enhancements for the recipe organizing program could include:

1. Persistent Storage: Implement file-based or database storage to save recipes permanently, allowing users to retain their recipes between program runs.

2. Search Functionality: Introduce advanced search options such as filtering by ingredients, meal type, or dietary restrictions to improve recipe retrieval.

3. User Interface: Develop a graphical user interface (GUI) or a web-based interface to make the program more user-friendly and accessible.

4. Meal Planning: Integrate features for meal planning and generate shopping lists based on selected recipes to streamline meal preparation.

5. Recipe Sharing: Implement options for users to share recipes with others via email or social media, or within a community of users.

6. Mobile Compatibility: Develop mobile versions of the application to allow users to manage recipes on-the-go.

Incorporating these enhancements would significantly expand the capabilities of the recipe organizing program, making it user friendly and aligned with modern digital tools, thereby offering a comprehensive solution for managing and sharing personal recipes.