University of Petroleum and Energy Studies
Course Code: CSEG1032

Diary Management System

(Major project)

Instructor: Dr. Tanu Singh

Submitted By: Shashwat Tripathi

SAP ID: 590022932

Course Name: Programming in C

Abstract

The *Diary Management System* is a console-base application developed in the C programming language

The program enables users to digitally store and manage daily diary entries in a simple and efficient manner. Each diary entry consists of a date, title, and content, all stored in a structured format inside a text file.

Also this system provides five core functionalities adding new entries, viewing all saved entries, searching for a specific entry, editing an existing entry, and deleting an entry.

Problem Definition

The problem is to design and develop a **digital diary management system in the C programming language** that offers a user-friendly, menu-driven interface to store and manage personal diary entries. Each entry should contain essential information such as the date, title, and content. The system must allow users to add new entries, view all stored entries, search for specific entries based on date, edit existing records, and delete entries permanently when required.

System Design

Algorithm: (for main)

Step 1:

Start the program.

Step 2:

Display the main menu with the following options:

1. Add a new diary entry
2. View all saved entries
3. Search an entry
4. Edit an entry
5. Delete an entry
6. Exit

Step 3:

Prompt the user to enter their choice.

Step 4:

Use a switch statement to call the appropriate function based on the user's choice.

Step 5:

Loop back to the menu (due to while(1)), unless the user chooses to exit.

Step 6:

Stop the program.

Algorithm for ADDentry()

1. Open the file **diary.txt** in append mode.
2. If the file fails to open, display an error and stop the operation.
3. Ask the user to enter:
    - Date
    - Title
    - Content of the diary entry
4. Write the entered data to the file in the format: date|title|content
5. Close the file.
6. Display a success message.

```c
// this function adds new entry into the text file diary.txt
void ADDentry() {
    FILE *fp = fopen("diary.txt", "a");
    struct Diary d;

    if (!fp) {
        printf("\nError: Unable to open diary file.\n");
        return;
    }

    printf("\nEnter the date (DD-MM-YYYY): ");
    fgets(d.date, sizeof(d.date), stdin);

    printf("Enter the title: ");
    fgets(d.title, sizeof(d.title), stdin);

    printf("Enter the content:\n> ");
    fgets(d.content, sizeof(d.content), stdin);

    fprintf(fp, "%s|%s|%s", d.date, d.title, d.content);
    fclose(fp);

    printf("\nEntry saved successfully.\n");
}
```

Algorithm for VIEWentry()

1. Open **diary.txt** in read mode.
2. If the file does not exist or fails to open, display *"No entries found"* and stop.
3. Read each entry from the file using formatted input.
4. For every entry retrieved:
    - Display the date
    - Display the title
    - Display the content
5. Close the file.

```c
// This fuction reads and diaplays all saved entries
void VIEWentry() {
    FILE *fp = fopen("diary.txt", "r");
    struct Diary d;

    if (!fp) {
        printf("\nNo entries found.\n");
        return;
    }

    printf("\n========== SAVED DIARY ENTRIES ==========\n");

    while (fscanf(fp, "%19[^|]|%49[^|]|%499[^\n]\n", d.date, d.title, d.content) != EOF) {
        printf("\n----------------------------------------\n");
        printf("Date  : %s", d.date);
        printf("Title : %s", d.title);
        printf("Entry :\n%s\n", d.content);
    }

    fclose(fp);
}
```

Algorithm for SEARCHentry()

1. Open **diary.txt** in read mode.
2. If the file does not exist, display an error and stop.
3. Ask the user to enter the date they want to search for.
4. Read entries one by one from the file.
5. If any entry matches the given date:
    - Display the date, title, and content
    - Mark entry as found
    - Stop searching
6. If no matching entry is found, display *"No entry exists for that date"*.
7. Close the file.

```c
// this function searches all saved entries by date
void SEARCHentry() {
    FILE *fp = fopen("diary.txt", "r");
    struct Diary d;
    char searchDate[20];
    int found = 0;

    if (!fp) {
        printf("\nNo entries available to search.\n");
        return;
    }

    printf("\nEnter the date to search (DD-MM-YYYY): ");
    fgets(searchDate, sizeof(searchDate), stdin);

    while (fscanf(fp, "%19[^|]|%49[^|]|%499[^\n]\n", d.date, d.title, d.content) != EOF) {
        if (strcmp(searchDate, d.date) == 0) {
            printf("\nEntry Found:\n");
            printf("Date  : %s", d.date);
            printf("Title : %s", d.title);
            printf("Entry :\n%s\n", d.content);
            found = 1;
            break;
        }
    }

    fclose(fp);

    if (!found)
        printf("\nNo entry exists for that date.\n");
}
```

Algorithm for EDITentry()

1. Open **diary.txt** in read mode.
2. Open a temporary file **temp.txt** in write mode.
3. Ask the user for the date of the entry they want to edit.
4. For each entry in the main file:

   If the date matches:

   - Ask the user to enter a new title.
   - Ask for new content.
   - Write the updated entry to the temp file.
   - Mark entry as edited.

   Otherwise, write the original entry to the temp file.
5. Close both files.
6. Delete **diary.txt**.

7. Rename **temp.txt** to **diary.txt**.
8. Display whether the entry was updated or not.

```c
// this function edit entries by asking date when entry was written
void EDITentry() {
    FILE *fp = fopen("diary.txt", "r");
    FILE *temp = fopen("temp.txt", "w");
    struct Diary d;
    char searchDate[20];
    int found = 0;

    if (!fp) {
        printf("\nNo entries available to edit.\n");
        return;
    }

    printf("\nEnter the date of the entry to edit: ");
    fgets(searchDate, sizeof(searchDate), stdin);

    while (fscanf(fp, "%19[^|]|%49[^|]|%499[^\n]\n", d.date, d.title, d.content) != EOF) {
        if (strcmp(searchDate, d.date) == 0) {
            printf("\nEditing Entry...\n");

            printf("Enter new title: ");
            fgets(d.title, sizeof(d.title), stdin);

            printf("Enter new content:\n> ");
            fgets(d.content, sizeof(d.content), stdin);

            found = 1;
        }
        fprintf(temp, "%s|%s|%s", d.date, d.title, d.content);
    }

    fclose(fp);
    fclose(temp);
```

Algorithm for DELETEentry()

1. Open **diary.txt** in read mode.
2. Open **temp.txt** in write mode.
3. Ask the user to enter the date of the entry they want to delete.
4. For each entry read from the main file:

   If the date does not match the search date:

   - Write the entry to the temp file.

   Otherwise:

   - Mark entry as found but do not write it (deleting).

5. Close both files.
6. Replace **diary.txt** with **temp.txt**.
7. Display success message if deleted, otherwise show *"No entry found"*.

```c
void DELETEentry() {
    FILE *fp = fopen("diary.txt", "r");
    FILE *temp = fopen("temp.txt", "w");
    struct Diary d;
    char searchDate[20];
    int found = 0;

    if (!fp || !temp) {
        printf("\nError: Unable to open file.\n");
        return;
    }

    printf("\nEnter the date of the entry to delete: ");
    fgets(searchDate, sizeof(searchDate), stdin);
    searchDate[strcspn(searchDate, "\n")] = '\0';

    while (fscanf(fp, "%19[^|]|%49[^|]|%499[^\n]\n", d.date, d.title, d.content) != EOF) {
        d.date[strcspn(d.date, "\n")] = '\0';
        d.title[strcspn(d.title, "\n")] = '\0';
        d.content[strcspn(d.content, "\n")] = '\0';

        if (strcmp(searchDate, d.date) != 0) {
            fprintf(temp, "%s|%s|%s\n", d.date, d.title, d.content);
        } else {
            found = 1;
        }
    }

    fclose(fp);
    fclose(temp);

    remove("diary.txt");
    rename("temp.txt", "diary.txt");

    if (found)
        printf("\nEntry deleted successfully.\n");
    else
```

Algorithm for Exit Option

1. Display a closing message.
2. Terminate the program using exit(0).

Testing & Results

```
PS C:\Users\Shashwat Tripathi\Downloads\C major project> gcc mp.c
PS C:\Users\Shashwat Tripathi\Downloads\C major project> ./a.exe

========================================
        PERSONAL DIARY MANAGEMENT
========================================

Options:
1. Write a new entry
2. View all saved entries
3. Search an entry
4. Edit an existing entry
5. Delete an entry
6. Exit

Enter your choice:1

Enter the date (DD-MM-YYYY): 11-11-2025
Enter the title: todays entry
Enter the content:
> today i developed a new skill

Entry saved successfully.

========================================
        PERSONAL DIARY MANAGEMENT
========================================

Options:
1. Write a new entry
2. View all saved entries
3. Search an entry
4. Edit an existing entry
5. Delete an entry
6. Exit
```

```
Enter your choice:2

========== SAVED DIARY ENTRIES ==========

---------------------------------------
Date  : 11-12-2025Title : asdfghjklEntry :
sdfghjkfghjk

---------------------------------------
Date  : 11-11-2025
Title : todays entry
Entry :
today i developed a new skill

=======================================
        PERSONAL DIARY MANAGEMENT
=======================================

Options:
1. Write a new entry
2. View all saved entries
3. Search an entry
4. Edit an existing entry
5. Delete an entry
6. Exit

Enter your choice:3

Enter the date to search (DD-MM-YYYY): 11-11-2025

Entry Found:
Date  : 11-11-2025
Title : todays entry
Entry :
today i developed a new skill

=======================================
        PERSONAL DIARY MANAGEMENT
=======================================
```

Conclusion & Future Work

Conclusion:

The *Personal Diary Management System* developed in the C programming language successfully demonstrates how fundamental programming concepts can be applied to create a functional, real-world application. By using structures, file handling, and modular coding techniques, the system provides users with an efficient way to store, retrieve, search, edit, and delete diary entries in a persistent text-based format.

Future work:

1. Password protection
2. Graphical interfaces
3. Date validation

References

1. **Let Us C** – Yashavant Kanetkar
2. GeeksforGeeks
3. Class notes by **DR. TANU SINGH**