C programming Project

Submitted By : Pralav Goyal

Sap Id : 590023121

Submitted To : Mr. Rahul Prasad

1. Project Title

Digital Assignment Tracker

- Introduction

  The *Digital Assignment Tracker* is a mini-project developed in the C programming language using file handling.

  This system helps students and educators organize assignment submissions digitally instead of tracking them manually.

  The project includes functionalities like:

  • Adding assignment records
  • Viewing pending/completed assignments
  • Searching assignments
  • Editing/updating assignment details
  • Deleting assignments
  • Login authentication
  • Permanent storage using .dat files

2. Abstract

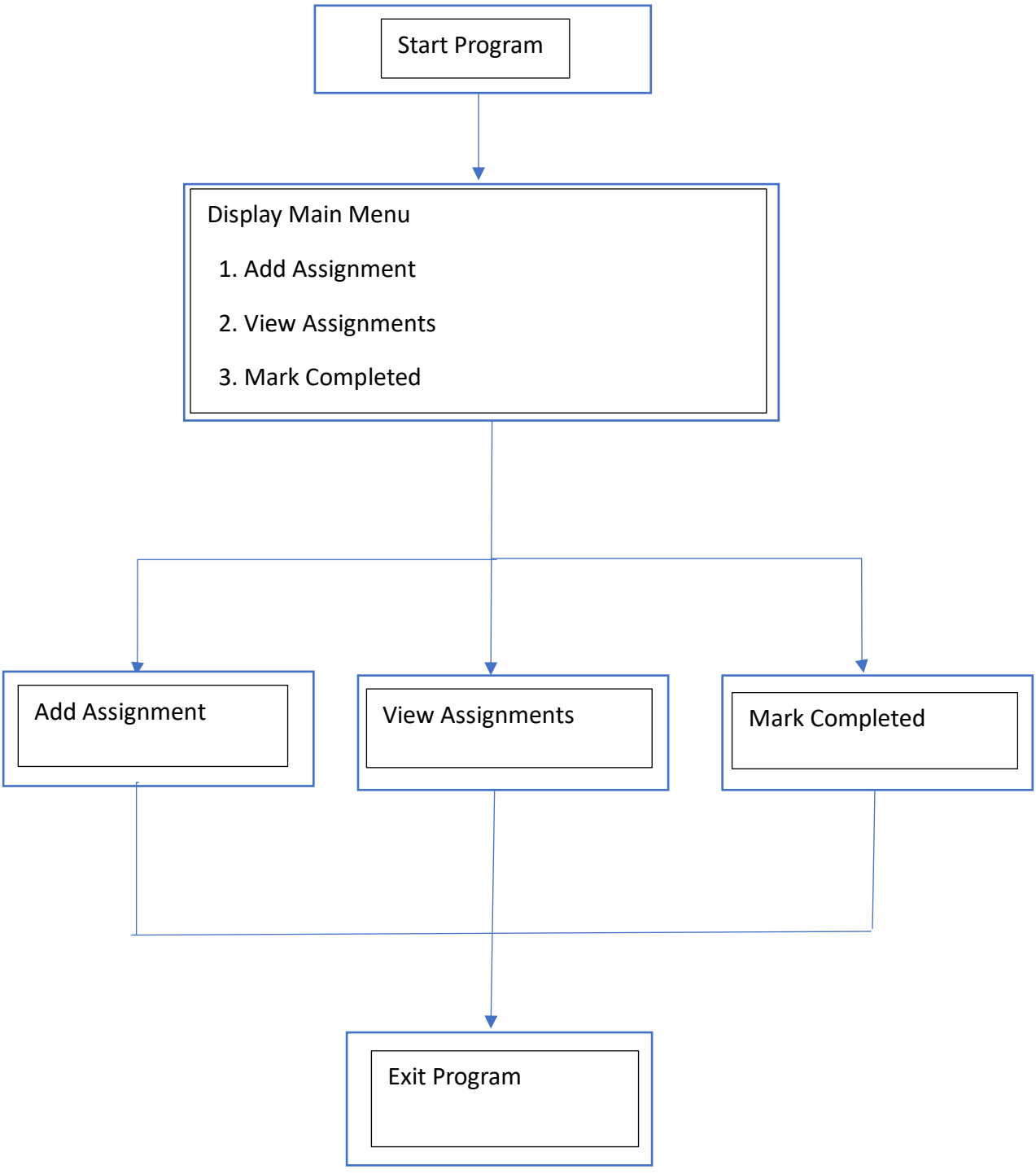
The Digital Assignment Tracker is a simple C program to add, view, and update assignments with due dates. It uses structures and arrays to manage tasks, demonstrating core C concepts while improving productivity and time management.

3. Problem Definition

Managing multiple assignments and deadlines often causes confusion and missed tasks.

- Traditional paper notes or manual lists are unreliable and lack updates.
- Users need a simple digital tool to record assignments with due dates.
- The tool should allow viewing pending/completed tasks and updating progress.
- The Digital Assignment Tracker solves this using a menu-driven C program with arrays and structures.

4. System Design (Flowcharts, Algorithms)



Algorithm: Digital Assignment Tracker

1. **Start**
2. Initialize an array of structures to store assignments.
3. Display the **Main Menu** with options:
    - Add Assignment
    - View Assignments
    - Mark Assignment Completed
    - Exit
4. **If user selects Add Assignment:**
    - Input title and due date.
    - Store in array with status = Pending.
5. **If user selects View Assignments:**
    - Display all assignments with title, due date, and status.
6. **If user selects Mark Completed:**
    - Show list of assignments.
    - Input assignment number.
    - Update status = Completed.
7. **If user selects Exit:**
    - Terminate program.
8. **Repeat steps 3–7** until Exit is chosen.
9. **Stop.**

5. Implementation Details (with snippets)

The project is implemented in C language using structures, arrays, and a menu-driven approach. The design is modular, with separate functions for adding, displaying, updating, and searching assignments.

1. Data Structure

Assignments are stored using a struct with fields for ID, subject, due date, and status.

```
struct Assignment {
    int id;
    char subject[50];
    char dueDate[20];
    char status[20];
};
```

2. Display Module

Displays details of a single assignment in a formatted way.

```
void displayAssignment(struct Assignment a) {
    printf("\nAssignment ID : %d", a.id);
    printf("\nSubject       : %s", a.subject);
    printf("\nDue Date      : %s", a.dueDate);
    printf("\nStatus        : %s\n", a.status);
}
```

3. Add Assignment Module

Allows the user to input assignment details and stores them in the tracker array.

```
tracker[n].id = n + 1;
printf("Enter subject: ");
```

```
scanf("%s", tracker[n].subject);
printf("Enter due date (DD/MM/YYYY): ");
scanf("%s", tracker[n].dueDate);
strcpy(tracker[n].status, "Pending");
n++;
```

4. Update Status Module

Updates the status of a selected assignment by ID.

```
void updateStatus(struct Assignment tracker[], int n, int id) {
    for (int i = 0; i < n; i++) {
        if (tracker[i].id == id) {
            printf("Enter new status (Pending/Completed): ");
            scanf("%s", tracker[i].status);
            printf("Status updated successfully!\n");
            return;
        }
    }
    printf("Assignment ID not found!\n");
}
```

5. Search Module

Searches assignments by subject name.

```
void searchBySubject(struct Assignment tracker[], int n, char
subject[]) {
```

```
    int found = 0;
    for (int i = 0; i < n; i++) {
        if (strcmp(tracker[i].subject, subject) == 0) {
            displayAssignment(tracker[i]);
            found = 1;
        }
    }
    if (!found) {
        printf("No assignment found for subject: %s\n", subject);
    }
}
```

6. Control Loop (Main Menu)

The program runs in a loop until the user chooses to exit.

```
do {
    printf("\n--- Digital Assignment Tracker ---\n");
    printf("1. Add Assignment\n");
    printf("2. Display All Assignments\n");
    printf("3. Update Assignment Status\n");
    printf("4. Search by Subject\n");
    printf("5. Exit\n");
    printf("Enter your choice: ");
```

```
        scanf("%d", &choice);


        switch (choice) {
            case 1: /* Add Assignment */ break;
            case 2: /* Display All */ break;
            case 3: /* Update Status */ break;
            case 4: /* Search Subject */ break;
            case 5: printf("Exiting... Goodbye!\n"); break;
            default: printf("Invalid choice! Try again.\n");
        }
    } while (choice != 5);
```

6. Testing & Results

Testing

The project was tested using black-box testing (checking outputs for given inputs) and functional testing (verifying each module works as intended).

Test Cases

1. **Add Assignment**
    - **Input:** Subject = “Math”, Due Date = “25/11/2025”
    - **Expected Output:** Assignment added with ID = 1, Status = Pending.
2. **Display All Assignments**
    - **Input:** Choice = 2

    - **Expected Output:** List of all assignments with ID, Subject, Due Date, and Status.
3. **Update Assignment Status**
    - **Input:** ID = 1, New Status = Completed
    - **Expected Output:** Status updated successfully.
4. **Search by Subject**
    - **Input:** Subject = “Math”
    - **Expected Output:** Display assignment details for subject “Math”.
5. **Invalid ID Update**

- **Input:** ID = 99
- **Expected Output:** “Assignment ID not found!”

Results

Sample Run 1: Adding and Viewing Assignments

--- Digital Assignment Tracker ---

1. Add Assignment

2. Display All Assignments

3. Update Assignment Status

4. Search by Subject

5. Exit

Enter your choice: 1

Enter subject: Math

Enter due date (DD/MM/YYYY): 25/11/2025

Assignment added successfully!

Enter your choice: 2

=== Assignment List ===

Assignment ID : 1

Subject : Math

Due Date : 25/11/2025

Status : Pending

7. Conclusion & Future Work

Conclusion:-

The Digital Assignment Tracker successfully demonstrates how C programming can be applied to solve practical problems in academic and professional settings. By using structures and arrays, the system organizes assignments with unique IDs, subjects, due dates, and status indicators. The menu-driven interface makes it simple to add, view, update, and search assignments, ensuring users can manage tasks efficiently. Testing confirmed that all modules function correctly, handling both valid and invalid inputs gracefully. Overall, the project highlights the effectiveness of modular programming and user interaction in building lightweight task management tools.

Future Work

While the current version provides essential functionality, several enhancements can make the tracker more powerful and user-friendly:

- **Persistent Storage:** Save assignments to a file or database so data is retained after program exit.
- **Enhanced Search:** Allow searching by due date, status, or keywords instead of only subject.
- **Sorting Features:** Implement sorting by due date or subject for better organization.
- **User Interface:** Add a graphical interface (GUI) for improved usability beyond the console.
- **Notifications/Reminders:** Integrate alerts for upcoming deadlines.
- **Multi-user Support:** Extend functionality to handle multiple users with separate assignment lists.

8. References

- Yashavant Kanetkar. (2024). *Let Us C* (18th Edition). BPB Publications.

- Kernighan, B.W., & Ritchie, D.M. (1988). *The C Programming Language* (2nd Edition). Prentice Hall.

Whole Code:-

```
#include <stdio.h>

#include <string.h>


#define MAX_ASSIGNMENTS 100


struct Assignment {

    int id;

    char subject[50];

    char dueDate[20];

    char status[20];

};


void displayAssignment(struct Assignment a) {

    printf("\n-----------------------------");

    printf("\nAssignment ID : %d", a.id);

    printf("\nSubject       : %s", a.subject);

    printf("\nDue Date      : %s", a.dueDate);

    printf("\nStatus        : %s", a.status);

    printf("\n-----------------------------\n");

}


void updateStatus(struct Assignment tracker[], int n, int id) {

    for (int i = 0; i < n; i++) {

        if (tracker[i].id == id) {

            printf("Enter new status (Pending/Completed): ");

            scanf("%s", tracker[i].status);
```

```
            printf(" Status updated successfully!\n");
            return;
        }
    }
    printf(" Assignment ID not found!\n");
}

void searchBySubject(struct Assignment tracker[], int n, char subject[]) {
    int found = 0;
    for (int i = 0; i < n; i++) {
        if (strcmp(tracker[i].subject, subject) == 0) {
            displayAssignment(tracker[i]);
            found = 1;
        }
    }
    if (!found) {
        printf(" No assignment found for subject: %s\n", subject);
    }
}

void displayAll(struct Assignment tracker[], int n) {
    if (n == 0) {
        printf(" No assignments added yet!\n");
    } else {
        printf("\n=== Assignment List ===\n");
        for (int i = 0; i < n; i++) {
            displayAssignment(tracker[i]);
```

```
        }
    }
}

int main() {
    struct Assignment tracker[MAX_ASSIGNMENTS];
    int n = 0, choice, id;
    char subject[50];

    do {
        printf("\n--- Digital Assignment Tracker ---\n");
        printf("1. Add Assignment\n");
        printf("2. Display All Assignments\n");
        printf("3. Update Assignment Status\n");
        printf("4. Search by Subject\n");
        printf("5. Exit\n");
        printf("Enter your choice: ");
        scanf("%d", &choice);

        switch (choice) {
            case 1:
                if (n >= MAX_ASSIGNMENTS) {
                    printf(" Tracker full! Cannot add more assignments.\n");
                } else {
                    tracker[n].id = n + 1;
                    printf("Enter subject: ");
                    scanf("%s", tracker[n].subject);
```

```
        printf("Enter due date (DD/MM/YYYY): ");
        scanf("%s", tracker[n].dueDate);
        strcpy(tracker[n].status, "Pending");
        printf(" Assignment added successfully!\n");
        n++;
    }
    break;

case 2:
    displayAll(tracker, n);
    break;

case 3:
    printf("Enter Assignment ID to update: ");
    scanf("%d", &id);
    updateStatus(tracker, n, id);
    break;

case 4:
    printf("Enter subject to search: ");
    scanf("%s", subject);
    searchBySubject(tracker, n, subject);
    break;

case 5:
    printf(" Exiting... Goodbye!\n");
    break;
```

```
            default:
                printf("Invalid choice! Try again.\n");
        }
    } while (choice != 5);

    return 0;
}
```