GYM MEMBERSHIP MANAGEMENT

Name- Jacob Biju
Roll no – 39
Course name- C program
Date -

Introduction

Project overview:- The gym membership management system in C aims to help gym administrators manage member data efficiently. The system will allow adding new members, updating member details, deleting members, and viewing member information.

Problem statement :-The gym seeks to improve efficiency by replacing its error-prone, time-consuming manual system with a computerized Gym Membership Management System. The new system, developed in C, will streamline tracking, updating, and maintaining accurate member data.

Objective :-The objective of this project is to design and implement a Gym Membership Management System that allows gym administrators to efficiently manage member information, including adding new members, updating member details, deleting members, and viewing member information.

System Requirements

Hardware Requirement

Processor: Any processor with at least 1 GHz speed.
Storage: Minimum 1GB free disk space.
RAM: Minimum 2GB (4GB recommended).

Software Requirement

Operating System: Windows 7 or high (Windows 10 recommended).
Compiler: GCC or any compatible C compiler.
IDE: Any lightweight C IDE like Visual Studio, code blocks.

Program Logic

Member Structure:The Member structure stores member information including username, password, attendance count, and membership status.
Function: registerMemberThis function collects the new member's details (username, password) and initializes their attendance to 0 and membership status to active (1).The member's details are appended to the "members.txt" file.
Function: validateLoginThis function checks if the provided username and password match any registered member.It reads member records from "members.txt" and compares the input credentials.If a match is found, the function returns 1 and copies the member's details to the loggedInMember pointer; otherwise, it returns 0.
Function: trackAttendanceThis function increments the attendance count for the logged-in member.It reads all member records from "members.txt", updates the attendance for the logged-in member, and writes the updated records to a temporary file.The temporary file is then renamed to "members.txt" to replace the original file.
Function: manageRenewalsAndPaymentsThis function allows the administrator to manage membership renewals.It reads member records, checks if any membership is inactive (0), and prompts the admin to renew it.Updated records are written to a temporary file, which replaces the original file.
Function: adminOptionsThis function presents the admin menu with options to register members, track attendance, and manage renewals.It handles the admin's choices and calls the appropriate functions.
Function: memberOptionsThis function allows members to log in and track their attendance.It handles the member's choices and calls the appropriate functions.
Function: mainThis function presents the main menu for the system, providing options for admin login, member login, member registration, and exiting the system.It handles the user's choices and calls the appropriate functions.

Pseudocode

```
  STRUCT Member
   STRING username[30]
   STRING password[20]
    INTEGER attendance
   INTEGER membershipStatus
  END STRUCT
FUNCTION main
    INTEGER choice
    STRING adminUsername = “admin”
    STRING adminPassword = “admin”
    STRING username[30]
    STRING password[20]
DO
        PRINT “Gym Management System”
        PRINT “1. Admin login”
        PRINT “2. Member login”
        PRINT “3. Register a new member”
        PRINT “4. Exit”
        PRINT “Enter your choice: “
```

- READ choice
-
- SWITCH choice
- CASE 1:
- PRINT “Enter admin username: “
- READ username
- PRINT “Enter admin password: “
- READ password
- IF username == adminUsername AND password == adminPassword THEN
- PRINT “Admin login successful!”
- CALL adminOptions
- ELSE
- PRINT “Invalid admin credentials.”
- END IF
- CASE 2:
- CALL memberOptions
- CASE 3:
- CALL registerMember
- CASE 4:
- PRINT “Exiting system. Goodbye!”
- DEFAULT:
- PRINT “Invalid choice. Please try again.”
- END SWITCH
- UNTIL choice == 4
- END FUNCTION

- FUNCTION registerMember
- DECLARE Member member
- OPEN file “members.txt” FOR appending AS file
-
- IF file IS NULL THEN
- PRINT “Error opening file for registration.”
- RETURN
- END IF
-
- PRINT “Enter username: “
- READ member.username
- PRINT “Enter password: “
- READ member.password
- member.attendance = 0
- member.membershipStatus = 1
-
- WRITE member TO file
- CLOSE file
- PRINT “Member registered successfully!”
- END FUNCTION

- FUNCTION validateLogin(loggedInMember)
- DECLARE Member member
- STRING username[30]
- STRING password[20]
- OPEN file “members.txt” FOR reading AS file
-
- IF file IS NULL THEN
- PRINT “Error reading member data. Please register first.”
- RETURN 0
- END IF
-
- PRINT “Enter username: “
- READ username
- PRINT “Enter password: “
- READ password
-
- WHILE READ member FROM file
- IF username == member.username AND password == member.password THEN
- loggedInMember = member
- CLOSE file
- PRINT “Login successful!”
- RETURN 1
- END IF
- END WHILE
-
- CLOSE file
- PRINT “Invalid credentials. Please try again.”
- RETURN 0
- END FUNCTION

```
FUNCTION trackAttendance(member)
    DECLARE Member temp
    OPEN file “members_temp.txt” FOR writing AS file
    OPEN readFile “members.txt” FOR reading AS readFile

    IF file IS NULL OR readFile IS NULL THEN
        PRINT “Error opening file.”
        RETURN
    END IF

    WHILE READ temp FROM readFile
        IF temp.username == member.username THEN
            temp.attendance = temp.attendance + 1
        END IF
        WRITE temp TO file
    END WHILE

    CLOSE file
    CLOSE readFile
    DELETE “members.txt”
    RENAME “members_temp” TO “members.txt”
    PRINT “Attendance tracked successfully! Total attendance: “ + (member.attendance + 1)
END FUNCTION
```

```
FUNCTION manageRenewalsAndPayments
  DECLARE Member member
  OPEN file “members_temp.txt” FOR writing AS file
  OPEN readFile “members.txt” FOR reading AS readFile

  IF file IS NULL OR readFile IS NULL THEN
    PRINT “Error opening file.”
    RETURN
  END IF

  WHILE READ member FROM readFile
    IF member.membershipStatus == 0 THEN
      PRINT “Renew membership for “ + member.username + “? (1 for Yes, 0 for No): “
      READ member.membershipStatus
      IF member.membershipStatus == 1 THEN
        PRINT “Payment received. Membership renewed for “ + member.username
      END IF
    END IF
    WRITE member TO file
  END WHILE

  CLOSE file
  CLOSE readFile
  DELETE “members.txt”
  RENAME “members_temp” TO “members.txt”
  PRINT “Membership renewals and payments managed successfully.”
END FUNCTION
```

```
FUNCTION adminOptions
  INTEGER choice
  DECLARE Member loggedInMember

  DO
    PRINT “Admin Menu:”
    PRINT “1. Register a new member”
    PRINT “2. Track attendance”
    PRINT “3. Manage renewals and payments”
    PRINT “4. Logout”
    PRINT “Enter your choice: “
    READ choice

    SWITCH choice
      CASE 1:
        CALL registerMember
      CASE 2:
        IF CALL validateLogin(loggedInMember) THEN
          CALL trackAttendance(loggedInMember)
        END IF
      CASE 3:
        CALL manageRenewalsAndPayments
      CASE 4:
        PRINT “Logging out…”
      DEFAULT:
        PRINT “Invalid choice. Please try again.”
    END SWITCH
  UNTIL choice == 4
```

```
FUNCTION memberOptions
  INTEGER choice
  DECLARE Member loggedInMember

  IF NOT CALL validateLogin(loggedInMember) THEN
    RETURN
  END IF

  DO
    PRINT “Member Menu:”
    PRINT “1. Track attendance”
    PRINT “2. Logout”
    PRINT “Enter your choice: “
    READ choice

    SWITCH choice
      CASE 1:
        CALL trackAttendance(loggedInMember)
      CASE 2:
        PRINT “Logging out…”
      DEFAULT:
        PRINT “Invalid choice. Please try again.”
    END SWITCH
  UNTIL choice == 2
END FUNCTION
```

• Test Cases

1. Register Member

Test Case 1: Successful Registration,Input:Enter username: jacob,Enter password: 123
Expected Output: Member registered successfully!
Description: This test case checks if a new member can be registered successfully.

Test Case 2.1: Successful Member Login
Precondition: Ensure Jacob is already registered with passwor123.
Input:Enter username: Jacob Enter password: 123
Expected Output:Login successful!
Description: This test case checks if a member can log in successfully with valid credentials.

Test Case 2.2: Invalid Credentials
Input:Enter username:Jacob ,Enter password: wrongpassword
Expected Output:Invalid credentials.
Description: This test case checks how the system handles invalid login credentials.

- Test Case 3.1: Successful Attendance Tracking

Precondition: Ensure Jacob is already registered and logged in.

Input:Logged in as Jacob Select option to track attendance

Expected Output:Attendance tracked successfully! Total attendance: 1

Description: This test case checks if attendance tracking works correctly for a logged-in member.

Test Case 4.1: Successful Membership Renewal

Precondition: Ensure a member Jacob with inactive membership (membershipStatus = 0) exists.

Input:Renew membership for Jacob: 1

Expected Output:Payment received. Membership renewed for Jacob.Membership renewals and payments managed successfully.

Description: This test case checks if the system can successfully handle membership renewals and payments.

Test Case 5.1: Successful Admin Login

Input:Enter admin username: 0 Enter admin password: 0

Expected Output:Admin login successful!

Description: This test case checks if the admin can log in successfully with valid credentials.

Test Case 5.2: Invalid Admin Credentials

Input :Enter admin username: admin Enter admin password: wrongpassword

Expected Output:Invalid admin credentials.

Description: This test case checks how the system handles invalid admin login credentials.

Test Case 6.1: Member Menu Navigation

Precondition: Ensure Jacob is already registered and logged in.

Input:Logged in as Jacob Select option to track attendance,Select option to logout

Expected Output:Attendance tracked successfully!, Total attendance: 1,Logging out...

Description: This test case checks if the member menu navigation works correctly and the member can track attendance and log out.

.

Output Screenshot or results

- **Test case 1.1**

Register Member

```
Gym Management System
1. Admin login
2. Member login
3. Register a new member
4. Exit
Enter your choice: 3
Enter username: Jacob
Enter password: 123
Member registered successfully!
```

Test case 2.1

Succesful member login

```
Enter your choice: 2
Enter username: Jacob
Enter password: 123
Login successful!
```

Test case 2.2

```
Enter your choice: 2
Enter username: Jacob
Enter password: 3
Invalid credentials. Please try again.
```

Test case 3.1

Track attendance

```
Member Menu:
1. Track attendance
2. Logout
Enter your choice: 1
Attendance tracked successfully! Total attend
ance: 1
```

Test case 4.1
Manage renewals and payments

```
Admin Menu:
1. Register a new member
2. Track attendance
3. Manage renewals and payments
4. Logout
Enter your choice: 3
Membership renewals and payments managed succ
essfully.
```

Test case 5.2
Invalid admin credentials

```
Enter your choice: 1
Enter admin username: sjs
Enter admin password: wka
Invalid admin credentials.
```

Test case 5.1
Succesful admin login

```
Gym Management System
1. Admin login
2. Member login
3. Register a new member
4. Exit
Enter your choice: 1
Enter admin username: 0
Enter admin password: 0
Admin login successful!
```

Test case 6.1
Member menu navigation

```
Enter your choice: 2
Enter username: Jacob
Enter password: 123
Login successful!
Attendance tracked successfully! Total attend
ance: 1

Admin Menu:
1. Register a new member
2. Track attendance
3. Manage renewals and payments
4. Logout
Enter your choice: 4
Logging out...
```

Discussion of results

RegistrationSuccess: Members are registered correctly.

LoginSuccess: Members and admins can log in with valid credentials.Security: Denies access for incorrect credentials.

Attendance TrackingSuccess: Updates attendance records accurately.

Membership RenewalsSuccess: Processes renewals and payments effectively.

Admin FunctionsLogin: Secure access to admin functions.Menu: Admins can manage members and renewals efficiently.

Member FunctionsMenu: Members can track attendance and log out easily.

• Conclusion

Summary of the project

The Gym Membership Management System effectively automates key gym operations, reducing manual workload and improving data accuracy. The system is secure, reliable, and easy to use, with potential for future enhancements like database integration and improved security measures.

Future Enhancements

Future enhancements for the Gym Membership Management System include integrating a database for better scalability, improving security with stronger encryption and two-factor authentication, and developing a GUI or mobile app for enhanced user experience. Additional features like automated notifications, advanced reporting, and integration with fitness trackers will further enhance functionality.

Reference

https://membershipworks.com/membership-management-software/

Appendices

Source code

```c
#include <stdio.h>
#include <stdlib.h>
#include <string.h>

struct Member {
    char username[30];
    char password[20];
    int attendance;
    int membershipStatus;
};

void registerMember() {
    struct Member member;
    FILE *file = fopen("members.txt", "a");

    if (file == NULL) {
        printf("Error opening file for registration.\n");
        return;
    }

    printf("Enter username: ");
    scanf("%s", member.username);
    printf("Enter password: ");
    scanf("%s", member.password);
    member.attendance = 0;
    member.membershipStatus = 1;

    fwrite(&member, sizeof(struct Member), 1, file);
    fclose(file);
    printf("Member registered successfully!\n");
}
```

```c
int validateLogin(struct Member *loggedInMember) {
    struct Member member;
    char username[30], password[20];
    FILE *file = fopen("members.txt", "r");

    if (file == NULL) {
        printf("Error reading member data. Please register first.\n");
        return 0;
    }

    printf("Enter username: ");
    scanf("%s", username);
    printf("Enter password: ");
    scanf("%s", password);

    while (fread(&member, sizeof(struct Member), 1, file)) {
        if (strcmp(username, member.username) == 0 && strcmp(password, member.p
            *loggedInMember = member;
            fclose(file);
            printf("Login successful!\n");
            return 1;
        }
    }

    fclose(file);
    printf("Invalid credentials. Please try again.\n");
    return 0;
}

void trackAttendance(struct Member *member) {
    FILE *file = fopen("members_temp.txt", "w");
    FILE *readFile = fopen("members.txt", "r");
    struct Member temp;

    if (file == NULL || readFile == NULL) {
        printf("Error opening file.\n");
        return;
    }

    while (fread(&temp, sizeof(struct Member), 1, readFile)) {
        if (strcmp(temp.username, member->username) == 0) {
            temp.attendance++;
        }
        fwrite(&temp, sizeof(struct Member), 1, file);
    }

    fclose(file);
    fclose(readFile);
    remove("members.txt");
    rename("members_temp", "members.txt");
    printf("Attendance tracked successfully! Total attendance: %d\n", member->a
}
```

```c
void manageRenewalsAndPayments() {
    struct Member member;
    FILE *file = fopen("members_temp.txt", "w");
    FILE *readFile = fopen("members.txt", "r");

    if (file == NULL || readFile == NULL) {
        printf("Error opening file.\n");
        return;
    }

    while (fread(&member, sizeof(struct Member), 1, readFile)) {
        if (member.membershipStatus == 0) {
            printf("Renew membership for %s? (1 for Yes, 0 for No): ", member.u
            scanf("%d", &member.membershipStatus);
            if (member.membershipStatus == 1) {
                printf("Payment received. Membership renewed for %s.\n", member
            }
        }
        fwrite(&member, sizeof(struct Member), 1, file);
    }

    fclose(file);
    fclose(readFile);
    remove("members.txt");
    rename("members_temp", "members.txt");
    printf("Membership renewals and payments managed successfully.\n");
}

void adminOptions() {
    int choice;
    struct Member loggedInMember;

    do {
        printf("\nAdmin Menu:\n");
        printf("1. Register a new member\n");
        printf("2. Track attendance\n");
        printf("3. Manage renewals and payments\n");
        printf("4. Logout\n");
        printf("Enter your choice: ");
        scanf("%d", &choice);

        switch (choice) {
            case 1:
                registerMember();
                break;
            case 2:
                if (validateLogin(&loggedInMember)) {
                    trackAttendance(&loggedInMember);
                }
                break;
            case 3:
                manageRenewalsAndPayments();
                break;
            case 4:
                printf("Logging out...\n");
                break;
            default:
                printf("Invalid choice. Please try again.\n");
        }
    } while (choice != 4);
}
```

```c
void memberOptions() {
    int choice;
    struct Member loggedInMember;

    if (!validateLogin(&loggedInMember)) {
        return;
    }

    do {
        printf("\nMember Menu:\n");
        printf("1. Track attendance\n");
        printf("2. Logout\n");
        printf("Enter your choice: ");
        scanf("%d", &choice);

        switch (choice) {
            case 1:
                trackAttendance(&loggedInMember);
                break;
            case 2:
                printf("Logging out...\n");
                break;
            default:
                printf("Invalid choice. Please try again.\n");
        }
    } while (choice != 2);
}

int main() {
    int choice;
    char adminUsername[] = "0"; // admin pass and username mattan
    char adminPassword[] = "0";
    char username[30], password[20];

    do {
        printf("\nGym Management System\n");
        printf("1. Admin login\n");
        printf("2. Member login\n");
        printf("3. Register a new member\n");
        printf("4. Exit\n");
        printf("Enter your choice: ");
        scanf("%d", &choice);

```

```
190
191         switch (choice) {
192             case 1:
193                 printf("Enter admin username: ");
194                 scanf("%s", username);
195                 printf("Enter admin password: ");
196                 scanf("%s", password);
197                 if (strcmp(username, adminUsername) == 0 && strcmp(password, ad
198                     printf("Admin login successful!\n");
199                     adminOptions();
200                 } else {
201                     printf("Invalid admin credentials.\n");
202                 }
203                 break;
204             case 2:
205                 memberOptions();
206                 break;
207             case 3:
208                 registerMember();
209                 break;
210             case 4:
211                 printf("Exiting system. Goodbye!\n");
212                 break;
213             default:
214                 printf("Invalid choice. Please try again.\n");
215         }
216     } while (choice != 4);
217
218     return 0;
219 }
220
```