MICROPROJECT

COURSE MANAGEMENT SYSTEM

MEHNAZ ABDUL

54

PROGRAMMING IN C

17-07-2024

INTRODUCTION

This project implements a Course Management System in C programming language. The system allows administrators to enroll students in various courses and view their academic progress. It efficiently manages student enrollments and provides a user-friendly interface for tracking course progress

Key Features:

- **Enrollment Management:** Enables administrators to enroll students in predefined courses.
- **Progress Tracking:** Facilitates viewing and monitoring of student progress in enrolled courses.
- **Efficiency:** Ensures efficient management of student enrollments and course progress tracking

SYSTEM REQUIREMENTS

SOFTWARE COMPONENTS : GCC (GNU Compiler Collection): Widely used open-source compiler suite for C, C++, and other languages

Visual Studio Code: A lightweight but powerful IDE with support for various languages including C/C++

HARDWARE COMPONENTS : OS

DESIGN AND DEVELOPMENT

Program Logic:
The program uses structures to define `Course` and `Student` entities. It manages enrollments through user input and displays student progress based on enrolled courses

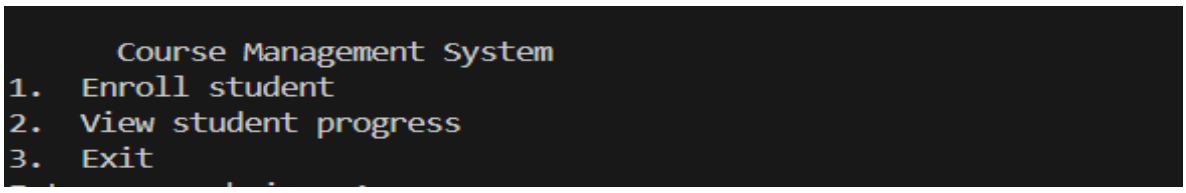
PSEUDOCODE

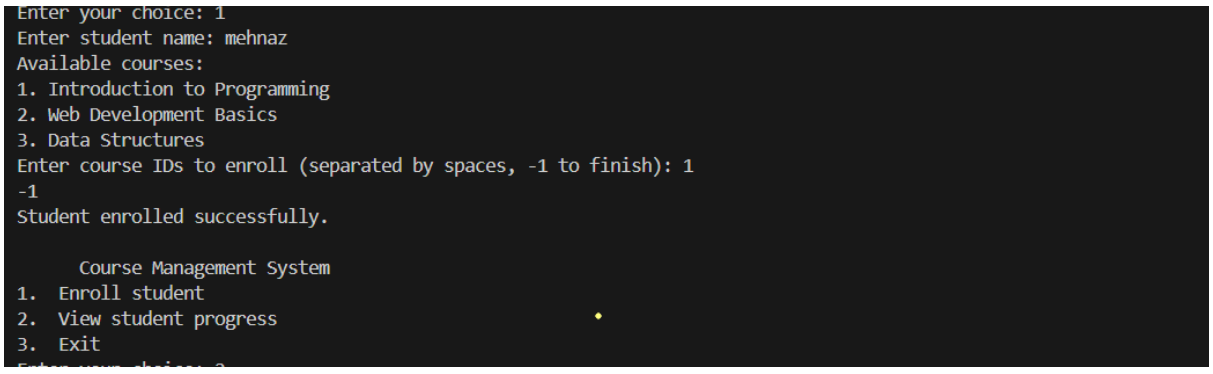
```
struct Course {
   int courseId;
   char courseName[50];
};
struct Student {
   char studentName[50];
   int studentId;
   int enrolledCourses[MAXIMUM_COURSES];
   int numCourses;
};
Course courses[MAXIMUM_COURSES] = {
   {1, "Introduction to Programming"},
   {2, "Web Development Basics"},
   {3, "Data Structures"}
};
Student students[MAXIMUM_STUDENTS];
```

```
int numStudents = 0;
void enrollStudent() {
   // Implement enrollment logic
}
void viewStudentProgress(); {
   // Implement progress viewing logic
}
int main(); {
   // Display menu and handle user choices
   return 0;
}
```

TESTING AND RESULTS

TEST CASES





```
Enter your choice: 2
Enter student ID (1 to 1) to view progress: 1
Student Name: mehnaz
Enrolled Courses:
- Introduction to Programming

     Course Management System
1.  Enroll student
2.  View student progress
3.  Exit
```

```
Enter your choice: 3
- Introduction to Programming

      Course Management System
1.  Enroll student
2.  View student progress
3.  Exit
Enter your choice: 3
1.  Enroll student
2.  View student progress
3.  Exit
Enter your choice: 3
Exiting program.
PS C:\LOCAL DISK A\C PROGRAMMINGS>
Exiting program.
Exiting program.
PS C:\LOCAL DISK A\C PROGRAMMINGS>
```

CONCLUSION

This project demonstrates a basic COURSE MANAGEMENT SYSTEM in C, showcasing usage of structure, function.

ENHANCEMENT : we can modify this system by specifying course enrollment dates, adding registrations details etc..

APPENDICES

CODE LISTING:

course_.c × _course.c Settings

course_.c > Course > courseName

```c
#include <stdio.h>
#include <string.h>
#define MAXIMUM_COURSES 3
#define MAXIMUM_STUDENTS 5

// Course structure
struct Course {
    int courseId;
    char courseName[50];
};

// Student structure
struct Student {
    char studentName[50];
    int studentId;
    int enrolledCourses[MAXIMUM_COURSES];
    int numCourses;
};

// Global variables
struct Course courses[MAXIMUM_COURSES] = {
    {1, "Introduction to Programming"},
    {2, "Web Development Basics"},
    {3, "Data Structures"}
};

struct Student students[MAXIMUM_STUDENTS];
int numStudents = 0;

// Function prototypes
void enrollStudent();
void viewStudentProgress();

int main() {
    int choice;
    do {
        printf("\n      Course Management System     \n");
```

course_.c × _course.c Settings

course_.c > Course > courseName

```c
int main() {
        printf("\n      Course Management System     \n");
        printf("1.  Enroll student\n");
        printf("2.  View student progress\n");
        printf("3.  Exit\n");
        printf("Enter your choice: ");
        scanf("%d", &choice);

        switch (choice) {
            case 1:
                enrollStudent();
                break;
            case 2:
                viewStudentProgress();
                break;
            case 3:
                printf("Exiting program.\n");
                break;
            default:
                printf("Invalid choice. Please try again.......\n");
                break;
        }
    } while (choice != 3);

    return 0;
}

// Function to enroll a student in courses
void enrollStudent() {
    if (numStudents >= MAXIMUM_STUDENTS) {
        printf("Maximum number of students reached.\n");
        return;
    }

    struct Student newStudent;
    printf("Enter student name: ");
```

```c
    scanf(" %[^\n]", newStudent.studentName); // Corrected scanf format

    printf("Available courses:\n");
    for (int i = 0; i < MAXIMUM_COURSES; ++i) {
        printf("%d. %s\n", courses[i].courseId, courses[i].courseName);
    }
    printf("Enter course IDs to enroll (separated by spaces, -1 to finish): ");
    newStudent.numCourses = 0;
    int courseId;
    do {
        scanf("%d", &courseId);
        if (courseId == -1 || newStudent.numCourses >= MAXIMUM_COURSES)
            break;
```

```c
        int found = 0;
        for (int i = 0; i < MAXIMUM_COURSES; ++i) {
            if (courses[i].courseId == courseId) {
                newStudent.enrolledCourses[newStudent.numCourses++] = courseId;
                found = 1;
                break;
            }
        }
```

course_.c | _course.c | Settings

course_.c > viewStudentProgress()

```
64  void enrollStudent() {
95          if (!found)
96              printf("Course ID %d not found. Please enter a valid ID (-1 to finish): ", courseId);
97      } while (courseId != -1);
98
99      newStudent.studentId = numStudents;
100     students[numStudents++] = newStudent;
101     printf("Student enrolled successfully.\n");
102 }
103
104 // Function to view student progress
105 void viewStudentProgress() {
106     if (numStudents == 0) {
107         printf("No students enrolled.\n");
108         return;
109     }
110
111     printf("Enter student ID (1 to %d) to view progress: ", numStudents);
112     int studentId;
113     scanf("%d", &studentId);
114
115     if (studentId < 1 || studentId > numStudents) {
116         printf("Invalid student ID.\n");
117         return;
118     }
119
120     struct Student student = students[studentId - 1];
121     printf("Student Name: %s\n", student.studentName);
122     printf("Enrolled Courses:\n");
123
124     for (int i = 0; i < student.numCourses; ++i) {
125         int courseId = student.enrolledCourses[i];
126         for (int j = 0; j < MAXIMUM_COURSES; ++j) {
127             if (courses[j].courseId == courseId) {
128                 printf("- %s\n", courses[j].courseName);
129                 break;
130             }
131         }
132     }
133 }
```