#include <iostream>

#include <vector>

#include <string>

using namespace std;

// Structure to hold assignment data

struct Assignment {

    string courseName;

    string deadline;

    string submissionStatus;

};

// Function prototypes

void createAssignment(vector<Assignment> &assignments);

void displayAssignments(const vector<Assignment> &assignments);

void updateAssignment(vector<Assignment> &assignments);

void deleteAssignment(vector<Assignment> &assignments);

int main() {

    vector<Assignment> assignments;  // Vector to store assignments

    int choice;

    do {

        cout << "\nAssignment Submission Tracker\n";

        cout << "1. Add Assignment\n";

        cout << "2. View All Assignments\n";

        cout << "3. Update Assignment\n";

        cout << "4. Delete Assignment\n";

        cout << "5. Exit\n";

        cout << "Choose an option: ";

        cin >> choice;

        switch (choice) {

            case 1:

                createAssignment(assignments);

                break;

            case 2:

                displayAssignments(assignments);

                break;

            case 3:

                updateAssignment(assignments);

                break;

            case 4:

                deleteAssignment(assignments);

                break;

            case 5:

                cout << "Exiting the program...\n";

                break;

            default:

                cout << "Invalid option. Please try again.\n";

        }

    } while (choice != 5);

    return 0;

}

// Function to add a new assignment

void createAssignment(vector<Assignment> &assignments) {

    Assignment newAssignment;

    cout << "\nEnter Course Name: ";

    cin.ignore();

    getline(cin, newAssignment.courseName);

    cout << "Enter Deadline (e.g., 2024-10-07): ";

    getline(cin, newAssignment.deadline);

    cout << "Enter Submission Status (e.g., Submitted, Not Submitted): ";

    getline(cin, newAssignment.submissionStatus);

    assignments.push\_back(newAssignment);

    cout << "Assignment added successfully!\n";

}

// Function to display all assignments

void displayAssignments(const vector<Assignment> &assignments) {

    if (assignments.empty()) {

        cout << "\nNo assignments to display.\n";

        return;

    }

    cout << "\nList of Assignments:\n";

    for (size\_t i = 0; i < assignments.size(); ++i) {

        cout << "Assignment " << i + 1 << ":\n";

        cout << "Course Name: " << assignments[i].courseName << "\n";

        cout << "Deadline: " << assignments[i].deadline << "\n";

        cout << "Submission Status: " << assignments[i].submissionStatus << "\n";

        cout << "--------------------------\n";

    }

}

// Function to update an existing assignment

void updateAssignment(vector<Assignment> &assignments) {

    if (assignments.empty()) {

        cout << "\nNo assignments to update.\n";

        return;

    }

    int index;

    cout << "\nEnter the assignment number to update: ";

    cin >> index;

    if (index <= 0 || index > assignments.size()) {

        cout << "Invalid assignment number.\n";

        return;

    }

    Assignment &assignmentToUpdate = assignments[index - 1];

    cout << "Enter new Course Name (" << assignmentToUpdate.courseName << "): ";

    cin.ignore();

    getline(cin, assignmentToUpdate.courseName);

    cout << "Enter new Deadline (" << assignmentToUpdate.deadline << "): ";

    getline(cin, assignmentToUpdate.deadline);

    cout << "Enter new Submission Status (" << assignmentToUpdate.submissionStatus << "): ";

    getline(cin, assignmentToUpdate.submissionStatus);

    cout << "Assignment updated successfully!\n";

}

// Function to delete an assignment

void deleteAssignment(vector<Assignment> &assignments) {

    if (assignments.empty()) {

        cout << "\nNo assignments to delete.\n";

        return;

    }

    int index;

    cout << "\nEnter the assignment number to delete: ";

    cin >> index;

    if (index <= 0 || index > assignments.size()) {

        cout << "Invalid assignment number.\n";

        return;

    }

    assignments.erase(assignments.begin() + (index - 1));

    cout << "Assignment deleted successfully!\n";

}