#include <iostream>

#include <string>

using namespace std;

// Node structure to store member information

struct Member {

int prn;

string name;

Member\* next;

};

// Function to add a member to the club

void addMember(Member\*& head, int prn, string name) {

Member\* newMember = new Member{prn, name, nullptr};

if (head == nullptr) {

head = newMember;

} else {

Member\* temp = head;

while (temp->next != nullptr) {

temp = temp->next;

}

temp->next = newMember;

}

}

// Function to delete a member

void deleteMember(Member\*& head, int prn) {

if (head == nullptr) {

cout << "No members in the club." << endl;

return;

}

if (head->prn == prn) {

Member\* temp = head;

head = head->next;

delete temp;

cout << "Member with PRN " << prn << " deleted." << endl;

return;

}

Member\* temp = head;

while (temp->next != nullptr && temp->next->prn != prn) {

temp = temp->next;

}

if (temp->next == nullptr) {

cout << "Member with PRN " << prn << " not found." << endl;

} else {

Member\* toDelete = temp->next;

temp->next = temp->next->next;

delete toDelete;

cout << "Member with PRN " << prn << " deleted." << endl;

}

}

// Function to display all members

void displayMembers(Member\* head) {

if (head == nullptr) {

cout << "No members in the club." << endl;

return;

}

cout << "Club Members:" << endl;

while (head != nullptr) {

cout << "PRN: " << head->prn << ", Name: " << head->name << endl;

head = head->next;

}

}

// Function to count total members

int countMembers(Member\* head) {

int count = 0;

while (head != nullptr) {

count++;

head = head->next;

}

return count;

}

// Function to concatenate two lists

void concatenateLists(Member\*& head1, Member\*& head2) {

if (head1 == nullptr) {

head1 = head2;

return;

}

Member\* temp = head1;

while (temp->next != nullptr) {

temp = temp->next;

}

temp->next = head2;

}

int main() {

Member\* division1 = nullptr; // Head for Division 1

Member\* division2 = nullptr; // Head for Division 2

// Adding some initial members

addMember(division1, 1, "President");

addMember(division1, 101, "Alice");

addMember(division1, 102, "Bob");

addMember(division1, 2, "Secretary");

addMember(division2, 201, "Charlie");

addMember(division2, 202, "Dave");

int choice;

do {

cout << "\n1. Add Member\n2. Delete Member\n3. Display Members\n4. Count Members\n5. Concatenate Divisions\n6. Exit\nEnter your choice: ";

cin >> choice;

switch (choice) {

case 1: {

int prn;

string name;

cout << "Enter PRN: ";

cin >> prn;

cout << "Enter Name: ";

cin.ignore();

getline(cin, name);

addMember(division1, prn, name);

break;

}

case 2: {

int prn;

cout << "Enter PRN of member to delete: ";

cin >> prn;

deleteMember(division1, prn);

break;

}

case 3:

displayMembers(division1);

break;

case 4:

cout << "Total Members: " << countMembers(division1) << endl;

break;

case 5:

concatenateLists(division1, division2);

cout << "Divisions concatenated successfully." << endl;

break;

case 6:

cout << "Exiting..." << endl;

break;

default:

cout << "Invalid choice. Try again!" << endl;

}

} while (choice != 6);

return 0;

}