**EMPLOYEE MANAGEMENT SYSTEM**

#include <bits/stdc++.h>

using namespace std;

struct Employee {

string firstName;

string lastName;

string dateOfBirth;

string title;

bool isActive;

};

vector<Employee> employees;

void addEmployeeInput(string &firstName, string &lastName, string &dateOfBirth, string &title) {

// Prompt the user to input the required values

cout << "Enter first name: ";

cin >> firstName;

cout << "Enter last name: ";

cin >> lastName;

cout << "Enter date of birth (dd/mm/yyyy): ";

cin >> dateOfBirth;

cout << "Enter title: ";

cin >> title;

}

void addEmployee(string firstName, string lastName, string dateOfBirth, string title) {

// Create a new employee object

Employee employee;

employee.firstName = firstName;

employee.lastName = lastName;

employee.dateOfBirth = dateOfBirth;

employee.title = title;

employee.isActive = true;

// Add the employee to the vector of employees

employees.push\_back(employee);

}

void filter(int filterChoice) {

// Filter the employees based on the selected filter

if (filterChoice == 1) {

// Filter by isActive as of a date

cout << "Enter a date (dd/mm/yyyy): ";

string date;

cin >> date;

for (int i = 0; i < employees.size(); i++) {

if (employees[i].isActive && employees[i].dateOfBirth < date) {

cout << employees[i].firstName << " " << employees[i].lastName << " (" << employees[i].dateOfBirth << ")" << endl;

}

}

} else if (filterChoice == 2) {

// Filter by title

cout << "Enter a title: ";

string title;

cin >> title;

for (int i = 0; i < employees.size(); i++) {

if (employees[i].title == title) {

cout << employees[i].firstName << " " << employees[i].lastName << " (" << employees[i].title << ")" << endl;

}

}

}

}

void searchEmployee(string kerberos) {

// Search for the employee with the given kerberos

for (int i = 0; i < employees.size(); i++) {

if (employees[i].kerberos == kerberos) {

cout << employees[i].firstName << " " << employees[i].lastName << " (" << employees[i].kerberos << ")" << endl;

return;

}

}

cout << "Employee not found." << endl;

}

void terminate(string kerberos) {

// Terminate the employee with the given kerberos

for (int i = 0; i < employees.size(); i++) {

if (employees[i].kerberos == kerberos) {

employees[i].isActive = false;

cout << "Employee terminated." << endl;

return;

}

}

cout << "Employee not found." << endl;

}

int main() {

cout<<"\n------------------Employee Management----------------------\n"<<endl;

cout<<"Press 1 to Add an Employee\n"<<endl;

cout<<"Press 2 to Search an Employee\n"<<endl;

cout<<"Press 3 to Filter Employees\n"<<endl;

cout<<"Press 4 to Terminate an Employee\n"<<endl;

int choice;

cin>>choice;

string firstName,lastName, dateOfBirth, title, kerberos;

switch(choice) {

case 1:

addEmployeeInput(firstName, lastName, dateOfBirth, title);

addEmployee(firstName, lastName, dateOfBirth, title);

break;

case 2:

cout<<"Enter a kerberos id to search: "<<endl;

cin>>kerberos;

searchEmployee(kerberos);

break;

case 3:

cout<<"Press 1 to filter by isActive as of a date\n"<<endl;

cout<<"Press 2 to filter by title\n"<<endl;

int filterChoice;

cin>>filterChoice;

filter(choice);

break;

case 4:

cout<<"Enter a kerberos id to terminate: "<<endl;

cin>>kerberos;

terminate(kerberos);

break;

default:

cout<<"Please select a valid choice"<<endl;

break;

}

}