Scenario:

Imagine you're developing a university management system. You have a base class named Person that stores basic information about individuals associated with the university, such as:

name (string)

id (int)

Question:

Design a class hierarchy using inheritance to represent different types of people within the university. Consider the following categories:

Student: Inherits from Person and has additional attributes like:

major (string)

gpa (double)

A method calculateSemesterGPA(vector<double> grades) that takes a vector of grades (doubles) and calculates the semester GPA.

Faculty: Inherits from Person and has additional attributes like:

department (string)

title (string) - e.g., "Professor", "Lecturer"

A method teachCourse(string courseName) that simulates assigning a faculty member to teach a specific course.

Additional Considerations:

You can introduce further derived classes if you think of more specific roles within the university (e.g., Staff, Administrator).

Think about access specifiers (public, private, protected) for member variables and methods in the base and derived classes.

Consider virtual functions (especially in the context of polymorphism) if there's common functionality that might have different implementations in derived classes.

Guiding Tips:

Focus on code clarity and maintainability.

Use meaningful variable and method names.

Add comments to explain your design choices.

Test your code to ensure it works as expected.

Explanation:

1. Create Base class person and this class contains common attributes name and id.
2. Create a derived class student which is inherits from person .There are addditional attributes department and tittle.There is a method calculatesemestergpa which calculate and sets student semester gpa.
3. Create a derived class faculty inherits from person which has additional attributes like department and tittle.Here is a method teachcourse which is assign a faculty member to teach a specific course.
4. Then we give protected access for name and id in the person class allow derived class to access the attributes directly.
5. Then method calculatesemestergpa demonstrate the functionality to the student class and method teachcourse demonstrate the functionality to the faculty class.

Code:

#include <iostream>

#include <vector>

#include <string>

using namespace std;

class Person {

protected:

string name;

int id;

public:

Person(string name, int id) : name(name), id(id) {}

string getName() const {

return name;

}

int getId() const {

return id;

}

void setName(string name) {

this->name = name;

}

void setId(int id) {

this->id = id;

}

};

class Student : public Person {

private:

string major;

double gpa;

public:

Student(string name, int id, string major, double gpa)

: Person(name, id), major(major), gpa(gpa) {}

string getMajor() const {

return major;

}

double getGpa() const {

return gpa;

}

void setMajor(string major) {

this->major = major;

}

void setGpa(double gpa) {

this->gpa = gpa;

}

double calculateSemesterGPA(vector<double> grades) {

double total = 0;

for(double grade : grades) {

total += grade;

}

gpa = total / grades.size();

return gpa;

}

};

class Faculty : public Person {

private:

string department;

string title;

public:

Faculty(string name, int id, string department, string title)

: Person(name, id), department(department), title(title) {}

string getDepartment() const {

return department;

}

string getTitle() const {

return title;

}

void setDepartment(string department) {

this->department = department;

}

void setTitle(string title) {

this->title = title;

}

void teachCourse(string courseName) {

cout << title << " " << name << " is teaching " << courseName << endl;

}

};

int main() {

Student student("Souvik", 12345, "ECE", 3.5);

vector<double> grades = {3.7, 3.9, 4.0, 3.8};

student.calculateSemesterGPA(grades);

cout << "Student " << student.getName() << " with ID " << student.getId()

<< " has a GPA of " << student.getGpa() << endl;

Faculty faculty("Baban", 67890, "Engineering", "Professor");

faculty.teachCourse("Data Structures");

cout << "Faculty " << faculty.getName() << " with ID " << faculty.getId()

<< " is from the " << faculty.getDepartment() << " department and holds the title of " << faculty.getTitle() << endl;

return 0;

}

Output:

