**Gợi ý Thiết kế cho Bài toán quản lý sinh viên (Student Management)**

Trong thực tế, có mối quan hệ n-n giữa 2 thực thể môn học (subject) và sinh viên (Student) vì:

* Một sinh viên học nhiều (n) môn học.
* Một môn học được nhiều sinh viên (n) theo học.

Như vậy, có mối quan hệ hợp tác (association) giữa hai lớp Subject và Student. Trong thiết kế OOP, mối quan hệ n-n này được thể hiện bằng một lớp mới được phát sinh, bảng điểm, transcript, và cấu trúc được gợi ý như sau:

**Gợi ý 1:**

**Class Subject**

String subID;

String subName

int credit

Contructor()

Getters/setters

**Class Student**

String stID

String fName

String lName

boolean gender

Date bDate;

String email;

String phone

Contructor()

Getters/setters

**Class Transcript**

Subject sObj

Student stObj;

Double labMark

Double testMark

Double finalMark

Contructor()

Getters/setters

// tính điểm trung bình

Double average()

Các danh sách của bài toán- dùng HashMap theo yêu cầu của đề bài:

**Class SubjectList**

Extends HashMap <String, Subject>

**Class StudentList**

Extends HashMap <String, Student>

**Class TranscriptList**

Extends ArrayList <Transcript>

Với thiết kế này, trong transcript đã có tham chiếu đến đối tượng Subject và Student có liên quan. Việc truy xuất các đối tượng liên quan này dễ dàng.

**Gợi ý 2:**

**Class Transcript**

String subID // mã môn

String stID// mã SV

Double labMark

Double testMark

Double finalMark

Contructor()

Getters/setters

// tính điểm trung bình

Double average()

**Class Student**

String stID

String fName

String lName

boolean gender

Date bDate;

String email;

String phone

Contructor()

Getters/setters

**Class Subject**

String subID;

String subName

int credit

Contructor()

Getters/setters

Các danh sách của bài toán - dùng HashMap theo yêu cầu của đề bài:

**Class SubjectList**

Extends HashMap <String, Subject>

**Class StudentList**

Extends HashMap <String, Student>

**Class TranscriptList**

Extends ArrayList <Transcript>

Với thiết kế này, trong transcript KHÔNG CÓ tham chiếu đến đối tượng Subject và Student có liên quan. Như vậy, từ String subID và String stID, chúng ta phải tốn chi phí cho tác vụ tìm kiếm mới xác định được các đối tượng có liên quan.

**Đánh giá: Thiết kế 1 tốt hơn thiết kế 2.**

**Một số gợi ý về phân tích hành vi:**

**Với các lớp cơ bản**: Chỉ cần hiện thực constructor và các getters/ setters. Không nên hiện thực các hành vi thêm/xóa/cập nhật gì cả.

**Với các lớp nhóm**: PHẢI hiện thực các hành vi thêm phần tử/ tìm phần tử/ xóa phần tử/ cập nhật phần tử/ duyệt xuất/ duyệt tính toán vì đây là các tác vụ cơ bản trên nhóm.

* Hành vi thêm phần tử: Nhập có kiểm tra validation các thành phần dữ liệu của phần tử . Thông qua constructor của lớp phần tử, tạo mới phần tử rồi thêm vpha62n tử này vào danh sách.
* Các hành vi khác trên nhóm cũng nên hiện thực theo suy nghĩa vừa được giới thiệu.

**Việc nhập dữ liệu ngày tháng, boolean, kiểm tra validation**

Xem phần hướng dẫn đã được để trong LMS/Lab211-SuTV