

ĐẠI HỌC QUỐC GIA THÀNH PHỐ HỒ CHÍ MINH
TRƯỜNG ĐẠI HỌC BÁCH KHOA



BÁO CÁO
LẬP TRÌNH NÂNG CAO (CO2039)

Đề tài:
ỨNG DỤNG QUẢN LÝ SINH VIÊN

Giáo viên hướng dẫn: LÊ ĐÌNH THUẬN

Sinh viên thực hiện: HOÀNG CHÍ NHÂN

MSSV: 2111898

Hồ Chí Minh, ngày 25 tháng 4 năm 2023

Mục lục

1. Giới thiệu.....	1
2. Hiện thực ứng dụng.....	2
2.1. Công cụ, phương pháp.....	2
2.1.1. Lập trình hệ thống - Java.....	2
2.1.2. Lưu trữ dữ liệu - SQLite.....	3
2.1.3. Giao diện đồ họa người dùng - Java Swing.....	3
2.2. Thiết kế.....	4
2.2.1. Yêu cầu tính năng.....	4
2.2.2. Class Diagrams.....	5
2.3. Xây dựng và kiểm thử.....	6
2.3.1. Mã nguồn.....	6
2.3.2. Kiểm thử chương trình.....	6
3. Kết quả.....	7
3.1. Tính năng của ứng dụng.....	7
3.1.1. Thông tin sinh viên.....	7
3.1.2. Thông tin giảng viên.....	8
3.1.3. Thông tin khóa học.....	9
3.1.4. Sinh viên tham gia khóa học.....	10
3.1.5. Bảng điểm sinh viên.....	10
3.2. Giao diện và hướng dẫn sử dụng.....	12
3.2.1. Trang sinh viên (Student).....	12
3.2.2. Trang giảng viên (Lecturer).....	13
3.2.3. Trang khóa học (Course).....	14
3.2.4. Trang tham gia khóa học (Enroll).....	15
3.2.5. Trang bảng điểm (Grade).....	16
Tài liệu tham khảo.....	17
Phụ lục.....	18
Các tập tin mã nguồn tiêu biểu:.....	18
Một số hình ảnh chức năng của ứng dụng:.....	26

1. Giới thiệu

Trong thời đại công nghệ số hiện nay, các ứng dụng quản lý sinh viên đã trở thành một công cụ hữu ích và không thể thiếu trong các trường đại học và cao đẳng. Quản lý thông tin của hàng ngàn sinh viên là một thách thức lớn đối với các trường và việc sử dụng các ứng dụng quản lý sinh viên có thể giúp đơn giản hóa quy trình quản lý, giảm thiểu sai sót và tiết kiệm thời gian cho các nhân viên quản lý. Với sự tiện lợi và hiệu quả mà nó mang lại, ứng dụng quản lý sinh viên là một giải pháp hoàn hảo cho các trường đại học và cao đẳng trong việc quản lý sinh viên của mình.

Thông qua Bài tập lớn môn Lập trình nâng cao (CO2039) niên khóa 2022-2023 của trường Đại học Bách Khoa - Đại học Quốc gia TP.HCM (HCMUT), tác giả xin giới thiệu về ứng dụng *Quản lý sinh viên* được xây dựng với các tính năng cơ bản và có triển vọng mở rộng thêm trong tương lai.

Các thành phần tạo nên ứng dụng bao gồm:

- Hệ thống được thiết kế bằng ngôn ngữ lập trình Java, chạy trên Java Virtual Machine (JVM).
- Hệ cơ sở dữ liệu được xây dựng bằng SQLite.
- Giao diện đồ họa người dùng (GUI) được thiết kế với Java Swing - thư viện GUI được tích hợp sẵn Java.

2. Hiện thực ứng dụng

2.1. Công cụ, phương pháp

2.1.1. Lập trình hệ thống - Java

Ngôn ngữ lập trình Java® là ngôn ngữ hướng đối tượng. Java được thiết kế đơn giản để nhiều lập trình viên có thể sử dụng thành thạo ngôn ngữ này. Ngôn ngữ lập trình Java có liên quan đến C và C++ nhưng được thiết kế khá khác biệt, với một số khía cạnh của C và C++ bị lược bỏ và đưa vào thêm một vài ý tưởng từ các ngôn ngữ khác. Nó được dự định là ngôn ngữ sản xuất, không phải ngôn ngữ nghiên cứu, vì vậy, như C. A. R. Hoare đã đề xuất trong bài báo cổ điển của ông về thiết kế ngôn ngữ, thiết kế của Java đã tránh đưa vào các tính năng mới và chưa được kiểm nghiệm.

Ngôn ngữ lập trình Java có kiểu dữ liệu mạnh và tĩnh. Điều này phân biệt rõ ràng giữa các lỗi biên dịch có thể và phải được phát hiện tại thời điểm biên dịch và các lỗi xảy ra trong thời gian chạy. Biên dịch thường bao gồm việc dịch các chương trình thành ngôn ngữ máy. Các hoạt động trong thời gian chạy bao gồm tải và liên kết các lớp cần thiết để thực thi chương trình, tạo “mã máy tùy chọn” (optional machine code) và “tối ưu hóa động” (dynamic optimization) của chương trình cũng như thực thi chương trình trong thực tế.

Ngôn ngữ lập trình Java là ngôn ngữ cấp cao tương đối, trong đó các chi tiết về “biểu diễn máy” (machine representation) không có sẵn thông qua ngôn ngữ này. Java bao gồm quản lý lưu trữ tự động, điển hình là sử dụng trình thu gom rác, để tránh các vấn đề an toàn của việc phân bổ rõ ràng (như C free hoặc C++ delete). Việc triển khai thu gom rác hiệu suất cao có thể có các khoảng dừng giới hạn để hỗ trợ lập trình hệ thống và các ứng dụng thời gian thực. Ngôn ngữ không bao gồm bất kỳ cấu trúc không an toàn, chẳng hạn như truy cập mảng mà không kiểm tra chỉ số (index), vì các cấu trúc không an toàn như vậy sẽ khiến chương trình hoạt động theo cách không xác định.

Ngôn ngữ lập trình Java thường được biên dịch thành tập lệnh mã byte và định dạng nhị phân được xác định trong *The Java Virtual Machine Specification, Java SE 20 Edition*.¹

2.1.2. Lưu trữ dữ liệu - SQLite

SQLite là một thư viện ngôn ngữ C triển khai một công cụ cơ sở dữ liệu SQL nhỏ, nhanh, khép kín, có độ tin cậy cao và đầy đủ tính năng. SQLite là hệ thống quản lý cơ sở dữ liệu được sử dụng nhiều nhất trên thế giới. SQLite được tích hợp vào tất cả các điện thoại di động và hầu hết các máy tính, đi kèm với vô số ứng dụng khác mà mọi người sử dụng hàng ngày.

SQLite là một hệ thống quản lý cơ sở dữ liệu SQL nhưng. Không giống như hầu hết các cơ sở dữ liệu SQL khác, SQLite không có quy trình máy chủ riêng biệt. SQLite đọc và ghi trực tiếp vào các tệp đĩa thông thường. Định dạng tệp cơ sở dữ liệu là đa nền tảng, có thể tự do sao chép cơ sở dữ liệu giữa các hệ thống 32-bit và 64-bit hoặc giữa các kiến trúc big-endian và little-endian. Các tính năng này làm cho SQLite trở thành lựa chọn phổ biến dưới dạng “Định dạng tệp ứng dụng” (Application File Format). Tệp cơ sở dữ liệu SQLite là định dạng lưu trữ được đề xuất bởi US Library of Congress.²

2.1.3. Giao diện đồ họa người dùng - Java Swing

Java Swing là bộ công cụ giao diện đồ họa người dùng Java (GUI) nhẹ bao gồm nhiều thành phần giao diện phong phú. Swing là một phần của Java Foundation Classes (JFC) và bao gồm một số “gói” (package) để phát triển các ứng dụng máy tính phong phú trong Java. Swing bao gồm các điều khiển tích hợp sẵn như nút hình ảnh, ngăn theo thẻ, thanh trượt, thanh công cụ, bộ chọn màu, bảng và vùng văn bản. Các thành phần của Swing được viết hoàn toàn bằng Java và do đó đặt được thuộc tính “không phụ thuộc nền tảng” (platform-independent).³

¹ Gosling, James; Joy, Bill; Steele, Guy; and Bracha, Gilad. [The Java Language Specification, 2nd Edition](#).

² [About SQLite](#). [sqlite.org](#). Truy cập 16/4/2023

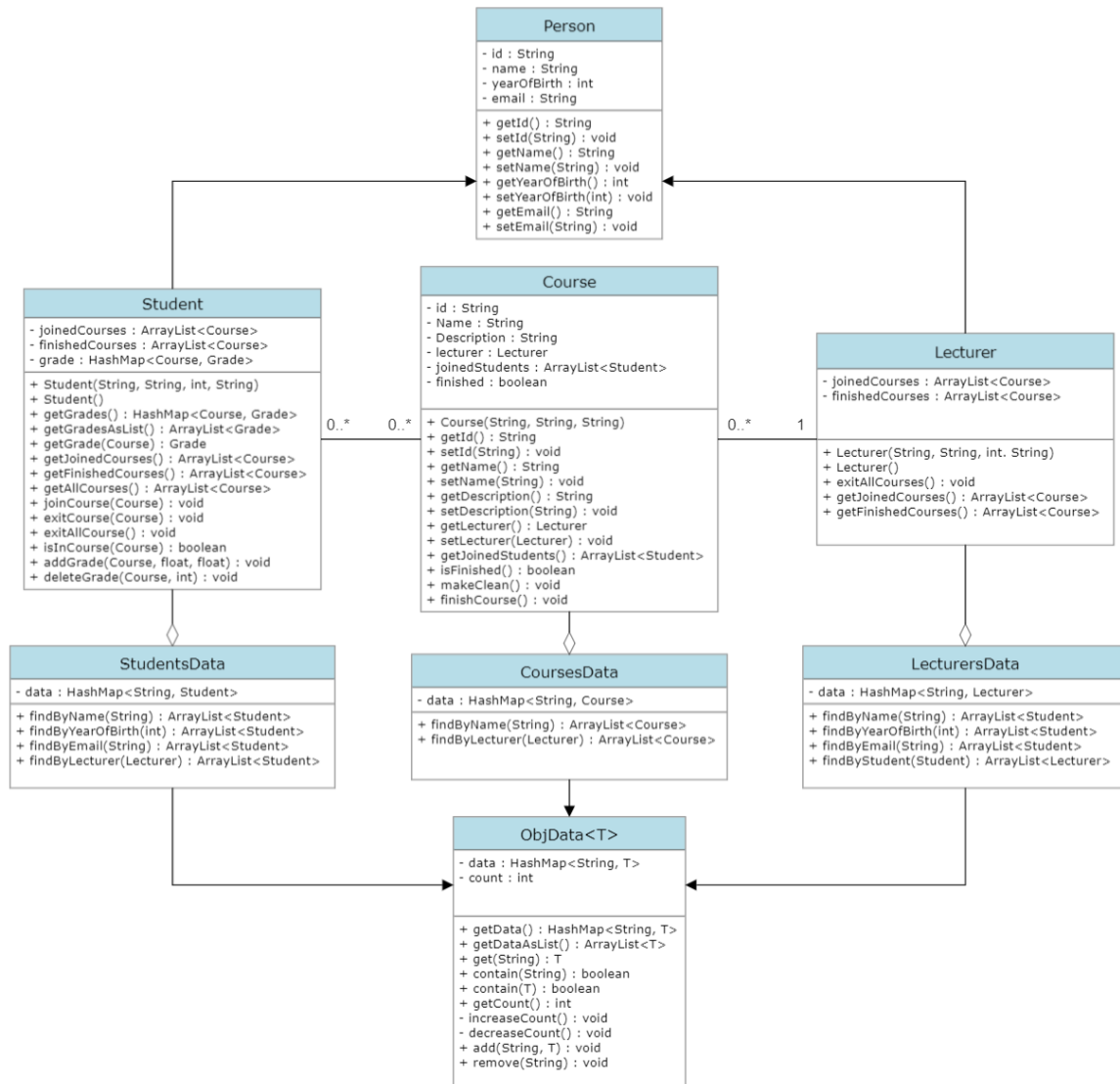
³ [What is Java Swing? - Definition from Techopedia](#). Techopedia Inc. Truy cập 16/4/2023.

2.2. Thiết kế

2.2.1. Yêu cầu tính năng

- ☒ Thêm, xóa, chỉnh sửa thông tin sinh viên.
- ☒ Thêm, xóa, chỉnh sửa thông tin giảng viên.
- ☒ Thêm, xóa, chỉnh sửa thông tin khóa học.
- ☒ Thêm, thay đổi giảng viên của khóa học.
- ☒ Thêm hoặc xóa sinh viên tham gia khóa học.
- ☒ Thêm điểm của sinh viên trong khóa học.
- ☒ Tính điểm trung bình của sinh viên.
- ☒ Lọc sinh viên theo tên, năm sinh, email, giảng viên hay khóa học.
- ☒ Lọc giảng viên theo tên, năm sinh, email hay sinh viên.
- ☒ Lọc khóa học theo tên, sinh viên hay giảng viên.
- ☒ Kết thúc một khóa học.
- ☐ Sao lưu và phục hồi dữ liệu.
- ☐ Phân quyền người dùng ứng dụng.

2.2.2. Class Diagrams



2.3. Xây dựng và kiểm thử

2.3.1. Mã nguồn

Mã nguồn của ứng dụng được lưu tại [github/chinhan1405/StudentManager_Java](https://github.com/chinhan1405/StudentManager_Java).

Xem một vài class chính của ứng dụng tại [phụ lục](#).

2.3.2. Kiểm thử chương trình

Phương pháp:

- Sử dụng Java Virtual Machine (JVM) để chạy thử chương trình.
- Thử tất cả các tính năng của ứng dụng (lưu, sửa, xóa sinh viên, ...).

Kết quả: Tất cả test case đã chạy thành công và không có lỗi nào được phát hiện.

Kết luận: Phần kiểm thử đã chứng minh rằng ứng dụng Quản lý sinh viên hoạt động tốt và sẵn sàng để đưa vào sử dụng trong thực tiễn.

3. Kết quả

3.1. Tính năng của ứng dụng

3.1.1. Thông tin sinh viên

Thông tin chung của sinh viên có 4 trường chính, bao gồm:

- ID (Identification): Mã ký tự độc nhất của từng sinh viên để phân biệt các sinh viên với nhau. ID là công cụ chính để chọn sinh viên nhằm mục đích chỉnh sửa, xóa, đăng ký khóa học hay ghi điểm cho sinh viên.
- Tên sinh viên (Name): Bao gồm tên đầy đủ của sinh viên.
- Năm sinh (Year Of Birth).
- Email sinh viên (Email).

Ngoài ra, mỗi sinh viên có một danh sách các khóa học đã đăng ký (joined courses), một danh sách các khóa học đã kết thúc (finished courses) và một danh sách điểm của các khóa học.

Các tính năng hệ thống cung cấp đối với thông tin sinh viên bao gồm:

- Thêm sinh viên: Người dùng cần cung cấp các trường ID, Name, Year Of Birth và Email để thêm một sinh viên mới.
- Xóa sinh viên: Thông qua trường ID, người dùng có thể xác định sinh viên cần xóa.
- Chỉnh sửa thông tin sinh viên: Người dùng có thể sửa đổi các thông tin như tên sinh viên (Name), năm sinh (Year Of Birth) hoặc email sinh viên (Email). Thông qua trường ID, người dùng xác định sinh viên cần chỉnh sửa thông tin.
- Hiển thị tất cả sinh viên.
- Tìm sinh viên theo tên.
- Tìm sinh viên theo năm sinh.
- Tìm sinh viên theo email.

- Tìm sinh viên theo giảng viên. Người dùng cần cung cấp ID của giảng viên để tìm các sinh viên theo học giảng viên đó.
- Tìm sinh viên theo khoá học. Người dùng cần cung cấp ID của khóa học để tìm các sinh viên theo học khóa học đó.

3.1.2. Thông tin giảng viên

Thông tin chung của giảng viên có 4 trường chính, bao gồm:

- ID (Identification): Mã ký tự độc nhất của từng giảng viên để phân biệt các giảng viên với nhau. ID là công cụ chính để chọn giảng viên nhằm mục đích chỉnh sửa, xóa hay thêm khóa học cho giảng viên.
- Tên giảng viên (Name): Bao gồm tên đầy đủ của giảng viên.
- Năm sinh (Year Of Birth).
- Email giảng viên (Email).

Ngoài ra, mỗi giảng viên có một danh sách các khóa học đã tham gia (joined courses) và một danh sách các khóa học đã kết thúc (finished courses).

Các tính năng hệ thống cung cấp đối với thông tin giảng viên bao gồm:

- Thêm giảng viên: Người dùng cần cung cấp các trường ID, Name, Year Of Birth và Email để thêm một giảng viên mới.
- Xóa giảng viên: Thông qua trường ID, người dùng có thể xác định giảng viên cần xóa.
- Chỉnh sửa thông tin giảng viên: Người dùng có thể sửa đổi các thông tin như tên giảng viên (Name), năm sinh (Year Of Birth) hoặc email giảng viên (Email). Thông qua trường ID, người dùng xác định giảng viên cần chỉnh sửa thông tin.
- Hiển thị tất cả giảng viên.
- Tìm giảng viên theo tên.
- Tìm giảng viên theo năm sinh.
- Tìm giảng viên theo email.

- Tìm giảng viên theo sinh viên. Người dùng cần cung cấp ID của sinh viên để tìm giảng viên mà sinh viên đó theo học.

3.1.3. Thông tin khóa học

Thông tin chung của khóa học có 3 trường chính, bao gồm:

- ID (Identification): Mã ký tự độc nhất của từng giảng viên để phân biệt các khóa học. ID là công cụ chính để chọn khóa học nhằm mục đích chỉnh sửa, xóa, thêm giảng viên, đăng ký sinh viên hay kết thúc khóa học đó.
- Tên khóa học (Name).
- Mô tả khóa học (Description).
- Giảng viên của khóa học (Lecturer).

Ngoài ra, mỗi khóa học có một danh sách các sinh viên tham gia (enrolled students).

Các tính năng hệ thống cung cấp đối với thông tin khóa học bao gồm:

- Thêm khóa học: Người dùng cần cung cấp các trường ID, Name, Description để thêm một khóa học mới.
- Xóa khóa học: Thông qua trường ID, người dùng có thể xác định khóa học cần xóa.
- Chỉnh sửa thông tin khóa học: Người dùng có thể sửa đổi các thông tin như tên khóa học (Name) và mô tả khóa học (Description). Thông qua trường ID, người dùng xác định khóa học cần chỉnh sửa thông tin.
- Hiển thị tất cả khóa học.
- Tìm khóa học theo tên.
- Tìm khóa học theo giảng viên. Người dùng cần cung cấp ID của giảng viên để tìm khóa học mà giảng viên đó tham gia.
- Thêm giảng viên cho khóa học: Thông qua ID giảng viên được cung cấp, hệ thống sẽ thêm giảng viên đó vào khóa học.
- Tìm khóa học theo sinh viên. Người dùng cần cung cấp ID của sinh viên để tìm khóa học mà sinh viên đó theo học.

- Kết thúc khóa học.

3.1.4. Sinh viên tham gia khóa học

Thông tin sinh viên tham gia khóa học là liên kết giữa một sinh viên và một khóa học thông qua ID.

Các tính năng hệ thống cung cấp đối với thông tin đăng ký khóa học bao gồm:

- Thêm sinh viên vào khóa học. Người dùng cung cấp ID sinh viên và ID khóa học để hệ thống thêm sinh viên vào khóa học đó.
- Xóa sinh viên khỏi khóa học. Bằng ID sinh viên và ID khóa học, hệ thống xác định được khóa học và sinh viên cần xóa khỏi khóa học đó.
- Hiển thị tất cả các mối liên kết bao gồm ID sinh viên, ID khóa học và tình trạng khóa học.
- Tìm theo sinh viên. Kết quả sẽ trả về những liên kết sinh viên - khóa học mà sinh viên có ID được cung cấp tham gia.
- Tìm theo khóa học. Kết quả sẽ trả về những liên kết sinh viên - khóa học theo ID của khóa học được cung cấp.

3.1.5. Bảng điểm sinh viên

Bảng điểm của một sinh viên bao gồm:

- Khóa học (course).
- Danh sách các cặp (điểm - hệ số) trong khóa học của sinh viên.

Các tính năng hệ thống cung cấp đối với bảng điểm sinh viên bao gồm:

- Thêm cột điểm cho sinh viên. Người dùng cung cấp ID sinh viên và ID khóa học để xác định khóa học cần ghi điểm. Điểm (Point) và hệ số (Coef) tương ứng sẽ được lưu vào bảng điểm của sinh viên.
- Xóa cột điểm của sinh viên.
- Hiển thị bảng điểm của tất cả sinh viên.

- Hiển thị bảng điểm của một sinh viên. Người dùng cần cung cấp ID sinh viên để xác định sinh viên cần xem bảng điểm.
- Hiển thị điểm trung bình các môn và điểm trung bình tích lũy của sinh viên thông qua ID sinh viên người dùng cung cấp.

3.2. Giao diện và hướng dẫn sử dụng⁴

Yêu cầu hệ thống đã cài đặt Java Virtual Machine (JVM).

Khởi động ứng dụng bằng cách mở file *StudentManager_Java.jar* bằng JVM.

Tip: Để reset dữ liệu, người dùng chỉ cần xóa file *studentmanagerdata.db*.

3.2.1. Trang sinh viên (Student)

ID	Name	YearOfBirth	Email
1	Emily Smith	2004	emilysmith2004@email.com
2	William Johnson	2003	wjohnson2003@email.com
3	Sophia Garcia	2005	sophiagarcia2005@email.com
4	Michael Rodriguez	2004	mrodriguez2004@email.com
5	Isabella Hernandez	2003	isabellaahernandez2003@em...
6	Ethan Martinez	2004	ethanmartinez2004@email.c...
7	Mia Davis	2005	miadavis2005@email.com
8	Alexander Wilson	2003	awilson2003@email.com
9	Olivia Anderson	2004	oliviaanderson2004@email.c...
10	Daniel Thomas	2003	danielthomas2003@email.com

- **Thêm sinh viên:** Điền đầy đủ thông tin vào các ô *ID*, *Name* (tên sinh viên), *Year of Birth* (năm sinh) và *Email*, nhấn nút ADD để thêm sinh viên mới.
- **Xóa sinh viên:** Nhập ID của sinh viên vào ô *ID* và nhấn nút REMOVE. Các ô khác không ảnh hưởng đến kết quả.
- **Sửa thông tin sinh viên:** Nhập ID của sinh viên cần sửa vào ô *ID*, điền thông tin cần sửa vào ô *Name*, *Year of Birth* hay *Email* tương ứng và nhấn nút UPDATE. Lưu ý đối với những trường thông tin không cần thay đổi, người dùng có thể để trống.
- **Hiển thị tất cả sinh viên:** Nhấn nút SHOW ALL.
- **Tìm sinh viên theo tên:** Nhập tên sinh viên muốn tìm vào ô *Name*, nhấn nút FIND BY NAME.

⁴ Dữ liệu trong hình minh họa đều không có trong thực tế, được [ChatGPT](#) tạo ra một cách ngẫu nhiên.

- **Tìm sinh viên theo năm sinh:** Nhập năm sinh muốn tìm vào ô *Year of Birth*, nhấn nút FIND BY YEAR.
- **Tìm sinh viên theo email:** Nhập email sinh viên vào ô *Email*, nhấn nút FIND BY EMAIL.
- **Tìm sinh viên theo giảng viên:** Nhập ID giảng viên vào ô *Lecturer ID*, nhấn nút FIND BY LECTURER.
- **Tìm sinh viên theo khóa học:** Nhập ID khóa học vào ô *Course ID*, nhấn nút FIND BY COURSE.

3.2.2. Trang giảng viên (Lecturer)

The screenshot shows the 'Student Manager' application with the 'Lecturer' tab selected. On the left, there are input fields for 'ID:', 'Name:', 'Year of Birth:', and 'Email:'. Below these are buttons: 'ADD', 'REMOVE', 'UPDATE', 'SHOW ALL', 'FIND BY NAME', 'FIND BY YEAR', 'FIND BY EMAIL', and 'FIND BY STUDENT'. The 'FIND BY STUDENT' button is disabled. On the right, a table displays the following data:

ID	Name	YearOfBirth	Email
1	Dr. Amanda Carter	1980	acarter1980@university.edu
2	Professor John Lee	1965	jlee1965@university.edu
3	Dr. Elizabeth Evans	1972	eevans1972@university.edu
4	Professor David ...	1958	dsmith1958@university.edu

- **Thêm giảng viên:** Điền đầy đủ thông tin vào các ô *ID*, *Name* (tên giảng viên), *Year of Birth* (năm sinh) và *Email*, nhấn nút ADD để thêm giảng viên mới.
- **Xóa giảng viên:** Nhập ID của giảng viên vào ô *ID* và nhấn nút REMOVE. Các ô khác không ảnh hưởng đến kết quả.
- **Sửa thông tin giảng viên:** Nhập ID của giảng viên cần sửa vào ô *ID*, điền thông tin cần sửa vào ô *Name*, *Year of Birth* hay *Email* tương ứng và nhấn nút UPDATE. Lưu ý đối với những trường thông tin không cần thay đổi, người dùng có thể để trống.
- **Hiển thị tất cả giảng viên:** Nhấn nút SHOW ALL.

- **Tìm giảng viên theo tên:** Nhập tên giảng viên muốn tìm vào ô *Name*, nhấn nút FIND BY NAME.
- **Tìm giảng viên theo năm sinh:** Nhập năm sinh muốn tìm vào ô *Year of Birth*, nhấn nút FIND BY YEAR.
- **Tìm giảng viên theo email:** Nhập email giảng viên vào ô *Email*, nhấn nút FIND BY EMAIL.
- **Tìm giảng viên theo sinh viên:** Nhập ID sinh viên vào ô *Student ID*, nhấn nút FIND BY STUDENT.

3.2.3. Trang khóa học (Course)

ID	Name	Description	Lecturer ID	Finished
1	Introducti...	This cours...	1	true
2	Principles ...	This cours...	2	false
3	Financial ...	This cours...	2	false
4	Macroeco...	This cours...	4	false
5	Social Psy...	This cours...	3	false
6	Introducti...	This cours...	4	false

- **Thêm khóa học:** Điền đầy đủ thông tin vào các ô *ID*, *Name* (tên khóa học) và *Description* (mô tả), nhấn nút ADD để thêm khóa học mới.
- **Xóa khóa học:** Nhập ID của khóa học vào ô *ID* và nhấn nút REMOVE. Các ô khác không ảnh hưởng đến kết quả.
- **Sửa thông tin khóa học:** Nhập ID khóa học cần sửa vào ô *ID*, điền thông tin cần sửa vào ô *Name* và *Description* tương ứng và nhấn nút UPDATE. Lưu ý đối với những trường thông tin không cần thay đổi, người dùng có thể để trống.
- **Hiển thị tất cả khóa học:** Nhấn nút SHOW ALL.
- **Tìm khóa học theo tên:** Nhập tên khóa học muốn tìm vào ô *Name*, nhấn nút FIND BY NAME.

- **Tìm khóa học theo giảng viên:** Nhập ID giảng viên vào ô *Lecturer ID*, nhấn nút FIND BY EMAIL.
- **Thêm giảng viên vào khóa học:** Nhập ID giảng viên vào ô *Lecturer ID*, nhập ID khóa học vào ô *ID*, nhấn SET LECTURER.
- **Tìm giảng viên theo sinh viên:** Nhập ID sinh viên vào ô *Student ID*, nhấn nút FIND BY STUDENT.
- **Kết thúc khóa học:** Nhập ID khóa học vào ô ID, nhấn nút FINISH COURSE. Khóa học khi kết thúc sẽ không thể thêm, xóa sinh viên hay thay đổi giảng viên.

3.2.4. Trang tham gia khóa học (Enroll)

Student ID	Course ID	Finished
1	2	false
1	3	false
1	4	false
1	6	false
1	1	true
2	3	false
2	5	false
2	6	false
3	2	false
3	3	false
3	5	false
3	1	true
4	2	false
4	3	false
4	6	false
4	1	true
5	2	false
5	3	false
5	5	false
5	6	false
5	1	true
6	4	false
6	5	false
6	6	false
7	3	false

- **Đăng ký sinh viên vào khóa học:** Nhập ID sinh viên vào ô *Student ID*, nhập ID khóa học vào ô *Course ID*, nhấn nút ENROLL để đăng ký.
- **Xóa đăng ký:** Nhập index (thứ tự khóa học mà sinh đăng ký) vào ô *Student ID* và nhấn nút REMOVE. Các ô khác không ảnh hưởng đến kết quả.
- **Hiển thị tất cả các đăng ký:** Nhấn nút SHOW ALL.
- **Hiển thị các đăng ký theo sinh viên:** Nhập ID sinh viên vào ô *Student ID* và nhấn nút FIND BY STUDENT.
- **Hiển thị các đăng ký theo khóa học:** Nhập ID khóa học vào ô *Course ID* và nhấn nút FIND BY COURSE.

3.2.5. Trang bảng điểm (Grade)

Student ID	Course ID	point	coef
1	1	8.5	0.3
1	1	9.0	0.7
1	2	6.0	0.2
1	2	4.0	0.3
1	2	9.0	0.5
3	1	9.0	0.3
3	1	6.0	0.7
3	2	8.0	0.2
3	2	8.0	0.3
3	2	5.0	0.5

- **Thêm cột điểm:** Nhập ID sinh viên vào ô *Student ID*, nhập ID khóa học vào ô *Course ID*, nhập điểm vào ô *Point*, nhập hệ số vào ô *Coef* và nhấn nút ADD để thêm cột điểm.
- **Xóa đăng ký:** Nhập index (thứ tự khóa học mà sinh đăng ký) vào ô *Student ID* và nhấn nút REMOVE. Các ô khác không ảnh hưởng đến kết quả.
- **Hiển thị điểm của tất cả sinh viên:** Nhấn nút SHOW ALL.
- **Hiển thị điểm của một sinh viên:** Nhập ID sinh viên vào ô *Student ID* và nhấn nút STUDENT GRADES.
- **Hiển thị điểm trung bình của sinh viên:** Nhập ID sinh viên vào ô *Student ID* và nhấn nút STUDENT AVERAGE. Điểm trung bình tích lũy của sinh viên sẽ hiện lên ở dòng đầu tiên, là điểm trung bình các khóa học đã kết thúc của sinh viên.

Tài liệu tham khảo

1. Gosling, James; Joy, Bill; Steele, Guy; and Bracha, Gilad. “[The Java Language Specification, 2nd Edition](#)”.
2. [About SQLite](#). sqlite.org. Truy cập 16/4/2023.
3. [What is Java Swing? - Definition from Techopedia](#). Techopedia Inc. Truy cập 16/4/2023

Phụ lục

Các tập tin mã nguồn tiêu biểu:

Person.java
<pre>package StudentManager; public class Person { private String id; private String name; private int yearOfBirth; private String email; public String getId() { return id; } public void setId(String id) { this.id = id; } public String getName() { return name; } public void setName(String name) { this.name = name; } public int getYearOfBirth() { return yearOfBirth; } public void setYearOfBirth(int yearOfBirth) { this.yearOfBirth = yearOfBirth; } public String getEmail() { return email; } public void setEmail(String email) { this.email = email; } }</pre>

```
}
```

Student.java

```
package StudentManager;
import java.util.ArrayList;
import java.util.HashMap;

public class Student extends Person {
    private ArrayList<Course> joinedCourses = new ArrayList<>();
    private ArrayList<Course> finishedCourses = new ArrayList<>();
    private HashMap<Course, Grade> grades = new HashMap<>();

    public Student(String id, String name, int yearOfBirth, String email) {
        setId(id);
        setName(name);
        setYearOfBirth(yearOfBirth);
        setEmail(email);
    }

    public Student() {
        this("", "", 0, "");
    }

    public HashMap<Course, Grade> getGrades() {
        return grades;
    }

    public ArrayList<Grade> getGradeAsList() {
        ArrayList<Grade> gradesList = new ArrayList<>();
        for (Course key : this.grades.keySet()) {
            gradesList.add(grades.get(key));
        }
        return gradesList;
    }

    public Grade getGrade(Course course) {
        return this.grades.get(course);
    }

    public ArrayList<Course> getJoinedCourses() {
```

```

        return joinedCourses;
    }

    public ArrayList<Course> getFinishedCourses() {
        return finishedCourses;
    }

    public ArrayList<Course> getAllCourse() {
        ArrayList<Course> courses = new ArrayList<>();
        courses.addAll(getJoinedCourses());
        courses.addAll(getFinishedCourses());
        return courses;
    }

    public void joinCourse(Course course) {
        if (!course.getJoinedStudents().contains(this)) {
            this.joinedCourses.add(course);
            course.getJoinedStudents().add(this);
            this.grades.put(course, new Grade(course));
        }
    }

    public void exitCourse(Course course) {
        if (course.getJoinedStudents().contains(this)) {
            this.joinedCourses.remove(course);
            course.getJoinedStudents().remove(this);
            grades.remove(course);
        }
    }

    public void exitAllCourse() {
        for (Course course : getJoinedCourses()) {
            course.getJoinedStudents().remove(this);
            grades.remove(course);
        }
        for (Course course : getFinishedCourses()) {
            course.getJoinedStudents().remove(this);
            grades.remove(course);
        }
    }

    public boolean isInCourse(Course course) {
        return joinedCourses.contains(course);
    }

```

```

public void addGrade(Course course, float point, float coefficient) {
    this.getGrade(course).addPoint(point, coefficient);
}

public void deleteGrade(Course course, int index) {
    this.getGrade(course).deletePoint(index);
}

public float getAllAverage() {
    if (finishedCourses.size()==0) return 0;
    float sum = 0f;
    for (Course course : finishedCourses) {
        sum += grades.get(course).average();
    }
    return sum / (float) finishedCourses.size();
}
}

```

Lecturer.java

```

package StudentManager;
import java.util.ArrayList;

public class Lecturer extends Person{
    private ArrayList<Course> joinedCourses = new ArrayList<>();
    private ArrayList<Course> finishedCourses = new ArrayList<>();

    public Lecturer(String id, String name, int yearOfBirth, String email) {
        setId(id);
        setName(name);
        setYearOfBirth(yearOfBirth);
        setEmail(email);
    }

    public Lecturer() {
        this("", "", 0, "");
    }

    public void exitAllCourse() {
        for (Course course : getJoinedCourses()) {
            course.setLecturer(new Lecturer());
        }
    }
}

```

```

    }
    for (Course course : getFinishedCourses()) {
        course.setLecturer(new Lecturer());
    }
}

public ArrayList<Course> getJoinedCourses() {
    return joinedCourses;
}

public ArrayList<Course> getFinishedCourses() {
    return finishedCourses;
}

public void joinCourse(Course course) {
    if (!this.joinedCourses.contains(course)) {
        this.joinedCourses.add(course);
        course.setLecturer(this);
    }
}
}
}

```

Course.Java

```

package StudentManager;
import java.util.ArrayList;

public class Course {
    private String id;
    private String name;
    private String description;
    private Lecturer lecturer;
    private ArrayList<Student> joinedStudents = new ArrayList<>();
    private boolean finished;

    public Course(String id, String name, String description) {
        setId(id);
        setName(name);
        setDescription(description);
        setLecturer(new Lecturer());
        this.finished = false;
    }
}

```



```

    }

    public String getId() {
        return id;
    }

    public void setId(String id) {
        this.id = id;
    }

    public String getName() {
        return name;
    }

    public void setName(String name) {
        this.name = name;
    }

    public String getDescription() {
        return description;
    }

    public void setDescription(String description) {
        this.description = description;
    }

    public Lecturer getLecturer() {
        return lecturer;
    }

    public void setLecturer(Lecturer lecturer) {
        this.lecturer = lecturer;
        lecturer.getJoinedCourses().add(this);
    }

    public boolean isFinished() {
        return finished;
    }

    public void makeClean() {
        if (finished) {
            getLecturer().getFinishedCourses().remove(this);
            for (Student student : getJoinedStudents()) {
                student.getFinishedCourses().remove(this);
            }
        }
    }

```

```

    }
    else {
        getLecturer().getJoinedCourses().remove(this);
        for (Student student : getJoinedStudents()) {
            student.getJoinedCourses().remove(this);
        }
    }
}

public void finishCourse() {
    this.finished = true;
    this.lecturer.getJoinedCourses().remove(this);
    this.lecturer.getFinishedCourses().add(this);
    for (Student student : getJoinedStudents()) {
        student.getJoinedCourses().remove(this);
        student.getFinishedCourses().add(this);
    }
}

public ArrayList<Student> getJoinedStudents() {
    return joinedStudents;
}
}

```

Grade.java

```

package StudentManager;
import java.util.ArrayList;

public class Grade {
    private Course course;
    private ArrayList<Pair> points;

    public Grade(Course course) {
        this.course = course;
        this.points = new ArrayList<>();
    }

    public Course getCourse() {
        return course;
    }
}

```

```

public void setCourse(Course course) {
    this.course = course;
}

public ArrayList<Pair> getPoints() {
    return points;
}

public void addPoint(float point, float coefficient) {
    this.points.add(new Pair(point, coefficient));
}

public void deletePoint(int index) {
    this.points.remove(index);
}

public float average() {
    float sum = 0, sumCoef = 0;
    for (Pair pair : this.points) {
        sum += pair.point * pair.coefficient;
        sumCoef += pair.coefficient;
    }
    if (sumCoef==0) return 0;
    return sum/sumCoef;
}

}

class Pair {
    public float point;
    public float coefficient;

    public Pair(float point, float coefficient) {
        this.point = point;
        this.coefficient = coefficient;
    }
}

```

Một số hình ảnh chức năng của ứng dụng:

Tìm sinh viên theo tên:

The screenshot shows the 'Student Manager' application window. The 'Student' tab is selected. On the left, there are input fields for 'ID:', 'Name:', 'Year of Birth:', and 'Email:'. The 'Name' field contains 'Mia Davis'. Below these fields are buttons for 'ADD', 'REMOVE', 'UPDATE', 'SHOW ALL', 'FIND BY NAME', 'FIND BY YEAR', and 'FIND BY EMAIL'. Further down are 'Lecturer ID:', 'FIND BY LECTURER', 'Course ID:', and 'FIND BY COURSE'. On the right, a table displays the search results:

ID	Name	YearOfBirth	Email
6	Mia Davis	2005	miadavis200...

Tìm sinh viên theo năm sinh:

The screenshot shows the 'Student Manager' application window. The 'Student' tab is selected. On the left, the 'Year of Birth' field is filled with '2005'. The 'FIND BY YEAR' button is highlighted. On the right, a table displays the search results:

ID	Name	YearOfBirth	Email
2	Sophia Garcia	2005	sophiagarcia...
6	Mia Davis	2005	miadavis200...

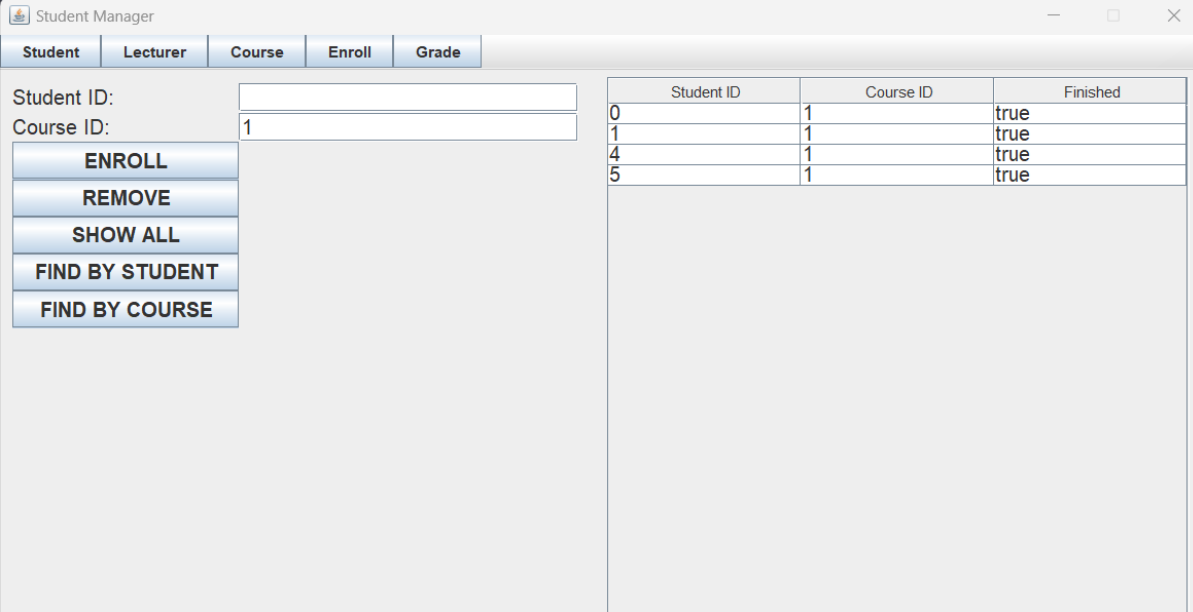
Tìm sinh viên theo email:

ID	Name	YearOfBirth	Email
4	Isabella Hern...	2003	isabellaherna...

Tìm khóa học theo giảng viên:

ID	Name	Description	Lecturer ID	Finished
0	Introducti...	This cours...	1	false
1	Principles ...	This cours...	1	true

Tìm các sinh viên tham gia một khóa học:



Student Manager

Student Lecturer Course **Enroll** Grade

Student ID:

Course ID:

ENROLL

REMOVE

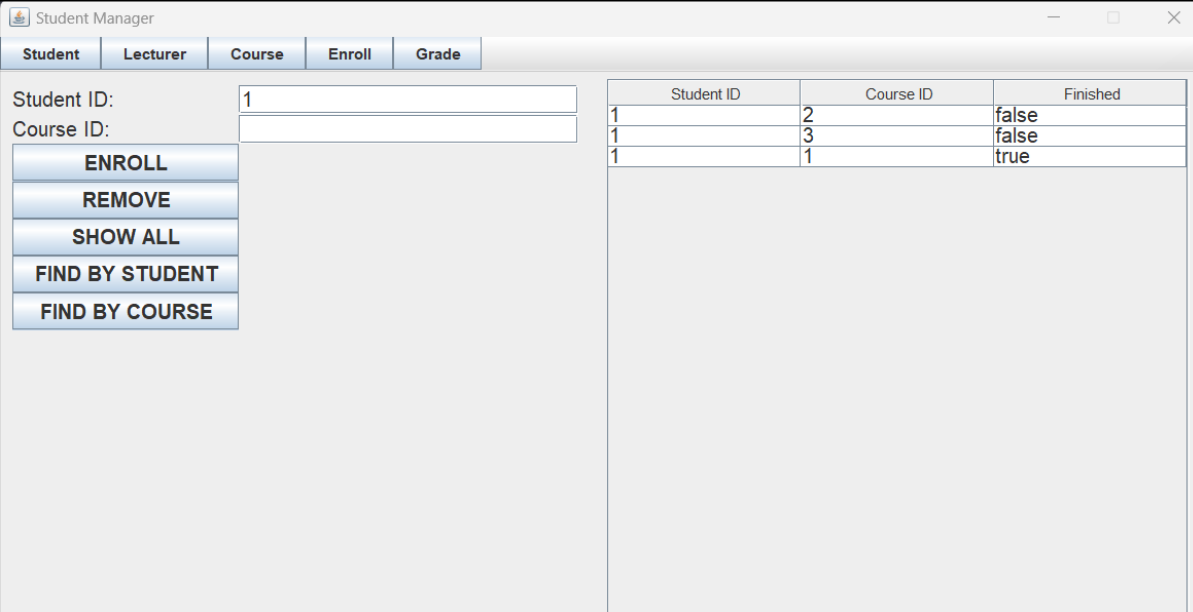
SHOW ALL

FIND BY STUDENT

FIND BY COURSE

Student ID	Course ID	Finished
0	1	true
1	1	true
4	1	true
5	1	true

Tìm các khóa học mà sinh viên tham gia:



Student Manager

Student Lecturer Course **Enroll** Grade

Student ID:

Course ID:

ENROLL

REMOVE

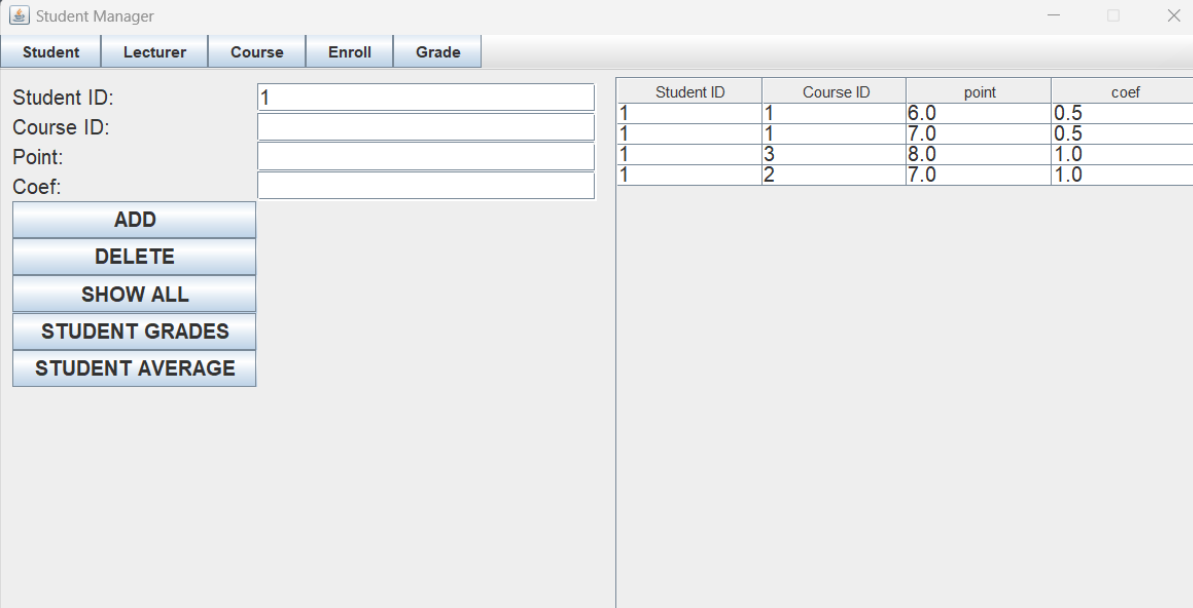
SHOW ALL

FIND BY STUDENT

FIND BY COURSE

Student ID	Course ID	Finished
1	2	false
1	3	false
1	1	true

Xem điểm thành phần của sinh viên:



Student Manager

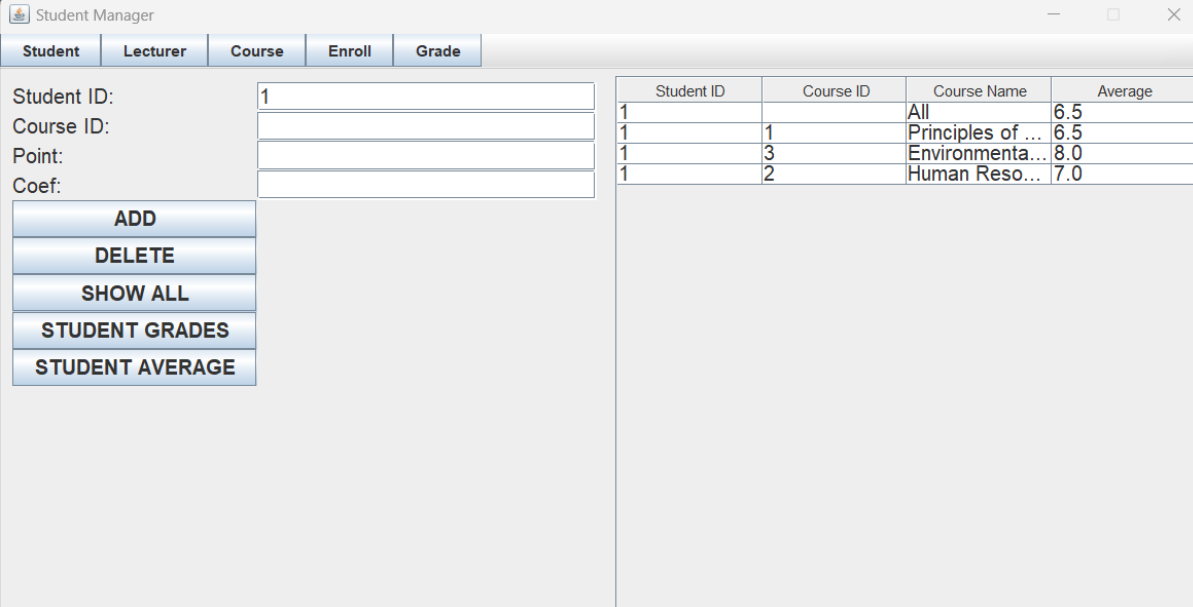
Student | Lecturer | Course | Enroll | **Grade**

Student ID: 1
Course ID:
Point:
Coef:

ADD
DELETE
SHOW ALL
STUDENT GRADES
STUDENT AVERAGE

Student ID	Course ID	point	coef
1	1	6.0	0.5
1	1	7.0	0.5
1	3	8.0	1.0
1	2	7.0	1.0

Xem điểm trung bình của sinh viên (điểm trung bình tổng là trung bình của các khóa học đã kết thúc):



Student Manager

Student | Lecturer | Course | Enroll | **Grade**

Student ID: 1
Course ID:
Point:
Coef:

ADD
DELETE
SHOW ALL
STUDENT GRADES
STUDENT AVERAGE

Student ID	Course ID	Course Name	Average
1		All	6.5
1	1	Principles of ...	6.5
1	3	Environmenta...	8.0
1	2	Human Reso...	7.0