**TRƯỜNG ĐẠI HỌC SƯ PHẠM KỸ THUẬT TP. HỒ CHÍ MINH**

**KHOA ĐÀO TẠO CHẤT LƯỢNG CAO**

**NGÀNH CÔNG NGHỆ THÔNG TIN**

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_



**BÁO CÁO ĐỒ ÁN**

**CẤU TRÚC DỮ LIỆU & GIẢI THUẬT**

**Thiết kế chương trình quản lý sinh viên bằng danh sách liên kết (Linked List)**

**SVTH : NGUYỄN XUÂN ĐIỀN**

**MSSV : 17110121**

**SVTH : VŨ HOÀNG HIỆP**

**MSSV : 17110138**

**GVPT : ThS. Trần Công Tú**

**TP. Hồ Chí Minh, tháng 11 năm 2018**

MỤC LỤC

[I. GIỚI THIỆU ĐỀ TÀI 1](#_Toc531900083)

[1. Lý do chọn đề tài 1](#_Toc531900084)

[2. Hoàn cảnh ra đời của danh sách liên kết 1](#_Toc531900085)

[3. Giới thiệu chương trình quản lý sinh viên 1](#_Toc531900086)

[4. Lựa chọn ngôn ngữ cài đặt chương trình 2](#_Toc531900087)

[II. NỘI DUNG 3](#_Toc531900088)

[1. Giới thiệu về Linked List (Danh sách liên kết) 3](#_Toc531900089)

[a. Ưu điểm: 3](#_Toc531900090)

[b. Khuyết điểm: 3](#_Toc531900091)

[c. Các dạng danh sách liên kết: 3](#_Toc531900092)

[d. Một số hàm quan trọng 4](#_Toc531900093)

[2. Áp dụng danh sách liên kết vào chương trình 4](#_Toc531900094)

[a. Khai báo 4](#_Toc531900095)

[b. Khởi tạo danh sách liên kết ban đầu là rỗng 5](#_Toc531900096)

[c. Hàm thêm vào danh sách, thích ứng với cả kiểu Sinh viên, Lớp và Ngành (Add) 5](#_Toc531900097)

[d. Hàm xóa một nút khỏi danh sách liên kết (Remove) 6](#_Toc531900098)

[e. Hàm tìm kiếm một nút trong danh sách liên kết (Search) 8](#_Toc531900099)

[3. Thiết kế giao diện 8](#_Toc531900100)

[4. Cài đặt chương trình 11](#_Toc531900101)

[a. Các hàm chức năng 11](#_Toc531900102)

[b. Hàm thiết kế chương trình 11](#_Toc531900103)

[i. Quản lý sinh viên 11](#_Toc531900104)

[ii. Quản lý lớp học 21](#_Toc531900105)

[iii. Quản lý ngành học 23](#_Toc531900106)

[c. Các hàm thêm 23](#_Toc531900107)

[5. Các thuật lợi và khó khăn 24](#_Toc531900108)

[6. Bảng phân công công việc 24](#_Toc531900109)

[III. KẾT LUẬN 25](#_Toc531900110)

[e. Ưu điểm 25](#_Toc531900111)

[f. Nhược điểm 25](#_Toc531900112)

[g. Hướng phát triển 25](#_Toc531900113)

LỜI NÓI ĐẦU

Đầu tiên nhóm xin cảm ơn thầy vì những góp ý và hướng dẫn từ thầy trong suốt thời gian qua, đồng thời xin thầy bỏ qua những thiếu sót trong quá trình tìm hiểu và thực hiện đồ án. Tuy nhiên nhóm vẫn mong nhận được những lời góp ý, những phê bình cần thiết để phát triển chương trình tốt hơn, góp phần giúp các thành viên nhóm phát triển bản thân và ngày một hoàn thiện mình hơn.

DANH MỤC

[hình 1 Hình khởi tạo Linked List và các biến 4](file:///F:\New%20folder\UpGit\-n-c-u-tr-c-d-li-u\BaoCaoDoAnCTDL.docx#_Toc531900253)

[hình 2 Hàm CreateNode 5](file:///F:\New%20folder\UpGit\-n-c-u-tr-c-d-li-u\BaoCaoDoAnCTDL.docx#_Toc531900254)

[hình 3 Hàm add 6](file:///F:\New%20folder\UpGit\-n-c-u-tr-c-d-li-u\BaoCaoDoAnCTDL.docx#_Toc531900255)

[hình 4 Hàm xóa 7](file:///F:\New%20folder\UpGit\-n-c-u-tr-c-d-li-u\BaoCaoDoAnCTDL.docx#_Toc531900256)

[hình 5 Hàm tìm kiếm (Search) 8](file:///F:\New%20folder\UpGit\-n-c-u-tr-c-d-li-u\BaoCaoDoAnCTDL.docx#_Toc531900257)

[hình 6 Giao diện quản lý sinh viên 9](file:///F:\New%20folder\UpGit\-n-c-u-tr-c-d-li-u\BaoCaoDoAnCTDL.docx#_Toc531900258)

[hình 7 Giao diện giới thiệu của chương trình 9](file:///F:\New%20folder\UpGit\-n-c-u-tr-c-d-li-u\BaoCaoDoAnCTDL.docx#_Toc531900259)

[hình 8 Giao diện quản lý ngành 10](file:///F:\New%20folder\UpGit\-n-c-u-tr-c-d-li-u\BaoCaoDoAnCTDL.docx#_Toc531900260)

[hình 9 Giao diện tìm kiếm sinh viên 10](file:///F:\New%20folder\UpGit\-n-c-u-tr-c-d-li-u\BaoCaoDoAnCTDL.docx#_Toc531900261)

[hình 10 Hình minh họa nút lưu trước 12](file:///F:\New%20folder\UpGit\-n-c-u-tr-c-d-li-u\BaoCaoDoAnCTDL.docx#_Toc531900262)

[hình 11 Hình minh họa nút lưu sau 12](file:///F:\New%20folder\UpGit\-n-c-u-tr-c-d-li-u\BaoCaoDoAnCTDL.docx#_Toc531900263)

[hình 12 Nút lưu sinh viên 13](file:///F:\New%20folder\UpGit\-n-c-u-tr-c-d-li-u\BaoCaoDoAnCTDL.docx#_Toc531900264)

[hình 13 Nút xóa trước 14](file:///F:\New%20folder\UpGit\-n-c-u-tr-c-d-li-u\BaoCaoDoAnCTDL.docx#_Toc531900265)

[hình 14 Nút xóa sau 15](file:///F:\New%20folder\UpGit\-n-c-u-tr-c-d-li-u\BaoCaoDoAnCTDL.docx#_Toc531900266)

[hình 15 Nội dung xóa của nút xóa 15](file:///F:\New%20folder\UpGit\-n-c-u-tr-c-d-li-u\BaoCaoDoAnCTDL.docx#_Toc531900267)

[hình 16 Ví dụ click để sửa thông tin sinh viên 16](#_Toc531900268)

[hình 17 Định nghĩa phần nội dung chính để chỉnh sửa thông tin sinh viên 17](#_Toc531900269)

[hình 18 Hàm validation (Check lỗi trong chương trình) 19](#_Toc531900270)

[hình 19 Nút lưu của lớp-sau 21](file:///F:\New%20folder\UpGit\-n-c-u-tr-c-d-li-u\BaoCaoDoAnCTDL.docx#_Toc531900271)

[hình 20 Nút lưu của lớp-trước 21](file:///F:\New%20folder\UpGit\-n-c-u-tr-c-d-li-u\BaoCaoDoAnCTDL.docx#_Toc531900272)

[hình 21 Nút xóa của lớp-sau 22](file:///F:\New%20folder\UpGit\-n-c-u-tr-c-d-li-u\BaoCaoDoAnCTDL.docx#_Toc531900273)

[hình 22 Nút xóa của lớp-trước 22](file:///F:\New%20folder\UpGit\-n-c-u-tr-c-d-li-u\BaoCaoDoAnCTDL.docx#_Toc531900274)

[hình 23 Click sửa thông tin 23](file:///F:\New%20folder\UpGit\-n-c-u-tr-c-d-li-u\BaoCaoDoAnCTDL.docx#_Toc531900275)

[hình 24 Bảng phân công thực hiện đồ án 25](file:///F:\New%20folder\UpGit\-n-c-u-tr-c-d-li-u\BaoCaoDoAnCTDL.docx#_Toc531900276)

# GIỚI THIỆU ĐỀ TÀI

## Lý do chọn đề tài

Như chúng ta đã biết, ngày xưa, mô hình quản lý đã phổ biến rộng rãi. Có thể kể đến như việc quản lý đất nước của vua chúa là một hình thức quản lý, hay quản lý một chuỗi nhà hàng, quản lý thông tin của khách hàng ngân hàng, v.v…

Vậy từ đó, ta có thể thấy được rằng việc quản lý thông tin là vô cùng cần thiết, nhất là trong thời đại kĩ thuật số ngày nay. Vì vậy, nhóm muốn hướng đến đề tài này, với mong muốn hướng đến nhu cầu của người dùng, để sau chương trình này, nhóm sẽ có cơ hội mở rộng quy mô của chương trình hơn, phục vụ cho lợi ích của khách hàng nói riêng cũng như của xã hội loài người nói chung.

## Hoàn cảnh ra đời của danh sách liên kết

Trong lập trình căn bản, chúng ta đã được biết hầu hết các bài toán đều có thể giải quyết được bằng mảng, điển hình là trong việc lưu trữ thông tin và dữ liệu. Nhưng lưu trữ bằng mảng lại có nhiều nhược điểm:

- Vì nó là cấu trúc tĩnh, nên nó không thể dễ dàng mở rộng hay thu hẹp để phù hợp với dạng dữ liệu.

- Để duy trì việc thêm và bớt trong mảng cũng rất khó khan.

Để khắc phục các nhược điểm đó, người ta có thể dùng Danh sách liên kết (Linked List).

## Giới thiệu chương trình quản lý sinh viên

Chương trình quản lý thông tin sinh viên được viết trên giao diện winform.

Danh sách liên kết được đưa vào chương trình để quản lý toàn bộ thông tin sinh viên của các ngành, trong mỗi ngành đó sẽ có nhiều lớp, mỗi lớp sẽ có một số lượng sinh viên cụ thể. Các thông tin ấy cụ thể sẽ là:

Ngành:

1. Mã ngành
2. Tên ngành

Lớp:

1. Mã lớp
2. Tên lớp

Sinh viên:

1. Mã số sinh viên
2. Tên sinh viên
3. Ngày sinh
4. Phái
5. Điểm

**Các tác vụ cơ bản của chương trình:**

1. Sinh viên
2. Tìm kiếm 1 sinh viên theo mã số sinh viên
3. Thêm 1 sinh viên vào danh sách
4. Sửa thông tin 1 sinh viên cụ thể
5. Xóa toàn bộ thông tin của 1 sinh viên trong danh sách
6. Lớp
7. Thêm 1 lớp mới vào danh sách
8. Sửa thông tin của 1 lớp
9. Xóa toàn bộ 1 lớp trong dánh sách
10. Ngành
11. Thêm 1 ngành mới vào danh sách
12. Sửa thông tin của 1 ngành
13. Xóa toàn bộ 1 ngành trong danh sách

## Lựa chọn ngôn ngữ cài đặt chương trình

Mục tiêu của nhóm là tạo ra một chương trình gần gũi và thân thiện với người dùng, nên thiết nghĩ chương trình tạo ra trên winform là phù hợp nhất vì người dùng đến nay đã quá quen với giao diện của window. Microsoft C#.NET với phiên bản .NET Framework 4.5.2; IDE và compiler nằm trong bộ Microsoft Visual Studio 2017 để cài đặt.

# NỘI DUNG

### Giới thiệu về Linked List (Danh sách liên kết)

Vậy, danh sách liên kết là gì ?

Một danh sách liên kết là một cấu trúc dữ liệu tuyến tính trong đó mỗi phần tử là một đối tượng riêng biệt.

Trong đó, mỗi phần tử (Node - nút) của một danh sách được gồm hai mục - dữ liệu và một tham chiếu đến nút tiếp theo. Nút cuối cùng có tham chiếu đến giá trị null. Các điểm nhập vào một danh sách liên kết được gọi là điểm đầu danh sách. Cần lưu ý rằng điểm đứng đầu không phải là một nút riêng biệt, mà tham chiếu đến nút đầu tiên. Nếu danh sách rỗng thì điểm đứng đầu là một tham chiếu null.

#### Ưu điểm:

Danh sách liên kết là một cấu trúc dữ liệu động. Và … tất nhiên rồi, theo cách nói này thì nó là định nghĩa trái với định nghĩa trên của mảng, nghĩa là nó linh động hơn, ít tốn dữ liệu hơn, …Số lượng nút trong danh sách không phải là bất biến, chúng có thể tăng lên, làm danh sách giãn ra theo nhu cầu. Bất cứ chương trình nào cũng có thể giải quyết số lượng bất kì các qđối tượng cần thiết để dùng một danh sách liên kết.

Đó là ưu điểm. Còn khuyết điểm của nó là gì ?

#### Khuyết điểm:

Một trong những khuyết điểm của danh sách liên kết so với mảng đó là nó không cho phép truy xuất trực tiếp đến các biến. Nếu muốn truy cập vào cá nhân các mục, bạn phải bắt đầu duyệt từ đầu và tiếp tục đến khi duyệt đến được mục bạn muốn.

Một nhược điểm khác của danh sách liên kết là nó sử dụng nhiều bộ nhớ hơn so với mảng, ta cần thêm 4 bytes (trên CPU 32-bit) để lưu trữ đến nút tiếp theo.

#### Các dạng danh sách liên kết:

Danh sách liên kết có 3 dạng:

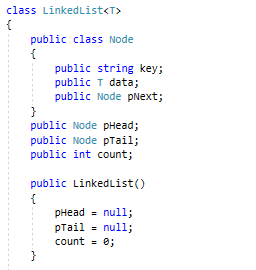
* Danh sách liên kết đơn (a singly linked list): là danh sách đã được miêu tả trên.
* Danh sách liên kết đôi (a doubly linked list): là danh sách có hai truy truy xuất, một đến nút tiếp theo và một cho nút trước đó.
* Danh sách liên kết vòng (circular linked list): nút cuối cùng trong danh sách sẽ trỏ ngược về nút đầu tiên trong danh sách.

#### Một số hàm quan trọng

Các hàm đặc trưng của danh sách liên kết bao gồm : Thêm, Sửa, Xóa phần tử trong mảng bằng danh sách liên kết, sẽ được áp dụng vào đồ án của nhóm, và miêu tả cụ thể ở trong chương trình sinh viên này.

### Áp dụng danh sách liên kết vào chương trình

##### Khai báo



hình 1 Hình khởi tạo Linked List và các biến

Giải thích :

Generic cho lớp : Để có thể tái sử dụng code nhiều lần cho Sinh viên, lớp và ngành mà không cần định nghĩa lại nhiều lần

key : biến kiểu string dùng như một dấu hiệu duy nhất, để phân biệt giữa các phần tử trong danh sách với nhau

data : dùng để lưu trữ các thông tin về bản thân phần tử

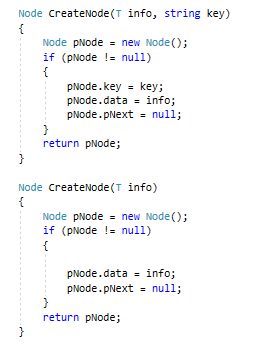
pNext : dùng để lưu địa chỉ của phần tử kế tiếp trong danh sách, hoặc lưu giá trị là NULL nếu là phần tử cuối trong danh sách

pHead : Con trỏ lưu giữ địa chỉ phần tử đầu chuỗi

pTail : Con trỏ lưu giữ địa chỉ phần tử cuối chuỗi

count : Đếm số phần tử trong danh sách

##### Khởi tạo danh sách liên kết ban đầu là rỗng



hình 2 Hàm CreateNode

Cách hoạt động:

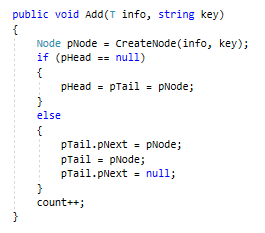
Tạo một node chứa thông tin trong danh sách để dễ dàng chèn, xóa và sửa danh sách hơn.

Gán node truyền vào vào dữ liệu của biến data, biến này chứa các thông tin cá nhân từng phẩn tử cụ thể.

Cấp phát vùng nhớ cho Node.

Đối với hàm có biến key thì chuyển nó sang kiểu Node

##### Hàm thêm vào danh sách, thích ứng với cả kiểu Sinh viên, Lớp và Ngành (Add)



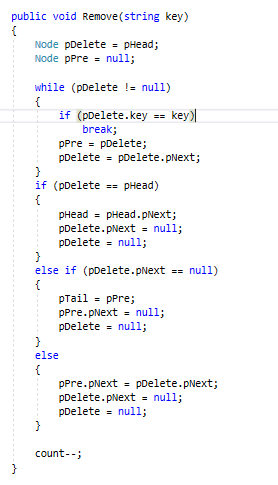
hình 3 Hàm add

Hàm này có sự cải tiến một chút so với hàm add thông thường bằng danh sách liên kết đơn, đó là ta có thể trực tiếp thêm sinh viên vào qua mã số sinh viên nhận được.

Gọi hàm CreateNode như đã nói ở trên và gán vào nút pNode

Kiểm tra xem danh sách ban đầu có rỗng hay không, nếu rỗng thì gán vị trí đầu và cuối bằng với pNode, nếu không rỗng thì trỏ pTail đến vị trí tiếp theo và cho vị trí đó bằng pNode, gán pTail bằng pNode. Việc làm này giúp mở rộng danh sách thêm 1.

##### Hàm xóa một nút khỏi danh sách liên kết (Remove)



hình 4 Hàm xóa

Trình tự duyệt của hàm:

- Khai báo pDelete bằng pHead để bắt đầu duyệt từ đầu danh sách, khai báo pPre khởi tạo bằng null. Sau đó duyệt vòng lặp nếu pDelete khác 0 thì kiểm tra xem pDelete có bằng pSearch không, nếu bằng thì break và gán pPre (previous) bằng pHead, còn pDelete lúc này sẽ trỏ tới nút tiếp theo, thao tác này chỉ nhằm duyệt đến vị trí muốn xóa. Sau khi tìm được vị trí cần xóa rồi thì pDelete chính là nút cần xóa.

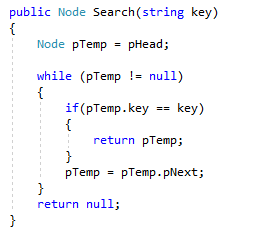
+ Kiểm tra xem pDelete có bằng pHead hay không, nếu pHead bằng pDelete thì nút cần xóa là nút đầu tiên trong danh sách, ta sẽ trỏ pHead đến nút tiếp theo, để xóa nút đầu đó. Sau đó, pDelete.Next và pDelete bằng null nghĩa là xóa thành công, kết thức thao tác xóa.

+ Nếu điều kiện đầu tiên không thỏa, xét tiếp điều kiện tiếp theo là pDelete.pNext có bằng null không, để kiểm tra xem nút cần xóa có phải nút cuối danh sách hay không. Nếu nút cần xóa là nút cuối, gán pTail vào nút kế cuối để cập nhật lại pTail, và cho pPre.pNext và pDelete bằng null, khi ấy, xóa thành công.

+ Nếu nút cần xóa không phải là nút đầu cũng không phải là nút cuối, xóa chính nút đó đi. Để thực hiện thao tác này, ta gán pPre=pDelete rồi cho pDelete.pNext và pDelete bằng null, kết thúc việc xóa (xóa thành công).

-Cuối cùng giảm count (số phần tử trong danh sách đi 1), thao tác này là cập nhật lại số phần tử trong mảng sau khi xóa.

##### Hàm tìm kiếm một nút trong danh sách liên kết (Search)

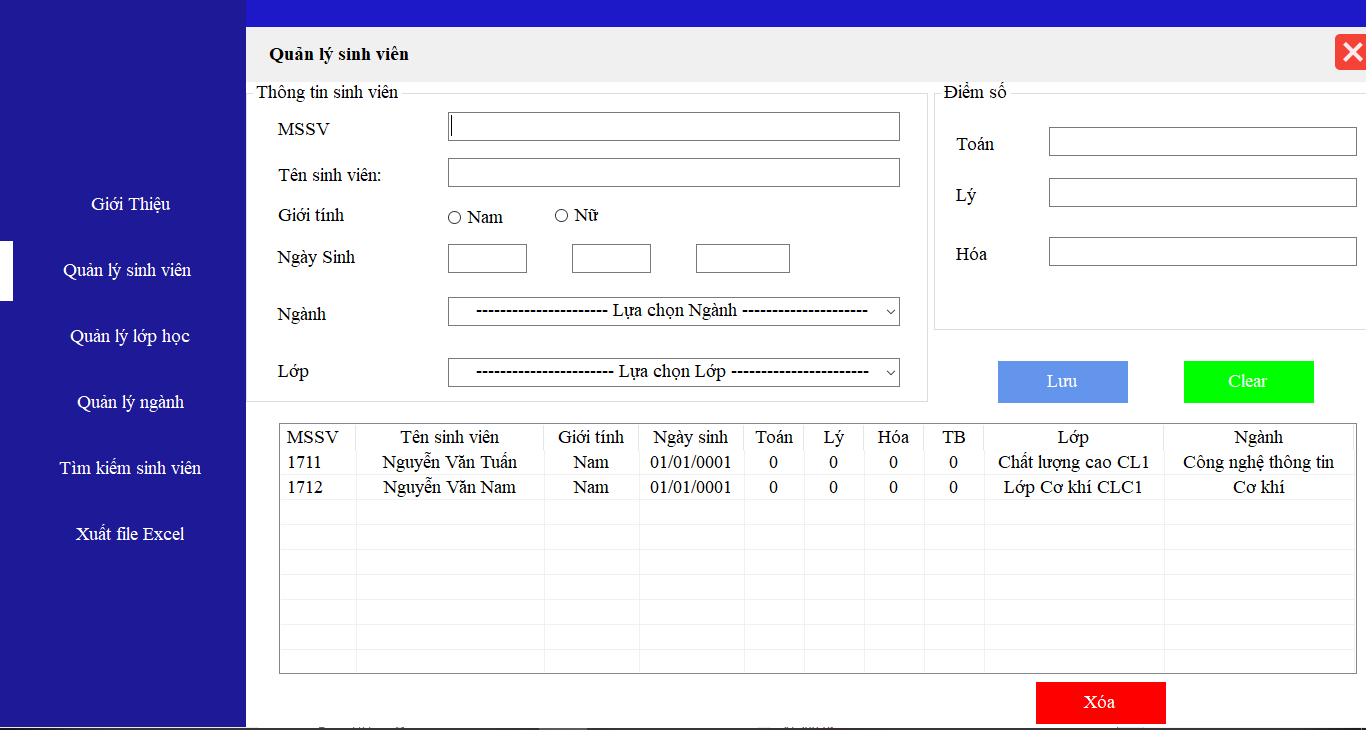


hình 5 Hàm tìm kiếm (Search)

Hàm tìm kiếm có kiểu dữ liệu truyền vào là string, biến key như đã nói ở trên là biến duy nhất, hay còn gọi là biến dùng để xác định, thường biến này trong chương trình sẽ là mã sinh viên hay mã ngành mã lớp, vì những thành phần này là khóa chính. Ví dụ khi tìm kiếm một sinh viên nào đó, ta tìm thành phần key là mã số sinh viên sẽ tìm ngay được chính xác sinh viên đó, vì mỗi sinh viên có một mã số khác nhau và không được trùng lặp.

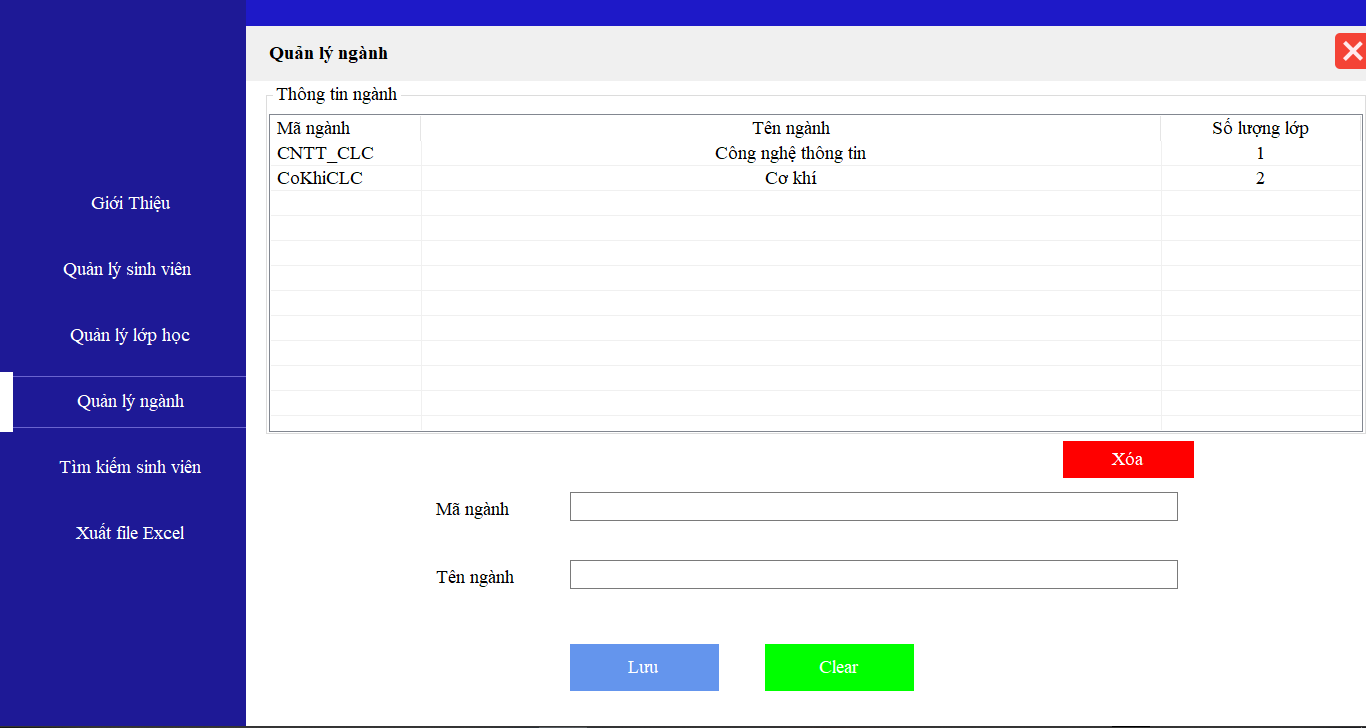
### Thiết kế giao diện





hình 6 Giao diện quản lý sinh viên

hình 7 Giao diện giới thiệu của chương trình



hình 8 Giao diện quản lý ngành



hình 9 Giao diện tìm kiếm sinh viên

Chương trình cho phép thực hiển chức năng thêm, sửa, xóa cho từng phần như sinh viên, lớp, ngành và tìm kiếm sinh viên theo mã số sinh viên. Ngoài ra, chương trình còn có chức năng rất hay đó là xuất dữ liệu trong bảng biểu thông tin ra thành file excel để lưu trữ.

### Cài đặt chương trình

1. Các hàm chức năng

Chương trình được thiết kế gồm 6 class và 1 form chứa 5 UserControls:

* Các class:

+ LinkedList.cs: để tạo các hàm quan trọng trong danh sách liên kết.

+ LopHoc.cs: chứa các giá trị sẽ sử dụng của lớp, lớp này quản lý các sinh viên.

+ Nganh.cs: chứa các giá trị sẽ sử dụng của ngành, ngành quản lý các lớp, trong các lớp lại chứa các sinh viên.

+ SinhVien.cs: chứa các giá trị sẽ sử dụng của sinh viên như mã số sinh viên, họ tên, ngày tháng năm sinh, v.v…

+ Validation.cs: class dùng để check lỗi, rất quan trọng trong việc thêm sửa xóa, cụ thể sẽ được mô tả bên dưới.

+ Program.cs: chương trình chính.

* Form1:

+ GioiThieuUC.cs: màn hình chào, giao diện khởi đầu.

+ QuanLyLopHocUC.cs: chứa các chức năng dùng cho quản lý dữ liệu trong lớp học.

+ QuanLyNganhUC.cs: chứa các chức năng dùng cho quản lý dữ liệu trong ngành học.

+ QuanLySinhVienUC.cs: chứa các chức năng dùng cho quản lý dữ liệu của sinh viên.

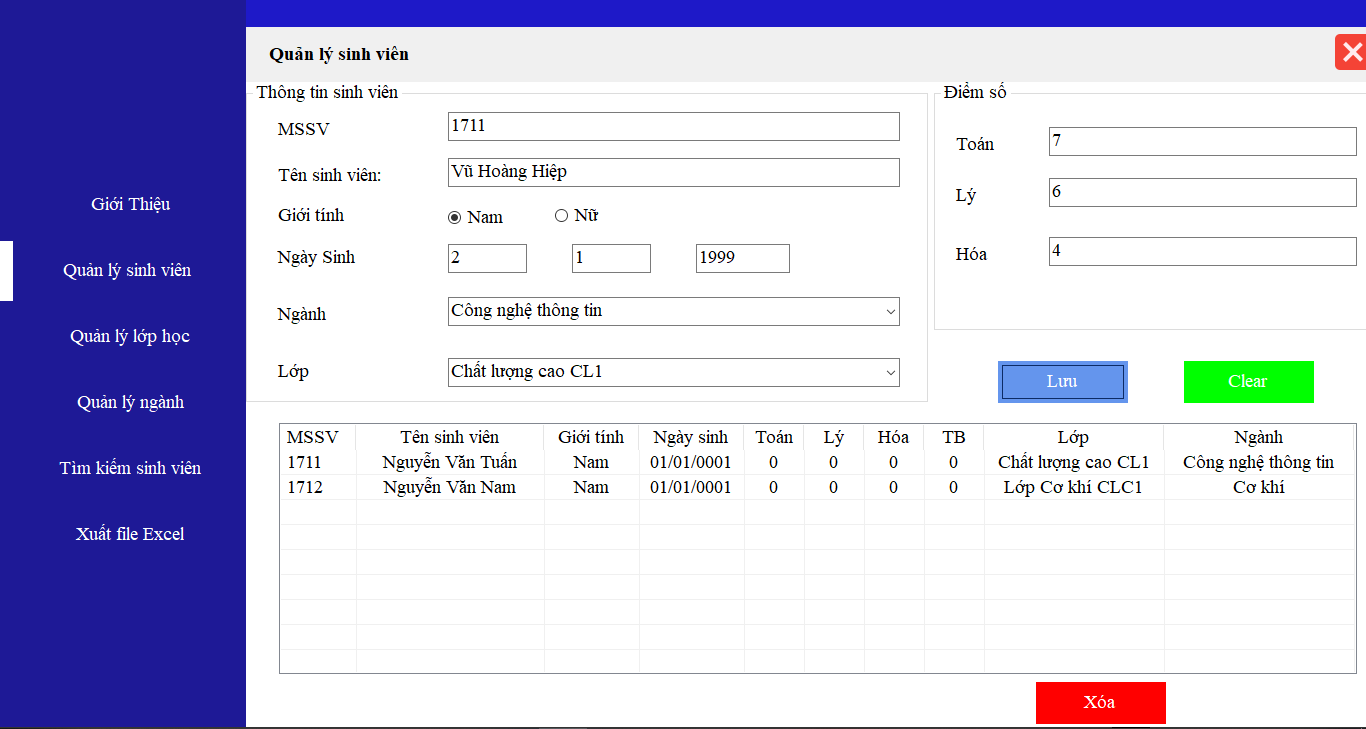
+ TimKiemSinhVien.cs: chức năng chính là tìm kiếm và show ra thông tin sinh viên.

UserControls là một khái niệm nhỏ hơn form, mục đích sử dụng UserControls là để có thể tái sử dụng các hàm nhiều lần.

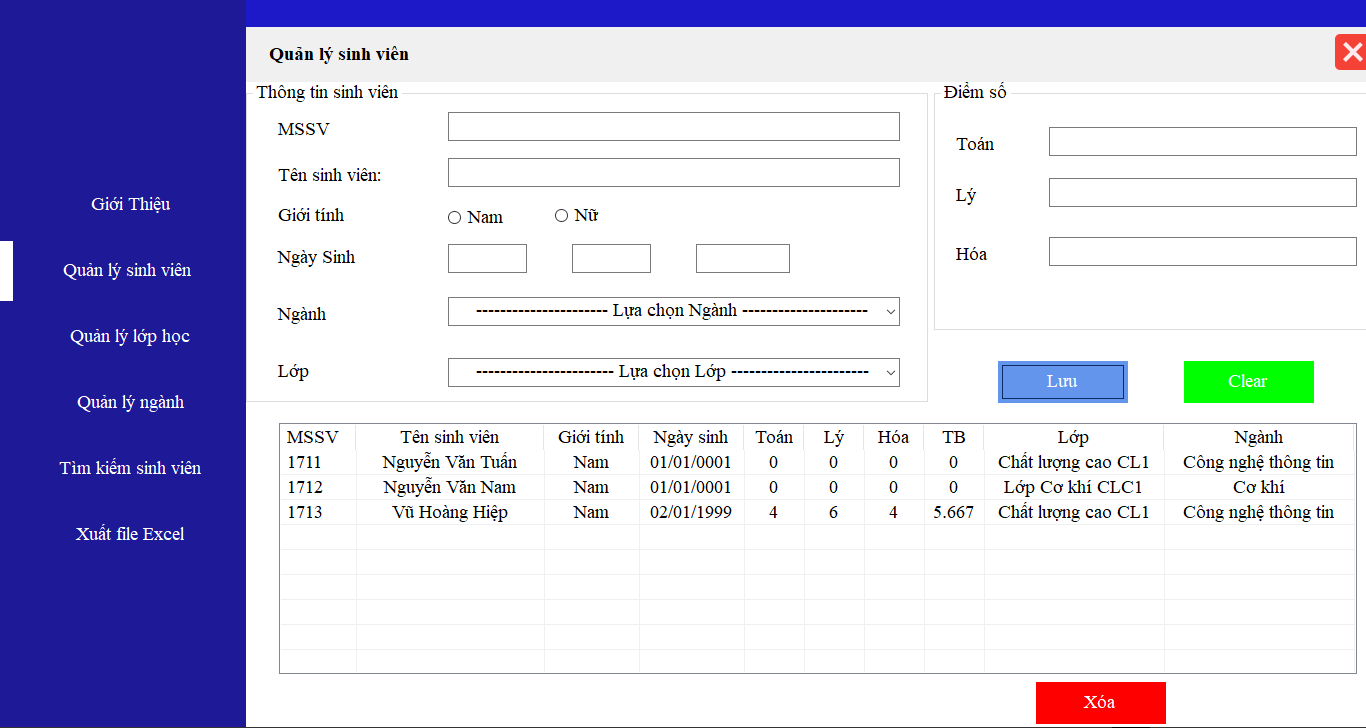
1. Hàm thiết kế chương trình

###### Quản lý sinh viên

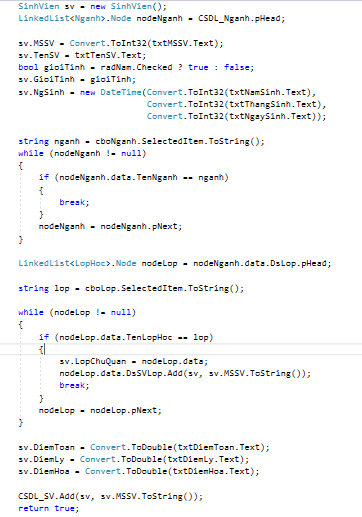
* Nút lưu



hình 10 Hình minh họa nút lưu trước



hình 11 Hình minh họa nút lưu sau

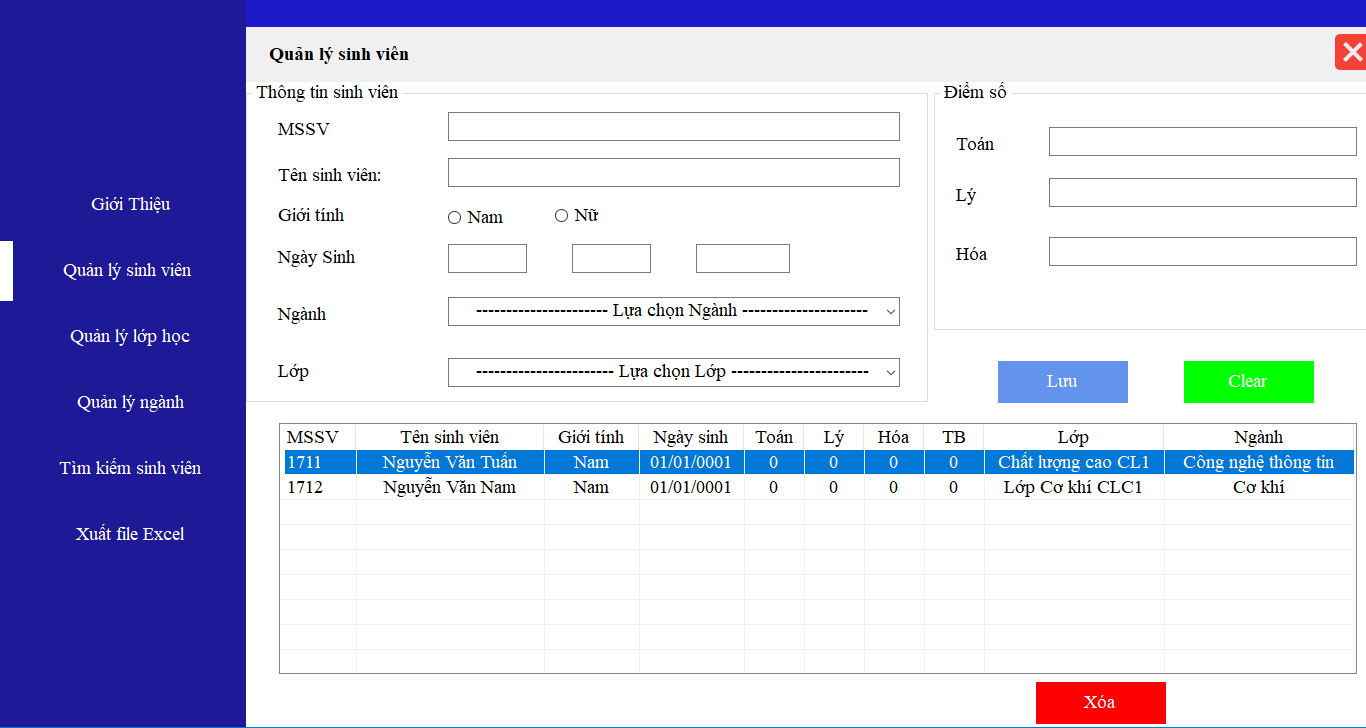


hình 12 Nút lưu sinh viên

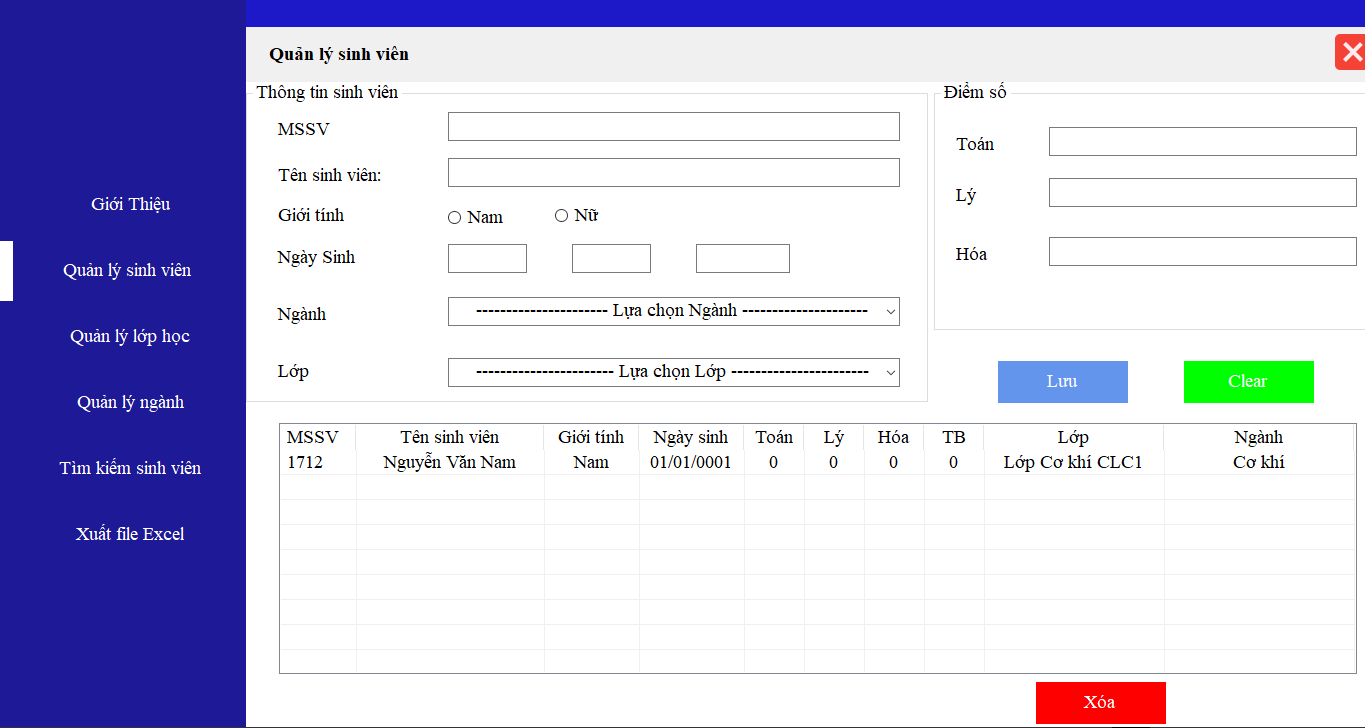
Cách hoạt động:

Lấy các thông tin mã số sinh viên, tên sinh viên, giới tính, ngày sinh, điểm các môn Toán, Lý, Hóa từ textbox, sau đó, nếu ngành hiện có không là null thì ta có thể chọn một ngành trong. Ngành đã chọn sẽ quản lý nhiều lớp, nếu số lớp ngành này quản lý không là null thì ta sẽ được lựa chọn sinh viên muốn thêm vào sẽ được thêm vào lớp nào. Tương tự như ngành, ta dùng vòng lặp để duyệt thao tác này, sau khi chọn lớp thì mới thêm toàn bộ thông tin của sinh viên đó vào lớp . Sau cùng mới thêm toàn bộ thông tin của sinh viên vừa rồi vào danh sách liên kết và hiển thị lên listView.

* Nút xóa

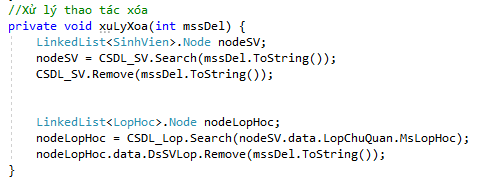


hình 13 Nút xóa trước



hình 14 Nút xóa sau

Phần nội dung chính dùng để xóa:



hình 15 Nội dung xóa của nút xóa

Giải thích hàm:

Đầu tiên kiểm tra xem đã chọn sinh viên muốn xóa bằng cách click chọn chưa (lvThongTinSV.SelectedItems.Count>0 có nghĩa là người dùng đã click chọn sinh viên muốn xóa). Hàm này thiết kế xóa theo mã số sinh viên, nên ta sẽ dựa vào mã số sinh viên để tìm được Node Sinh viên cần muốn xóa.

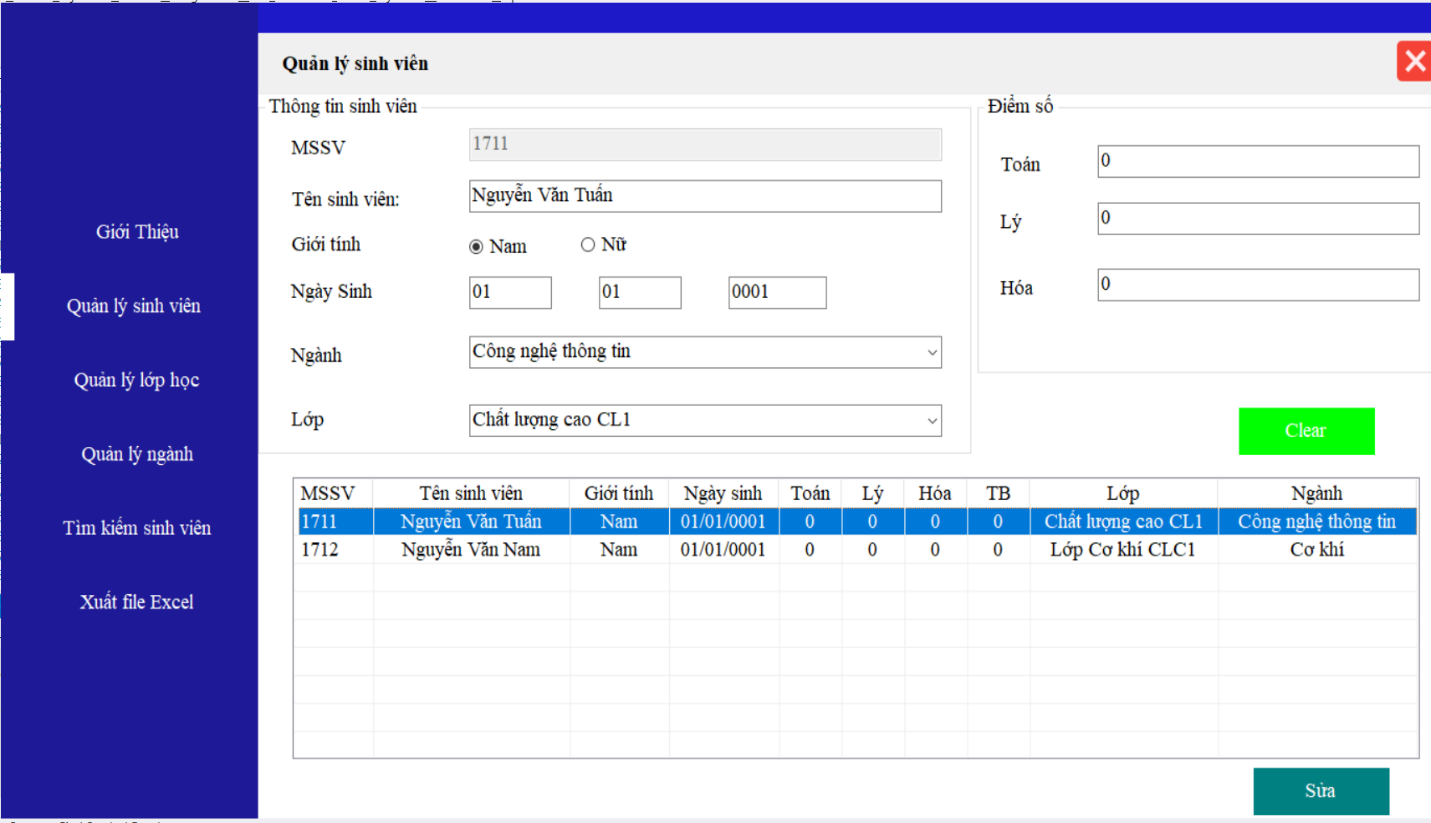
Kiểm tra xem nút được đang duyệt có null hay không, nếu không null thì ta kiểm tra xem đây có phải là nút muốn xóa không, nếu phải sẽ có một MessageBox hỏi lại người dùng có thực sự muốn xóa sinh viên này ra khỏi danh sách liên kết không, nếu người dùng xác nhận muốn (yes) thì ta gọi hàm Remove trong danh sách liên kết (được tạo sẵn để xóa Node sinh viên) và thông báo đã xóa thành công. Cuối cùng trỏ NodeSV đến nút tiếp theo.

Tiếp theo phải xóa thông tin của sinh viên vừa xóa trong lớp. Để thực hiện điều này, ta khai báo một nút gọi là nodeLopHoc theo cơ sở dữ liệu lớp để duyệt. Bắt đầu duyệt.

Dùng vòng lặp và bắt đầu kiểm tra xem mã số của lớp nào trong danh sách trùng với mã số lớp của sinh viên được xóa ( mỗi sinh viên đều có thông tin lớp chủ quản – lớp sinh viên đó học ) , nếu trùng rồi chỉ cần kiểm tra xem trong lớp đó, có sinh viên nào có mã số sinh viên trùng với mã số sinh viên muốn xóa đó thì xóa bằng hàm Remove một lần nữa.

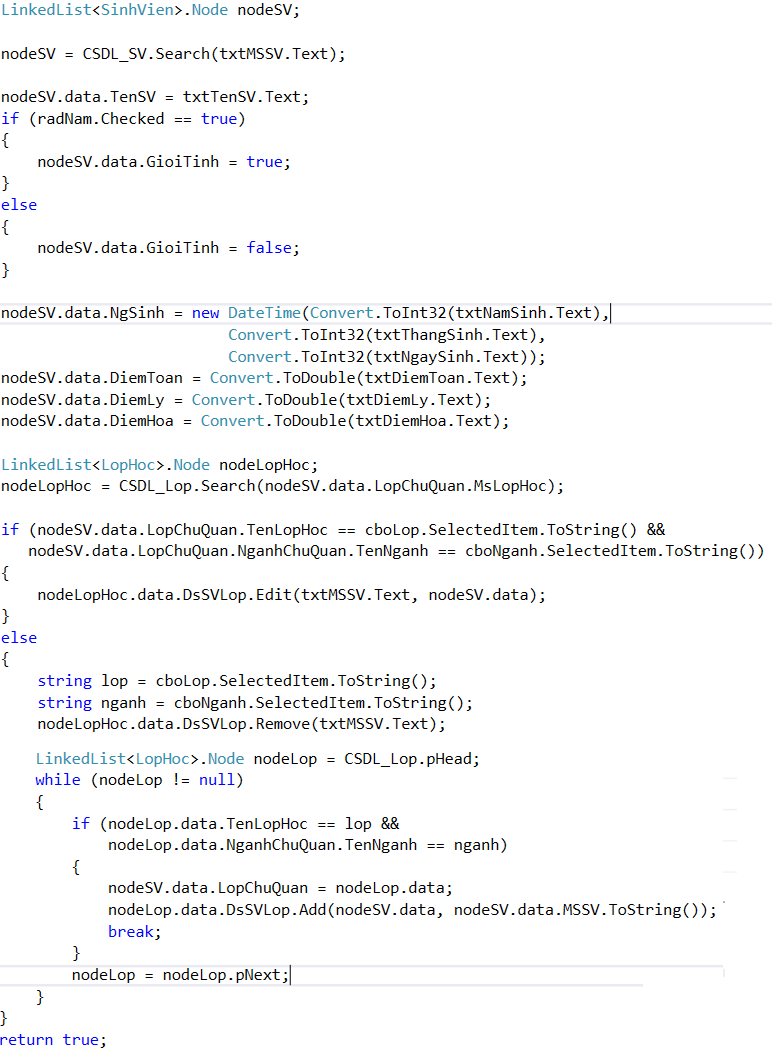
Kết thúc công việc xóa.

* Click để sửa



hình 16 Ví dụ click để sửa thông tin sinh viên

Phần nội dung chính dùng để sửa:



hình 17 Định nghĩa phần nội dung chính để chỉnh sửa thông tin sinh viên

Giải thích hàm:

Đầu tiên double click vào chọn sinh viên cần sửa, thông tin sinh viên được chọn đổ lên textbox, đồng thời ẩn đi các button Lưu, Xóa hiện lên button Sửa. Sau khi người dùng sửa, nhấn button Sửa. Hàm này thiết sửa dựa vào mã số sinh viên, nên ta sẽ dựa vào mã số sinh viên để tìm được Node Sinh viên cần muốn sửa.

Duyệt tìm node sinh viên cần sửa bằng mã số sinh viên. Sau khi tìm thấy gán các giá trị từ textbox mà người dùng sửa vào node sinh viên vừa tìm.

Sau khi cập nhập giá trị cho node sinh viên ta cũng cần phải cập lại giá trị sinh trong lớp học:

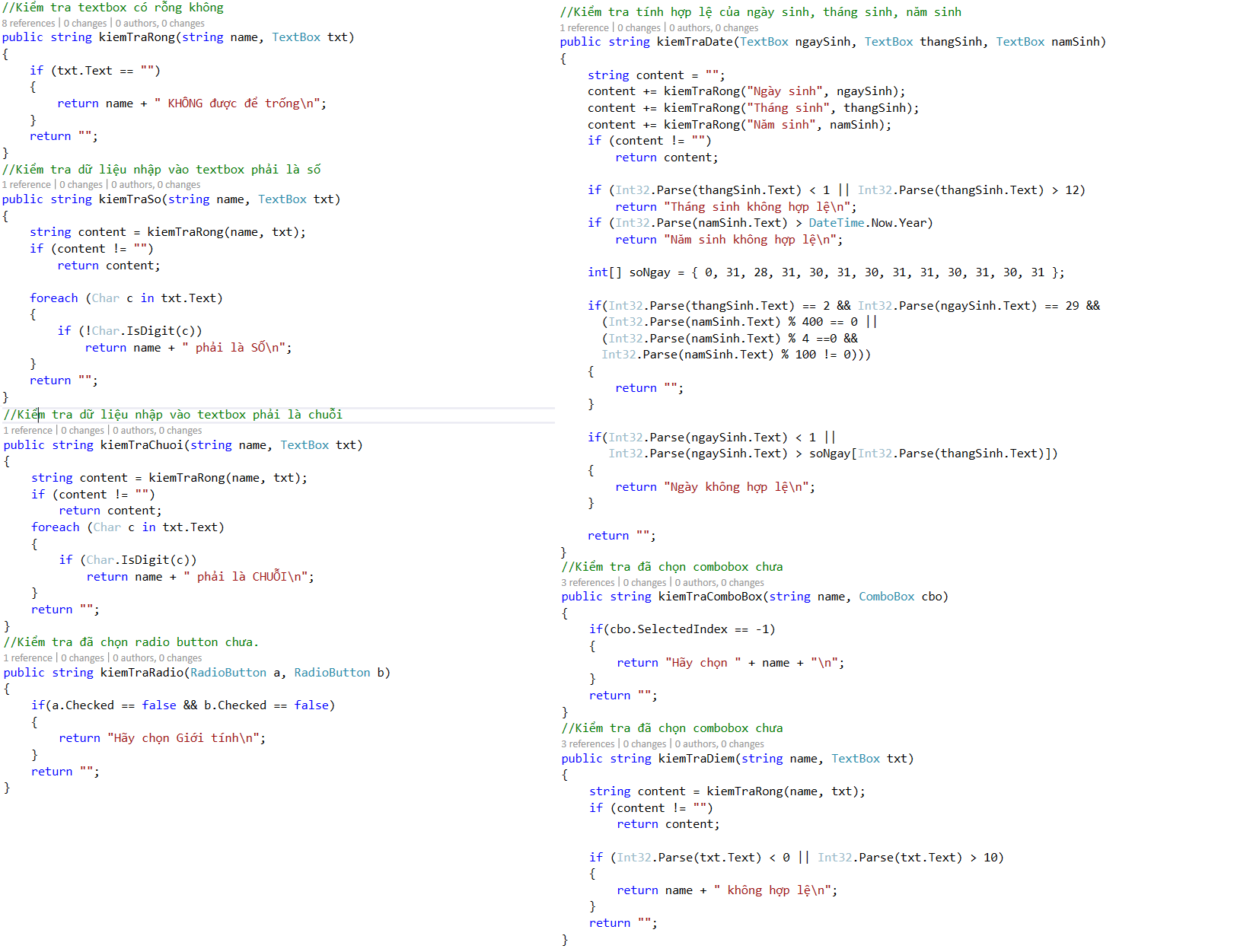
Có 2 trường hợp khi câp nhập lại trong lớp học:

+ Nếu không thay đổi giá trị ngành và lớp :

* + - Cập nhập giá trị node sinh viên trong lớp học bằng hàm edit

+ Nếu giá trị ngành hoặc lớp thay đổi:

* + - Xóa node sinh viên trong lớp cũ
    - Thêm node sinh viên vừa sửa vào lớp học mới thay đổi
* Validation

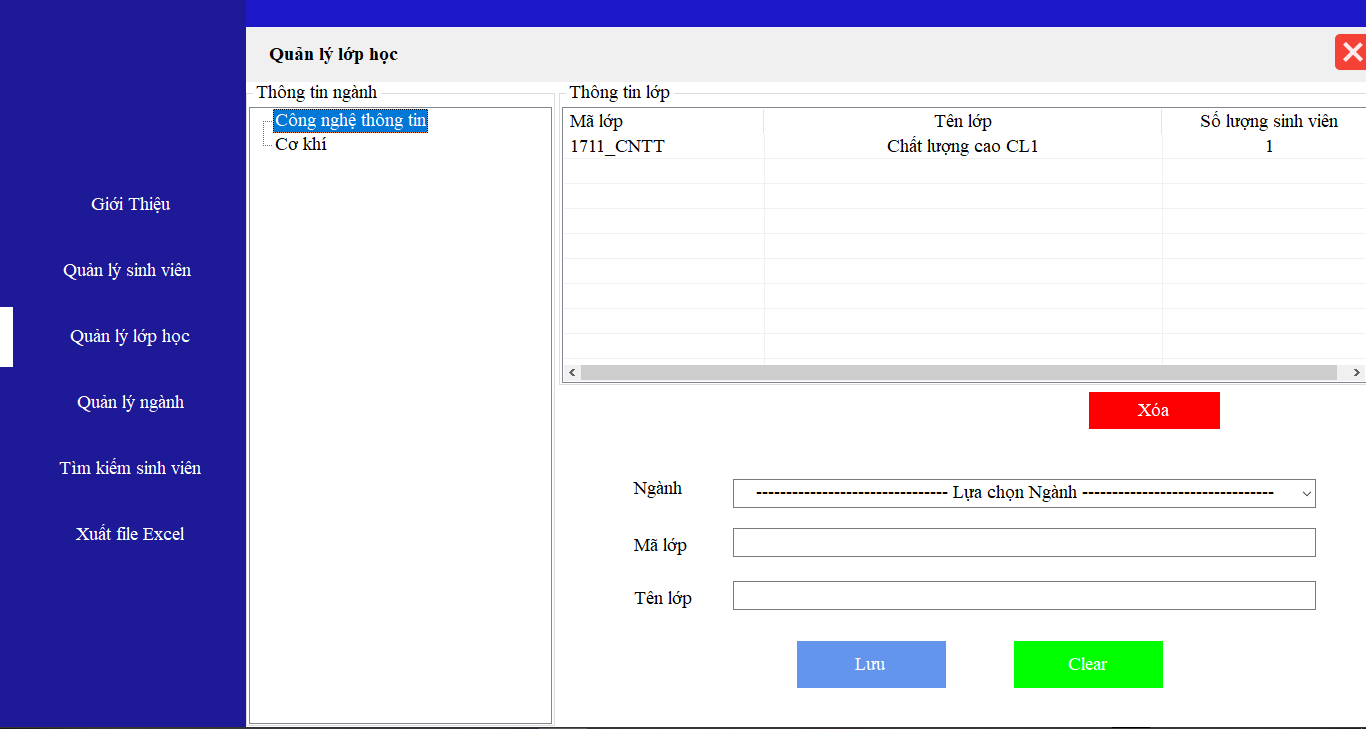
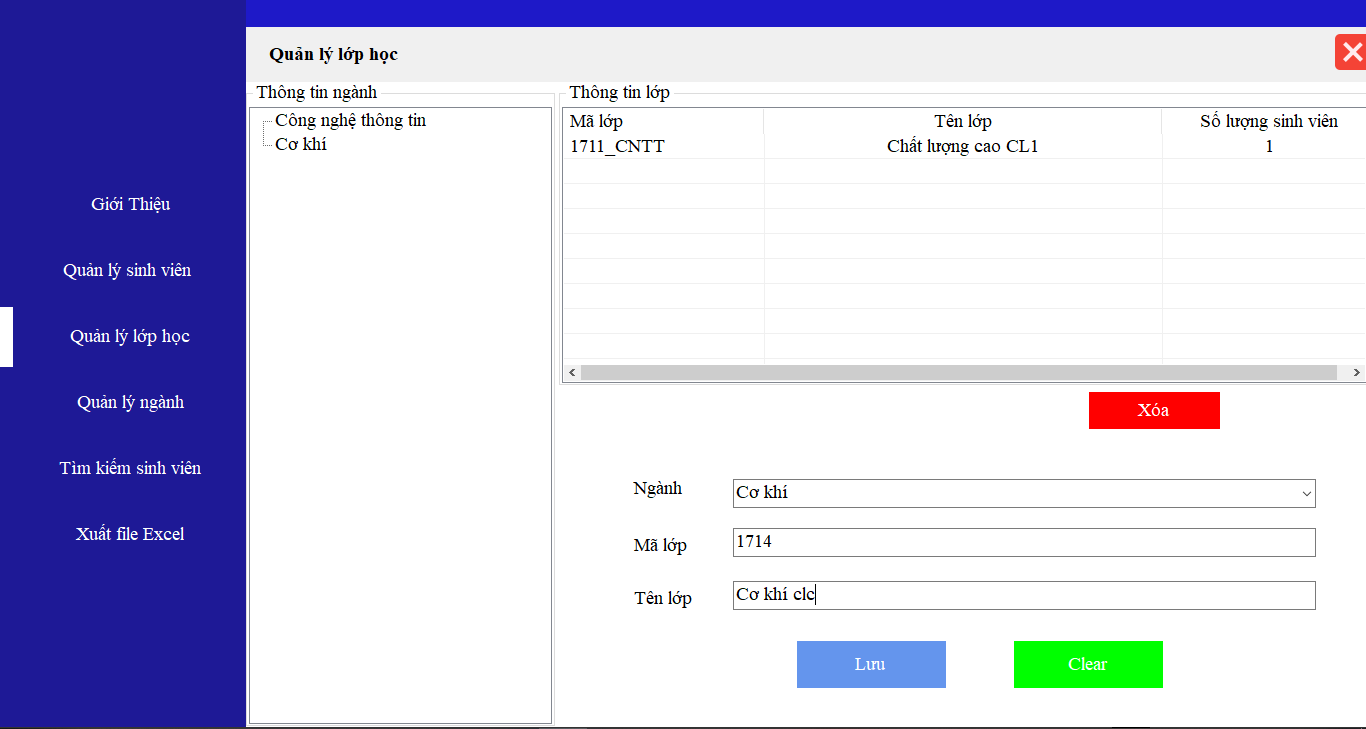


hình 18 Hàm validation (Check lỗi trong chương trình)

Tác dụng:

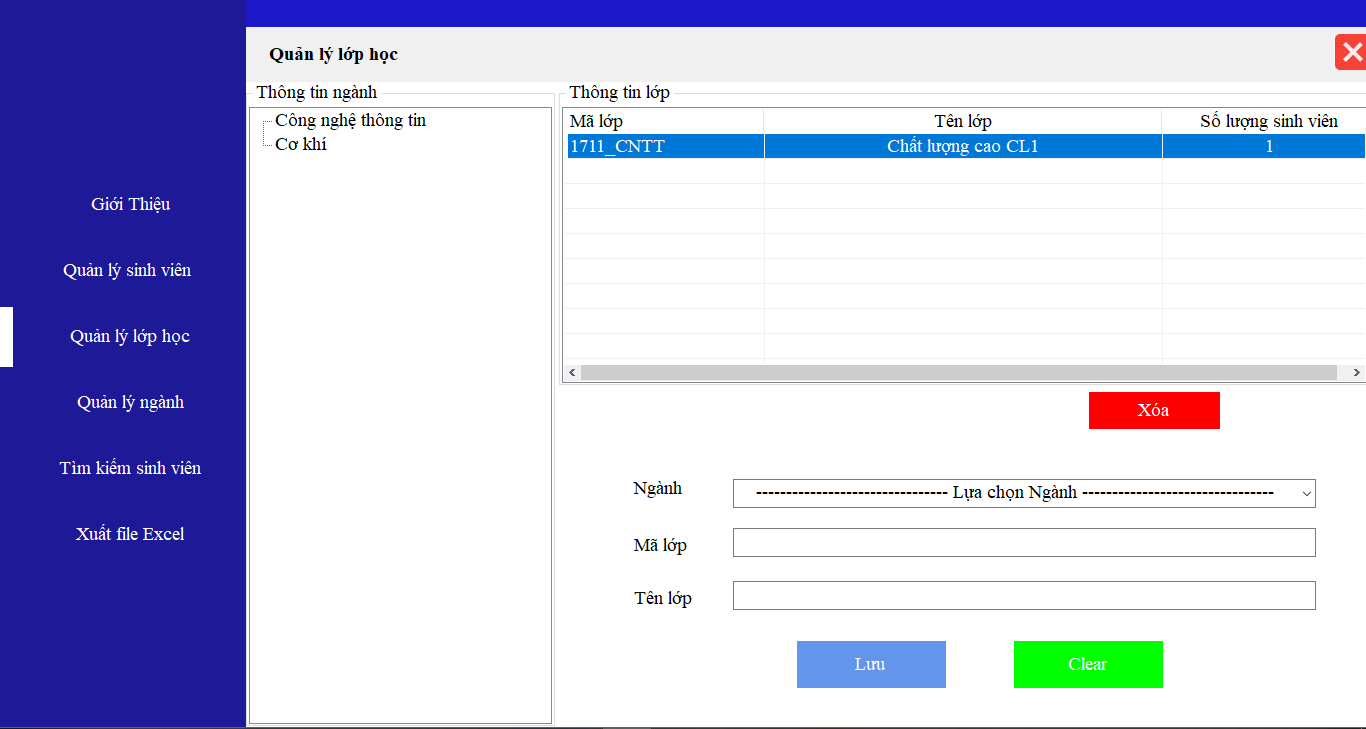
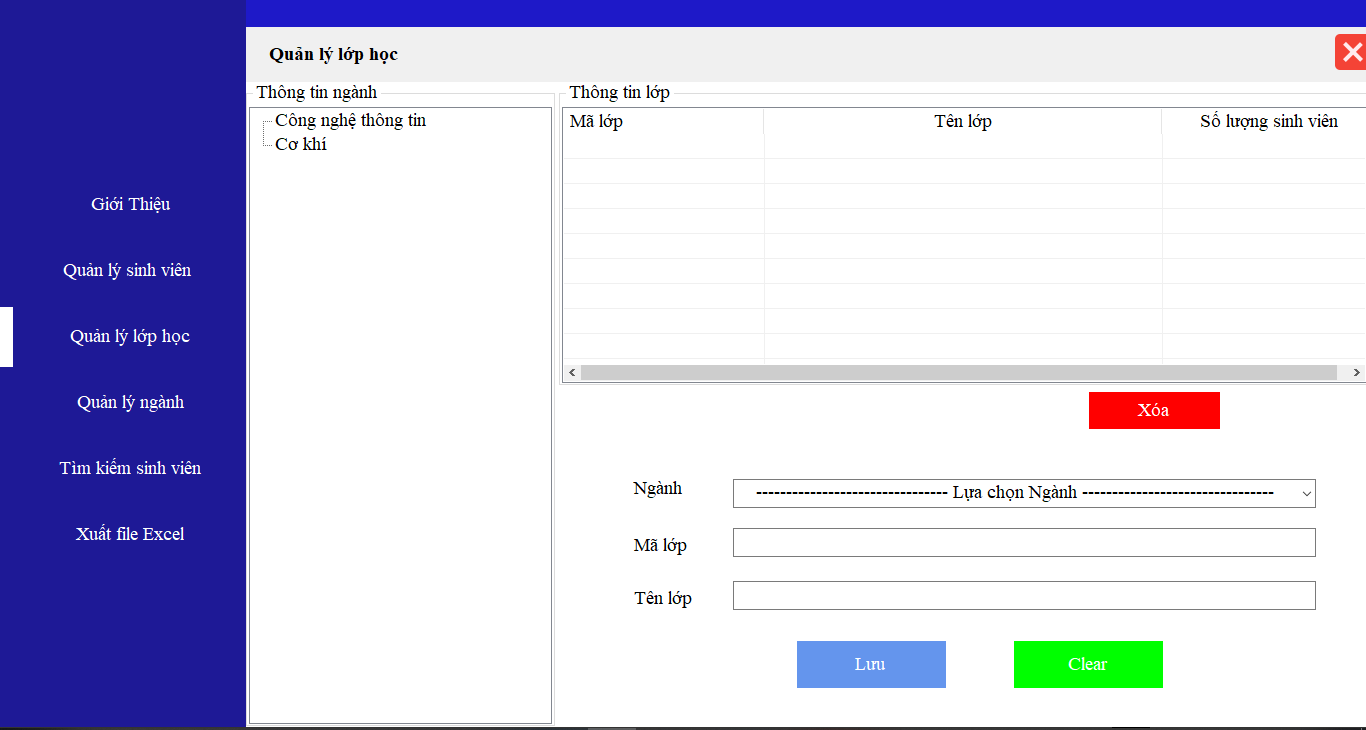
Sử dụng để xử lý khi người dùng nhập dữ liệu không hợp lệ. Ví dụ như mã sinh viên phải là số mà người dùng nhập chữ thì sẽ yêu cầu người dùng nhập số.

###### Quản lý lớp học

Các nút của quản lý lớp học và ngành học có thiết kế giống với nút ở quản lý sinh viên, nên nhóm sẽ không trình bày lại nội dung nút ở các phần này, mặc dù vậy, vẫn sẽ có ví dụ về cách hoạt động của mỗi nút

hình 19 Nút lưu của lớp-sau

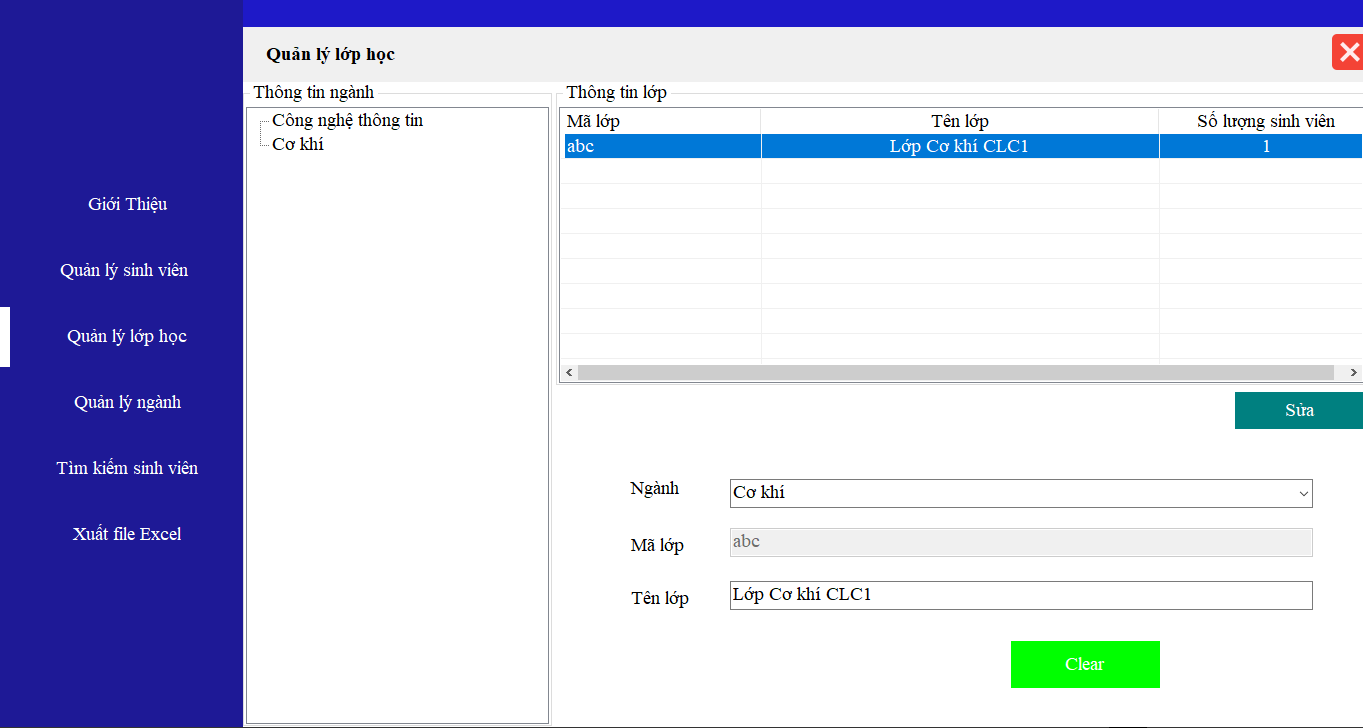
hình 20 Nút lưu của lớp-trước



hình 21 Nút xóa của lớp-sau

hình 22 Nút xóa của lớp-trước

Muốn sửa chúng ta cũng làm tương tự sinh viên, double click vào dòng muốn sửa thông tin sẽ hiện ra giao diện này, tiếp theo, click vào nút sửa để sửa thông tin.



hình 23 Click sửa thông tin

###### Quản lý ngành học

Các nút của ngành hoạt động hệt như của lớp

1. Các hàm thêm

Hàm xuất excel

* Sử dụng thư viện EPPlus
* Tham khảo video của Kteam <https://www.youtube.com/watch?v=EWYjU2yizL8>

### Các thuật lợi và khó khăn

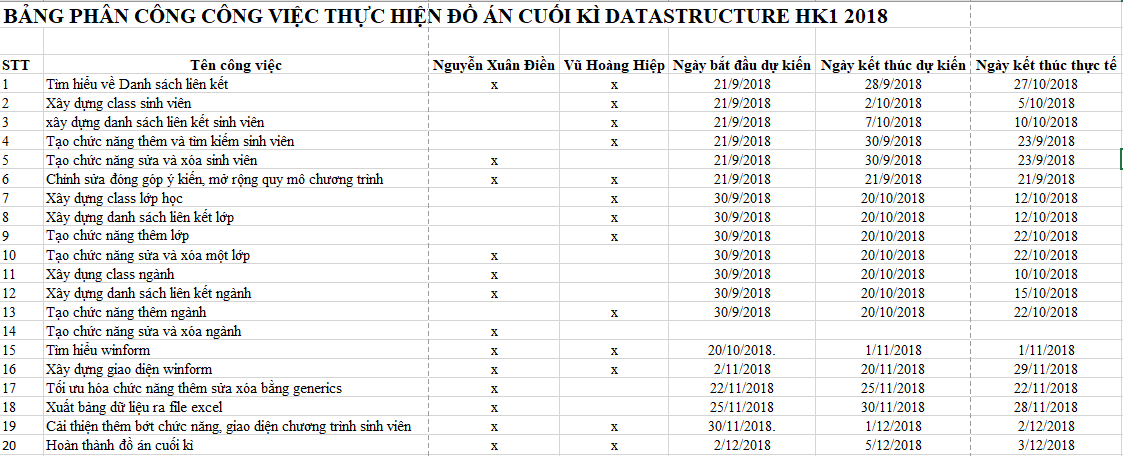
*Thuận lợi:*

* Kiến thức cơ bản dễ tiếp thu và phát triển.
* Đã được giảng viên hướng dẫn và cung cấp đầy đủ những thông tin, cách thức cần thiết để thiết kế chương trình.
* Chương trình khá là gần gũi với tất cả mọi người, nên cách thức thiết kế đã được định hướng rõ rang trước khi bắt tay vào làm việc

*Khó khăn:*

* Chênh lệch trình độ giữa 2 bạn trong nhóm ban đầu rất lớn, dẫn đến những khó khăn lúc mới bắt tay vào làm về tiến độ và chất lượng, cần nhiều thời gian hơn để tìm hiểu kiến thức nền so với dự kiến (cần nhiều thời gian tìm hiểu về Git, winform, các thao tác thiết kế, v.v…).
* Mất thời gian tìm cách tạo chương trình tương tự bằng struct, nhưng vì 3 đối tượng (Sinh viên, Lớp và Ngành) tương tác qua lại lẫn nhau, đối tượng này chứa đối tượng kia nên việc tạo chương trình bằng struct (vốn dự đoán là dễ hơn) lại không khả thi, phần này làm tiêu tốn hơn 1/3 thời gian thực hiện đồ án cho phép.
* Các khó khăn nhỏ trong việc tiếp cận Github thời gian đầu.

### Bảng phân công công việc



hình 24 Bảng phân công thực hiện đồ án

# KẾT LUẬN

#### Ưu điểm

* Giao diện thân thuộc, cách sử dụng đơn giản.
* Dễ phát triển thêm các chức năng khác.
* Có thể quản lý được nhiều thông tin, dữ liệu.
* Cấu trúc dữ liệu thêm vào chặt chẽ và dễ truy xuất.
* Có thể truy xuất thành file thông tin để lưu trữ (excel), có ích trong các hoạt động văn phòng hay quản lý của trường học.
* Có thể chuyển từ quản lý trong giáo dục sang các lĩnh vực khác dễ dàng với cách thiết kế tương tự

#### Nhược điểm

Dữ liệu truyền vào được truyền từ hàm có sẵn, nên sau khi tắt chương trình dữ liệu vừa nhập sẽ mất, không thể dùng lại cho những lần sau.

#### Hướng phát triển

Chương trình có thể được cải tiến với quy mô lớn hơn, sử dụng trong không chỉ cá nhân một trường, mà còn có thể dùng trong một sở giáo dục, thậm chí cả một hệ thống giáo dục trên cả nước, để hỗ trợ tốt việc lưu trữ thông tin của học sinh, giảm thời gian truy xuất thông tin học sinh trong các kì thi, đặc biệt là kì thi trung học phổ thông quốc gia hằng năm.

Ngoài ra, cũng với hình thức như trên, chương trình còn có thể được thay đổi đôi chút trở thành chương trình quản lý chuỗi nhà hàng khách sạn phục vụ cho kinh doanh thương mại, hay quản lý chuỗi cung ứng làm giảm hao phí như trong nhóm ngành Logitics, v.v…

Tóm lại, việc quản lý thông tin bằng danh sách liên kết là vô cùng hữu ích và đơn giản, vì thế ta có thể dùng danh sách liên kết cho vô vàn kiểu và dạng chương trình khác đa dạng hơn.

TÀI LIỆU THAM KHẢO