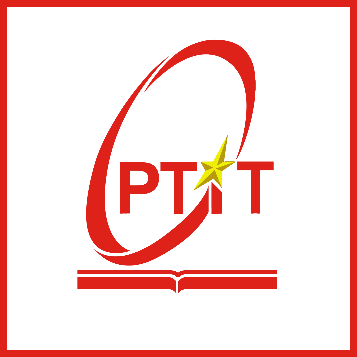
**BỘ KHOA HỌC VÀ CÔNG NGHỆ**

**HỌC VIỆN CÔNG NGHỆ BƯU CHÍNH VIỄN THÔNG**

**CƠ SỞ TẠI TP. HỒ CHÍ MINH**

**KHOA CÔNG NGHỆ THÔNG TIN 2**

-----------------------------



**BÁO CÁO   
ĐỒ ÁN MÔN HỌC**

**ĐỀ TÀI: XÂY DỰNG CHƯƠNG TRÌNH QUẢN LÝ SINH VIÊN VÀ GIẢI THUẬT MÃ ĐI TUẦN**

**Môn học:** Cấu trúc dữ liệu và giải thuật

**Giảng viên hướng dẫn:** Dương Thanh Thảo

**Thực hiện bởi nhóm sinh viên, bao gồm:**

1. Nguyễn Minh Khánh N23DCCN165
2. Lâm Thụy Khương N23DCCN169
3. Phan Trung Kiên N23DCCN170
4. Võ Nguyễn Bảo Hưng N23DCCN092

**TP.HCM, tháng 06/2025**

**MỤC LỤC**

[DANH MỤC HÌNH ẢNH 4](#_Toc200740124)

[TÓM TẮT 5](#_Toc200740125)

[1. Mục tiêu đồ án 5](#_Toc200740126)

[2. Các chức năng đã xây dựng 5](#_Toc200740127)

[3. Ngôn ngữ và công cụ sử dụng 5](#_Toc200740128)

[4. Kết quả đạt được 5](#_Toc200740129)

[CHƯƠNG I. TỔNG QUAN 6](#_Toc200740130)

[1.1. Giới thiệu đề tài 6](#_Toc200740131)

[1.1.1. Mục tiêu đề tài 6](#_Toc200740132)

[1.1.2. Liên hệ thực tế 6](#_Toc200740133)

[1.2. Cơ sở lý thuyết 6](#_Toc200740134)

[1.2.1. Mô tả về các cấu trúc dữ liệu 6](#_Toc200740135)

[1.2.2. Giới thiệu sơ bộ về các thuật toán sắp xếp 7](#_Toc200740136)

[1.2.3. Giới thiệu sơ bộ về thuật toán tìm kiếm (vét cạn, nhị phân) 7](#_Toc200740137)

[1.2.4. Thống kê dữ liệu 7](#_Toc200740138)

[1.2.5. Giới thiệu giải thuật mã đi tuần (Knight’s Tour) 7](#_Toc200740139)

[1.2.6. Phân tích độ phức tạp thuật toán 7](#_Toc200740140)

[CHƯƠNG II. THIẾT KẾ CHƯƠNG TRÌNH 10](#_Toc200740141)

[2.1. Tổng quan 10](#_Toc200740142)

[2.2. Thiết kế phần Quản lý sinh viên 10](#_Toc200740143)

[2.2.1. Giao diện chương trình (Qt GUI) 10](#_Toc200740144)

[2.2.2. Cấu trúc dữ liệu 12](#_Toc200740145)

[2.2.3. Các chức năng chính 12](#_Toc200740146)

[2.3. Thiết kế phần mã đi tuần 12](#_Toc200740147)

[2.3.1. Mô tả bài toán 12](#_Toc200740148)

[2.3.2. Input/Output 12](#_Toc200740149)

[2.3.3. Giải thuật sử dụng 13](#_Toc200740150)

[CHƯƠNG III. TRIỂN KHAI CHƯƠNG TRÌNH 14](#_Toc200740151)

[3.1. Quản lý sinh viên (Qt C++) 14](#_Toc200740152)

[3.1.1. Cấu trúc tổng thể 14](#_Toc200740153)

[3.1.2. Chức năng nhập liệu 14](#_Toc200740154)

[3.1.3. Quản lý danh sách sinh viên 15](#_Toc200740155)

[3.1.4. Sắp xếp và thống kê 20](#_Toc200740156)

[3.1.5. Tìm kiếm 24](#_Toc200740157)

[3.2. Mã đi tuần (console C++) 27](#_Toc200740158)

[3.2.1. Cài đặt thuật toán 27](#_Toc200740159)

[3.2.2. Quá trình thực thi 27](#_Toc200740160)

[3.2.3. Hiển thị kết quả 27](#_Toc200740161)

[CHƯƠNG IV. KẾT LUẬN 29](#_Toc200740162)

[4.1. Những nội dung đã hoàn thành 29](#_Toc200740163)

[4.2. Hướng phát triển 29](#_Toc200740164)

[PHỤ LỤC 30](#_Toc200740165)

[1. Bảng phân công công việc 30](#_Toc200740166)

[2. So sánh hiệu năng 30](#_Toc200740167)

[3. Đánh giá hiệu suất 31](#_Toc200740168)

[4. Những điều đã cải thiện sau buổi báo cáo 31](#_Toc200740169)

[TÀI LIỆU THAM KHẢO 32](#_Toc200740170)

# DANH MỤC HÌNH ẢNH

[Hình 1. Giao diện chính 11](#_Toc200739874)

[Hình 2. Form Thêm sinh viên 12](#_Toc200739875)

[Hình 3. Form Sửa thông tin sinh viên 12](#_Toc200739876)

[Hình 4. Các lỗi phát hiện được khi import file B.csv 16](#_Toc200739877)

[Hình 5. Thêm sinh viên mới có các thông tin không hợp lệ 17](#_Toc200739878)

[Hình 6. Thông báo các lỗi không hợp lệ khi nhập sai 17](#_Toc200739879)

[Hình 7. Thêm sinh viên hợp lệ 17](#_Toc200739880)

[Hình 8. Danh sách sinh viên sau khi thêm 18](#_Toc200739881)

[Hình 9. Chọn hàng cần xóa và nhấn nút Xóa 18](#_Toc200739882)

[Hình 10. Danh sách sau khi xóa hàng được chọn 19](#_Toc200739883)

[Hình 11. Chọn hàng cần sửa và nhấn nút Sửa 19](#_Toc200739884)

[Hình 12. Cửa sổ sửa thông tin sinh viên vừa được chọn 20](#_Toc200739885)

[Hình 13. Thông tin sau khi được sửa 20](#_Toc200739886)

[Hình 14. Sắp xếp theo mã sinh viên bằng Bubble Sort trên DSLK vòng 21](#_Toc200739887)

[Hình 15. Sắp xếp theo mã sinh viên bằng Merge Sort trên DSLK vòng 21](#_Toc200739888)

[Hình 16. Sắp xếp theo tên bằng Insertion Sort trên DSLK kép 22](#_Toc200739889)

[Hình 17. Sắp xếp theo tên bằng Quick Sort trên DSLK kép 22](#_Toc200739890)

[Hình 18. Thông báo khi chọn thuật toán sắp xếp không được hỗ trợ ở DSLK đơn 23](#_Toc200739891)

[Hình 19. Thông báo khi chọn thuật toán sắp xếp không được hỗ trợ ở DSLK vòng 23](#_Toc200739892)

[Hình 20. Danh sách sinh viên có điểm thấp nhất khi nhấn nút Điểm Thấp Nhất 24](#_Toc200739893)

[Hình 21. Danh sách sinh viên có điểm cao nhất khi nhấn nút Điểm Cao Nhất 24](#_Toc200739894)

[Hình 22. Cửa sổ danh sách điểm trung bình theo từng lớp khi nhấn nút Trung Bình 25](#_Toc200739895)

[Hình 23. Tìm kiếm theo tên “khánh” bằng phương pháp Vét Cạn trên DSLK kép 26](#_Toc200739896)

[Hình 24. Kết quả tìm kiếm được và tên đảo ngược khi người dùng chọn Yes 26](#_Toc200739897)

[Hình 25. Option đảo ngược họ tên 26](#_Toc200739898)

[Hình 26. Tìm kiếm theo tên “kho” bằng phương pháp Vét Cạn trên DSLK kép 27](#_Toc200739899)

[Hình 27. Tìm kiếm theo mã sinh viên bằng phương pháp Nhị Phân trên mảng đã được sắp xếp theo mã sinh viên, có chọn option đảo ngược họ tên 27](#_Toc200739900)

[Hình 28. Trường hợp n nhỏ (n = 10) 29](#_Toc200739901)

[Hình 29. Trường hợp n lớn (n = 40) 29](#_Toc200739902)

# TÓM TẮT

## 1. Mục tiêu đồ án

Đối với yêu cầu đồ án, nhóm em lựa chọn xây dựng hai chương trình nhỏ để quản lý sinh viên và giải bài toán mã đi tuần. Một hệ thống quản lý sinh viên với giao diện đồ họa sử dụng Qt C++, và một chương trình giải bài toán mã đi tuần (Knight’s Tour) bằng thuật toán đệ quy chạy trên nền console. Thông qua đồ án, nhóm nắm chắc kiến thức về cấu trúc dữ liệu và thuật toán được sử dụng, nâng cao kỹ năng lập trình và tư duy giải quyết vấn đề.

## 2. Các chức năng đã xây dựng

* Quản lý sinh viên (Qt C++):
* Nhập danh sách sinh viên từ file CSV, xử lý lỗi nhập liệu
* Thêm, xóa, cập nhật thông tin sinh viên
* Chọn cách lưu trữ: mảng, DSLK đơn, vòng, kép
* Tính điểm trung bình theo lớp, tìm sinh viên có điểm cao nhất/thấp nhất
* Sắp xếp sinh viên theo mã, tên hoặc điểm bằng các thuật toán: bubble sort, insertion sort, selection sort, quick sort, heap sort, merge sort
* Hiển thị kết quả sắp xếp và thời gian thực hiện
* Tìm kiếm sinh viên theo mã, tên, lớp hoặc điểm bằng tìm kiếm nhị phân/vét cạn; hỗ trợ tìm gần đúng
* Hiển thị kết quả tìm kiếm và thời gian thực hiện. Có chức năng cho người dùng chọn xuất đảo ngược thông tin họ và tên các sinh viên được tìm thấy
* Giải thuật mã đi tuần (console C++):
* Nhập kích thước bàn cờ và vị trí bắt đầu
* Hiển thị trình tự nước đi của con mã đi qua toàn bộ bàn cờ dạng k: (u,v) ra màn hình console bằng thuật toán Warnsdorff rule and Backtracking

## 3. Ngôn ngữ và công cụ sử dụng

- Ngôn ngữ: C++

- Công cụ: Qt Creator (giao diện đồ họa), Visual studio code (chạy console)

## 4. Kết quả đạt được

- Triển khai thành công toàn bộ chức năng bắt buộc và phần mở rộng mã đi tuần

- Giao diện chương trình thân thiện, có xử lý ngoại lệ

- Áp dụng hiệu quả các cấu trúc dữ liệu và thuật toán

- Chương trình hoạt động ổn định, xử lý nhanh, cho kết quả chính xác

# CHƯƠNG I. TỔNG QUAN

## 1.1. Giới thiệu đề tài

### 1.1.1. Mục tiêu đề tài

Vận dụng các cấu trúc dữ liệu như mảng, danh sách liên kết đơn, vòng, kép vào bài toán quản lý sinh viên.

Áp dụng các thuật toán sắp xếp (bubble, selection, insertion, quick, heap, merge) và tìm kiếm (tuyến tính, nhị phân) để xử lý và phân tích dữ liệu sinh viên.

Tìm hiểu để cài đặt thuật toán Warnsdorff rule and Backtracking nhằm giải bài toán mã đi tuần.

Nâng cao kỹ năng lập trình C++, làm quen với Qt và tư duy giải quyết vấn đề qua quá trình làm đồ án.

### 1.1.2. Liên hệ thực tế

Quản lý dữ liệu: Chương trình quản lý sinh viên giúp xử lý thông tin nhanh chóng, hỗ trợ thống kê, tìm kiếm, sắp xếp và phân tích dữ liệu hiệu quả, góp phần tối ưu hóa công tác quản lý giáo dục.

Tư duy thuật toán: Bài toán mã đi tuần là một ví dụ điển hình giúp rèn luyện khả năng lập trình đệ quy và tư duy thuật toán nâng cao.

Ứng dụng thực tiễn: Các cấu trúc và thuật toán sử dụng trong đề tài hoàn toàn có thể áp dụng vào các hệ thống thực tế lớn hơn trong quản lý nhân sự, học tập, hoặc các phần mềm điều phối khác.

## 1.2. Cơ sở lý thuyết

### 1.2.1. Mô tả về các cấu trúc dữ liệu

Mảng (Array): Là cấu trúc dữ liệu tuyến tính lưu trữ các phần tử liên tiếp trong bộ nhớ, cho phép truy cập trực tiếp thông qua chỉ số.

Danh sách liên kết đơn (Singly Linked List): Gồm các nút liên kết nhau bằng con trỏ, cho phép linh hoạt trong việc thêm/xóa phần tử.

Danh sách liên kết vòng (Circular Linked List): Giống danh sách đơn nhưng nút cuối trỏ về nút đầu, thích hợp cho các bài toán vòng lặp liên tục.

Danh sách liên kết kép (Doubly Linked List): Mỗi nút có hai con trỏ trỏ tới nút trước và sau, thuận tiện cho việc duyệt ngược và thao tác linh hoạt hơn.

### 1.2.2. Giới thiệu sơ bộ về các thuật toán sắp xếp

Bubble Sort: So sánh cặp kề và đổi chỗ nếu sai thứ tự, đơn giản nhưng chậm với dữ liệu lớn.

Insertion Sort: Chèn từng phần tử vào vị trí đúng trong phần đã sắp.

Selection Sort: Chọn phần tử nhỏ nhất (hoặc lớn nhất) và đưa về đầu dãy.

Merge Sort: Chia mảng thành các phần nhỏ rồi trộn lại theo thứ tự đúng, sử dụng đệ quy.

Quick Sort: Chọn pivot để chia mảng và sắp xếp hai phần, hiệu quả cao.

Heap Sort: Xây dựng heap và trích xuất phần tử lớn/nhỏ nhất theo thứ tự.

### 1.2.3. Giới thiệu sơ bộ về thuật toán tìm kiếm (vét cạn, nhị phân)

Tìm kiếm tuyến tính (vét cạn): Duyệt từng phần tử và so sánh, đơn giản nhưng không tối ưu.

Tìm kiếm nhị phân: Áp dụng với dữ liệu đã sắp xếp, chia đôi dãy và tìm kiếm nhanh hơn rất nhiều.

### 1.2.4. Thống kê dữ liệu

Tính điểm trung bình theo lớp: duyệt qua danh sách sinh viên để tính tổng điểm và sĩ số từng lớp, sau khi duyệt xong, tính điểm trung bình và in ra theo từng lớp.

Lấy danh sách sinh viên có điểm thấp/cao nhất:

* Đối với danh sách chưa có thứ tự, vét cạn hết danh sách, tức là tìm điểm nhỏ/cao nhất, sau đó duyệt lại danh sách và in ra các sinh viên có điểm bằng điểm nhỏ/cao nhất.
* Đối với danh sách đã sắp xếp theo thứ tự, ghi nhận điểm nhỏ/cao nhất, sau đó in ra các sinh viên có điểm nhỏ/cao nhất theo thứ tự đã sắp xếp.

### 1.2.5. Giới thiệu giải thuật mã đi tuần (Knight’s Tour)

Là bài toán tìm một đường đi của quân mã trên bàn cờ vua sao cho đi qua mỗi ô đúng một lần. Sử dụng giải thuật Warnsdorff’s Rule kết hợp với Backtracking để giải bài toán Knight’s Tour.

Bài toán thể hiện khả năng áp dụng thuật toán đệ quy và kiểm tra ràng buộc trong tìm kiếm lời giải.

### 1.2.6. Phân tích độ phức tạp thuật toán

* **Các thuật toán thống kê**
* Tính điểm trung bình theo lớp:
  + Độ phức tạp thời gian: O(n)
  + Không gian: O(k) (k là số lớp)
* Liệt kê các sinh viên có điểm thấp/cao nhất:
  + Độ phức tạp thời gian: O(n) (trường hợp xấu nhất), O(1) (trường hợp tốt nhất – danh sách đã được sắp xếp)
  + Không gian: O(1)
* **Các thuật toán Sắp Xếp**
* Bubble Sort:
  + Độ phức tạp thời gian: O(n2) (trường hợp xấu nhất và trung bình).
  + Không gian: O(1) (tại chỗ).
* Insertion Sort:
  + Độ phức tạp thời gian: O(n2) (trường hợp xấu nhất), O(n) (trường hợp tốt nhất).
  + Không gian: O(1).
* Selection Sort:
  + Độ phức tạp thời gian: O(n2) (mọi trường hợp).
  + Không gian: O(1).
* Merge Sort:
  + Độ phức tạp thời gian: O() (mọi trường hợp).
  + Không gian: O(n) (do cần mảng phụ).
* Quick Sort:
  + Độ phức tạp thời gian: O(n2) (trường hợp xấu nhất), O() (trung bình).
  + Không gian: O() (do đệ quy).
* Heap Sort:
  + Độ phức tạp thời gian: O()
  + Không gian: O(1).
* **Các thuật toán Tìm Kiếm**
* Tìm kiếm tuyến tính (Vét cạn):
  + Độ phức tạp thời gian: O(n).
  + Không gian: O(1).
* Tìm kiếm nhị phân:
  + Độ phức tạp thời gian: O() (yêu cầu dữ liệu đã sắp xếp).
  + Không gian: O(1).
* **Thuật Toán Mã Đi Tuần (Knight's Tour)**
* Warnsdorff’s rule : O(n3)
* Tốn n x n ô phải duyệt, mỗi lần duyệt qua 1 ô phải xét 8 trường hợp để chọn ra k ô phù hợp điều kiện, tốn tiếp k lần để chọn ra ô có ít đường đi nhất là ô tiếp theo.
* Backtracking: O()
* Tốn 8 trường hợp phải duyệt cho mỗi ô trong bàn cờ n x n.

# CHƯƠNG II. THIẾT KẾ CHƯƠNG TRÌNH

## 2.1. Tổng quan

Gồm hai phần chính:

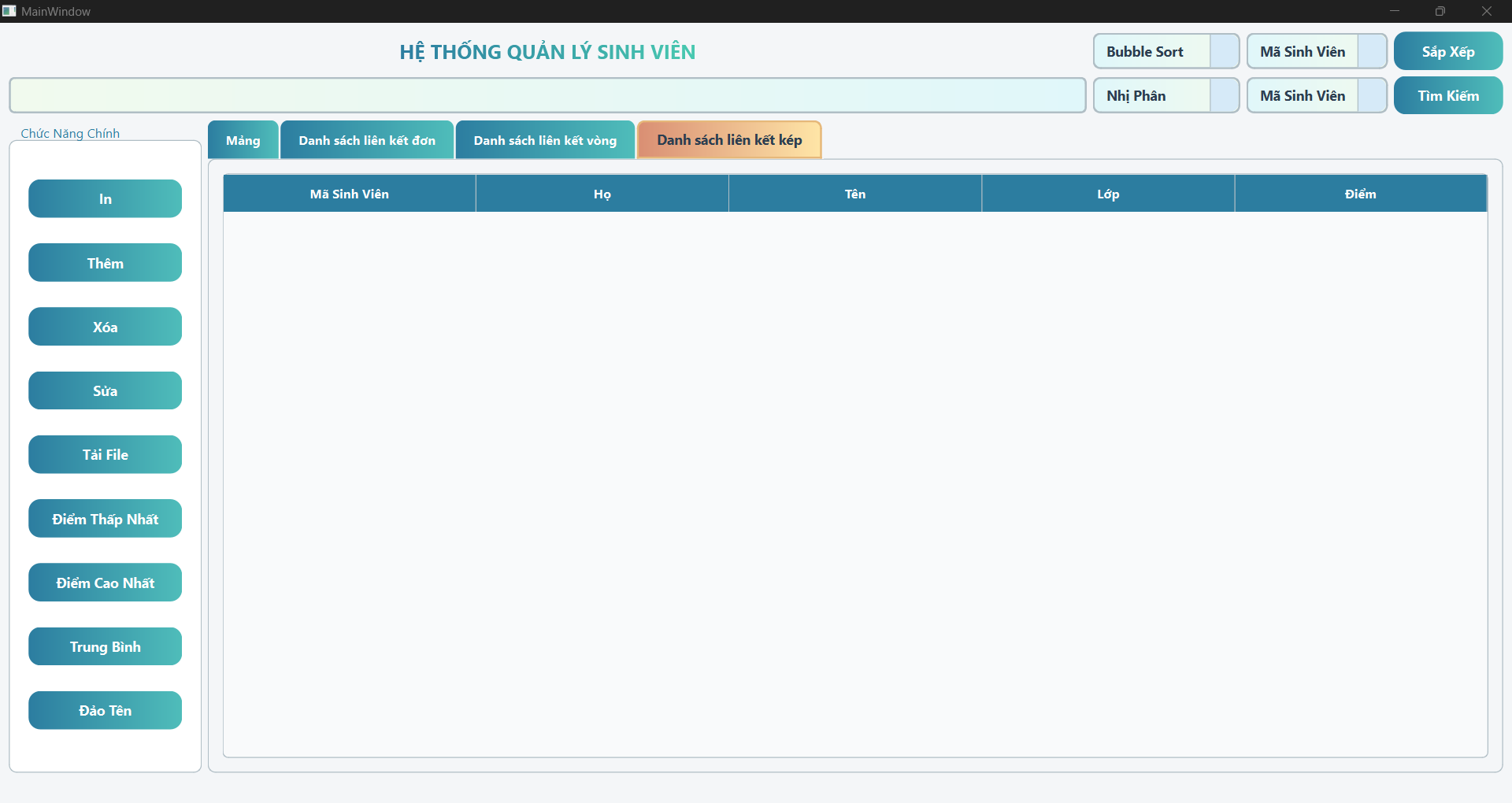
* Phần 1: Chương trình quản lý sinh viên với giao diện đồ họa sử dụng Qt C++.
* Phần 2: Chương trình giải thuật mã đi tuần (Knight’s Tour) chạy trên môi trường console với Visual Studio Code.

Người dùng có thể thao tác trực tiếp trên giao diện (đối với phần quản lý sinh viên), hoặc nhập đầu vào trực tiếp từ bàn phím (đối với phần mã đi tuần).

## 2.2. Thiết kế phần Quản lý sinh viên

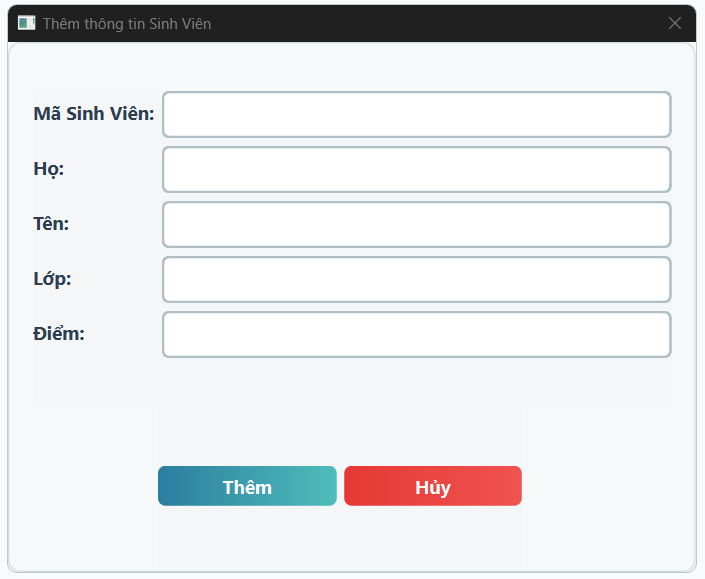
### 2.2.1. Giao diện chương trình (Qt GUI)

Màn hình chính: menu chọn kiểu lưu trữ (mảng, DSLK đơn/vòng/kép). Mỗi kiểu lưu trữ gồm các menu chức năng như: Import File, In danh sách, Thêm sinh viên, Xóa sinh viên, Sửa thông tin, Sắp xếp, Tìm kiếm, Thống kê



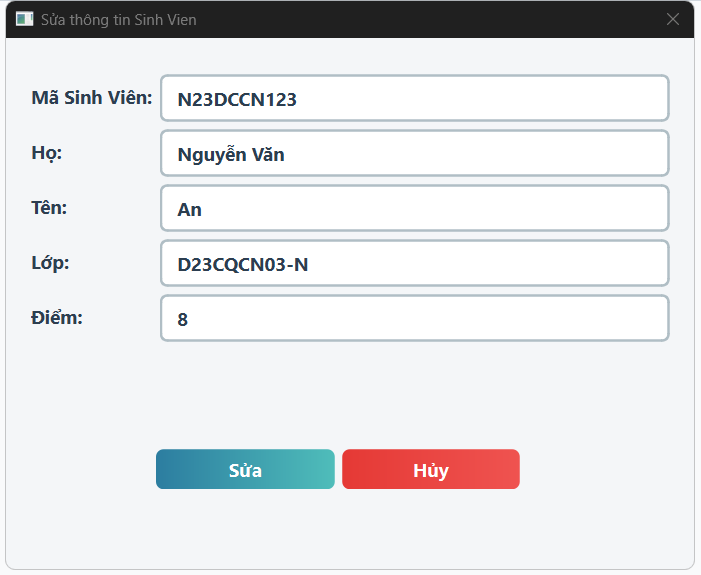
Hình 1. Giao diện chính

Form Thêm sinh viên gồm các trường: Mã Sinh Viên, Họ, Tên, Lớp, Điểm. Có kiểm tra tính hợp lệ của dữ liệu nhập vào.



Hình 2. Form Thêm sinh viên

Form Sửa thông tin sinh viên gồm các trường: Mã Sinh Viên, Họ, Tên, Lớp, Điểm. Có kiểm tra tính hợp lệ của dữ liệu nhập vào.



Hình 3. Form Sửa thông tin sinh viên

### 2.2.2. Cấu trúc dữ liệu

Người dùng có thể chọn cấu trúc lưu trữ danh sách sinh viên:

* Mảng
* Danh sách liên kết đơn
* Danh sách liên kết vòng
* Danh sách liên kết kép

Thao tác trên các kiểu lưu trữ là độc lập, không ảnh hưởng nhau; không thiết kế theo dạng chuyển dữ liệu từ kiểu lưu trữ này sang kiểu lưu trữ khác. Tùy theo lựa chọn, chương trình sẽ sử dụng các hàm xử lý tương ứng để thao tác thêm, xóa, cập nhật, thống kê, sắp xếp và tìm kiếm.

### 2.2.3. Các chức năng chính

Thêm/Xóa/Sửa: Thực hiện các thao tác trên danh sách sinh viên theo cấu trúc dữ liệu được chọn.

Sắp xếp: Cho phép người dùng chọn trường để sắp xếp (mã SV, tên, điểm) và thuật toán (bubble, insert, select, merge, quick, heap).

* DSLK đơn/vòng: bubble sort, insertion sort, selection sort, merge sort
* Mảng/DSLK kép: bubble sort, insertion sort, selection sort, merge sort, quick sort, heap sort

Tìm kiếm: Hỗ trợ tìm theo nhiều trường (mã SV, họ, tên, lớp, điểm), có thể tìm chính xác hoặc gần đúng (bỏ dấu, không phân biệt hoa thường). Không hỗ trợ tìm kiếm Nhị phân với DSLK đơn/vòng.

Thống kê: Tính điểm trung bình, tìm sinh viên có điểm cao/thấp nhất.

Kết quả: Hiển thị kết quả và thời gian xử lý từng thao tác sắp xếp, tìm kiếm.

## 2.3. Thiết kế phần mã đi tuần

### 2.3.1. Mô tả bài toán

Cho trước một bàn cờ kích thước n × n ô (2 ≤ n ≤ 200) và một vị trí bắt đầu của quân mã ở ô (u, v), với u, v ≤ n – 1. Liệt kê các bước đi để quân mã đi qua tất cả các ô trên bàn cờ đúng một lần.

### 2.3.2. Input/Output

**Input**

Kích thước bàn cờ (2 ≤ n ≤ 200)

Vị trí xuất phát (u, v)

**Output**

Hiển thị các nước đi theo thứ tự dạng k: (u, v)

### 2.3.3. Giải thuật sử dụng

Kết hợp Warnsdorff’s rule và Backtracking:

* Tại mỗi bước, tạo danh sách các bước đi hợp lệ từ vị trí hiện tại.
* Sắp xếp danh sách theo số bước đi khả thi tiếp theo (Warnsdorff's rule).
* Lần lượt thử từng bước theo thứ tự đó.
* Nếu không có hướng đi phù hợp, quay lui về bước trước để thử hướng khác.

# CHƯƠNG III. TRIỂN KHAI CHƯƠNG TRÌNH

## 3.1. Quản lý sinh viên (Qt C++)

### 3.1.1. Cấu trúc tổng thể

Thư viện GUI: Qt (Qt Widgets)

Tổ chức mã nguồn:

* model/
  + model/Sinhvien.h: Định nghĩa class/struct Sinhvien gồm các trường (msv, ho, ten, lop, diem) và (nếu cần) các hàm thành viên thuần, như serialize thành CSV hoặc toString. Không chứa logic quản lý danh sách.
  + model/Danhsachsinhvien.h/.cpp: Interface chung cho danh sách sinh viên.
  + model/Sinhvien\_mang.h/.cpp, Sinhvien\_sll.h/.cpp, Sinhvien\_cll.h/.cpp, Sinhvien\_dll.h/.cpp
  + Mỗi cặp header/cpp chứa cài đặt class kế thừa từ DanhSachSinhVien.
  + Chứa toàn bộ logic thêm/xóa/sửa/truy xuất/ImportFromCSV/điểm cao nhất/điểm thấp nhất/điểm trung bình các lớp cho cấu trúc đó.
* view/
  + mainwindow.ui: chứa UI chính, bảng hiển thị danh sách, menu, toolbar, nút tải file, nút thêm/sửa/xóa/in/điểm thấp nhất/điểm cao nhất/trung bình.
  + mainwindow.cpp/.h Điều phối UI : khởi tạo đối tượng danh sách, gọi các phương thức model khi user thao tác, cập nhật hiển thị.
  + dialog\_them.ui/.h/.cpp, dialog\_sua.ui/.h/.cpp: giao diện lấy thông tin sinh viên mới hoặc sửa.
* main.cpp: khởi Qt app.

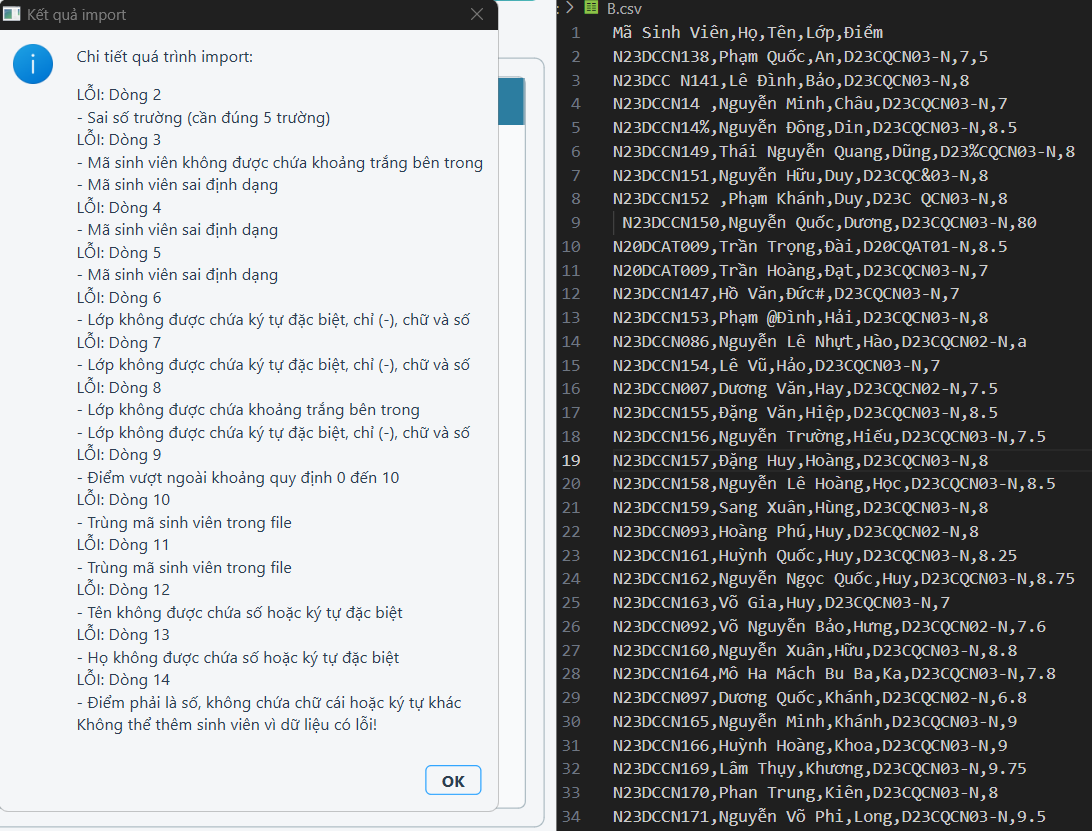
### 3.1.2. Chức năng nhập liệu

Nhập từ file CSV: Dữ liệu sinh viên được đọc và phân tích, xử lý lỗi định dạng.

Nhập thủ công: Thông qua form Thêm sinh viên trên giao diện với kiểm tra đầu vào.

Các lỗi trong mỗi trường phải được phát hiện gồm:

* Mã sinh viên:
* Sai định dạng mã sinh viên AXXAAAAXXX, với A là chữ cái, X là chữ số
* Trùng mã sinh viên
* Chứa khoảng trắng trong mã sinh viên (không bao gồm trường hợp khoảng trắng ở đầu và cuối mã), chứa ký tự đặc biệt
* Họ, tên: chứa chữ số, ký tự đặc biệt
* Lớp: chứa khoảng trắng (không bao gồm trường hợp khoảng trắng ở đầu và cuối), chứa ký tự đặc biệt
* Điểm:
* Vượt ngoài khoảng quy định 0 ≤ Điểm ≤ 10
* Là chữ cái
* Chứa dấu phẩy



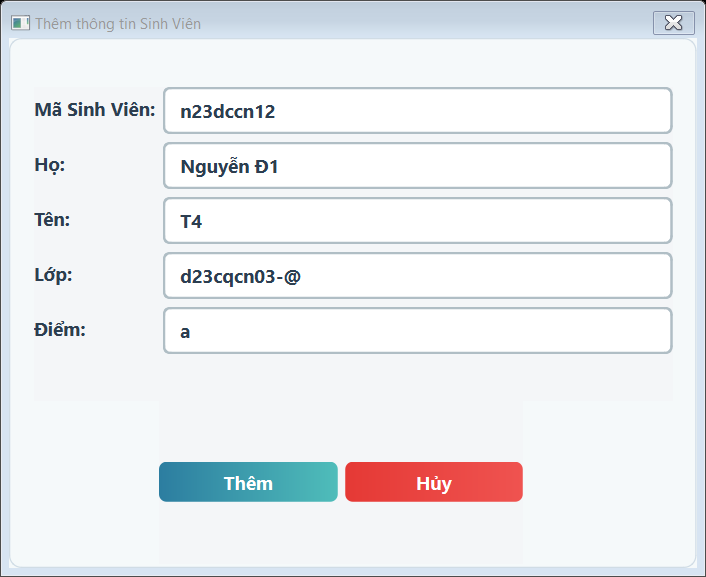
Hình 4. Các lỗi phát hiện được khi import file B.csv

### 3.1.3. Quản lý danh sách sinh viên

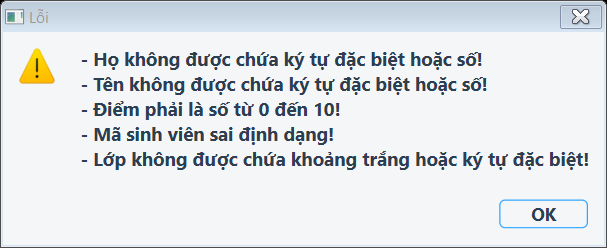
Hiển thị danh sách: Dữ liệu được hiển thị trên bảng Qt TableWidget khi nhấn nút In hoặc khi mới import file vào.

Sau khi import thành công file T.csv, thực hiện các thao tác sau:

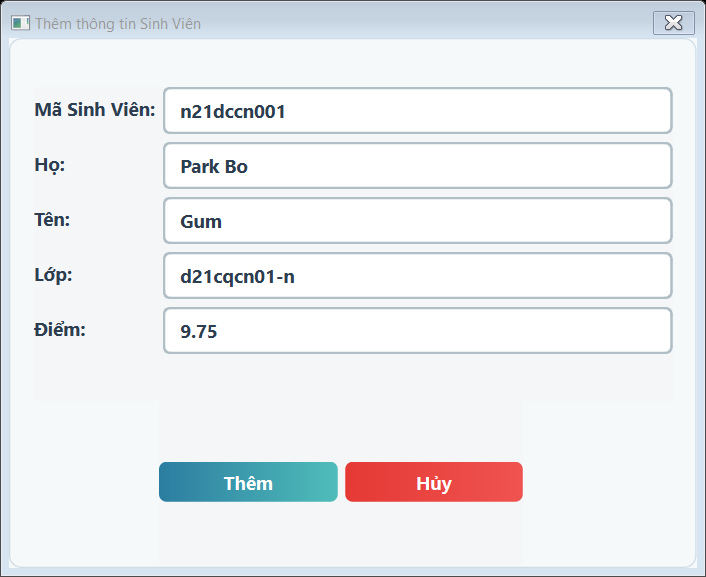
Thêm sinh viên: chọn cấu trúc lưu trữ hiện hành, gọi hàm thêm tương ứng, sinh viên mới sẽ được thêm vào cuối danh sách. Nếu nhập không hợp lệ sẽ thông báo lỗi.



Hình 5. Thêm sinh viên mới có các thông tin không hợp lệ



Hình 6. Thông báo các lỗi không hợp lệ khi nhập sai



Hình 7. Thêm sinh viên hợp lệ



Hình 8. Danh sách sinh viên sau khi thêm

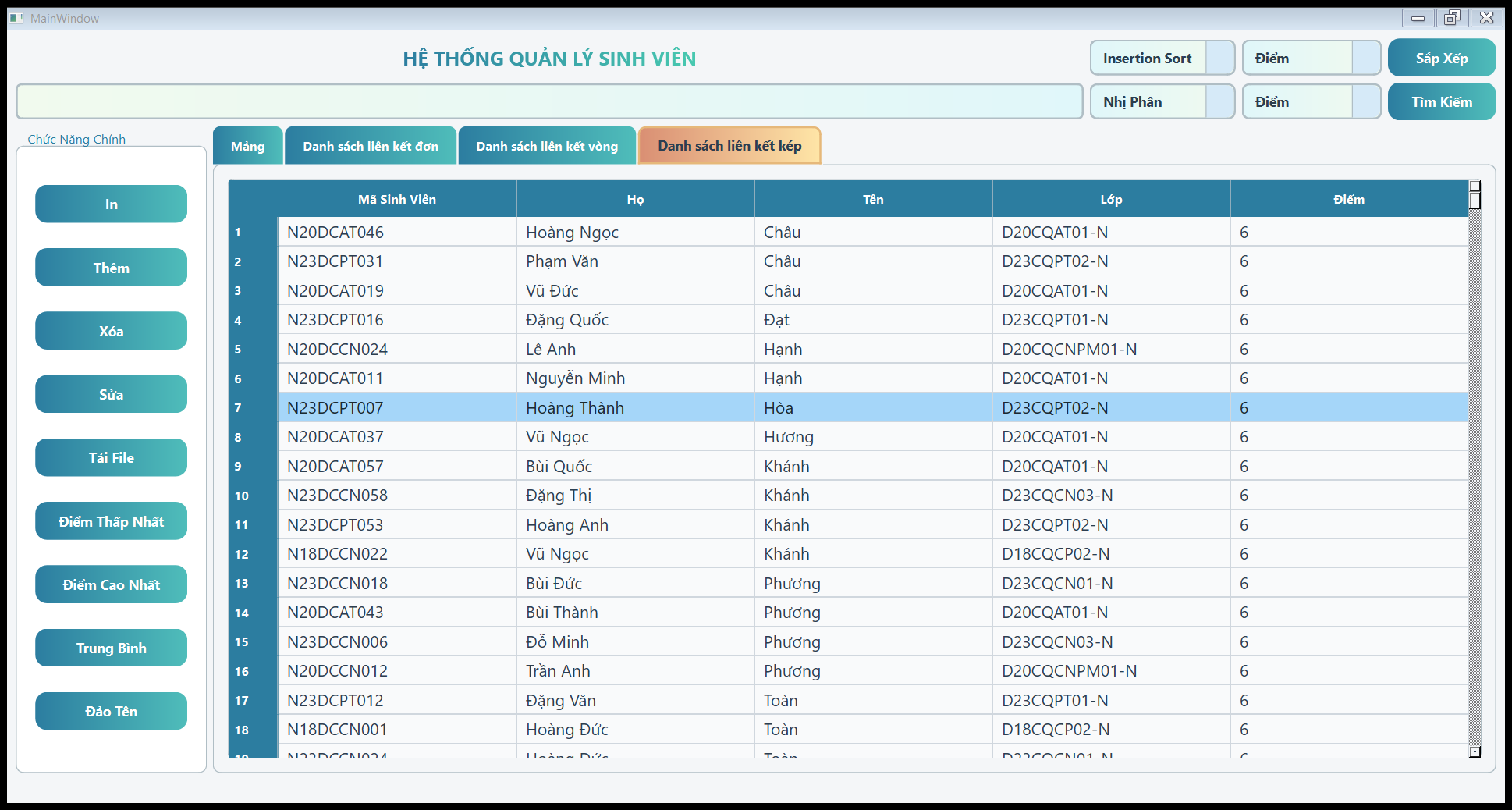
Xóa/Sửa thông tin: Hiển thị danh sách sinh viên, chọn hàng để thao tác xóa/sửa thông tin sinh viên trên cấu trúc dữ liệu được chọn.



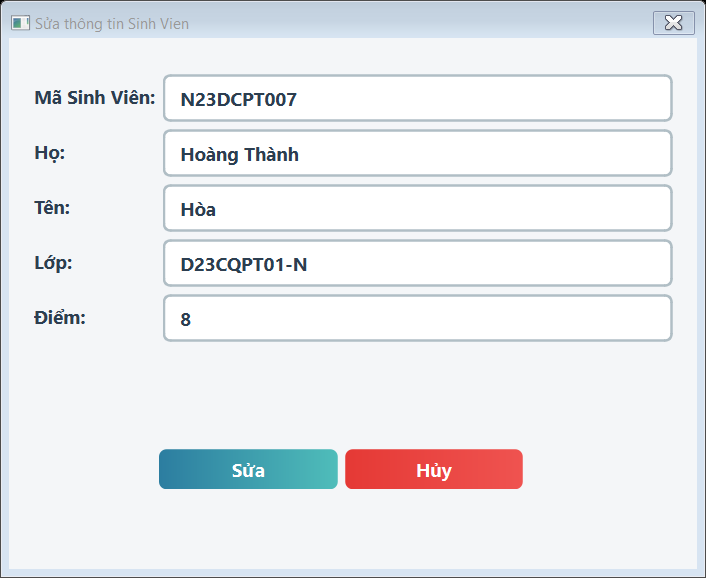
Hình 9. Chọn hàng cần xóa và nhấn nút Xóa



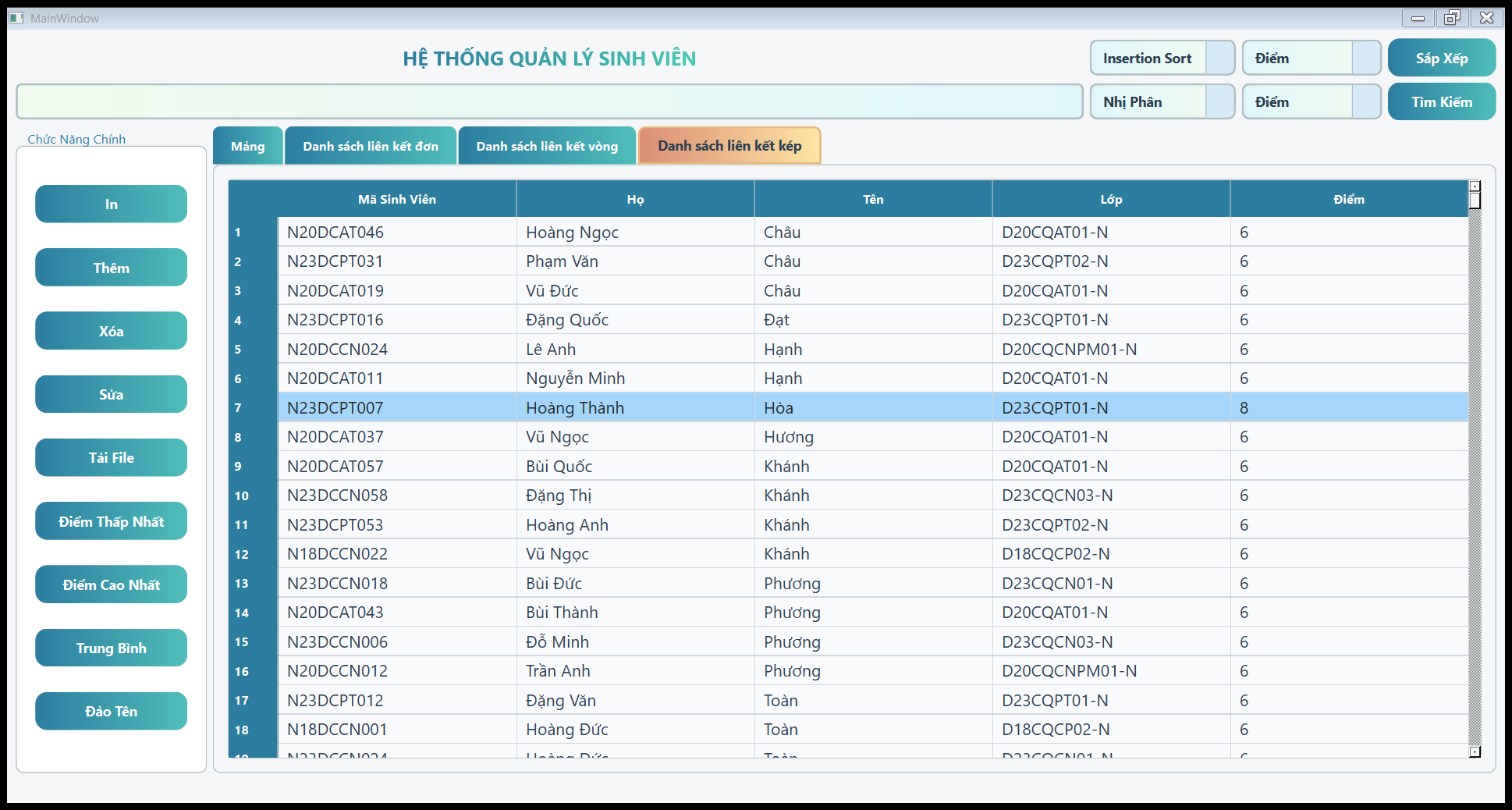
Hình 10. Danh sách sau khi xóa hàng được chọn



Hình 11. Chọn hàng cần sửa và nhấn nút Sửa



Hình 12. Cửa sổ sửa thông tin sinh viên vừa được chọn

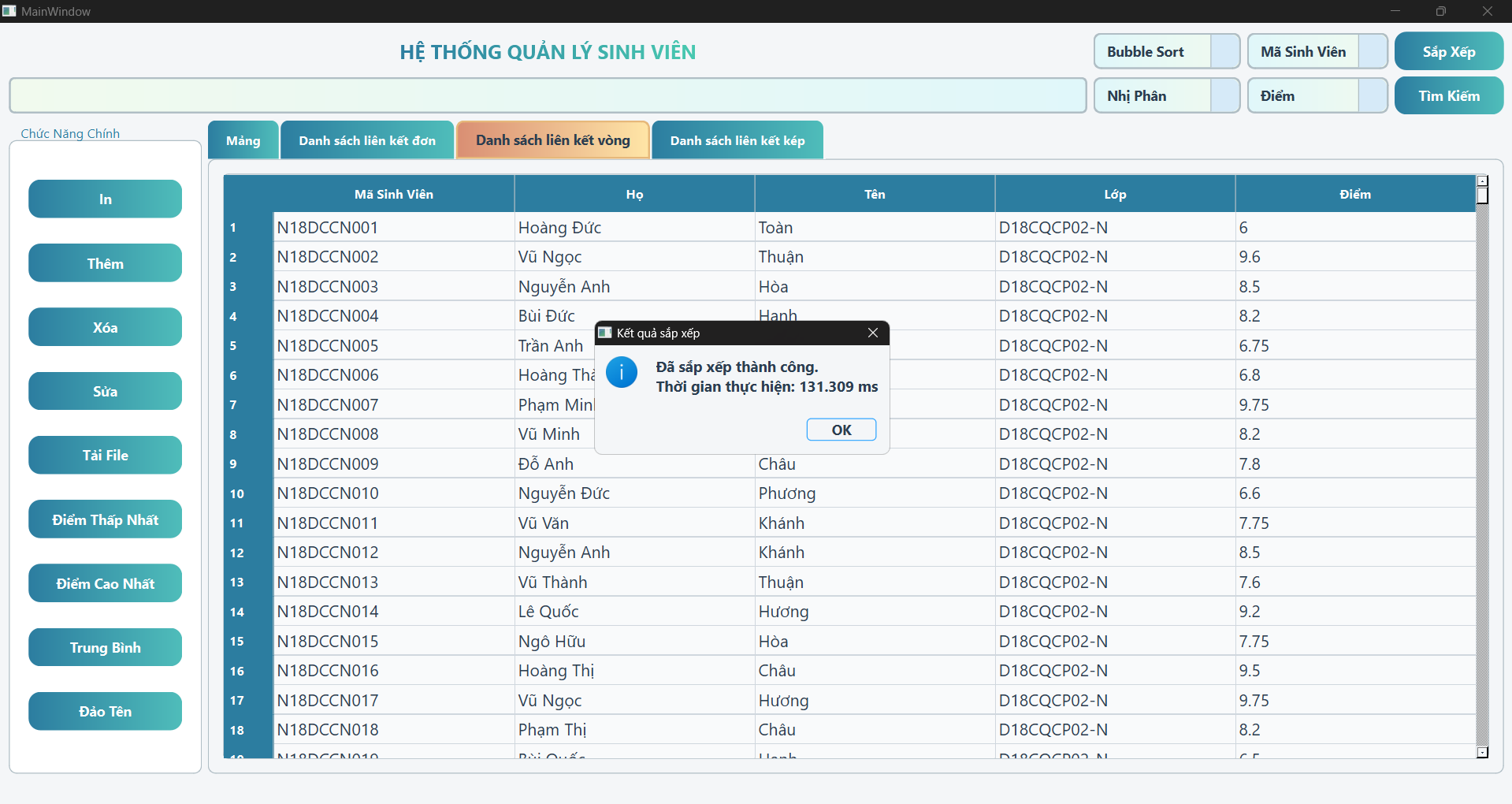


Hình 13. Thông tin sau khi được sửa

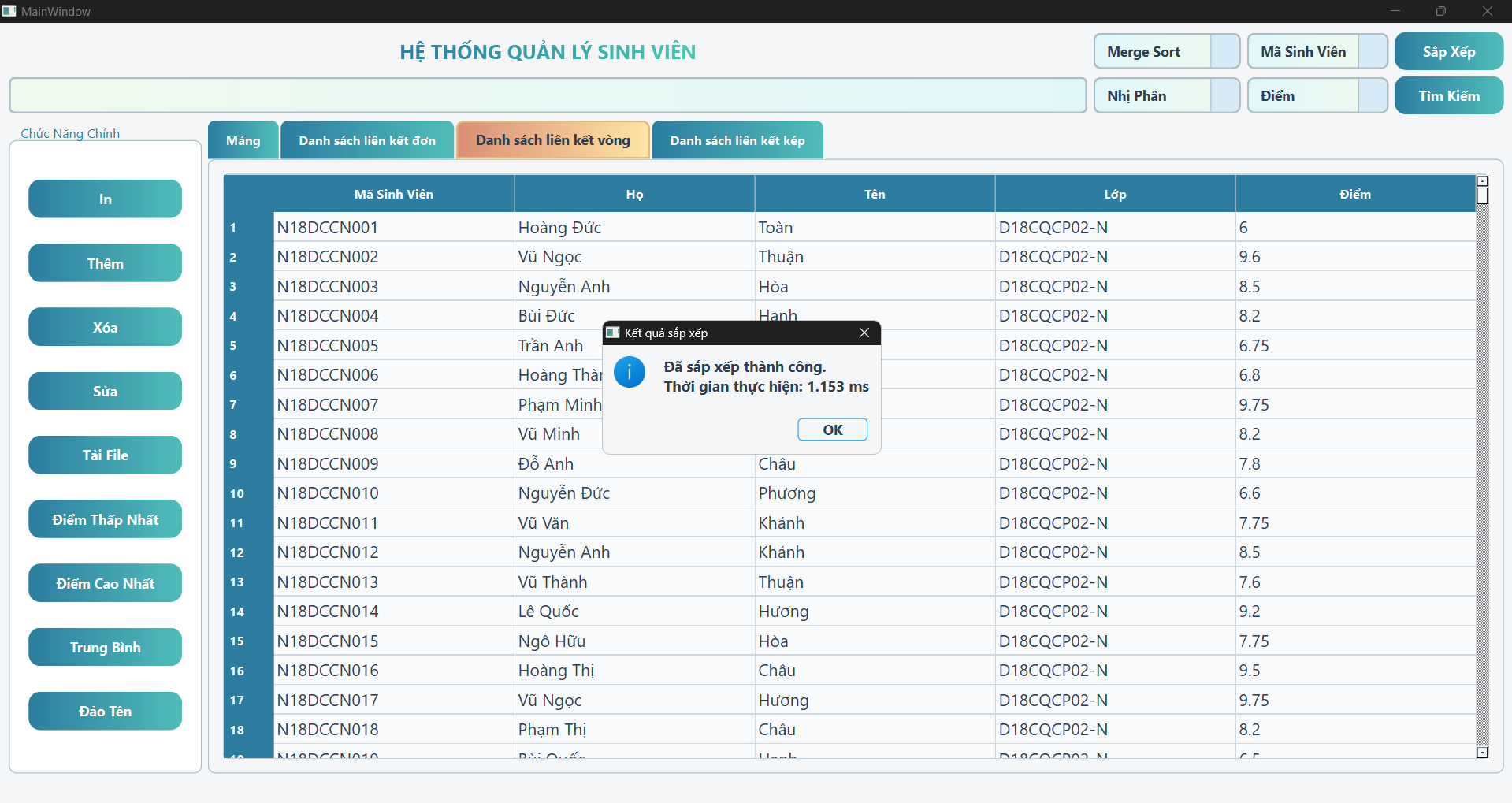
### 3.1.4. Sắp xếp và thống kê

Thuật toán sắp xếp: cho phép người dùng chọn trường (mã, tên, điểm) và thuật toán (bubble, selection, insertion, merge, quick, heap). Tùy loại danh sách đang thao tác là gì sẽ thực hiện được thuật toán sắp xếp và tìm kiếm được phép (đề cập ở mục 2.2.3).

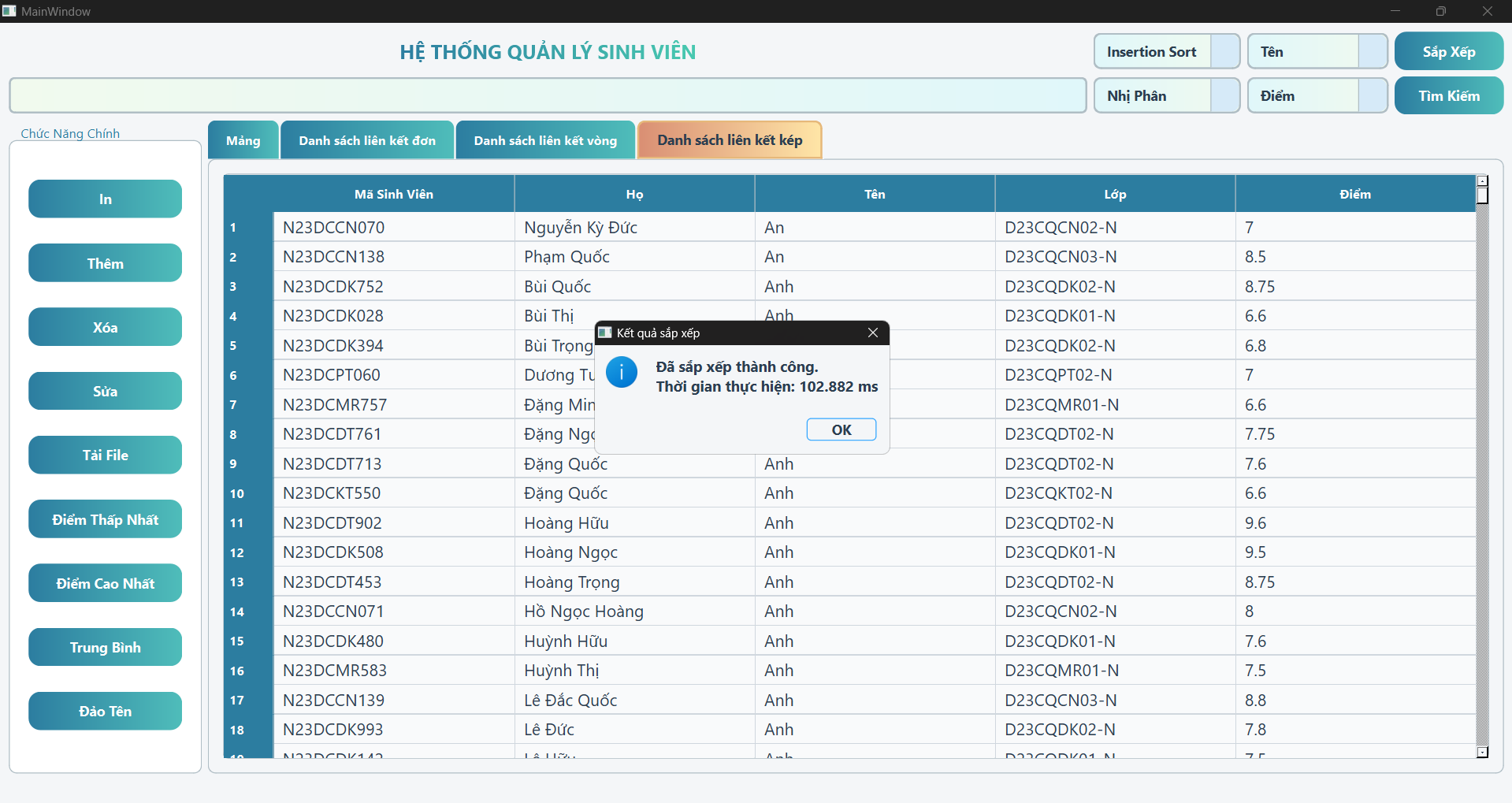
Demo một vài trường hợp tiêu biểu:



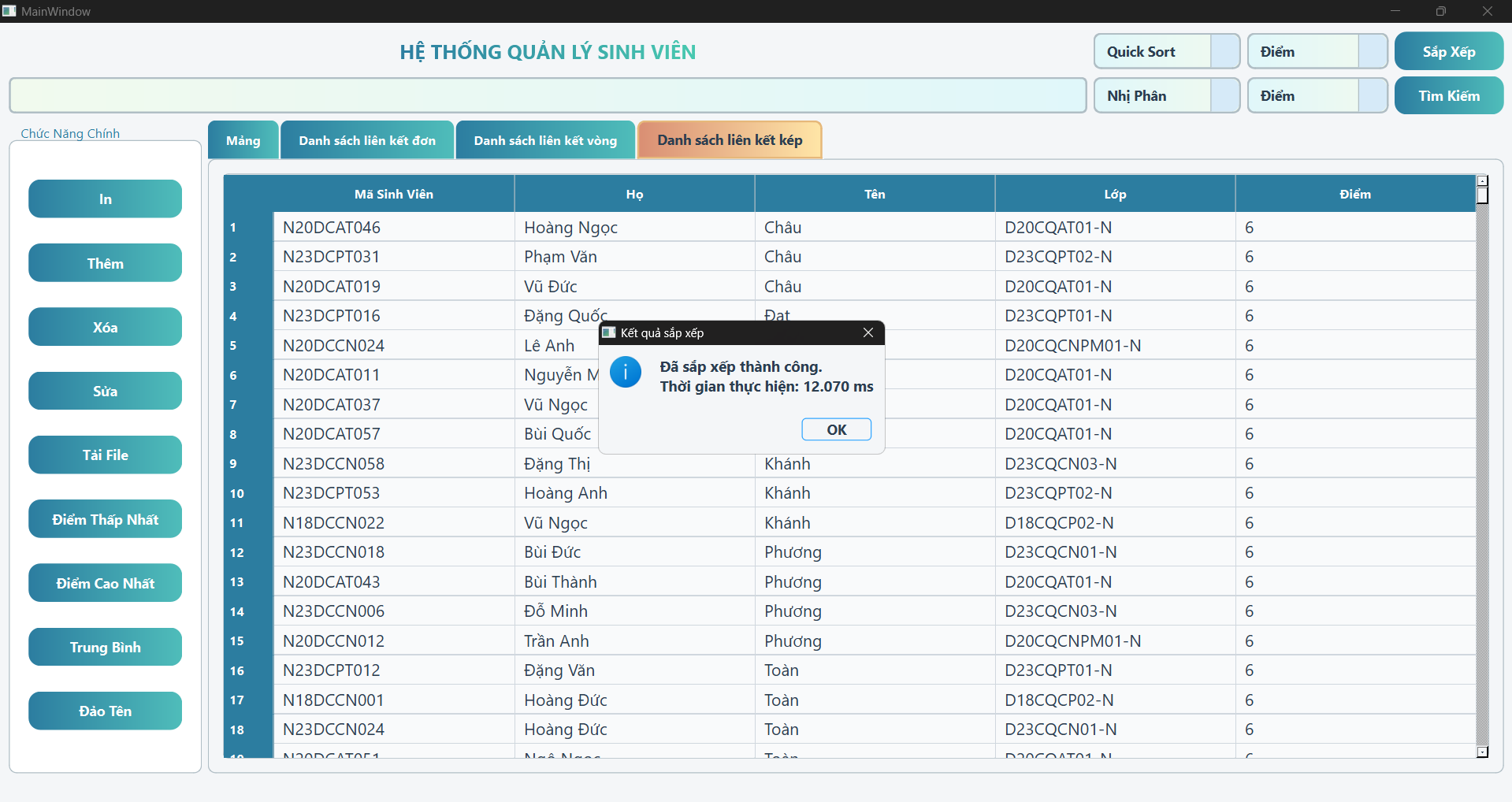
Hình 14. Sắp xếp theo mã sinh viên bằng Bubble Sort trên DSLK vòng



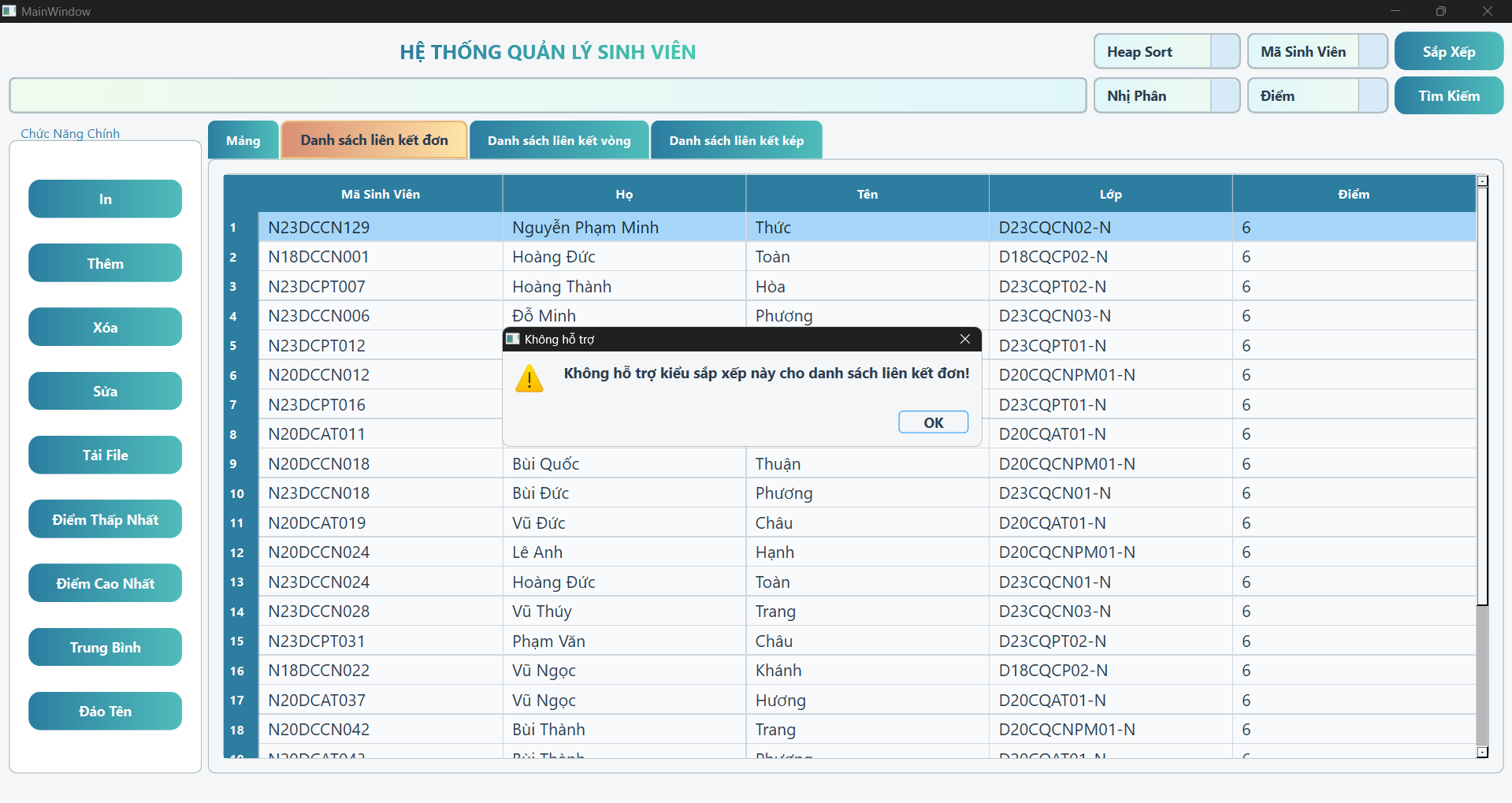
Hình 15. Sắp xếp theo mã sinh viên bằng Merge Sort trên DSLK vòng



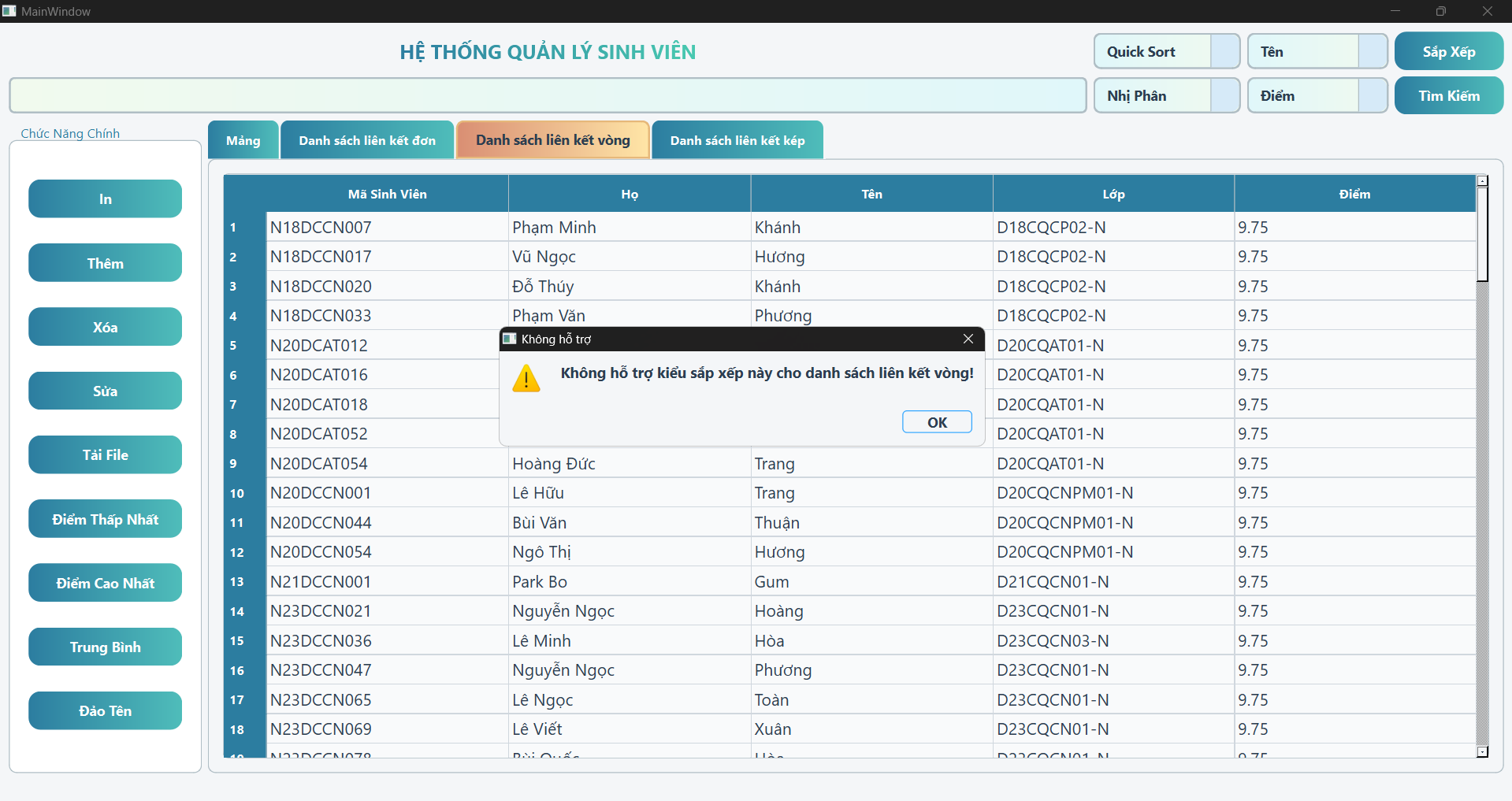
Hình 16. Sắp xếp theo tên bằng Insertion Sort trên DSLK kép



Hình 17. Sắp xếp theo tên bằng Quick Sort trên DSLK kép



Hình 18. Thông báo khi chọn thuật toán sắp xếp không được hỗ trợ ở DSLK đơn



Hình 19. Thông báo khi chọn thuật toán sắp xếp không được hỗ trợ ở DSLK vòng

Thống kê:

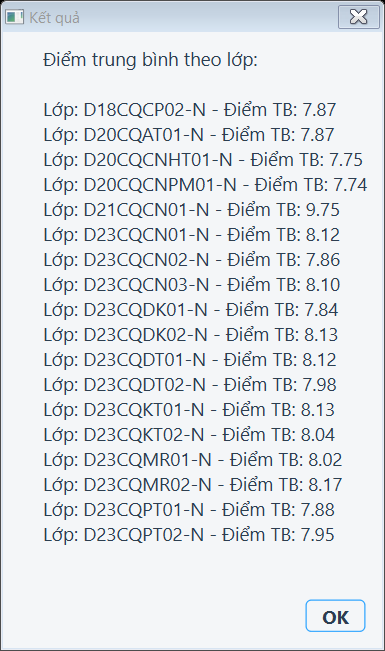
* Tìm các sinh viên có điểm thấp nhất, cao nhất
* Tính điểm trung bình từng lớp
* In kết quả lên bảng



Hình 20. Danh sách sinh viên có điểm thấp nhất khi nhấn nút Điểm Thấp Nhất



Hình 21. Danh sách sinh viên có điểm cao nhất khi nhấn nút Điểm Cao Nhất



Hình 22. Cửa sổ danh sách điểm trung bình theo từng lớp khi nhấn nút Trung Bình

### 3.1.5. Tìm kiếm

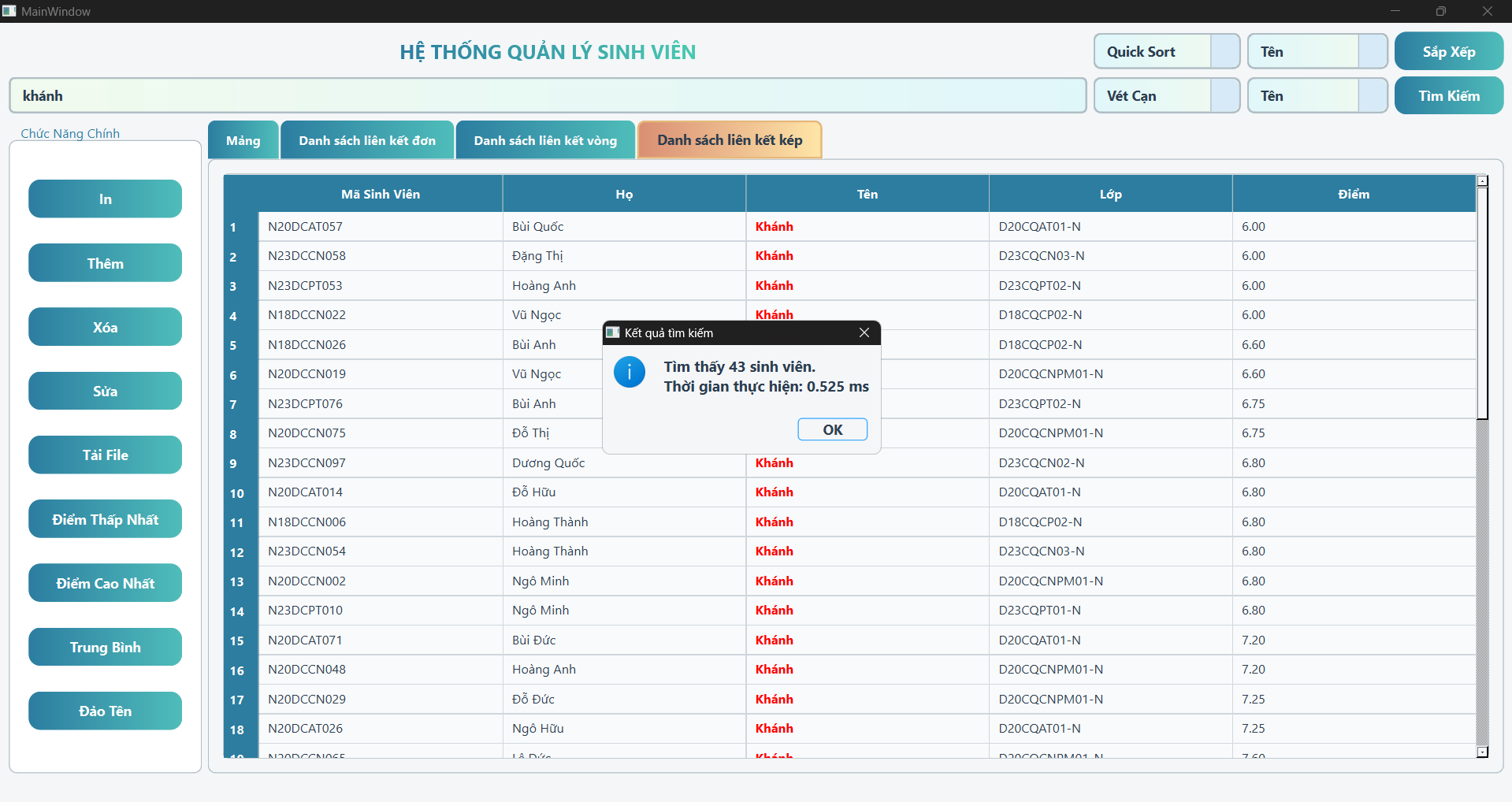
Tìm tuyến tính (vét cạn): Tìm tất cả các sinh viên phù hợp với từ khóa nhập vào.

Tìm nhị phân: Áp dụng sau khi danh sách được sắp xếp theo trường phù hợp.

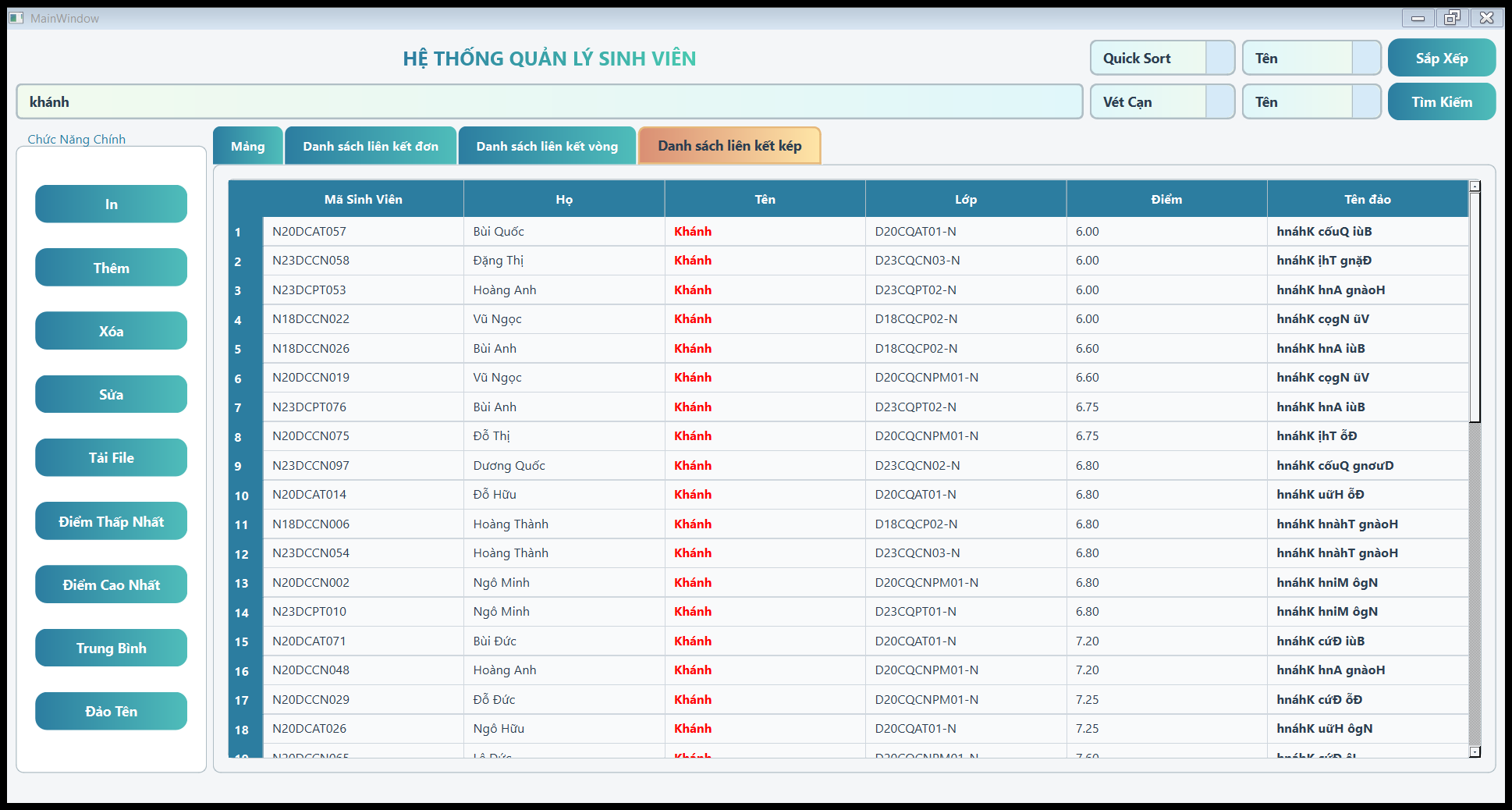
Tìm gần đúng: Hỗ trợ bỏ dấu, không phân biệt chữ hoa – thường.

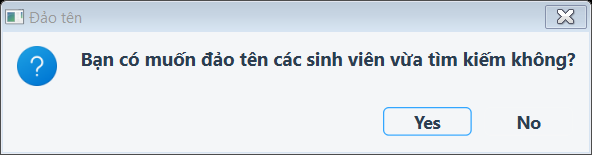
Kết quả tìm kiếm: In ra danh sách các sinh viên phù hợp, có thể chọn xuất họ tên đảo ngược (VD: "Nguyễn Văn A" → "A Văn Nguyễn").

Demo một vài trường hợp:

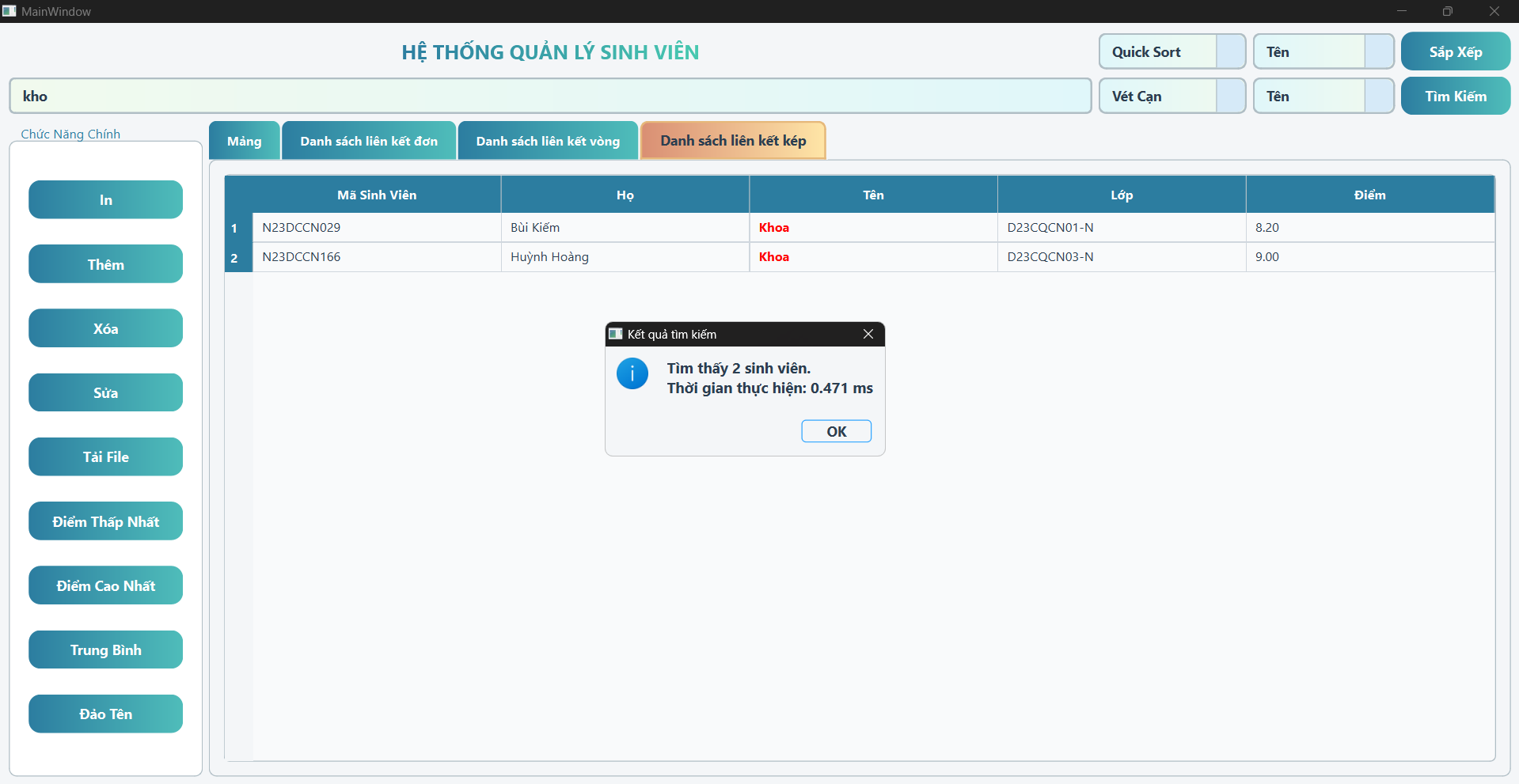


Hình 23. Tìm kiếm theo tên “khánh” bằng phương pháp Vét Cạn trên DSLK kép



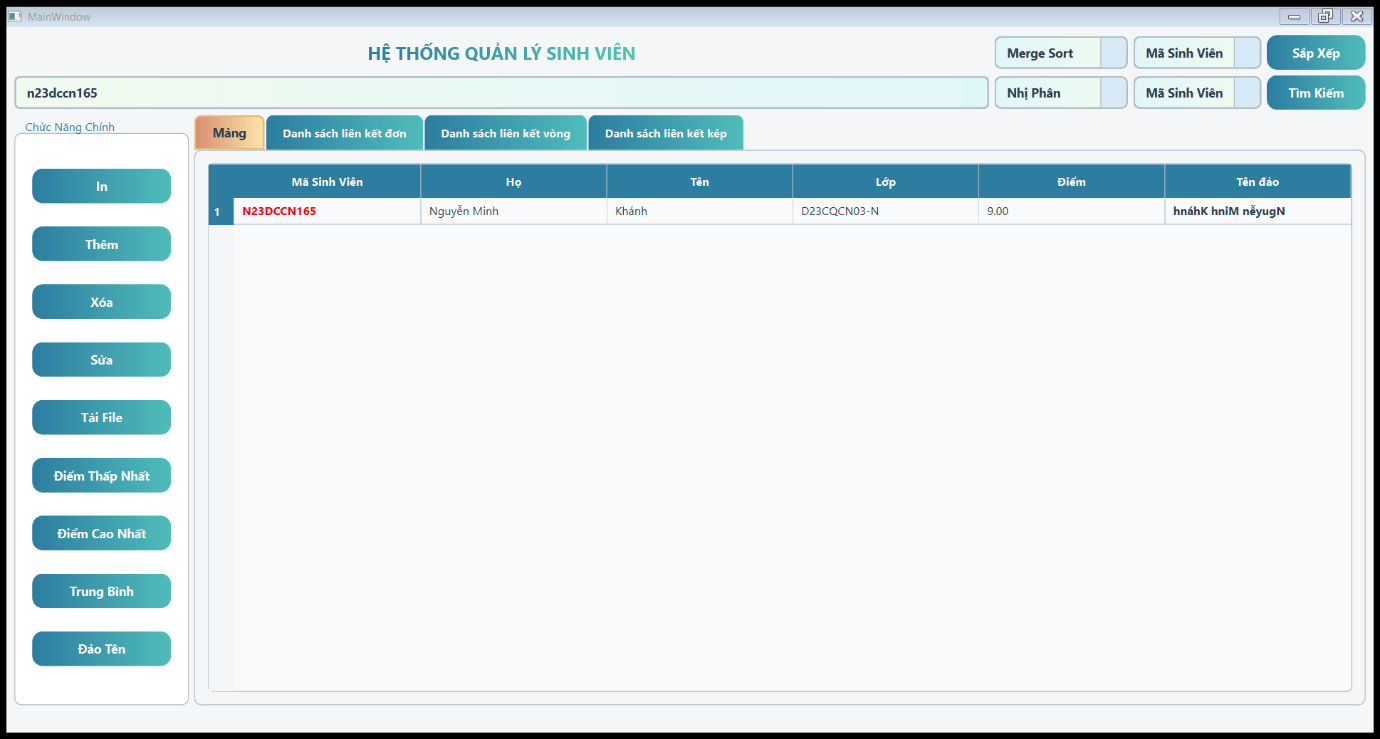
Hình 24. Kết quả tìm kiếm được và tên đảo ngược khi người dùng chọn Yes

Hình 25. Option đảo ngược họ tên



Hình 26. Tìm kiếm theo tên “kho” bằng phương pháp Vét Cạn trên DSLK kép

Trước khi thực hiện tìm kiếm bằng phương pháp Nhị Phân trên mảng theo mã sinh viên phải sắp xếp lại danh sách theo mã sinh viên.



Hình 27. Tìm kiếm theo mã sinh viên bằng phương pháp Nhị Phân trên mảng đã được sắp xếp theo mã sinh viên, có chọn option đảo ngược họ tên

## 3.2. Mã đi tuần (console C++)

### 3.2.1. Cài đặt thuật toán

* back\_track(i, j, move, n) – Backtracking

Áp dụng khi n ≤ 5, chỉ dùng cho bàn cờ nhỏ do bài toán phức tạp với n lớn

* Thử tất cả 8 hướng đi của mã
* Đánh dấu ô đã đi, gọi đệ quy với bước tiếp theo
* Nếu đi đủ n\*n ô, trả về true
* get\_steps(i, j, n) – Warnsdorff’s Rule

Áp dụng khi n > 5, nhanh hơn quay lui, phù hợp cho n lớn

* Tại mỗi bước, chọn nước đi có **ít hướng tiếp theo nhất**
* Nếu không còn nước đi → thất bại
* Nếu đi đủ n\*n ô → thành công

### 3.2.2. Quá trình thực thi

1. Nhập kích thước bàn cờ n

2. Nhập vị trí bắt đầu (u, v)

3. Dựa vào n:

* Nếu n > 5 → dùng thuật toán heuristic (get\_steps)
* Nếu n ≤ 5 → dùng quay lui (back\_track)

4. Nếu tìm được đường đi → in thứ tự nước đi (output)

Ngược lại → in "No way for the knight"

5. Hỏi người dùng có muốn tiếp tục chương trình không

### 3.2.3. Hiển thị kết quả

Output: Hiển thị thứ tự nước đi theo định dạng k: (u, v) – nước đi thứ k tại ô (u, v)

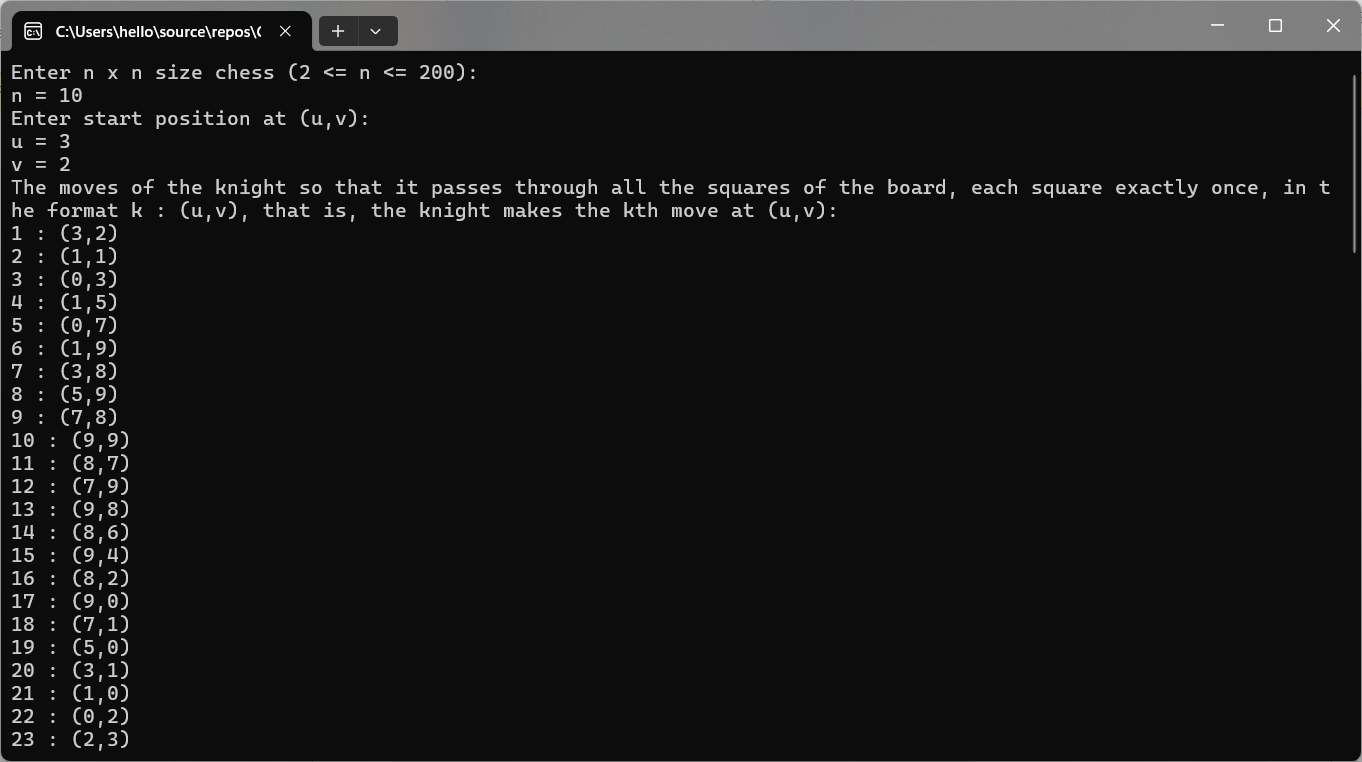
Ví dụ:

1: (0,0)

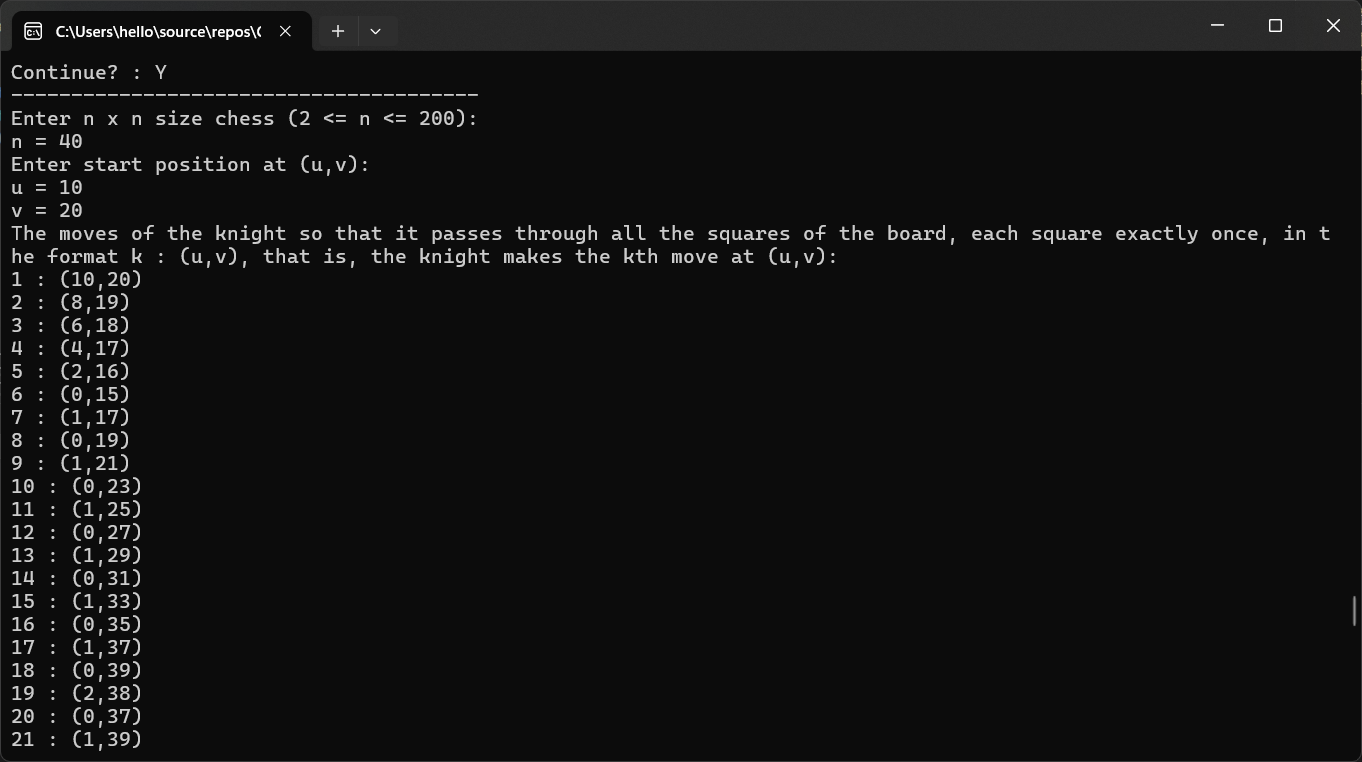
2: (2,1)

...

n²: (x,y)



Hình 28. Trường hợp n nhỏ (n = 10)



Hình 29. Trường hợp n lớn (n = 40)

# CHƯƠNG IV. KẾT LUẬN

## 4.1. Những nội dung đã hoàn thành

Sau quá trình thực hiện đề tài, nhóm đã hoàn thành toàn bộ các yêu cầu đề ra và một số phần mở rộng:

* Xây dựng thành công chương trình quản lý sinh viên với giao diện Qt, hỗ trợ đầy đủ các chức năng thêm, xóa, sửa, thống kê, sắp xếp và tìm kiếm sinh viên với các cấu trúc dữ liệu khác nhau như mảng, danh sách liên kết đơn, vòng, kép.
* Cài đặt các thuật toán sắp xếp gồm: bubble sort, insertion sort, selection sort, quick sort, heap sort và merge sort, cho phép người dùng lựa chọn thuật toán phù hợp.
* Thực hiện tìm kiếm bằng hai phương pháp tuyến tính (vét cạn) và nhị phân, hỗ trợ tìm gần đúng và xử lý tiếng Việt có dấu.
* Hoàn thành chương trình giải bài toán mã đi tuần sử dụng thuật toán Warnsdorff’s Rule kết hợp Backtracking, hoạt động ổn định với các giá trị kích thước bàn cờ khác nhau trong khoảng 2 ≤ n ≤ 200, tối ưu thuật toán cho bàn cờ có kích thước lớn (n = 200).
* Xây dựng giao diện người dùng thân thiện, trực quan và dễ sử dụng.

## 4.2. Hướng phát triển

Mở rộng và cải tiến chương trình theo các hướng sau:

* Lưu kết quả tìm kiếm/sắp xếp ra file định dạng CSV hoặc PDF để hỗ trợ thống kê và báo cáo.
* Mở rộng giao diện: Nâng cấp giao diện đồ họa đẹp hơn, tương thích đa nền tảng, thêm biểu đồ thống kê.

# PHỤ LỤC

## 1. Bảng phân công công việc

|  |  |  |  |
| --- | --- | --- | --- |
| **NHÓM 9** | | | |
| **STT** | **Họ và tên** | **Nhiệm vụ** | **Hoàn thành** |
| 1 | Nguyễn Minh Khánh | Phân công nhiệm vụ cho từng thành viên Viết code về mảng, danh sách liên kết đơn trong chương trình quản lý sinh viên | 100% |
| 2 | Lâm Thụy Khương | Làm giao diện cho chương trình quản lý sinh viên bằng QT | 100% |
| 3 | Phan Trung Kiên | Viết code về danh sách liên kết vòng và danh sách liên kết kép trong chương trình quản lý sinh viên | 100% |
| 4 | Võ Nguyễn Bảo Hưng | Viết chương trình cho giải thuật mã đi tuần (Knight's Tour) | 100% |

## 2. So sánh hiệu năng

Xét trường hợp tệ nhất, các thuật toán sẽ có độ phức tạp như sau:

Sắp xếp

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Thuật toán | Mảng | DSLK đơn | DSLK vòng | DSLK kép |
| Bubble sort | O(n2) | O(n2) | O(n2) | O(n2) |
| Insertion sort | O(n2) | O(n2) | O(n2) | O(n2) |
| Selection sort | O(n2) | O(n2) | O(n2) | O(n2) |
| Merge sort | O(n log n) | O(n log n) | O(n log n) | O(n log n) |
| Quick sort | O(n2) |  |  | O(n2) |
| Heap sort | O(n log n) |  |  | O(n2 log n) |

Tìm kiếm

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Phương pháp | Mảng | DSLK đơn | DSLK vòng | DSLK kép |
| Tuyến tính | O(n) | O(n) | O(n) | O(n) |
| Nhị phân (yêu cầu dữ liệu đã sắp xếp) | O(n) |  |  | O(n2) |

## 3. Đánh giá hiệu suất

**Quản lý sinh viên**: Các thao tác cơ bản như tìm kiếm, duyệt, thêm, xóa được xử lý ổn định với O(n) thời gian. Các thuật toán sắp xếp có nhiều lựa chọn (Merge, Quick, Heap), phù hợp xử lý mảng lớn. Dễ mở rộng thêm chức năng khác. Tốc độ xử lý tốt với dữ liệu nhỏ - trung bình.

**Mã đi tuần**: Hoạt động ổn định với các giá trị kích thước bàn cờ khác nhau trong khoảng yêu cầu 2 ≤ n ≤ 200, tối ưu thuật toán cho bàn cờ có kích thước lớn (n = 200).

## 4. Những điều đã cải thiện sau buổi báo cáo

* Kiểm tra lỗi khi import file .csv vào gồm:
  + Hiển thị lỗi trùng mã sinh viên ở các dòng bị trùng
  + Báo lỗi ở các trường không hợp lệ: họ, tên, lớp không được chứa ký tự đặc biệt, điểm phải là chữ số từ 0 – 10, báo lỗi khi điểm chứa dấu phẩy
* Cửa sổ sửa sinh viên:
  + Sửa được nhiều lần
  + Kiểm tra lỗi ở các trường nếu không hợp lệ
* Hiển thị được thời gian sắp xếp, tìm kiếm theo đơn vị mili giây
* Xử lý cách lấy danh sách điểm thấp/cao nhất trong trường hợp danh sách đã có thứ tự theo điểm nhằm tối ưu thời gian thực hiện.

# TÀI LIỆU THAM KHẢO

*1. Slide bài giảng môn học Cấu trúc dữ liệu và giải thuật*

*2. Qt documentation*

*3. Youtube: Qt With Ketan*