**A white rectangular frame with black border

AI-generated content may be incorrect. TRƯỜNG ĐẠI HỌC GIAO THÔNG VẬN TẢI**

**PHÂN HIỆU TẠI THÀNH PHỐ HỒ CHÍ MINH**

**KHOA CÔNG NGHỆ THÔNG TIN**

**A blue and yellow logo

AI-generated content may be incorrect.**

**MÔN: NHẬP MÔN CÔNG NGHỆ THÔNG TIN**

**ĐỀ TÀI: BLOCKCHAIN FOR PRODUCT**

Giảng viên hướng dẫn: Th.S TRẦN PHONG NHÃ

Sinh viên thực hiện: NGUYỄN DƯƠNG NHẬT MINH

Mã sinh viên: 6551071055

Lớp: CQ.CNTT.65

Khóa: 65

Tp.Hồ Chí Minh, năm 2025

**A white rectangular frame with black border

AI-generated content may be incorrect. TRƯỜNG ĐẠI HỌC GIAO THÔNG VẬN TẢI**

**PHÂN HIỆU TẠI THÀNH PHỐ HỒ CHÍ MINH**

**KHOA CÔNG NGHỆ THÔNG TIN**

**A blue and yellow logo

AI-generated content may be incorrect.**

**MÔN: NHẬP MÔN CÔNG NGHỆ THÔNG TIN**

**ĐỀ TÀI: BLOCKCHAIN FOR PRODUCT**

Giảng viên hướng dẫn: Th.S TRẦN PHONG NHÃ

Sinh viên thực hiện: NGUYỄN DƯƠNG NHẬT MINH

Mã sinh viên: 6551071055

Lớp: CQ.CNTT.65

Khóa: 65

Tp.Hồ Chí Minh, năm 2025

# LỜI CẢM ƠN

Lời nói đầu tiên, em xin gửi tới Quý Thầy Cô Bộ môn Công nghệ Thông tin Trường Đại học Giao thông vận tải phân hiệu tại thành phố Hồ Chí Minh lời chúc sức khỏe và lòng biết ơn sâu sắc.

Em xin chân thành cảm ơn quý thầy cô đã giúp đỡ tạo điều kiện để em hoàn thành báo cáo với đề tài *“Quản lý điểm thi đại học”*. Đặc biệt em xin cảm ơn thầy cô hướng dẫn đã nhiệt tình giúp đỡ, hướng dẫn cho em kiến thức, định hướng và kỹ năng để có thể hoàn thành bài báo cáo này. Tuy đã cố gắng trong quá trình nghiên cứu tìm hiểu tuy nhiên do kiến thức còn hạn chế nên vẫn còn tồn tại nhiều thiếu sót. Vì vậy em rất mong nhận được sự đóng góp ý kiến của Quý thầy cô bộ môn để đề tài của em có thể hoàn thiện hơn.

Lời sau cùng, em xin gửi lời chúc tới Quý Thầy Cô Bộ môn Công nghệ thông tin có thật nhiều sức khỏe, có nhiều thành công trong công việc. Em xin chân thành cảm ơn!

# NHẬN XÉT CỦA GIẢNG VIÊN HƯỚNG DẪN

.....................................................................................................................................

.....................................................................................................................................

.....................................................................................................................................

....................................................................................................................................

.....................................................................................................................................

.....................................................................................................................................

.....................................................................................................................................

.....................................................................................................................................

.....................................................................................................................................

.....................................................................................................................................

.....................................................................................................................................

.....................................................................................................................................

.....................................................................................................................................

.....................................................................................................................................

.....................................................................................................................................

.....................................................................................................................................

.....................................................................................................................................

Ngày.........Tháng.........Năm 2025

**Giảng viên hướng dẫn**

**KS. Trần Phong Nhã**

# MỤC LỤC

[LỜI CẢM ƠN i](#_Toc197894371)

[NHẬN XÉT CỦA GIẢNG VIÊN HƯỚNG DẪN ii](#_Toc197894372)

[MỤC LỤC iii](#_Toc197894373)

[LỜI MỞ ĐẦU v](#_Toc197894374)

[CHƯƠNG 1: TỔNG QUAN VỀ ĐỀ TÀI 1](#_Toc197894375)

[1.1. Đặt vấn đề 1](#_Toc197894376)

[1.2. Mục tiêu đề tài 1](#_Toc197894377)

[1.3. Phạm vi đề tài 2](#_Toc197894378)

[1.4. Phương pháp nghiên cứu 3](#_Toc197894379)

[CHƯƠNG 2: CƠ SỞ LÝ THUYẾT 4](#_Toc197894380)

[2.1. Hàm 4](#_Toc197894381)

[2.2. Con trỏ 5](#_Toc197894382)

[2.3. Con trỏ mảng 6](#_Toc197894383)

[2.4. Mảng con trỏ 6](#_Toc197894384)

[2.5. Con trỏ hàm 7](#_Toc197894385)

[2.6. Cấp phát động 8](#_Toc197894386)

[2.7. Xử lý tệp 9](#_Toc197894387)

[2.8. Kiểu cấu trúc 11](#_Toc197894388)

[2.9. Danh sách liên kết 12](#_Toc197894389)

[CHƯƠNG 3: THIẾT KẾ VÀ CÀI ĐẶT CHƯƠNG TRÌNH 15](#_Toc197894390)

[3.1. Phân tích yêu cầu 15](#_Toc197894391)

[3.2. Thiết kế chương trình 16](#_Toc197894392)

[3.2.1. Thiết kế cấu trúc dữ liệu 16](#_Toc197894393)

[3.2.2. Thiết kế module chức năng 17](#_Toc197894394)

[CHƯƠNG 4: KẾT QUẢ ĐẠT ĐƯỢC 19](#_Toc197894395)

[4.1. Kết quả đạt được 19](#_Toc197894396)

[4.2. Đánh giá 19](#_Toc197894397)

[4.2.1. Ưu điểm 19](#_Toc197894398)

[4.2.2. Nhược điểm 20](#_Toc197894399)

[4.3. Hướng phát triển 20](#_Toc197894400)

# LỜI MỞ ĐẦU

Trong thời đại công nghệ thông tin phát triển mạnh mẽ như hiện nay, việc ứng dụng công nghệ thông tin vào các lĩnh vực của đời sống xã hội ngày càng trở nên phổ biến và cần thiết. Đặc biệt trong lĩnh vực giáo dục, việc quản lý thông tin học sinh, sinh viên và kết quả học tập đóng vai trò quan trọng trong công tác quản lý và đánh giá chất lượng đào tạo.

Kỳ thi đại học là một trong những kỳ thi quan trọng đối với học sinh Trung học phổ thông, quyết định tương lai và hướng đi của mỗi học sinh. Việc quản lý điểm thi đại học một cách hiệu quả, chính xác và nhanh chóng là yêu cầu cấp thiết đối với các cơ sở giáo dục. Từ đó, đề tài "Chương trình quản lý điểm thi đại học" được đề xuất nhằm xây dựng một hệ thống quản lý điểm thi đại học hiệu quả, giúp các cơ sở giáo dục dễ dàng quản lý, tra cứu và thống kê điểm thi của thí sinh.

Báo cáo này trình bày quá trình nghiên cứu, thiết kế và cài đặt chương trình quản lý điểm thi đại học bằng ngôn ngữ lập trình C. Chương trình được xây dựng dựa trên các kiến thức cơ bản về lập trình C như: hàm, con trỏ, cấp phát động, xử lý tệp, kiểu cấu trúc và danh sách liên kết. Đồng thời, chương trình cũng được thiết kế với giao diện thân thiện, dễ sử dụng, đáp ứng đầy đủ các yêu cầu về quản lý điểm thi đại học.

# TỔNG QUAN VỀ ĐỀ TÀI

## Đặt vấn đề

Hàng năm, kỳ thi tốt nghiệp Trung học phổ thông và xét tuyển đại học diễn ra trên toàn quốc với số lượng thí sinh tham gia rất lớn. Việc quản lý thông tin thí sinh và điểm thi của họ là một công việc phức tạp và đòi hỏi độ chính xác cao. Các phương pháp quản lý thủ công truyền thống như sử dụng sổ sách, bảng biểu không còn phù hợp với khối lượng dữ liệu lớn và yêu cầu xử lý nhanh chóng.

Hiện nay, nhiều trường đại học và cơ sở giáo dục đã áp dụng các phần mềm quản lý điểm thi, tuy nhiên, các phần mềm này thường có chi phí cao và đòi hỏi cấu hình máy tính mạnh. Đối với các đơn vị có ngân sách hạn chế hoặc cơ sở vật chất chưa đáp ứng, việc sử dụng các phần mềm này gặp nhiều khó khăn.

Từ thực tế trên, việc xây dựng một chương trình quản lý điểm thi đại học đơn giản, hiệu quả, có thể chạy trên các máy tính có cấu hình thấp là rất cần thiết. Chương trình này sẽ giúp các cơ sở giáo dục quản lý thông tin thí sinh, điểm thi các môn, tính toán tổng điểm, xét tuyển và thống kê kết quả một cách nhanh chóng và chính xác.

## Mục tiêu đề tài

Mục tiêu chính của đề tài là xây dựng một chương trình quản lý điểm thi đại học bằng ngôn ngữ lập trình C với các chức năng cơ bản sau:

1. Quản lý thông tin thí sinh: Thêm, sửa, xóa, hiển thị thông tin thí sinh bao gồm: mã thí sinh, họ tên, ngày sinh, địa chỉ, giới tính.

2. Quản lý điểm thi: Nhập, sửa, hiển thị điểm các môn thi (Toán, Lý, Hóa, Sinh, Văn, Anh) và tính tổng điểm.

3. Tìm kiếm thông tin: Tìm kiếm thí sinh theo mã hoặc theo tên.

4. Sắp xếp dữ liệu: Sắp xếp danh sách thí sinh theo tổng điểm từ cao đến thấp.

5. Thống kê điểm thi: Hiển thị thí sinh có điểm cao nhất và thấp nhất.

6. Xét tuyển: Xuất danh sách thí sinh trúng tuyển dựa trên điểm chuẩn.

7. Lưu trữ dữ liệu: Lưu danh sách thí sinh và điểm thi vào file và đọc dữ liệu từ file.

## Phạm vi đề tài

Đề tài tập trung vào việc xây dựng một chương trình quản lý điểm thi đại học với các chức năng cơ bản, phù hợp với môn học Nhập môn lập trình C. Chương trình được thiết kế để chạy trên môi trường console (dòng lệnh) của hệ điều hành Windows.

Phạm vi của đề tài bao gồm:

- Quản lý thông tin cơ bản của thí sinh

- Quản lý điểm thi các môn và tính tổng điểm

- Tìm kiếm, sắp xếp và thống kê dữ liệu

- Lưu trữ và đọc dữ liệu từ file

Đề tài không bao gồm các chức năng nâng cao như:

- Giao diện đồ họa

- Kết nối cơ sở dữ liệu

- Bảo mật và phân quyền người dùng

- Xử lý dữ liệu trực tuyến

## Phương pháp nghiên cứu

Để thực hiện đề tài, các phương pháp nghiên cứu sau đã được áp dụng:

1. Phương pháp nghiên cứu lý thuyết: Tìm hiểu và nghiên cứu các tài liệu liên quan đến ngôn ngữ lập trình C, đặc biệt là các kiến thức về hàm, con trỏ, cấp phát động, xử lý tệp, kiểu cấu trúc và danh sách liên kết.

2. Phương pháp phân tích và thiết kế hệ thống: Phân tích yêu cầu, thiết kế cấu trúc dữ liệu và thuật toán phù hợp cho chương trình.

3. Phương pháp lập trình cấu trúc: Áp dụng phương pháp lập trình cấu trúc để xây dựng chương trình theo các module chức năng rõ ràng, dễ bảo trì và phát triển.

# CƠ SỞ LÝ THUYẾT

## Hàm

Hàm là một khối lệnh thực hiện một nhiệm vụ cụ thể và có thể được gọi từ các phần khác của chương trình. Hàm giúp chia nhỏ chương trình thành các module, làm cho mã nguồn dễ đọc, dễ bảo trì và tái sử dụng.

**Cú pháp khai báo hàm**

|  |
| --- |
| kiểu\_trả\_về tên\_hàm(danh\_sách\_tham\_số) {      // Thân hàm      return giá\_trị; // Nếu hàm có giá trị trả về  } |

**Ví dụ minh họa**

|  |
| --- |
| //  // Hàm tính tổng hai số  int tinhTong(int a, int b) {      return a + b;  }  // Hàm tính tổng điểm các môn  float tinhTongDiem(float toan, float ly, float hoa, float sinh, float van, float anh) {      return toan + ly + hoa + sinh + van + anh;  } |

Trong chương trình quản lý điểm thi đại học, hàm được sử dụng rộng rãi để thực hiện các chức năng như nhập thông tin thí sinh, hiển thị danh sách, tìm kiếm, sắp xếp, thống kê và xử lý file.

## Con trỏ

Con trỏ là một biến đặc biệt dùng để lưu trữ địa chỉ của một biến khác. Con trỏ cho phép truy cập và thay đổi giá trị của biến thông qua địa chỉ của nó.

**Cú pháp**:

|  |
| --- |
| kiểu\_dữ\_liệu \*tên\_con\_trỏ; |

Các phép toán với con trỏ

- `&`: Lấy địa chỉ của biến

- `\*`: Lấy giá trị tại địa chỉ (toán tử giải tham chiếu)

**Ví dụ minh họa**

|  |
| --- |
| **//**  int a = 10;  int \*p = &a; // p trỏ đến a  printf("Gia tri cua a: %d\n", a);     // 10  printf("Dia chi cua a: %p\n", &a);    // Địa chỉ của a  printf("Gia tri cua p: %p\n", p);     // Địa chỉ của a  printf("Gia tri ma p tro den: %d\n", \*p); // 10  \*p = 20; // Thay đổi giá trị của a thông qua p  printf("Gia tri moi cua a: %d\n", a); // 20 |

Trong chương trình quản lý điểm thi đại học, con trỏ được sử dụng để:

- Truy cập và thay đổi thông tin thí sinh

- Xây dựng danh sách liên kết

- Cấp phát bộ nhớ động cho các đối tượng thí sinh

## Con trỏ mảng

Con trỏ mảng là con trỏ trỏ đến phần tử đầu tiên của mảng. Trong C, tên mảng chính là con trỏ hằng trỏ đến phần tử đầu tiên của mảng.

**Ví dụ minh họa**

|  |
| --- |
| **//**  int arr[5] = {1, 2, 3, 4, 5};  int \*p = arr; // p trỏ đến phần tử đầu tiên của mảng arr  // Truy cập phần tử mảng thông qua con trỏ  printf("%d ", \*p);     // 1 (phần tử đầu tiên)  printf("%d ", \*(p+1)); // 2 (phần tử thứ hai)  printf("%d ", \*(p+2)); // 3 (phần tử thứ ba)  // Duyệt mảng bằng con trỏ  for (int i = 0; i < 5; i++) {      printf("%d ", \*(p+i)); // 1 2 3 4 5  } |

Trong chương trình quản lý điểm thi đại học, con trỏ mảng được sử dụng để:

- Truy cập và thao tác với mảng các thí sinh

- Duyệt qua danh sách thí sinh để tìm kiếm, sắp xếp và thống kê

## Mảng con trỏ

Mảng con trỏ là một mảng mà mỗi phần tử của nó là một con trỏ. Mảng con trỏ cho phép lưu trữ và quản lý nhiều con trỏ cùng một lúc.

**Cú pháp**

|  |
| --- |
| kiểu\_dữ\_liệu \*tên\_mảng[kích\_thước]; |

**Ví dụ**

|  |
| --- |
| int a = 10, b = 20, c = 30;  int \*arr[3]; // Mảng 3 con trỏ kiểu int  arr[0] = &a;  arr[1] = &b;  arr[2] = &c;  // Truy cập giá trị thông qua mảng con trỏ  printf("%d %d %d\n", \*arr[0], \*arr[1], \*arr[2]); // 10 20 30  // Thay đổi giá trị thông qua mảng con trỏ  \*arr[0] = 100;  printf("Gia tri moi cua a: %d\n", a); // 100 |

Trong chương trình quản lý điểm thi đại học, mảng con trỏ được sử dụng để:

- Lưu trữ danh sách các con trỏ trỏ đến các đối tượng thí sinh

- Sắp xếp danh sách thí sinh mà không cần di chuyển dữ liệu thực

|  |
| --- |
| // Khai báo  ThiSinh\* danhSachThiSinh[MAX\_STUDENTS];  // Cấp phát bộ nhớ  danhSachThiSinh[soLuongThiSinh] = (ThiSinh\*)malloc(sizeof(ThiSinh));  \*danhSachThiSinh[soLuongThiSinh] = thiSinh;  soLuongThiSinh++; |

## Con trỏ hàm

Con trỏ hàm là con trỏ trỏ đến một hàm. Con trỏ hàm cho phép lưu trữ địa chỉ của hàm và gọi hàm thông qua con trỏ.

**Cú pháp khai báo**

|  |
| --- |
| kiểu\_trả\_về (\*tên\_con\_trỏ)(danh\_sách\_tham\_số); |

**Ví dụ minh họa**

|  |
| --- |
| // Khai báo hai hàm  int cong(int a, int b) {      return a + b;  }  int tru(int a, int b) {      return a - b;  }  // Khai báo con trỏ hàm  int (\*ptrFunc)(int, int);  // Gán địa chỉ hàm cho con trỏ  ptrFunc = cong;  printf("Ket qua: %d\n", ptrFunc(5, 3)); // 8  ptrFunc = tru;  printf("Ket qua: %d\n", ptrFunc(5, 3)); // 2 |

Trong chương trình quản lý điểm thi đại học, con trỏ hàm có thể được sử dụng để:

- Tạo các hàm callback cho các thao tác như sắp xếp, tìm kiếm

- Tạo menu động với các chức năng khác nhau

## Cấp phát động

Cấp phát động là quá trình cấp phát bộ nhớ cho biến trong thời gian chạy chương trình. Trong C, cấp phát động được thực hiện thông qua các hàm "malloc()", "calloc()", "realloc()" và "free()".

Các hàm cấp phát động:

- "malloc(size) ": Cấp phát một khối bộ nhớ có kích thước `size` byte

- "calloc(n, size) ": Cấp phát `n` khối bộ nhớ, mỗi khối có kích thước "size " byte và khởi tạo giá trị 0

- "realloc(ptr, size) ": Thay đổi kích thước của khối bộ nhớ đã cấp phát

- "free(ptr) ": Giải phóng bộ nhớ đã cấp phát

**Ví dụ minh họa**:

|  |
| --- |
| // Cấp phát động cho một biến int  int \*p = (int\*)malloc(sizeof(int));  \*p = 10;  printf("Gia tri: %d\n", \*p); // 10  free(p); // Giải phóng bộ nhớ  // Cấp phát động cho một mảng  int \*arr = (int\*)malloc(5 \* sizeof(int));  for (int i = 0; i < 5; i++) {      arr[i] = i + 1;  }  for (int i = 0; i < 5; i++) {      printf("%d ", arr[i]); // 1 2 3 4 5  }  free(arr); // Giải phóng bộ nhớ |

Trong chương trình quản lý điểm thi đại học, cấp phát động được sử dụng để:

- Cấp phát bộ nhớ cho các đối tượng thí sinh

- Tạo các node trong danh sách liên kết

- Mở rộng danh sách thí sinh khi cần thiết

## Xử lý tệp

Xử lý tệp trong C cho phép đọc và ghi dữ liệu từ/vào các tệp trên đĩa. Các thao tác cơ bản với tệp bao gồm mở tệp, đọc/ghi dữ liệu và đóng tệp.

**Các hàm xử lý tệp**:

- "fopen(filename, mode) ": Mở tệp với chế độ xác định

- "fprintf(file, format, ...) ": Ghi dữ liệu định dạng vào tệp

- "fscanf(file, format, ...) ": Đọc dữ liệu định dạng từ tệp

- "fgets(str, n, file) ": Đọc một dòng từ tệp

- "fputs(str, file) ": Ghi một chuỗi vào tệp

- "fclose(file) ": Đóng tệp

**Ví dụ minh họa**:

|  |
| --- |
| // Ghi dữ liệu vào tệp  FILE \*fp = fopen("data.txt", "w");  if (fp == NULL) {      printf("Khong the mo file!\n");      return;  }  fprintf(fp, "Hello, World!\n");  fclose(fp);  // Đọc dữ liệu từ tệp  fp = fopen("data.txt", "r");  if (fp == NULL) {      printf("Khong the mo file!\n");      return;  }  char buffer[100];  fgets(buffer, sizeof(buffer), fp);  printf("Doc tu file: %s", buffer);  fclose(fp); |

Trong chương trình quản lý điểm thi đại học, xử lý tệp được sử dụng để:

- Lưu danh sách thí sinh và điểm thi vào tệp

- Đọc danh sách thí sinh và điểm thi từ tệp

- Xuất báo cáo và thống kê ra tệp

## Kiểu cấu trúc

Kiểu cấu trúc (struct) trong C cho phép tạo ra một kiểu dữ liệu mới bằng cách kết hợp các kiểu dữ liệu khác nhau. Cấu trúc giúp tổ chức dữ liệu liên quan thành một đơn vị logic.

**Cú pháp**

|  |
| --- |
| struct tên\_cấu\_trúc {      kiểu\_dữ\_liệu1 tên\_thành\_viên1;      kiểu\_dữ\_liệu2 tên\_thành\_viên2;      // ...  }; |

**Ví dụ minh họa**

|  |
| --- |
| // Khai báo cấu trúc Sinh viên  struct SinhVien {      char maSV[10];      char hoTen[50];      int tuoi;      float diemTB;  };  // Sử dụng cấu trúc  struct SinhVien sv1;  strcpy(sv1.maSV, "SV001");  strcpy(sv1.hoTen, "Nguyen Van A");  sv1.tuoi = 20;  sv1.diemTB = 8.5;  // Hiển thị thông tin  printf("Ma SV: %s\n", sv1.maSV);  printf("Ho ten: %s\n", sv1.hoTen);  printf("Tuoi: %d\n", sv1.tuoi);  printf("Diem TB: %.2f\n", sv1.diemTB); |

Trong chương trình quản lý điểm thi đại học, kiểu cấu trúc được sử dụng để:

- Định nghĩa cấu trúc dữ liệu cho thí sinh và điểm thi

- Tổ chức thông tin thí sinh thành một đơn vị logic

- Tạo các node trong danh sách liên kết

## Danh sách liên kết

Danh sách liên kết là một cấu trúc dữ liệu động, trong đó mỗi phần tử (node) chứa dữ liệu và một con trỏ trỏ đến node tiếp theo. Danh sách liên kết cho phép thêm, xóa và truy cập các phần tử một cách linh hoạt.

**Cấu trúc node**

|  |
| --- |
| struct Node {      kiểu\_dữ\_liệu data;      struct Node\* next;  }; |

**Các thao tác cơ bản với danh sách liên kết**:

- Tạo node mới

- Thêm node vào đầu/cuối danh sách

- Xóa node khỏi danh sách

- Duyệt danh sách

- Tìm kiếm node trong danh sách

**Ví dụ minh họa**:

|  |
| --- |
| // Khai báo cấu trúc node  struct Node {      int data;      struct Node\* next;  };  // Tạo node mới  struct Node\* createNode(int value) {      struct Node\* newNode = (struct Node\*)malloc(sizeof(struct Node));      if (newNode == NULL) {          printf("Loi cap phat bo nho!\n");          return NULL;      }      newNode->data = value;      newNode->next = NULL;      return newNode;  }  // Thêm node vào cuối danh sách  void addNode(struct Node\*\* head, int value) {      struct Node\* newNode = createNode(value);      if (\*head == NULL) {          \*head = newNode;          return;      }        struct Node\* current = \*head;      while (current->next != NULL) {          current = current->next;      }      current->next = newNode;  }  // Hiển thị danh sách  void displayList(struct Node\* head) {      struct Node\* current = head;      while (current != NULL) {          printf("%d -> ", current->data);          current = current->next;      }      printf("NULL\n");  }  // Sử dụng danh sách liên kết  struct Node\* head = NULL;  addNode(&head, 10);  addNode(&head, 20);  addNode(&head, 30);  displayList(head); // 10 -> 20 -> 30 -> NULL |

Trong chương trình quản lý điểm thi đại học, danh sách liên kết được sử dụng để:

- Lưu trữ danh sách thí sinh

- Thêm, sửa, xóa thí sinh một cách linh hoạt

- Duyệt danh sách để tìm kiếm, sắp xếp và thống kê

# THIẾT KẾ VÀ CÀI ĐẶT CHƯƠNG TRÌNH

## Phân tích yêu cầu

Dựa trên mục tiêu đề tài, chương trình quản lý điểm thi đại học cần đáp ứng các yêu cầu chức năng sau:

1. **Quản lý thông tin thí sinh**:

- Thêm thí sinh mới vào danh sách

- Hiển thị danh sách thí sinh

- Sửa thông tin thí sinh

- Xóa thí sinh khỏi danh sách

2. **Quản lý điểm thi**:

- Nhập điểm các môn thi (Toán, Lý, Hóa, Sinh, Văn, Anh)

- Tính tổng điểm tự động

- Hiển thị điểm thi của thí sinh

3. **Tìm kiếm thông tin**:

- Tìm kiếm thí sinh theo mã

- Tìm kiếm thí sinh theo tên

4. **Sắp xếp dữ liệu**:

- Sắp xếp danh sách thí sinh theo tổng điểm từ cao đến thấp

5. **Thống kê điểm thi**:

- Thống kê số lượng thí sinh theo các ngưỡng điểm

- Hiển thị thí sinh có điểm cao nhất

- Hiển thị thí sinh có điểm thấp nhất

6. **Xét tuyển**:

- Xuất danh sách thí sinh trúng tuyển dựa trên điểm chuẩn

- Tính tỷ lệ trúng tuyển

7. **Lưu trữ dữ liệu**:

- Lưu danh sách thí sinh và điểm thi vào file

- Đọc danh sách thí sinh và điểm thi từ file

Ngoài ra, chương trình cần có giao diện menu thân thiện, dễ sử dụng, cho phép người dùng lựa chọn các chức năng một cách thuận tiện.

## Thiết kế chương trình

### Thiết kế cấu trúc dữ liệu

Để lưu trữ thông tin thí sinh và điểm thi, chương trình sử dụng các cấu trúc dữ liệu sau:

**Cấu trúc DiemThi**: Lưu trữ điểm các môn thi và tổng điểm

|  |
| --- |
| typedef struct {      float toan;      float ly;      float hoa;      float sinh;      float van;      float anh;      float tong;  } DiemThi; |

**Cấu trúc ThiSinh**

|  |
| --- |
| typedef struct {      char maThiSinh[MAX\_ID];      char hoTen[MAX\_NAME];      char ngaySinh[15];      char diaChi[MAX\_ADDRESS];      int gioiTinh; // 0: Nam, 1: Nu      DiemThi diem;  } ThiSinh; |

**Cấu trúc Node**

|  |
| --- |
| typedef struct Node {      ThiSinh data;      struct Node\* next;  } N |

**Mảng con trỏ:** Lưu trữ danh sách các con trỏ trỏ đến các đối tượng thí sinh

|  |
| --- |
| ThiSinh\* danhSachThiSinh[MAX\_STUDENTS]; |

### Thiết kế module chức năng

Chương trình được thiết kế theo các module chức năng sau:

**Module quản lý thí sinh**:

- Hàm nhập thông tin thí sinh

- Hàm thêm thí sinh vào danh sách

- Hàm hiển thị thông tin thí sinh

- Hàm hiển thị danh sách thí sinh

- Hàm sửa thông tin thí sinh

- Hàm xóa thí sinh

**Module tìm kiếm**:

- Hàm tìm kiếm thí sinh theo mã

- Hàm tìm kiếm thí sinh theo tên

**Module sắp xếp và thống kê**:

- Hàm sắp xếp thí sinh theo tổng điểm

- Hàm thống kê điểm

- Hàm xuất danh sách thí sinh trúng tuyển

**Module xử lý file**:

- Hàm lưu danh sách thí sinh ra file

- Hàm đọc danh sách thí sinh từ file

**Module quản lý bộ nhớ**:

- Hàm tạo node mới

- Hàm giải phóng bộ nhớ

**Module giao diện**:

- Hàm hiển thị menu

- Hàm xử lý lựa chọn của người dùng

# KẾT QUẢ ĐẠT ĐƯỢC

## Kết quả đạt được

Chương trình đã thể hiện chức năng quản lý điểm thi đại học của các thí sinh, cũng như thực hiện các công việc tìm kiếm, sắp xếp cơ bản.

Ngoài ra, chương trình cũng cho phép lưu trữ dữ liệu từ file và đọc từ file lên

Một số hình ảnh demo về chương trình

A screen shot of a computer

AI-generated content may be incorrect.

A black screen with white text

AI-generated content may be incorrect.

## Đánh giá

### Ưu điểm

* Chương trình đáp ứng đầy đủ các yêu cầu đề ra
* Sử dụng cả danh sách liên kết và mảng con trỏ để tận dụng ưu điểm của cả hai cấu trúc dữ liệu
* Giao diện thân thiện, dễ sử dụng
* Xử lý dữ liệu hiệu quả, đặc biệt là trong việc sắp xếp và tìm kiếm
* Có khả năng lưu trữ và đọc dữ liệu từ file

### Nhược điểm

* Chưa có giao diện đồ họa
* Chưa có chức năng bảo mật và phân quyền người dùng
* Thuật toán sắp xếp sử dụng Bubble Sort có độ phức tạp O(n²), chưa tối ưu cho dữ liệu lớn
* Chưa có chức năng sao lưu dữ liệu tự động

## Hướng phát triển

* Cải thiện thuật toán sắp xếp, sử dụng Quick Sort hoặc Merge Sort để tăng hiệu suất
* Bổ sung chức năng tìm kiếm nâng cao với nhiều tiêu chí kết hợp
* Phát triển giao diện đồ họa để tăng tính thân thiện với người dùng
* Bổ sung chức năng bảo mật và phân quyền người dùng
* Tích hợp với cơ sở dữ liệu để quản lý dữ liệu hiệu quả hơn