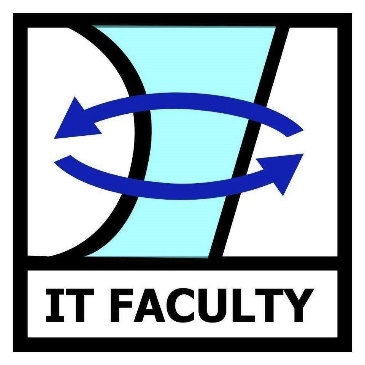
 **TRƯỜNG ĐẠI HỌC BÁCH KHOA**

**KHOA CÔNG NGHỆ THÔNG TIN**

 **🙢🕮🙠**



ĐỒ ÁN LẬP TRÌNH TÍNH TOÁN

**TÊN ĐỀ TÀI**

*XÂY DỰNG ỨNG DỤNG*

*QUẢN LÍ DANH SÁCH SINH VIÊN*

Người hướng dẫn**: ThS.** **NGUYỄN THỊ MINH HỶ**

Sinh viên thực hiện**:**

**Tên sinh viên 1: PHẠM VĂN TIẾN TRƯỞNG**

**LỚP: 20TCLC-DT4**

**Tên sinh viên 2:PHẠM CÔNG HUY**

**LỚP: 20TCLC-DT4**



MỤC LỤC

[MỤC LỤC 2](#_Toc77776039)

[DANH MỤC HÌNH VẼ 1](#_Toc77776040)

[LỜI NÓI ĐẦU 1](#_Toc77776041)

[1. TỔNG QUAN ĐỀ TÀI: 1](#_Toc77776042)

[2. CƠ SỞ LÝ THUYẾT 1](#_Toc77776043)

[2.1. Ý tưởng: 1](#_Toc77776044)

[2.2. Cơ sở lý thuyết: 2](#_Toc77776045)

[3. TỔ CHỨC CẤU TRÚC DỮ LIỆU VÀ THUẬT TOÁN 6](#_Toc77776046)

[3.1. Phát biểu bài toán 6](#_Toc77776047)

[3.2. Cấu trúc dữ liệu: 7](#_Toc77776048)

[3.3. Thuật toán 8](#_Toc77776049)

[4. CHƯƠNG TRÌNH VÀ KẾT QUẢ 14](#_Toc77776050)

[4.1. Tổ chức chương trình: 14](#_Toc77776051)

[4.2. Ngôn ngữ cài đặt: 14](#_Toc77776052)

[4.3. Kết quả 15](#_Toc77776053)

[4.3.1. Giao diện chính của chương trình 15](#_Toc77776054)

[4.3.2. Kết quả thực thi của chương trình 15](#_Toc77776055)

[4.3.3. Nhận xét đánh giá: 15](#_Toc77776056)

[5. KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN 15](#_Toc77776057)

[5.1. Kết luận: 15](#_Toc77776058)

[5.2. Hướng phát triển: 15](#_Toc77776059)

[- Tiếp thu ý kiến đóng góp và phản hồi của mọi người để phát triển thêm về chức năng và giao diện. 15](#_Toc77776060)

[TÀI LIỆU THAM KHẢO 16](#_Toc77776061)

[PHỤ LỤC 17](#_Toc77776062)

DANH MỤC HÌNH VẼ

1. Hình 1: Minh Họa Kiểu dữ liệu struct. ………………………………. Trang 2
2. Hình 2: Minh Họa Tìm Kiếm Tuần Tự. ………………………..…….. Trang 3
3. Hình 3: Minh Họa Tìm Kiếm Nhị Phân. …………………………..…. Trang 4

.

1. Hình 4: Minh Họa Sắp Xếp Quick Sort. ……………………………... Trang 5
2. Hình 5: Minh Họa Cánh Xóa Một Phần Tử Trong mảng. …………… Trang 6

1. Hình 6: Quá Trình Tìm Kiếm Một Sinh Viên. ………………………. Trang 8

1. Hình 7: Quá Trình Tìm Kiếm Sinh Viên theo Thuật Toán Tìm kiếm Nhị Phân

……………………………………………………………………….. Trang 9

1. Hình 8: Quá Trình Tìm Kiếm Tuần Tự……………………………… Trang 11
2. Hình 9: Lưu Đồ Thuật Toán Phân Đoạn Mảng……………………… Trang 14,15
3. Hình 10: Menu chính của chương trình……………………………… Trang 20
4. Hình 11: Menu lựa chọn cánh thêm sinh viên. ……………………… Trang 21
5. Hình 12: Màn hình nhập thông tin sinh viên trực tiếp. ……………… Trang 21
6. Hình 13: Màn hình nhập thông tin từ file. …………………………… Trang 22
7. Hình 14: Màn hình nhập thông tin để tìm kiếm hoặc xóa. …………… Trang 22
8. Hình 15: Màn hình hiện kết quả khi tìm thấy sinh viên. ……………... Trang 23
9. Hình 16: màn hình lựa chọn xóa sinh viên. ………………………….. Trang 23
10. Hình 17: Màn hình hiển thị danh sách sinh viên. ……………………. Trang 24
11. Hình 18: Dữ liệu đầu vào là danh sách sinh viên. …………………… Trang 24
12. Hình 19: Kết quả tìm kiếm theo tên. …………………………………. Trang 24
13. Hình 20: Kết quả tìm kiếm theo mã sinh viên. ……………………….. Trang 25
14. Hình 21: Kết quả xóa sinh viên theo tên……………………………… Trang 25
15. Hình 22: Kết quả xóa sinh viên theo mã sinh viên. ………………….. Trang 26
16. Hình 23: Kết quả sau khi sắp xếp. …………………………………… Trang 26
17. Hình 24: Kết quả sau khi cấp mã sinh viên và email. ………………… Trang 26
18. Hình 25: Kết quả khi in ra file. ……………………………………….. Trang 27

LỜI NÓI ĐẦU

Ngày nay công nghệ thông tin đóng vai trò cực kỳ quan trọng và trở thành một phần không thể thiếu trong đời sống. Việc ứng dụng một cách rộng rãi vào mọi lĩnh vực đã đem lại hiệu quả, năng suất công việc khá cao. Điều đó đòi hỏi ngày càng cải tiến công nghệ, tối ưu hóa thuật toán để phát triển nhiều tính năng hơn nữa. Có rất nhiều công cụ, môn học để giải quyết vấn đề này. Một trong những môn học nền tảng quan trọng ảnh hưởng trực tiếp đến thuật toán đó là **“*Cấu trúc dữ liệu*”**.

Để nghiên cứu kỹ hơn và xây dựng hợp lý cấu trúc dữ liệu và thuật toán, nhóm chúng em đã chọn đề tài về **“*Xây dựng ứng dụng quản lý danh sách sinh viên***.**”.**

Mục đích thực hiện đề tài: Áp dụng được kiến thức môn học cấu trúc dữ liệu vào đồ án, luyện tập khả năng làm việc nhóm.

Mục tiêu: Hiểu và trình bày được cấu trúc dữ liệu và phần code của đồ án.

Phạm vi và đối tượng nghiên cứu: Đối tượng là sinh viên, phạm vi nghiên cứu gồm: Quản lí thông tin sinh viên,sắp xếp danh sách, tìm kiếm (xóa) sinh viên, cấp mã sinh viên và tên email, cập nhật sinh viên.

Phương pháp nghiên cứu: Kết hợp kiến thức môn học cấu trúc dữ liệu, hiểu biết của bản thân và tham khảo thêm thông tin trên internet.

Ứng dụng quản lý học sinh – sinh viên đang là phần mềm mang lại nhiều lợi ích cho việc quản lý phù hợp sử dụng và hỗ trợ cho các giảng viên, giáo viên giúp quản lý các học sinh, học viên của mình được tốt hơn.

Trong quá trình thực hiện, mặc dù đã có nhiều cố gắng song không tránh khỏi những thiếu sót, chúng em rất mong nhận được sự chỉ dẫn, đóng góp của quý thầy cô để đề tài của chúng em ngày càng hoàn thiện hơn.

Đồng thời, chúng em cũng gửi lời cảm ơn chân thành đến

Cô **Nguyễn Thị Minh Hỷ** đã giúp đỡ chúng em hoàn thành đề tài này.

# TỔNG QUAN ĐỀ TÀI:

**1.1, Mục đích của đề tài**

Phần mềm hỗ trợ phòng giáo vụ trong công tác quản lý dữ liệu sinh viên, môn học, giảng viên, dễ dàng thống kê dữ liệu khi cần thiết. Bài toán “ Quản lý Sinh Viên” nhằm giải quyết và đáp ứng một cách hiệu quả các nhu cầu về mặt quản lý thông tin trong các trường Đại Học.

**1. 2, Ý nghĩa của đề tài**

Tin học hoá trong công tác quản lý nhằm giảm bớt sức lao động của con người, tiết kiệm được thời gian, độ chính xác cao, gọn nhẹ và tiện lợi hơn rất nhiều so với việc làm thủ công quản lý trên giấy tờ như trước đây. Tin học hoá giúp thu hẹp không gian lưu trữ, tránh được thất lạc dữ liệu, tự động hệ thống hoá và cụ thể hoá các thông tin theo nhu cầu của con người.

Bên cạnh đó, thông qua đề tài này giúp chúng em phát triển các kỹ năng: phân tích, thiết kế, lựa chọn cấu trúc dữ liệu và xây dựng thuật toán, lập trình ,...

# CƠ SỞ LÝ THUYẾT

## Ý tưởng

- Dùng kiểu dữ liệu Struct để định nghĩa một cấu trúc sinh viên, và một mảng struct để lưu trữ danh sách sinh viên.

- Cấu trúc sinh viên gồm :

+ Họ Lót

+ Tên

+ Giới Tính

+ Ngày Sinh

+ Địa Chỉ

+ Mã Sinh Viên

+ Email

- Công việc mà chúng ta cần làm ở mỗi yêu cầu :

+ Thêm sinh viên: Dùng các hàm nhập chuỗi để nhập thông tin sinh viên

+ Tìm Kiếm Sinh Viên: Chúng ta sẽ tìm kiếm bằng các thuật toán tìm kiếm như: Nhị phân ,Tuần tự .

+ Xóa Sinh Viên: Chúng ta cần tìm kiếm sinh viên và xóa sinh viên đó bằng cách dịch chuyển mảng lùi 1 chỉ số – phần phía sau nơi xóa.

+ Sắp xếp danh sách sinh viên : Đầu tiên chúng ta sẽ sắp xếp tên theo Alpha Beta bằng phương pháp quick sort sau đó sẽ sắp xếp các sinh viên có tên trùng nhau theo họ lót bằng phương pháp Quick Sort.

+ Xuất danh sách: Sẽ dùng vòng lặp để hiện thị lên màn hình thông tin của từng sinh viên.

+ Cấp mã sinh viên: Ban Đầu Mỗi Sinh Viên Sẽ Có Mã Sinh Viên Là 1022000, sau khi sắp xếp thì mã sinh viên sẽ tự động cấp 3 kí tự cuối theo thứ tự sau khi sắp xếp.

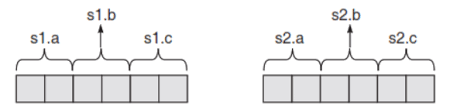
+ Cấp email: Sẽ dùng hàm xử lí chuỗi để ghép mã sinh viên với chuỗi “@gmail.com”.

## Cơ sở lý thuyết

**2.2.1, Giới thiệu về kiểu dữ liệu struct**

Structure trong C (cấu trúc trong C) là một kiểu dữ liệu người dùng tự định nghĩa (user defined datatype) cho phép bạn lưu trữ các loại phần tử khác nhau.

Mỗi phần tử của một cấu trúc được gọi là một thành viên (member).



Hình 1: Minh họa kiểu dữ liệu struct.

2.2.2, Các phương pháp tìm kiếm

2.2.2.1, **Tìm kiếm tuyến tính: Là kiểm tra tuần tự từng phần tử của mảng, đến khi nào giống thì thôi**

A picture containing text

Description automatically generated

Hình 2: Minh Họa Tìm Kiếm Tuần Tự.

int linearSearch (int a[], int n, int x) {

for (int i = 0; i < n; i++) {

if (a[i] == x) return i;

else return -1;

}

}

2.2.2.2, **Tìm kiếm nhị phân:** **Điều kiện của thuật toán này là mảng đã được sắp xếp tăng dần. So sánh x với giá trị của phần tử nằm ở giữa mảng (mid=(left+right)/2). Nếu x nhỏ hơn a[mid] thì nó chỉ có thể nằm ở nửa bên trái, ngược lại x lớn hơn a[mid] thì x nằm ở nửa bên phải. Xác định x nằm ở nửa nào thì ta lặp lại thuật toán với nửa đó. Như vậy số lần kiểm tra giảm đi nhiều do không phải mất công kiểm tra những phần tử thuộc nửa còn lại.**

int binarySearch (int a[], int n, int x) {

   int left = 0, right = n - 1, mid;

   do {

     mid = (left + right) / 2;

     if (a[mid] == x) return mid;

     else if (a[mid] <= x) left = mid + 1;

     else right = mid - 1;

   } while (left <= right);

  return -1;

Timeline

Description automatically generated}

Hình 3: Minh Họa Tìm Kiếm Nhị Phân.

2.3 **Thuật toán sắp xếp Quick Sort:**

* Chọn một phần tử trong dãy và gọi nó là phần tử chốt ***p*** (pivot).
* Chia dãy đã cho ra thành hai dãy con: Dãy con trái (***L***) gồm những phần tử không lớn hơn phần tử chốt, còn dãy con phải (***R***) gồm các phần tử không nhỏ hơn phần tử chốt. Thao tác này được gọi là thao tác phân đoạn (Partition).

 Lặp lại một cách đệ qui thuật toán đối với hai dãy con ***L*** và ***R***.

Diagram

Description automatically generated

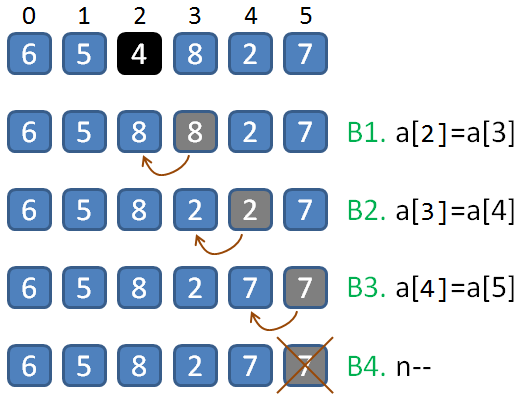
Hình 4: Minh họa sắp xếp Quick Sort.

2.4, Thuật toán xóa một phần tử

+ Đầu tiên sẽ xác định vị trí của sinh viên cần xóa.

+ Dịch chuyển mảng lùi 1 chỉ số – phần phía sau nơi xóa.

+ Giảm số lượng phần tử.



Hình 5: Minh họa cánh xóa một phần tử trong mảng .

2.5 Các kiến thức đã học được ở học phần kỹ thuật lập trình và cấu trúc dữ liệu.

# TỔ CHỨC CẤU TRÚC DỮ LIỆU VÀ THUẬT TOÁN

## Phát biểu bài toán

* Input: Nhập thủ công hoặc đọc file chứa số sinh viên cần thêm vào và thông tin cá nhân từng sinh viên. Chương trình cho phép quản lý khoản 100 sinh viên/mỗi lớp.
* Output:
* Cung cấp thông tin sinh viên mỗi một lớp, sẽ được lưu trên 1 file.
* Chức năng thêm sinh viên: nhập họ lót, tên, ngày sinh, giới tính, địa chỉ.
* Chức năng Cấp Mã Sinh Viên chương trình tự động cấp sau khi sắp xếp danh sách theo thứ tự Alphabe của Tên và Họ lót. Nếu chưa sắp xếp chương trình sẽ thông báo lỗi.
* Chức năng Địa Chỉ Email chương trình tự động cấp sau khi cấp mã sinh viên. Nếu chưa cấp mã chương trình sẽ thông báo lỗi.
* Chức năng sắp xếp danh sách.
* Chức năng tìm kiếm và xóa sinh viên.

## Cấu trúc dữ liệu

Dùng kiểu dữ liệu Struct để để định nghĩa một cấu trúc sinh viên, và một mảng Struct để lưu trữ danh sách sinh viên.

**Tổ chức lưu trữ dữ liệu:**

+ Định nghĩa một cấu trúc sinh viên:

struct first\_and\_last\_name{

char first\_name[25];

char last\_name[25];

};

struct student{

first\_and\_last\_name name;

char gender[10];

char Date\_of\_birth[15];

char address[50];

char student\_code[10];

char email[25];

};

+ Int total\_number\_of\_students: Dùng để lưu tổng số sinh viên trong lớp.

+ Bool check\_find , bool check\_innit , bool check\_sort , bool check\_provide\_student\_code , bool check\_provide\_email: Lần lượt dùng để lưu trạng thái của các hoạt động tìm kiếm, khởi tạo lớp, sắp xếp, cấp mã sinh viên, cấp email.

+ Int location\_to\_delete: Dùng để lưu vị trí sinh viên cần xóa đã tìm thấy.

## Thuật toán

**3.3.1 Tìm Kiếm Sinh Viên**

Bắt Đầu

Nhập mã sinh viên

hoặc tên sinh viên

Check\_sort ==True

Sai

Đúng

Linear\_Searnh();

inear\_Searnh();

Binary\_Searnh();

Check\_date()= =true

Sai

Đúng

Tìm kiếm theo tên hoặc mã sinh viên rồi hiển thị

Tìm kiếm theo mã sinh viên rồi hiển thị ra màn hình

Hình 6: Quá Trình Tìm Kiếm Một Sinh Viên.

Step 1: Nhập dữ liệu.

Step 2 : Kiểm tra danh sách đã sắp xếp chưa.

Step 3: Tìm kiếm theo thuật toán thích hợp.

a, Tìm kiếm Nhị Phân ( độ phức tạp: **Log2(n) )**

Bắt Đầu

int left =0

int right = total\_number\_of\_student-1

Check\_data(data)==0

Left<=right

return Liner\_Search(student,data);

mid = ( left + right )/2

Stricmp(student[mid].student\_code,data)==0

Stricmp(student[mid].student\_code,data) > 0

Location\_to\_delete = mid;

return true;

Return false

right =mid – 1;

Left =mid +1 ;

Đúng

Đúng

Sai

Đúng

Đúng

Sai

Sai

Sai

Hình 7: Quá trình tìm kiếm sinh viên theo thuật toán tìm kiếm nhị phân.

Step 1: Gán left = 0, right = total\_number\_of\_students , khởi tạo mid

Step 2: Kiểm tra dữ liệu là mã sinh viên hay là họ và tên.

Step 3: Nếu là tên thì chuyển sang tìm kiếm tuần tự, ngược lại thì bắt đầu tìm kiếm theo mã sinh viên.

Step 4:

while ( left <= right ){

mid = ( left + right ) / 2;

if ( stricmp(student[mid].student\_code,data) == 0 ){

location\_to\_delete = mid;

return true;

}

if ( stricmp(student[mid].student\_code,data) > 0 )

right = mid - 1;

else if ( stricmp(student[mid].student\_code,data) < 0 )

left = mid + 1;

}

return false;

}

b, Tìm kiếm tuần tự (độ phức tạp: **O(n)**)

Bắt đầu

int i=0;

char name[50];

char b[2] = {‘ ‘};

Check\_data(data)==false

strcpy( name, student[i].name.first\_name );

strcat( name, b);

strcat( name, student[i].name.last\_name );

i < total\_number\_of\_student

&& strcmp(name,data)!=0

i == total\_number\_of\_students

Location\_to\_delete=i;

Return true;

i++;

strcpy(name, student[i].name.first\_name);

strcat( name, b);

strcat(name, student[i].name.last\_name)

Return false

i < total\_number\_of\_student &&

strcmp(student[i].student\_code,data)!=0

i++;

Sai

Sai

Sai

Sai

Đúng

Đúng

Đúng

Đúng

Hình 8: Quá trình tìm kiếm tuần tự.

Step 1:Khởi tạo các biến

int i = 0;

char name[50];

char b[2] = {' '};

Step 2: Kiểm tra dữ liệu là mã sinh viên hay là họ và tên.

Step 3: Nếu dữ liệu là họ và tên thì tìm kiếm theo Tên

strcpy( name , student[i].name.last\_name );

strcat( name , b );

strcat( name , student[i].name. first \_name );

while( i < total\_number\_of\_students && stricmp(name,data) != 0 )

{ i++;

strcpy( name , student[i].name.last\_name );

strcat( name , b );

strcat( name , student[i].name. first \_name );

}

Step 4: Ngược lại nếu dữ liệu người dùng nhập là mã sinh viên .

while( i < total\_number\_of\_students && strcmp(student[i].student\_code,data) != 0)

i++;

Step 5: Kiểm tra xem có sinh viên trong lớp không.

if( i == total\_number\_of\_students ) return false;

location\_to\_delete = i;

return true;

3.3.2 Thuật toán sắp xếp Quick Sort (độ phức tạp **O(nlog(n))** )

- Thuật toán phân đoạn các sinh viên có tên ( Họ lót ) lớn hơn chốt sang trái, các sinh viên có tên ( Họ lót ) nhỏ hơn sang phải và trả về vị trị sau khi phân đoạn của phần tử chốt.

Bắt đầu

char pivot[10];

strcpy(pivot, student[hight].name.last\_name);

int left = low;

int right = high -1;

check=true;

Check= =true

flag= =1

swap( student[left], student[high] );

return left;

left <= right && stricmp(pivot, student[left].name.last\_name)> 0

left++

right >= left && stricmp(pivot, student[left].name.last\_name) < 0

right++;

left >= right

swap(student[left], student[right] );

left++;

right++;

check=false

Đúng

Đúng

Sai

Sai

Sai

Đúng

Đúng

Đúng

Sai

Bắt đầu

char pivot[10];

strcpy( pivot, student[high].name.first\_name ) ; int left = low;

int right = high -1;

check=true;

Check= =true

flag= =1

swap( student[left], student[high] );

return left;

left <= right && stricmp(pivot, student[left].name.firts\_name)> 0

left++

right >= left && stricmp(pivot, student[left].name. .firts\_name) < 0

right++;

left >= right

swap(student[left], student[right] );

left++;

right++;

check=false

Đúng

Sai

Sai

Sai

Sai

Đúng

Đúng

Đúng

Sai

Hình 9: Lưu đồ thuật toán phân đoạn mảng.

+ Khởi tạo các biến:

char pivot[10];

if(flag==1)

strcpy( pivot, student[high].name.last\_name ) ;

else strcpy( pivot, student[high].name.first\_name ) ;

int left = low;

int right = high - 1;

+ Tiến hành phân đoạn danh sách sinh viên:

while(true){

if(flag==1){

while( left <= right && stricmp(pivot,student[left].name.last\_name) >0);

left++;

while(right >= left && stricmp(pivot,student[right].name.last\_name) <0);

right--;

}

else {

while( left <= right && stricmp( pivot, student[left].name.first\_name )> 0);

left++;

while(right >= left && stricmp( pivot, student[right].name.first\_name ) <0);

right--;

}

if (left >= right) break;

swap( student[left] , student[right] );

left++;

right--;

}

swap( student[left] , student[high] );

}

+ Trả về vị trí phần tử chốt

return left;

* Đầu tiên sẽ tiến hành sắp xếp danh sách theo tên:

Step 1: Tiến hành phân đoạn mảng .

Step 2: Sau khi phân đoạn mảng ta sẽ có 2 mảng con: mảng bên trái của x

và mảng bên phải của x. Tiếp tục công việc với mỗi mảng con (chọn

pivot, phân đoạn) cho tới khi mảng được sắp xếp.

void quickSort\_ first\_name\_step1( SV student[] , int low , int high ){

if (low < high){

int pi = partition ( student, low, high,0);

quickSort\_ first\_name\_step1( student , low , pi-1 );

quickSort\_ first\_name\_step1( student , pi+1 , high );

}

}

* Tiếp theo sắp xếp các sinh viên có tên trùng nhau:

Step 1: Khởi tạo các biến

int bien\_trung\_gian=0;

char firstname[10];

Step 2: Tiến hành phân đoạn các sinh viên có tên trùng nhau để tiến hành so sánh. Sắp xếp theo họ lót các đoạn sinh viên trùng nhau cho đến hết mảng

strcpy(firstname,student[0].name. first\_name);

for(int i = 1 ; i < total\_number\_of\_students ; i++){

if( stricmp(firstname,student[i].name. first\_name) != 0 || i == total\_number\_of\_students-1 )

{

quickSort\_last\_name\_step2(student,bien\_trung\_gian,i-1);

bien\_trung\_gian = i;

strcpy(firstname,student[i].name. first\_name);

}

}

3.3.3 Thuật toán xóa một sinh viên

Step 1: Tìm kiếm sinh viên theo tên hoặc mã sinh viên.

Step 2: Dịch chuyển mảng lùi 1 chỉ số – phần phía sau nơi xóa.

for (int j = location\_to\_delete+1 ; j < total\_number\_of\_students ; j++ )

student[j-1] = student[j];

}

Step 3: Giảm số lượng sinh viên trong lớp đi 1

total\_number\_of\_students--;

# CHƯƠNG TRÌNH VÀ KẾT QUẢ

## Tổ chức chương trình:

Gồm menu sau:

1-Thêm sinh viên.

2-Đọc file chứa thông tin sinh viên.

3-Sắp xếp danh sách.

4-Xóa sinh viên.

5-Tìm sinh viên.

6-Cấp mã sinh viên

7-Cấp email

8-In ra danh sách.

9-Thoát.

Sau khi nghiên cứu bài toán, nhóm chúng em chia chương trình thành các hàm nhỏ để dễ tổ chức, quản lí và bảo trì chương trình.

- void mainloop(SV student[]): Hàm làm vòng lặp cho chương trình,hiển thị menu.

- void add\_students( SV student[] ): Hàm dùng để thêm sinh viên.

- void Filter\_the\_data(char \*p,char c): Hàm lọc dữ liệu từ file và dữ liệu nhận từ bàn phím.

- void Read\_in\_file(SV student[]): Hàm đọc thông tin từ file.

- void read\_data\_from\_keyboard(char \*p,int x,int y,int z,int w): Hàm đọc dữ liệu người dùng nhập từ bàn phím.

- int partition( SV student[] , int low , int high,int flag ): Hàm Phân đoạn mảng sinh viên.

- void quickSort\_last\_name\_step1( SV student[] ): Hàm sắp xếp danh sách sinh viên theo họ lót bước 1.

- void quickSort\_last\_name\_step2( SV student[] , int low , int high ): Hàm sắp xếp danh sách sinh viên theo họ lót bước 2.

- void quickSort\_ first\_name\_step1( SV student[] , int low , int high ): Hàm sắp xếp danh sách sinh viên theo tên bước 1.

- void swap( SV &a , SV &b ): Hàm hoán đổi vị trí hai sinh viên.

- void delete\_student( SV student[] ): Hàm xóa sinh viên.

-void find\_student( SV student[] ): Hàm tìm kiếm sinh viên.

- bool Linear\_Search( SV student[] ): Hàm tìm kiếm sinh viên bằng phương pháp tìm kiếm tuần tự.

- bool Binary\_Search( SV student[] ): Hàm tìm kiếm sinh viên bằng phương pháp tìm kiếm nhị phân.

- void provide\_student\_code( SV student[] ): Hàm cung cấp mã sinh viên.

- void provide\_email( SV student[] ): Hàm cung cấp email sinh viên.

- void print\_list\_step1( SV student[] ),void print\_list\_step2( SV sv ,int x ): Các hàm dùng để in danh sách ra màn hình.

- void output\_file( SV student[] ): Hàm xuất danh sách sinh viên ra file.

- bool check\_data( char data[] ): Hàm kiểm tra dữ liệu.

## Ngôn ngữ cài đặt:

Ngôn ngữ lập trình C.

## Kết quả

### Giao diện chính của chương trình

* Text

  Description automatically generated with medium confidenceMenu chính của chương trình

Hình 10: Menu chính của chương trình.

* Text, letter

  Description automatically generatedMenu lựa chọn cánh thêm sinh viên

Hình 11: Menu lựa chọn cánh thêm sinh viên.

* Màn hình nhập thông tin sinh viên trực tiếp

Graphical user interface, application

Description automatically generated

Hình 12: Màn hình nhập thông tin sinh viên trực tiếp.

* Graphical user interface, text, application

  Description automatically generatedMàn hình nhập thông tin từ file

Hình 13: Màn hình nhập thông tin từ file.

* Text

  Description automatically generated with medium confidenceMàn hình nhập thông tin để tìm kiếm hoặc xóa

Hình 14: Màn hình nhập thông tin để tìm kiếm hoặc xóa.

* Graphical user interface, text, application

  Description automatically generatedMàn hình hiện kết quả khi tìm thấy sinh viên

Hình 15: Màn hình hiện kết quả khi tìm thấy sinh viên.

* Graphical user interface, application

  Description automatically generated with medium confidenceMàn hình lựa chọn xóa sinh viên

Hình 16: màn hình lựa chọn xóa sinh viên.

* Màn Hình hiển thị Danh sách sinh viên

A screenshot of a computer

Description automatically generated with medium confidence

Hình 17: Màn hình hiển thị danh sách sinh viên.

### Kết quả thực thi của chương trình

* + - 1. A picture containing text, scoreboard, screen, screenshot

         Description automatically generatedDữ liệu đầu vào

Hình 18: Dữ liệu đầu vào là danh sách sinh viên.

* + - 1. Kết quả tìm kiếm
* Graphical user interface, text, application

  Description automatically generatedTìm kiếm sinh viên có tên là :”Pham Van Tien Truong”

Hình 19: Kết quả tìm kiếm theo tên.

* Graphical user interface, text, application

  Description automatically generatedTìm kiếm sinh viên có mã số sinh viên là 1022007 (tìm kiếm sau khi danh sách lớp đã được cấp mã sinh viên)

Hình 20: Kết quả tìm kiếm theo mã sinh viên.

* + - 1. Kết quả sau khi xóa một sinh viên
* A picture containing text, screen, scoreboard, screenshot

  Description automatically generatedXóa sinh viên có tên là: ”Pham Van Tien Truong”.

Hình 21: Kết quả xóa sinh viên theo tên.

- Xóa sinh viên có mã sinh viên là : 1022003 (Xóa sau khi danh sách lớp đã được cấp mã sinh viên). Đây là sinh viên có tên “Ho Ky Huy”A picture containing text, screen, screenshot

Description automatically generated

Hình 22: Kết quả xóa sinh viên theo mã sinh viên.

* + - 1. A picture containing text, screen, monitor, screenshot

         Description automatically generatedKết quả sau khi sắp xếp sinh viên

Hình 23: Kết quả sau khi sắp xếp.

* + - 1. A picture containing text, screen, scoreboard, displayed

         Description automatically generated Kết quả sau khi cung cấp mã sinh viên và email

Hình 24: Kết quả sau khi cấp mã sinh viên và email.

* + - 1. Graphical user interface

         Description automatically generated with low confidenceKết quả khi in ra file

Hình 25: Kết quả khi in ra file.

* + - 1. Các lỗi người dùng gặp phải

- Lớp rỗng: ” Empty class”.

- Danh sách lớp đã được sắp xếp: ”Arranged”.

- Mã sinh viên đã được cấp: ” Student Code Has Been Issued”.

- Danh sách lớp chưa sắp xếp nên không thể cấp mã sinh viên: ” Not Arranged”.

- Email đã được cấp: ” Email Has Been Issued”.

- Chưa cấp mã sinh viên nên không thể cấp Email: ” No Student Code Yet” .

### Nhận xét đánh giá:

Đáp ứng đầy đủ các yêu cầu về chức năng cần thiết.

Giao diện đơn giản dễ sử dụng nhưng vẫn cần được cải tiến thêm.

Thuật toán đơn giản nên thời gian chạy cho kết quả khá nhanh và ổn định.

# KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN

## Kết luận:

Thuật toán đáp ứng được nhu cầu của đề bài. Qua đồ án lần này, ta có thể hiểu được và sử dụng các mảng song song để lưu dữ liệu quan hệ, các mảng một chiều, các thuật toán thao tác với mảng, các hàm xử lý chuỗi, đồng thời có thể áp dụng nó vào bài toán thực tế. Hơn thế nữa, đồ án còn giúp sinh viên rèn luyện khả năng tư duy, sáng tạo, làm việc nhóm và khả năng thuyết trình.

## Hướng phát triển:

## Tiếp thu ý kiến đóng góp và phản hồi của mọi người để phát triển thêm về chức năng và giao diện.

* Nghiên cứu, phát triển thuật toán tối ưu hơn nữa.

TÀI LIỆU THAM KHẢO

**[1]**. Chương trình quản lý sinh viên sử dụng struct trong C,<https://nguyenvanhieu.vn/chuong-trinh-quan-ly-sinh-vien-su-dung-struct/>

**[2]**. Các hàm trong chế độ đồ họa (phần 1) <https://tuhoclaptrinh.cachhoc.net/2017/02/26/cac-ham-trong-che-do-do-hoa-phan-1/>

**[3]**. 5 Thuật Toán Tìm Kiếm Mọi LTV Nên Biết <https://codelearn.io/sharing/5-thuat-toan-tim-kiem-moi-ltv-nen-biet>

**[4]**. Đồ Án Game Rắn Săn Mồi, <https://github.com/nguyenvanhieuvn/Snake>

PHỤ LỤC

1. Nhập Thông Tin Sinh Viên Từ Bàn Phím

void add\_students( SV student[]){

settextstyle(1,0,3);

setfillstyle(1,15);

showText(50,50,"First Name:");

showText(50,150,"Last Name:");

showText(50,250,"Gender:");

showText(50,350,"Date\_of\_birth:");

showText(50,450,"Address:");

bar(250,50,600,75);

bar(250,150,600,175);

bar(250,250,600,275);

bar(300,350,650,375);

bar(200,450,600,475);

read\_data\_from\_keyboard(student[total\_number\_of\_students].name.first\_name,250,50,600,75); Filter\_the\_data(student[total\_number\_of\_students].name.first\_name,63); read\_data\_from\_keyboard(student[total\_number\_of\_students].name.last\_name,250,150,600,175);

Filter\_the\_data(student[total\_number\_of\_students].name.last\_name,63);

read\_data\_from\_keyboard(student[total\_number\_of\_students].gender,250,250,600,275);

Filter\_the\_data(student[total\_number\_of\_students].gender,63);

read\_data\_from\_keyboard(student[total\_number\_of\_students].Date\_of\_birth,300,350,650,375);

Filter\_the\_data(student[total\_number\_of\_students].Date\_of\_birth,63);

read\_data\_from\_keyboard(student[total\_number\_of\_students].address,200,450,600,475);

Filter\_the\_data(student[total\_number\_of\_students].address,63);

total\_number\_of\_students++;

clearmouseclick(WM\_LBUTTONDOWN);

while(!ismouseclick(WM\_LBUTTONDOWN) || mousex() < 100 || mousey() < 400 || mousex() > 400){

showText(150,550,"Back");

showText(300,550,"Add");

}

if(mousex()<=275){

showTextBackground(150,550,"Back",15);

}else {

showTextBackground(300,550,"Add",15);

cleardevice();

add\_students(student); }

delay(500);

cleardevice();}

**2. Đọc Dữ Liệu Từ Bàn Phím**

void read\_data\_from\_keyboard(char \*p,int x,int y,int z,int w){

char str[2];

str[1] = 0;

char key=0;

int m=0;

int r;

bool check=false;

bool check\_full=false;

while(key!=13){

key= getch();

if(key==8&&check\_full==false){

p[m]='?';

m--;

char b[51]={""};

strncat(b,p,m+1);

bar(x,y,z,w);

outtextxy(x,y,b);

if(p[0]=='?') check\_full=true;

}

else str[0] = key;

if(key!=13&&key!=8){

for(int i=0;i<strlen(p);i++){

if(p[i]=='?') {

p[i]=key;

check=true;

r=i;

if(i==strlen(p)-1) m=strlen(p)-1;

check\_full=false;

break;

}

}

if(check==true){

char b[51]={""};

strncat(b,p,r+1);

bar(x,y,z,w);

outtextxy(x,y,b);

check=false;

}

else {

str[0]=key;

strcat(p,str);

bar(x,y,z,w);

outtextxy(x,y,p);

m=strlen(p)-1;

}

}

}

}

**3. Hàm Lọc Dữ Liệu Đọc từ file Và bàn phím**

void Filter\_the\_data(char \*p,char c){

for(int i=0;i<strlen(p);i++){

if(p[i]==c) p[i]=' ';

}

char string[70];

int key;

for(int i=strlen(p)-1;i>=0;i--){

if(p[i]!=' ') {

key=i;

break;}

}

if(key==strlen(p)-1) return;

strncpy(string,p,key+1);

strcpy(p,string);

}

4. **Hàm Đọc Dữ Liệu Từ file**

void Read\_in\_file(SV student[]){

p=fopen("input.txt","r");

cleardevice();

if(p==NULL) {

PlaySound (TEXT ("error.wav"), NULL, SND\_FILENAME | SND\_ASYNC);

showTextBackground(250,200,"File Does Not Exist",15);

}

else {

char first\_name[50];

char last\_name[50];

char gender[50];

char d[50];

char ns[50];

int n=total\_number\_of\_students;

while( fscanf(p, "Last\_Name: %s First\_Name:%s Gender: %s Date\_of\_birth: %s Address: %s\n", student[n].name.last\_name,student[n].name. first \_name,student[n].gender,student[n].Date\_of\_birth,student[n].address) != EOF ){

Filter\_the\_data(student[n].name.first\_name,95);

Filter\_the\_data(student[n].name.last\_name,95);

Filter\_the\_data(student[n].address,95);

total\_number\_of\_students++;

n=total\_number\_of\_students;

}

showTextBackground(250,200,"Read Successfully",15);

fclose(p);

}

delay(500);

}

**5. Các Hàm dùng để in danh sách sinh viên**

void print\_list\_step1(SV student[]){

setbkcolor(0);

cleardevice();

setcolor(15);

rectangle(5,5,1470,30+25\*(total\_number\_of\_students));

rectangle(5,5,1470,30);

line(50,5,50,30+25\*(total\_number\_of\_students));

line(400,5,400,30+25\*(total\_number\_of\_students));

line(500,5,500,30+25\*(total\_number\_of\_students));

line(700,5,700,30+25\*(total\_number\_of\_students));

line(1100,5,1100,30+25\*(total\_number\_of\_students));

line(1250,5,1250,30+25\*(total\_number\_of\_students));

settextstyle(2,0,7);

outtextxy(7,7,"Stt");

outtextxy(140,7,"Ho Va Ten");

outtextxy(415,7,"Gioi Tinh");

outtextxy(550,7,"Ngay sinh");

outtextxy(840,7,"Dia Chi");

outtextxy(1110,7,"Ma Sinh Vien");

outtextxy(1280,7,"Email");

outtextxy(550,30+25\*(total\_number\_of\_students)+40,"Press enter to continue");

int x=5;

char s[5]={""};

for(int i = 0 ; i < total\_number\_of\_students ; i++ ){

print\_list\_step2( student[i],x);

sprintf (s,"%d",i);

outtextxy(7,x+25+2,s);

line(5,x+25,1470,x+25);

x+=25;

}

clearmouseclick(WM\_LBUTTONDOWN);

char key=0;

while(key!=13){

key=getch();

}

char key=0;

while(key!=13){

key=getch();

}

setbkcolor(7);

setbkcolor(0);

setbkcolor(7);

}

void print\_list\_step2( SV sv ,int x){

char name[50]={""};

char b[2] = {' '};

strcpy( name , sv.name.first\_name );

strcat( name , b );

strcat( name , sv.name.last\_name );

outtextxy(70,x+25+2,name);

outtextxy(415,x+25+2,sv.gender);

outtextxy(550,x+25+2,sv.Date\_of\_birth);

outtextxy(710,x+25+2,sv.address);

outtextxy(1110,x+25+2,sv.student\_code);

outtextxy(1260,x+25+2,sv.email);

}

6. Hàm tìm kiếm sinh viên bằng thuật toán nhị phân

bool Binary\_Search( SV student[] , char data[] ){

int left = 0;

int right = total\_number\_of\_students - 1;

int mid;

if( check\_data(data) ){

while ( left <= right ){

mid = ( left + right ) / 2;

if ( stricmp(student[mid].student\_code,data) == 0 ){

location\_to\_delete = mid;

return true;

}

if ( stricmp(student[mid].student\_code,data) > 0 )

right = mid - 1;

else if ( stricmp(student[mid].student\_code,data) < 0 )

left = mid + 1;

}

return false;

}

else {

return Linear\_Search(student,data);

}

}

7. Hàm tìm kiếm sinh viên bằng thuật toán tuần tự

bool Linear\_Search( SV student[] , char data[] ){

int i = 0;

char name[50];

char b[2] = {' '};

if( check\_data(data) == false ){

strcpy( name , student[i].name.last\_name );

strcat( name , b );

strcat( name , student[i].name. first \_name );

while( i < total\_number\_of\_students && stricmp(name,data) != 0 ) {

i++;

strcpy( name , student[i].name.last\_name );

strcat( name , b );

strcat( name , student[i].name. first\_name );

}

}

else {

while( i < total\_number\_of\_students && strcmp(student[i].student\_code,data) != 0) i++;

}

if( i == total\_number\_of\_students ) return false;

location\_to\_delete = i;

return true;

}

8. Hàm Xóa Một Sinh Viên

void delete\_student( SV student[] ){

find\_student( student );

if( check\_find ){

cleardevice();

settextstyle(1,0,5);

clearmouseclick(WM\_LBUTTONDOWN);

while(!ismouseclick(WM\_LBUTTONDOWN) || mousex() < 200 || mousey() > 600 || mousey() < 110 || mousex() > 580){

showText(300,100,"Delete");

showText(300,300,"Back");

}

if (mousey() <= 250){

showTextBackground(300,100,"Delete",15);

for(int j = location\_to\_delete+1 ; j < total\_number\_of\_students ; j+)

student[j-1] = student[j];

total\_number\_of\_students--;

}

else {

showTextBackground(300,300,"Back",15);

}

cleardevice();

}

}

9. Hàm Kiêm Tra dữ liệu

bool check\_data( char data[] ){

if( data[0] >= '0' && data[0] <= '9' ) return true;

return false;

}

10. Hàm Phân Đoạn mảng sinh viên

int partition( SV student[] , int low , int high ,int flag){

char pivot[10];

if(flag==1)

strcpy( pivot, student[high].name.last\_name ) ;

else strcpy( pivot, student[high].name. first\_name ) ;

int left = low;

int right = high - 1;

while(true){

if(flag==1){

while( left <= right && stricmp(pivot,student[left].name.last\_name)>0)

left++;

while(right >= left && stricmp(pivot,student[right].name.last\_name)<0)

right--;

}

else {

while( left<= right && stricmp( pivot, student[left].name.first\_name)>0)

left++;

while(right>=left && stricmp( pivot, student[right].name.first\_name)<0)

right--;

}

if (left >= right) break;

swap( student[left] , student[right] );

left++;

right--;

}

swap( student[left] , student[high] );

return left;

}

11. Hàm Sắp Xếp Danh Sách sinh viên Theo tên

void quickSort\_ first\_name\_step1( SV student[] , int low , int high )

{

if (low < high)

{

int pi = partition( student, low, high,0);

quickSort\_ first\_name\_step1( student , low , pi-1 );

quickSort\_ first\_name\_step1( student , pi+1 , high );

}

}

12. Hàm xác Định mảng Các Sinh viên có tên trùng nhau

void quickSort\_last\_name\_step1(SV student[]){

int bien\_trung\_gian=0;

char firstname[10];

strcpy(firstname,student[0].name. first\_name);

for(int i = 1 ; i < total\_number\_of\_students ; i++){

if( stricmp(firstname,student[i].name.first\_name) != 0 || i == total\_number\_of\_students-1 )

{

quickSort\_last\_name\_step2(student,bien\_trung\_gian,i-1);

bien\_trung\_gian = i;

strcpy(firstname,student[i].name. first\_name);

}

}

}

13. Hàm sắp xếp sinh viên Theo họ lót

void quickSort\_last\_name\_step2(SV student[],int low,int high){

if(low<high){

int pi=partition(student,low,high,1);

quickSort\_last\_name\_step2(student,low,pi-1);

quickSort\_last\_name\_step2(student,pi+1,high);

}

}

14. Hàm Cấp Mã Sinh Viên

void provide\_student\_code(SV student[])

int key=1022000;

for(int i=0;i<total\_number\_of\_students;i++){

itoa(key,student[i].student\_code,10);

key++;

}

}

15. Hàm Cấp Email

void provide\_email(SV student[]){

char b[11]="@gmail.com";

for(int i=0;i<total\_number\_of\_students;i++){

strcpy(student[i].email,student[i].student\_code);

strcat(student[i].email, b);

}

}

16. Hàm in danh sách sinh viên ra file

void output\_file(SV student[])

FILE \*fp;

char filename[13]="dssvlop .txt";

FILE \*p;

p=fopen("lop.txt","r");

int n;

char r='0';

fscanf(p,"%d",&n);

filename[7]=r+n+1;

fclose(p);

p=fopen("lop.txt","w");

fprintf(p,"%d",n+1);

fclose(p);

fp=fopen(filename,"w");

fprintf(fp,"%-4s%d\n\n","LOP:",n+1);

fprintf(fp,"%-145s\n"," \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_");

fprintf(fp,"%-145s\n","| STT | Ho Va Ten |Gioi Tinh| Ngay Sinh | Dia Chi |Ma Sinh Vien| Email |");

for(int i=0;i<total\_number\_of\_students;i++){

fprintf(fp,"%c%2s%-3d%c",'|'," ",i,'|');

char name[34];

char b[2] = {' '};

strcpy( name , student[i].name.first\_name );

strcat( name , b );

strcat( name ,student[i].name.last\_name );

fprintf(fp,"%-33s%c",name,'|');

fprintf(fp,"%-9s%c",student[i].gender,'|');

fprintf(fp,"%-12s%c",student[i].Date\_of\_birth,'|');

fprintf(fp,"%-50s%c",student[i].address,'|');

fprintf(fp,"%-12s%c",student[i].student\_code,'|');

fprintf(fp,"%-15s%c\n",student[i].email,'|');

}

fprintf(fp,"%-145s"," ----------------------------------------------------------------------------------------------------------------------------------------------");

fclose(fp);

}

17. Hàm Hiển Thị menu

void mainloop(SV student[]){

initwindow (800,600);

setbkcolor(7);

int x = 0;

char \*s = new char[5];

PlaySound (TEXT ("start.wav"), NULL, SND\_FILENAME | SND\_ASYNC);

while (x <= 100){

cleardevice();

setcolor (10);settextstyle(4,0,5);outtextxy (60,200," Loading...");

sprintf (s,"%d",x);

setcolor (4);

outtextxy (550,200,strcat(s,"%"));

if (x == 100)

delay (500);

delay (20);

x++;

}

cleardevice();

while(true){

setbkcolor(0);

setbkcolor(7);

clearmouseclick(WM\_LBUTTONDOWN);

while (!ismouseclick(WM\_LBUTTONDOWN) || mousex() < 200 || mousey() > 600 || mousey() < 20 || mousex() > 580 ){

settextstyle(1,0,4);

showText(120,10,"Student Management");

settextstyle(10,0,5);

showText(220,50,"Add");

showText(220,120,"Find");

showText(220,190,"Delete");

showText(220,260,"Print List");

showText(220,330,"Sort");

showText(220,400,"Provide Student Code");

showText(220,470,"Provide Email");

showText(220,540,"Exit");

}

clearmouseclick(WM\_LBUTTONDOWN);

if (mousey() <= 95){

PlaySound (TEXT ("beep.wav"), NULL, SND\_FILENAME | SND\_ASYNC);

showTextBackground(220,50,"Add",15);

cleardevice();

clearmouseclick(WM\_LBUTTONDOWN);

while (!ismouseclick(WM\_LBUTTONDOWN) || mousex() < 200 || mousey() > 600 || mousey() < 110 || mousex() > 580 ){

showText(220,200,"Read From File");

showText(220,270,"Enter Directly Info");

}

if (mousey() <= 270){

PlaySound (TEXT ("beep.wav"), NULL, SND\_FILENAME | SND\_ASYNC);

showTextBackground(220,200,"Read From File",15);

Read\_in\_file(student);

}else {

PlaySound (TEXT ("beep.wav"), NULL, SND\_FILENAME | SND\_ASYNC);

showTextBackground(220,270,"Enter Directly Info",15);

cleardevice();

add\_students(student);

}

check\_innit = true;

check\_sort = false;

check\_provide\_student\_code = false;

check\_provide\_email = false;

cleardevice();

clearmouseclick(WM\_LBUTTONDOWN);

} else if(mousey() <= 165){

showTextBackground(220,120,"Find",15);

PlaySound (TEXT ("beep.wav"), NULL, SND\_FILENAME | SND\_ASYNC);

setbkcolor(7);

cleardevice();

if(check\_innit==true){

find\_student( student );

}

else {

delay(300);

PlaySound (TEXT ("error.wav"), NULL, SND\_FILENAME | SND\_ASYNC);

showTextBackground(250,200,"Empty class",15);

}

delay(1000);

cleardevice();

clearmouseclick(WM\_LBUTTONDOWN);

}else if(mousey() <= 235){

showTextBackground(220,190,"Delete",15);

cleardevice();

PlaySound (TEXT ("beep.wav"), NULL, SND\_FILENAME | SND\_ASYNC);

if(check\_innit==true){

delete\_student(student);

}

else {

delay(300);

PlaySound (TEXT ("error.wav"), NULL, SND\_FILENAME | SND\_ASYNC);

showTextBackground(250,200,"Empty class",15);

}

delay(1000);

cleardevice();

clearmouseclick(WM\_LBUTTONDOWN);

}else if(mousey() <= 305){

showTextBackground(220,260,"Print List",15);

cleardevice();

if(check\_innit==true){

PlaySound (TEXT ("beep.wav"), NULL, SND\_FILENAME | SND\_ASYNC);

print\_list\_step1(student);

}else {

PlaySound (TEXT ("error.wav"), NULL, SND\_FILENAME | SND\_ASYNC);

showTextBackground(250,200,"Empty class",15);

}

cleardevice();

clearmouseclick(WM\_LBUTTONDOWN);

}else if(mousey() <= 375){

cleardevice();

showTextBackground(220,330,"Sort",15);

cleardevice();

PlaySound (TEXT ("beep.wav"), NULL, SND\_FILENAME | SND\_ASYNC);

if(check\_innit==false){

PlaySound (TEXT ("error.wav"), NULL, SND\_FILENAME | SND\_ASYNC);

showTextBackground(250,200,"Empty class",15);

}

else if(check\_sort==false){

quickSort\_last\_name\_step1(student,0,total\_number\_of\_students-1);

quickSort\_first\_name\_step1(student);

showTextBackground(250,200,"Sort Success",15);

check\_sort=true;

}

else if(check\_sort==true){

PlaySound (TEXT ("error.wav"), NULL, SND\_FILENAME | SND\_ASYNC);

showTextBackground(250,200,"Arranged",15);

}

delay(1000);

cleardevice();

clearmouseclick(WM\_LBUTTONDOWN);

}

else if(mousey() <= 445){

PlaySound (TEXT ("beep.wav"), NULL, SND\_FILENAME | SND\_ASYNC);

showTextBackground(220,400,"Provide Student Code",15);

cleardevice();

if(check\_innit==false){

PlaySound (TEXT ("error.wav"), NULL, SND\_FILENAME | SND\_ASYNC);

showTextBackground(250,200,"Empty class",15);

}

else if(check\_provide\_student\_code == true){

PlaySound (TEXT ("error.wav"), NULL, SND\_FILENAME | SND\_ASYNC);

showTextBackground(100,200,"Student Code Has Been Issued",15);

}

else if(check\_sort == false){

PlaySound (TEXT ("error.wav"), NULL, SND\_FILENAME | SND\_ASYNC);

showTextBackground(100,200,"Not Arranged",15);

}else {

showTextBackground(50,100,"Student Code has been issued successfully",15);

provide\_student\_code( student );

check\_provide\_student\_code = true;

}

delay(1000);

cleardevice();

clearmouseclick(WM\_LBUTTONDOWN);

}

else if(mousey() <= 515){

PlaySound (TEXT ("beep.wav"), NULL, SND\_FILENAME | SND\_ASYNC);

cleardevice();

showTextBackground(220,470,"Provide Email",15);

cleardevice();

if(check\_innit==false){

PlaySound (TEXT ("error.wav"), NULL, SND\_FILENAME | SND\_ASYNC);

PlaySound (TEXT ("error.wav"), NULL, SND\_FILENAME | SND\_ASYNC);

showTextBackground(250,200,"Empty class",15);

}else if(check\_provide\_email==true){

PlaySound (TEXT ("error.wav"), NULL, SND\_FILENAME | SND\_ASYNC);

showTextBackground(100,200,"Email Has Been Issued",15);

}

else if(check\_provide\_student\_code == false){

PlaySound (TEXT ("error.wav"), NULL, SND\_FILENAME | SND\_ASYNC);

showTextBackground(100,200,"No Student Code Yet",15);

}

else {

showTextBackground(100,200,"Email Issued Successfully",15);

provide\_email( student );

check\_provide\_email = true;

}

delay(1000);

cleardevice();

clearmouseclick(WM\_LBUTTONDOWN);

}

else {

PlaySound (TEXT ("beep.wav"), NULL, SND\_FILENAME | SND\_ASYNC);

showTextBackground(220,540,"Exit",15);

setbkcolor(7);

cleardevice();

clearmouseclick(WM\_LBUTTONDOWN);

output\_file( student );

exit(0);

}

}

}

/\* Do An Quan Li Hoc Sinh \*/