TRƯỜNG ĐẠI HỌC ĐIỆN LỰC

**KHOA CÔNG NGHỆ THÔNG TIN**



**BÁO CÁO CHUYÊN ĐỀ HỌC PHẦN**

**CẤU TRÚC DỮ LIỆU VÀ GIẢI THUẬT NÂNG CAO**

**ĐỀ TÀI:**

**ỨNG DỤNG KỸ THUẬT ĐỆ QUY VÀ ÁP DỤNG CÂY NHỊ**

**PHÂN TÌM KIẾM CHO BÀI TOÁN TÌM KIẾM THÍ SINH**

**THI ĐẠI HỌC**

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  | | --- | --- | | **Sinh viên thực hiện** | **: ĐẶNG VĂN HẢI**  **PHẠM QUẢNG ĐẠI** | | **Giảng viên hướng dẫn** | **: NGUYỄN THỊ THANH TÂN** | | | **Ngành** | **: CÔNG NGHỆ THÔNG TIN** | | | **Chuyên ngành** | **: CÔNG NGHỆ PHẦN MỀM** | | | **Lớp** | **: D13CNPM1** | | | **Khóa** | **: 2018-2022** | | |  |
|  |  |

***Hà Nội, tháng 6 năm 2020***

**PHIẾU CHẤM ĐIỂM**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **STT** | **Họ và tên sinh viên** | **Nội dung thực hiện** | **Điểm** | **Chữ ký** |
| 1 | Đặng Văn Hải | Viết chương trình, Viết báo cáo. |  |  |
| 2 | Phạm Quảng Đại | Góp ý sửa chương trình, viết báo cáo |  |  |

|  |  |  |
| --- | --- | --- |
| **Họ và tên giảng viên** | **Chữ ký** | **Ghi chú** |
| Giảng viên chấm 1: |  |  |
| Giảng viên chấm 2: |  |  |

MỤC LỤC

DANH MỤC HÌNH ẢNH………………………………………………………….2

[LỜI MỞ ĐẦU 3](#_Toc44102938)

[CHƯƠNG 1: TỔNG QUAN. 4](#_Toc44102939)

[1.1.Khái niệm. 4](#_Toc44102940)

[1.1.1.Đệ quy. 4](#_Toc44102941)

[1.1.2.Thuật toán đệ quy trong tin học. 4](#_Toc44102942)

[1.1.3. Cấu trúc của thuật toán đệ quy. 4](#_Toc44102943)

[1.1.4. Các loại đệ quy. 4](#_Toc44102944)

[1.1.5. Ưu và nhược điểm của thuật toán đệ quy. 4](#_Toc44102945)

[1.2. Cấu trúc dữ liệu cây. 5](#_Toc44102946)

[1.2.1. Cây tổng quát. 5](#_Toc44102947)

[1.2.2. Khái niệm. 5](#_Toc44102948)

[1.2.3. Các khái niệm liên quan. 5](#_Toc44102949)

[1.2.4. Các cách biểu diễn cây. 6](#_Toc44102950)

[CHƯƠNG 2: NỘI DUNG. 8](#_Toc44102951)

[2.1. Ý Tưởng. 8](#_Toc44102952)

[2.2. Thuật toán. 8](#_Toc44102953)

[2.2.1. Hàm tạo nút. 8](#_Toc44102954)

[2.2.2. Hàm tạo cây. 9](#_Toc44102955)

[2.2.3. Hàm hiển thi. 9](#_Toc44102956)

[2.2.4. Hàm tìm kiếm. 10](#_Toc44102957)

[2.2.5. Hàm xóa. 10](#_Toc44102958)

[2.2.6. Hàm ghi file. 11](#_Toc44102959)

[2.3. Các thao tác trong chương trình. 11](#_Toc44102960)

[CHƯƠNG 3: CÀI ĐẶT VÀ ĐÁNH GIÁ THỬ NGHIỆM. 12](#_Toc44102961)

[3.1. Chương trình. 12](#_Toc44102962)

[3.2. Kết quả cài đặt chương trình. 19](#_Toc44102963)

[3.3. Đánh giá kết quả thực hiện. 20](#_Toc44102964)

[KẾT LUẬN 21](#_Toc44102965)

[DANH MỤC THAM KHẢO 22](#_Toc44102966)

DANH MỤC HÌNH ẢNH

[Hình 1.1: Hình ảnh cây nhị phân. 5](#_Toc44103707)

[Hình 1.2: Hình ảnh cây nhị phân được đánh số. 6](#_Toc44103708)

[Hình 1.3: Hình ảnh hiển thị mảng. 6](#_Toc44103709)

[Hình 1.4: Cấu trúc nút của cây nhị phân. 7](#_Toc44103710)

[Hình 1.5: Biểu diễn cây bằng cấu trúc liên kết. 7](#_Toc44103711)

[Hình 2.1: Hàm tạo nút trong cây nhị phân. 8](#_Toc44103712)

[Hình 2.2: Hàm tạo cây trong cây nhị phân 9](#_Toc44103713)

[Hình 2.3: Hàm hiển thị cây nhị phân. 9](#_Toc44103714)

[Hình 2.4: Hàm tìm kiếm cây nhị phân 10](#_Toc44103715)

[Hình 2.5: Hàm tạo nút trong cây nhị phân. 10](#_Toc44103716)

[Hình 3.1: Chương trình chạy. 19](#_Toc44103717)

[Hình 3.2: Chương trình chạy lúc ấn số 1. 19](#_Toc44103718)

[Hình 3.3: Chương trình chạy lúc ấn số 2. 20](#_Toc44103719)

[Hình 3.4: Chương trình chạy lúc ấn số 3. 20](#_Toc44103720)

# LỜI MỞ ĐẦU

Trong thế giới hiện nay công nghệ thông tin ngày càng phát triển vượt bậc và ngày càng đạt được thành tựu to lớn trong việc phát triển kinh tế.Trên hầu hết tất cả lĩnh vực thì đều có mặt ngành công nghệ thông tin trong đó, nó đã trở thành một phần thiết yếu trong cuộc sống.

Chương trình quản lý thí sinh thi đại học là một chương trình xây dựng nhằm đáp ứng những nhu cầu quản lý như lưu trữ,tìm kiếm,…và rất nhiều công việc với một cách chính xác và nhanh chóng.

Nhóm em lựa chọn đề đề tài:” Ứng dụng kỹ thuật đệ quy áp dụng cây nhị phân tìm kiếm cho bài toán tìm kiếm thí sinh thi đại học”. Chương trình có nhiệm vụ đáp ứng nhu cầu lưu giữ liệu của thí sinh thi đại học và sẽ hiển thị ra thông tin chi tiết của thí sinh khi người dùng cần. Chúng em đã áp dụng kỹ thuật “đệ quy” cho cây nhị phân tìm kiếm vào chức năng tìm kiếm thông tin, giúp người dùng thao tối ưu, nhanh chóng hơn.

Với đề tài này giúp chúng ta củng cố kiến thức về cây nhị phân.Đề cương thực hiện dựa trên những kiến thức đã học và tìm kiếm trên internet. Do kiến thức và trình độ còn hạn hẹp nên trong quá trình thực hiện nếu có sai sót mong thầy cô thông cảm.

Trong quá trình hoàn thành, chúng em đã cố gắng, song vẫn không thể không tránh khỏi những thiếu sót, vì vậy rất mong được sự đóng góp ý kiến của thầy cô giáo. Chúng em xin chân thành cảm ơn cô giáo Nguyễn Thị Thanh Tân đã giúp chúng em hoàn thành bài báo cáo này một cách tốt nhất.

# CHƯƠNG 1: TỔNG QUAN.

## 1.1.Khái niệm.

### 1.1.1.Đệ quy.

Mặc dù lý thuyết đệ quy đã tồn tại rất lâu từ khi toán học ra đời và phát trời nhưng nó chỉ cho thấy tầm quan trọng và ứng dụng của mình khi con người phát minh ra máy vi tính và tin học. Trong toán tin một đối tượng là đệ quy nếu nó được định nghĩa qua chính nó hoặc một đối tượng khác cùng dạng bằng quy nạp.

### 1.1.2.Thuật toán đệ quy trong tin học.

Thuật toán đệ quy là một thuật ngữ tin học chỉ các bước thực hiện giải bài toán hoặc đối tượng nào đó bằng phương pháp đệ quy. Mỗi toán được giải thông qua cách xác định các trường hợp đặc biệt và tính quy nạp của nó được gọi là bài toán đệ quy. Nói cách khác để giải một bài toán đệ quy là việc chia nhỏ lời giải thành những bài toán con dễ giải hơn. Và thuật toán tương ứng với lời giải như vậy được gọi là thuật toán đệ quy.

### 1.1.3. Cấu trúc của thuật toán đệ quy.

Định nghĩa một hàm đệ quy gồm hai phần:

- Phần NEO: Xác định những trường hợp đặc biệt của bài toán,đối tượng.Là phần thực hiện những công việc đơn giản,có thể giaiar trực tiếp mà không cần đến bài toán con nào cả.Phần này cũng quyết định tính hữu hạn của thuật toán.

- Phần đệ quy: Để gọi đệ quy những bài toán con và phối hợp chúng nhằm tìm ra lời giải chính trong trường hợp chưa tìm ra lời giải từ phần NEO.

### 1.1.4. Các loại đệ quy.

Có hai loại đệ quy: Đệ quy trực tiếp và đệ quy gián tiếp

- Đệ quy trực tiếp là loại đệ quy mà đối tượng mô tả trực tiếp qua nó: A mô tả qua B, C…trong đó B, C không chứa A.2

- Đệ quy gián tiếp là đệ quy mà đối tượng được mô tả gián tiếp qua nó: A mô tả qua A1,A2,…,An. Trong đó một Ai đuộc mô tả qua A.

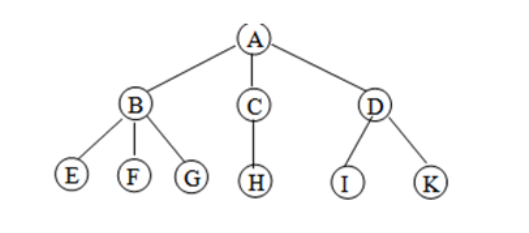
### 1.1.5. Ưu và nhược điểm của thuật toán đệ quy.

Bên cạnh nhiều giải thuật khác nhau như giải lặp,quy hoạc động,vét cạn…đệ quy vẫn là một công cụ hữu ích để xử lý các số liệu.Một ưu điểm quan trọng là không bị giới hạn vòng lặp nên sẽ mở rộng được khả năng xử lý số liệu dầu vào.Ngoài ra có nhiếu đối tượng mà việc xây dựng thuật toán đệ quy đơn giản hơn nhiều so với các thuật toán khác như lặp.

Mặc dù vây một số bài toán đệ qu hay bài toán khi được lập trình trên máy tính bằng thuật toán đệ quy thì gây tốn bộ nhớ và thời gian thực hiện quá lâu đối với những số liệu lớn.Nguyên nhân cơ bản là vì bản chất của đệ quy thực chất là một dây chuyền mà trong đó các lệnh đệ quy khi thực hiện thì trình dịch phải chuyển các mã lệnh thành các thủ tục được xếp chồng lên nhau rồi mới xử lý chúng theo thứ tự.Nếu một thuật toán đệ quy đòi hỏi máy tính thực hiện số lượng lớn các thủ tục đặc biệt như hàm mũ thì thời gan thực hiện và bộ nhớ tương đương cũng phải lớn.

## 1.2. Cấu trúc dữ liệu cây.

### 1.2.1. Cây tổng quát.



#### Hình 1.1: Hình ảnh cây nhị phân.

### 1.2.2. Khái niệm.

Cây là một cấu trúc lưu trữ trong đó các phân tử của cây (gọi là cá nốt) có cùng kiểu dữ liệu .Mỗi nốt gồm dữ liệu và các liên kết đến nốt khác.Giữa các nốt có quan hệ phân cấp gọi là “gọi là quan hệ cha con”. Trong cây có một nốt đặc biệt gọi là node gốc không là con của bất kỳ nốt nào.Cây có thể định nghĩa bằng đệ quy như sau:

-Mỗi nốt là một cây,nó cũng là nốt gốc của cây đó.

-Nếu n là một nốt và n1,n2,…nk lần lượt là nốt gốc của các cây T1,T2,…Tk.Và cho nốt cha của các nốt n1,n2,…nk,thì ta sẽ được một cây mới.Cây này có nốt gốc là n,các cây T1,T2,…Tk trở thành các cây con của nốt gốc n.

-Cây không có nốt nào gọi là cây rỗng.

### 1.2.3. Các khái niệm liên quan.

-Mức của cây**:**Người ta quy ước nốt gốc có mức 1,nếu nốt cha có mức i thì nốt con có mức i+1.

-Chiều cao của cây: Là mức cao nhất của các nốt trong cây.

-Bậc của nốt: Là số nốt con của cây đó.

-Nốt lá: Là nốt không có cây con (bậc bằng 0)

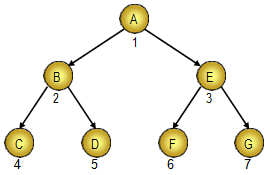
-Nốt nội(Nốt trong): Là nốt trên cây có ít nhất một con

### 1.2.4. Các cách biểu diễn cây.

Có 2 cách thông dụng để biểu diễn cấu trúc cây trên máy tính là bằng mảng hoặc bằng cấu trúc liên kết.

***\*Biểu diễn bằng mảng:***

Nếu có một cây nhị phân đầy đủ, ta có thể dễ dàng đánh số cho các nút trên cây đó theo thứ tự lần lượt từ mức 1 trở đi, hết mức này đi đến mức khác và từ trái sang phải đối với các nút ở mỗi nút.



#### Hình 1.2: Hình ảnh cây nhị phân được đánh số.

Với cách đánh số này, con của nút thứ i sẽ là các nút thứ 2i và 2i + 1. Cha của nút thứ j là nút j div 2. Từ đó có thể lưu chữ bằng một mảng T, nút thứ i của cây được lưu trữ bằng phần tử T[i].

Với cây đầy đủ như hình trên khi lưu trữ bằng mảng ta sẽ được mảng như sau:

giai-thuat-va-lap-trinh-bieu-dien-dang-mang-cay-nhi-phan

#### Hình 1.3: Hình ảnh hiển thị mảng.

Trong trường hợp cây nhị phân không đầy đủ, ta có thể thêm vào một số nút giả để được cây nhị phân đầy đủ, và gán những giá trị đặc biệt cho những phần tử trong mảng T tương ứng với những nút này. Hoặc dùng thêm một mảng phụ để đánh dấu những nút giả ta tự tạo thêm vào. Chính vì lý do này nên với cây nhị phân không đầy đủ, ta sẽ gặp phải sự lãng phí bộ nhớ vì có thể sẽ phải thêm rất nhiều nút giả vào thì mới được cây nhị phân đầy đủ.

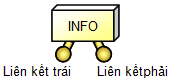
***\*Biểu diễn bằng cấu trúc liên kết:***

Khi biểu diễn cây nhị phân bằng cấu trúc liên kết, mỗi nút của cây là một bản ghi (record) gồm 3 trường:

-Trường Info: Chứa giá trị lưu tại nút đó.

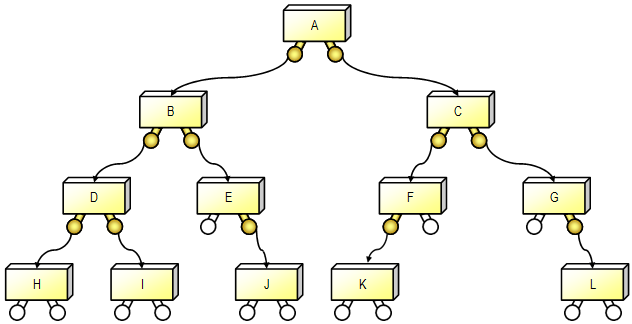
-Trường Left: Chứa liên kết (con trò) tới nút con trái, tức là chứa một thông tin đủ để biết nút con trái của nút đó là nút nào, trong trường hợp không có nút con trái, trường này được gán một giá trị đặc biệt.

-Trường Right: Chứa liên kết (Con trỏ) tới nút con phải, tức là chứa một thông tin đủ để biết nút con phải của nút đó là nút nào, trong trường hợp không có nút con phải, trường này được gán một giá trị đặc biệt.



#### Hình 1.4: Cấu trúc nút của cây nhị phân.

Đối với cây ta chỉ cần phải quan tâm giữ lại nút gốc, bởi từ nút gốc, đi theo các hướng liên kết Left, Right ta có thể duyệt mọi nút khác.



#### Hình 1.5: Biểu diễn cây bằng cấu trúc liên kết.

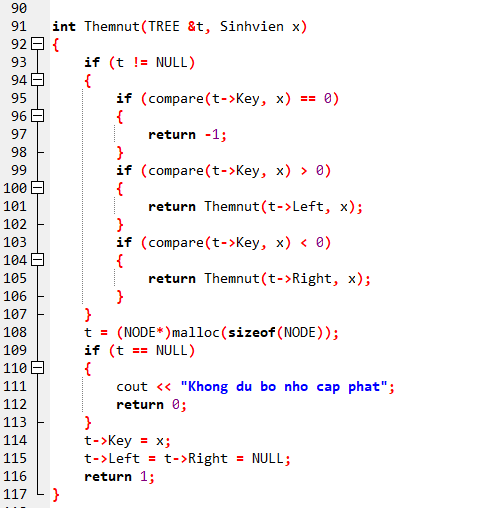
# CHƯƠNG 2: NỘI DUNG.

## 2.1. Ý Tưởng.

* Dùng ngôn ngữ lập trình c++ thiết lập một hệ thống nhập thông tin sinh viên, hiển thị toàn bộ thông tin đó ra màn hình.
* Dùng thuật toán cây nhị phân tạo bộ nhớ lưu trữ các dữ liệu đc nhập theo cấu trúc cây.
* Sau khi nhập dữ liệu thì người dùng sẽ dừng lại việc nhập, hệ thống sẽ tạm dừng, sau đó người dùng có thể nhấn để xem lại danh sách mình đã nhập.
* Người dùng có thể tìm kiếm thông tin thí sinh thi đại học qua số báo danh, sử dụng thuật toán tìm kiếm cây nhị phân để tìm ra thí sinh có số báo danh cần tìm.
* Sau khi tìm kiếm người dùng có thể xóa thí sinh bất kì nhờ vào cách tìm kiếm qua số báo danh.
* Sau khi làm các bước trên bạn bạn phải lưu lại các dữ liệu để không mất đi dữ liệu.
* Cuối cùng là chức năng thoát chương trình, thoát khỏi hệ thống.

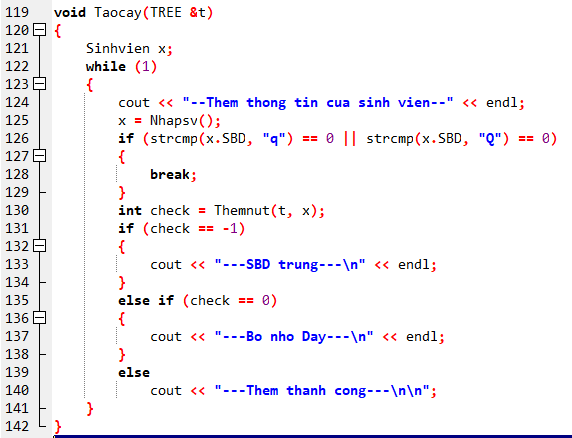
## 2.2. Thuật toán.

### 2.2.1. Hàm tạo nút.



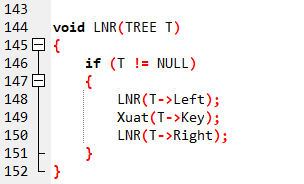
#### Hình 2.1: Hàm tạo nút trong cây nhị phân.

### 2.2.2. Hàm tạo cây.



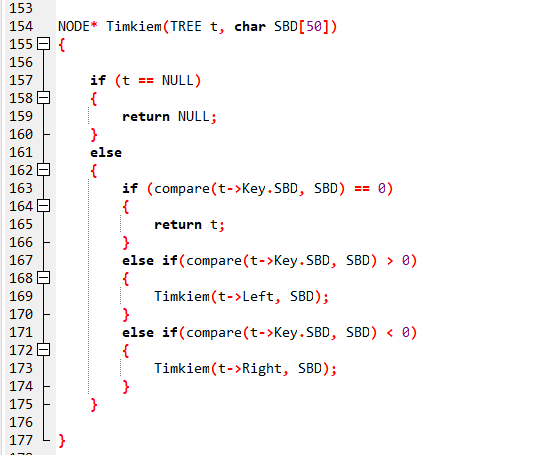
Hình 2.2: Hàm tạo cây trong cây nhị phân*.*

### 2.2.3. Hàm hiển thi.



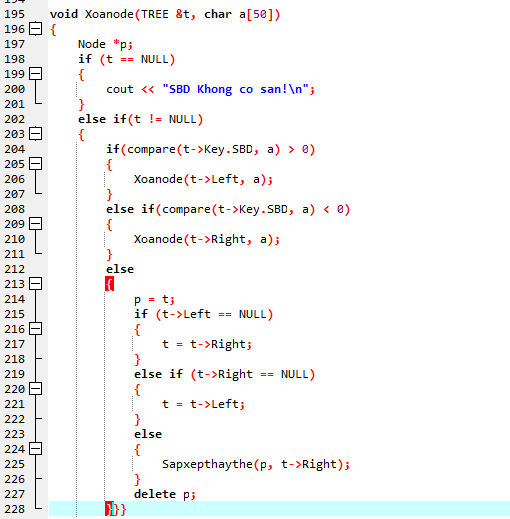
#### Hình 2.3: Hàm hiển thị cây nhị phân.

### 2.2.4. Hàm tìm kiếm.



Hình 2.4: Hàm tìm kiếm cây nhị phân*.*

### 2.2.5. Hàm xóa.



#### Hình 2.5: Hàm tạo nút trong cây nhị phân.

### 2.2.6. Hàm ghi file.

## 2.3. Các thao tác trong chương trình.

1. Khi chạy chương trình sẽ mở ra một cửa sổ console hiện thị một menu hướng dẫn nhập đến các thao tác trong chương trình.

2. Sau khi hiển thị chương trình sẽ yêu cầu bạn nhập số từ 1->5:

+ 1. Để thêm thí sinh.

+ 2. Hiển thị danh sách thí sinh đã nhập.

+ 3. Tìm kiếm thí sinh theo số báo danh.

+ 4. Xóa thí sinh theo số báo danh.

+ 5. Lưu ra file bên ngoài và thoát chương trình.

3. Sau khi nhập chọn thao tác:

+ Số 1: Thì người nhập phải thêm thông tin thí sinh bao gồm: Số báo danh, tên, quê quán, ngày tháng năm sinh, điểm toán, điểm lý, điểm hóa.

+ Số 2: Sẽ hiển thị lên tất cả các thông tin vừa nhập ở số 1.

+ Số 3: Đầu tiên sẽ hiển thị yêu cầu nhập số báo danh, người dùng phải nhập số báo danh vào để tìm kiếm, tìm kiếm xong hiển thị thông tin thí sinh theo số báo danh.

+ Số 4: Cũng giống số 3 đầu tiên là nhập số báo danh, sau đó xóa thông tin theo số báo danh.

+ Số 5: Lưu tất cả danh sách vừa nhập vào một file.txt, và kết thúc chương trình.

# CHƯƠNG 3: CÀI ĐẶT VÀ ĐÁNH GIÁ THỬ NGHIỆM.

## 3.1. Chương trình.

#include<iostream>

#include <stdlib.h>

#include <stdio.h>

#include <string.h>

#include <string>

using namespace std;

struct Date {

int Ngay, Thang, Nam;};

struct Sinhvien {

char SBD[50];

char Ten[50];

char Quequan[50];

Date Day;

double Toan;

double Ly;

double Hoa;};

struct Node

{

Sinhvien Key;

struct Node \*Left;

struct Node \*Right;};

typedef struct Node NODE;

typedef NODE\* TREE;

int compare(Sinhvien x, Sinhvien y){

return strcmp(x.SBD, y.SBD);}

int compare(char x[50], char y[50]){

return strcmp(x, y);

}

Sinhvien Nhapsv(){

Sinhvien x;

cout << "Nhap so bao danh (Q de quay lai): ";

gets(x.SBD);

if (strcmp(x.SBD, "q") == 0 || strcmp(x.SBD, "Q") == 0){

return x;}

cout << "Nhap ten: ";

gets(x.Ten);

cout << "Nhap que quan: ";

gets(x.Quequan);

cout << "Nhap ngay/ thang/ nam: \n";

cout << "Nhap ngay: ";

cin >> x.Day.Ngay;

cout << "Nhap thang: ";

cin >> x.Day.Thang;

cout << "Nhap nam: ";

cin >> x.Day.Nam;

cout << "Nhap diem toan (0->10) : ";

do {

cin>>x.Toan;

}while(x.Toan>10||x.Toan<0);

cout << "Nhap diem vat ly(0->10): ";

do{

cin>>x.Ly;

}while(x.Ly>10||x.Ly<0);

cout << "Nhap diem hoa(0->10): ";

do{ cin>>x.Hoa;

}while(x.Hoa>10||x.Hoa<0);

while (getchar() != '\n');

return x;}

void Xuat(Sinhvien x){

cout << "=================================================" << endl;

cout << "So Bao Danh: " << x.SBD << "\n";

cout << "Ten sinh vien: " << x.Ten << "\n";

cout << "Ngay-thang-nam: " << x.Day.Ngay << "/" << x.Day.Thang << "/" << x.Day.Nam << "\n";

cout << "Que Quan: " << x.Quequan << "\n";

cout << "Diem Toan: " << x.Toan << "\n";

cout << "Diem Ly: " << x.Ly << "\n";

cout << "Diem Hoa: " << x.Hoa << "\n";}

int Themnut(TREE &t, Sinhvien x){

if (t != NULL){

if (compare(t->Key, x) == 0){

return -1;}

if (compare(t->Key, x) > 0){

return Themnut(t->Left, x);}

if (compare(t->Key, x) < 0){

return Themnut(t->Right, x);}}

t = (NODE\*)malloc(sizeof(NODE));

if (t == NULL){

cout << "Khong du bo nho cap phat";

return 0;}

t->Key = x;t->Left = t->Right = NULL;return 1;}

void Taocay(TREE &t){

Sinhvien x;

while (1){

cout << "--Them thong tin cua sinh vien--" << endl;

x = Nhapsv();

if (strcmp(x.SBD, "q") == 0 || strcmp(x.SBD, "Q") == 0){

break;}

int check = Themnut(t, x);

if (check == -1){

cout << "---SBD Dung---\n" << endl;}

else if (check == 0){

cout << "---Bo nho Day---\n" << endl;}

else

cout << "---Them thanh cong---\n\n";}}

void LNR(TREE T){

if (T != NULL){

LNR(T->Left);

Xuat(T->Key);

LNR(T->Right);}}

NODE\* Timkiem(TREE t, char SBD[50]){

Sinhvien x;

if (t == NULL){

return NULL;}

else{

if (compare(t->Key.SBD, SBD) == 0){

return t;}

else if(compare(t->Key.SBD, SBD) > 0){

Timkiem(t->Left, SBD);}

else if(compare(t->Key.SBD, SBD) < 0){

Timkiem(t->Right, SBD);}}}

void Sapxepthaythe(TREE &X, TREE &Y) {

if (Y->Left != NULL){

Sapxepthaythe(X, Y->Left);}

else{

X->Key = Y->Key;

X = Y;

Y = Y->Right; }}

void Xoanode(TREE &t, char a[50]) {

Node \*p;

if (t == NULL){

cout << "SBD Khong co san!\n";}

else if(t != NULL){

if(compare(t->Key.SBD, a) > 0){

Xoanode(t->Left, a);}

else if(compare(t->Key.SBD, a) < 0){

Xoanode(t->Right, a); }

else {p = t;

if (t->Left == NULL){

t = t->Right; }

else if (t->Right == NULL){

t = t->Left;}

else{

Sapxepthaythe(p, t->Right);}delete p; }}}

void ghifile(char \*filename, TREE &T){

if (T != NULL){

FILE \*f = fopen(filename, "ab");

fprintf(f, " So bao danh: %s", T->Key.SBD);

fprintf(f, " Ten : %s \n", T->Key.Ten);

fprintf(f, " Ngay: %d \n", T->Key.Day.Ngay);

fprintf(f, " Thang: %d \n", T->Key.Day.Thang);

fprintf(f, " Nam: %d \n", T->Key.Day.Nam);

fprintf(f, " Que quan: %s \n", T->Key.Quequan);

fprintf(f, " TOan: %f \n", T->Key.Toan);

fprintf(f, " Ly: %f \n", T->Key.Ly);

fprintf(f, " Hoa: %f \n", T->Key.Hoa);

fprintf(f,"\n ", '\n');

ghifile(filename, T->Left);

ghifile(filename, T->Right);

fclose(f);}}

void Menu(TREE &t){

int so;

char \*f = "QLSV.TXT";

while (true)

{

system("cls");

cout << "===================================================";

cout << "\n\* -- 1. Them thong tin sinh vien -- \*";

cout << "\n\* -- 2. Hien thi thong tin sinh vien -- \*";

cout << "\n\* -- 3. Tim kiem sinh vien -- \*";

cout << "\n\* -- 4. Xoa sinh vien -- \*";

cout << "\n\* -- 5. Thoat -- \*";

cout << "\n=================================================";

cout << "\n\nChon mot so ma ban muon hien thi?(1->5): ";

cin >> so;

if (so == 1){

Sinhvien x;

gets(x.SBD);

Taocay(t);}

else if (so == 2){

cout << "\n\t HIEN THI CAY NHI PHAN TIM KIEM \n";

LNR(t);

system("pause");}

else if (so == 3){

char SBD[50];

cout << "\n TIM KIEM THEO SO BAO DANH ";

fflush(stdin);

gets(SBD);

NODE \*p = Timkiem(t, SBD);

if (p == NULL){

cout << "\nSBD " << SBD << " KHONG TIM THAY!\n";}

else{

Xuat(p->Key);}

system("pause");}

else if (so == 4){

char a[50];

cout << "\nNhap so bao danh can xoa : ";

fflush(stdin);

gets(a);

Xoanode(t, a);

cout<<"Xoa thanh cong!"<<endl;

system("pause");}

else if (so == 5){

break;}}}

int main(){

TREE t;

` t = NULL;

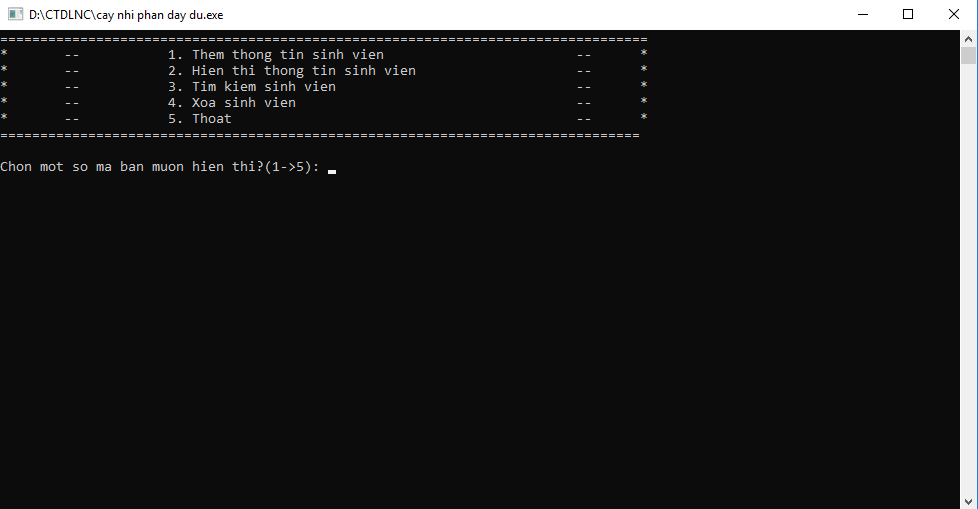
Menu(t);

system("pause");

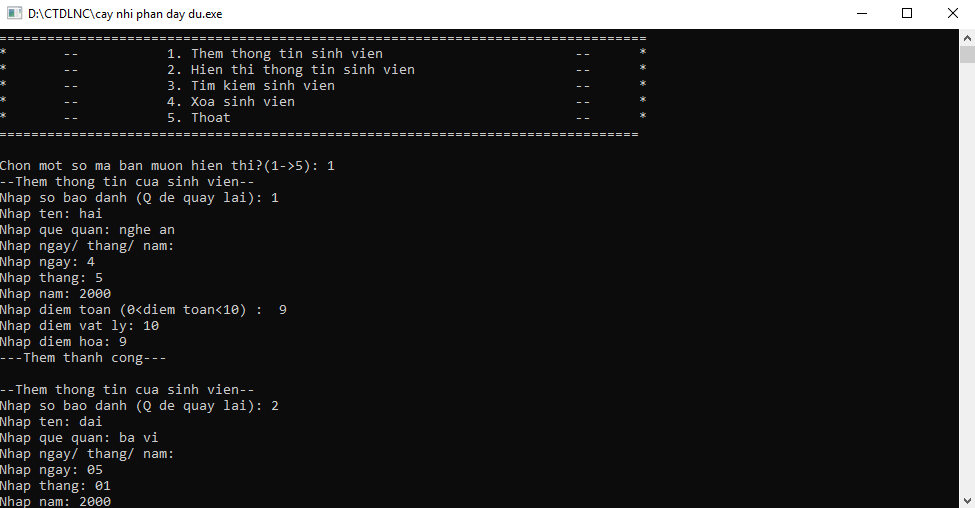
return 0;

}

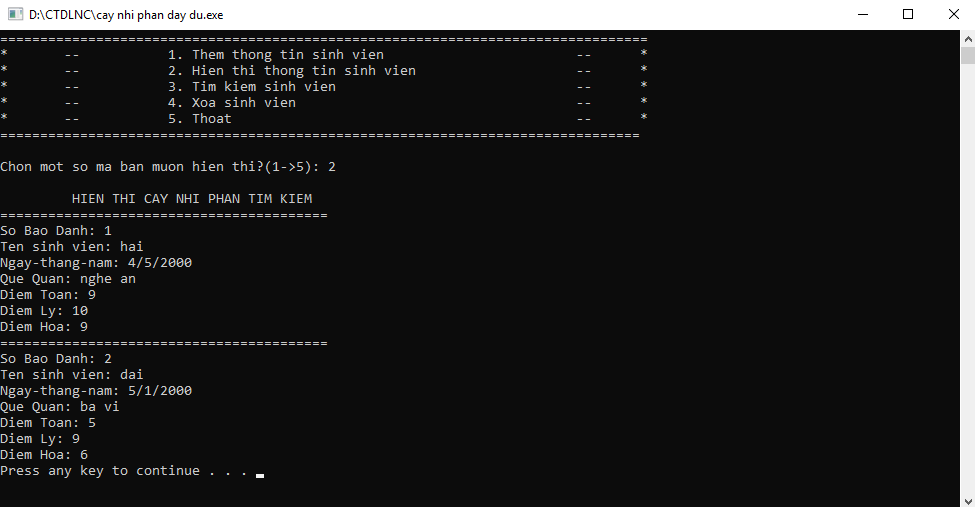
## 3.2. Kết quả cài đặt chương trình.



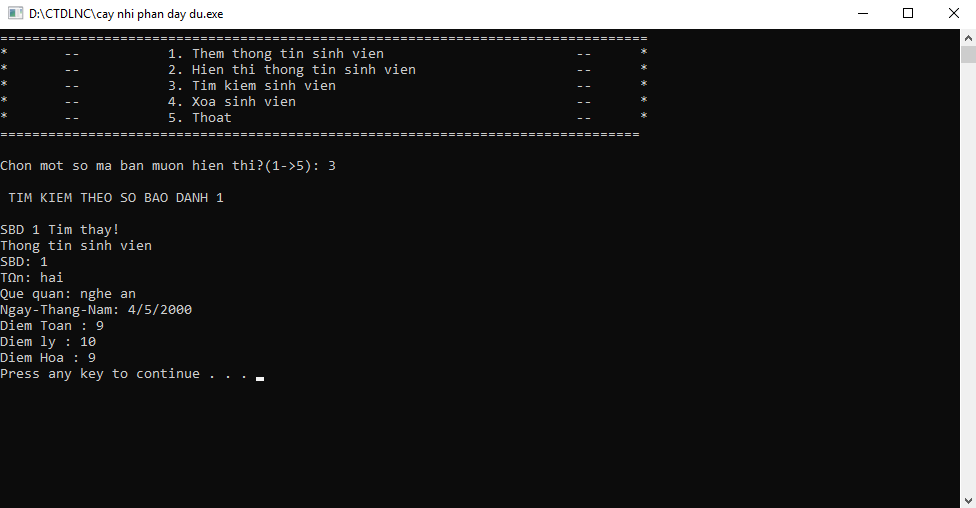
#### Hình 3.1: Chương trình chạy.



#### Hình 3.2: Chương trình chạy lúc ấn số 1.



#### Hình 3.3: Chương trình chạy lúc ấn số 2.



#### Hình 3.4: Chương trình chạy lúc ấn số 3.

## 3.3. Đánh giá kết quả thực hiện.

Sau một thời gian tìm hiểu , nghiên cứu và thực hiện đề tài . Các yêu cầu chính của đề tài cơ bản đã hoàn tất với nội dung sau:

* Ưu điểm:

- Chương trình xử lý nhanh và tương đối chính xác.

* + Khuyết điểm:
    - Chưa tối ưu hóa các chức năng, chưa có giao diện cụ thể.

# KẾT LUẬN

Qua việc thực hiện nghiên cứu đề tài “Áp dụng đệ quy và tìm kiếm nhị phân cho bài toán tìm thí sinh đại học”. Nhóm chúng em đã được biết thêm rất nhiều về môn cấu trúc dữ liệu giải thuật. Chúng em đã cố gắng hoàn thành tốt nhất bài của mình.

Sau một thời gian tìm hiểu, nghiên cứu và thực hiện đề tài chương trình sử dụng tìm kiếm nhị phân xử lý tìm kiếm rất nhanh và chính xác.

Trong quá trình thực hiện đề tài có rất nhiều ý tưởng hay, độc đáo. Nhưng do kiến thức của chúng em hạn hẹp và nhóm chúng em chưa có nhiều tư duy về thuật toán nên chưa thể thực hiện hết các ý tưởng đó. Tuy nhiên chúng em đã cố gắng hoàn thiện bài của mình tốt nhất có thể, nhóm chúng em đã cố gắng xây dựng 1 cách hoàn chỉnh nhất, đẹp nhất. Trong quá trình xây dựng chúng em khó có thể tránh khỏi những sai sót.

  Chúng em xin cảm ơn cô giáo Nguyễn Thị Thanh Tân giảng dạy chúng em trong môn cấu trúc và dữ liệu giải thuật nâng cao. Giúp đỡ chúng em trong quá trình nghiên cứu đề tài. Chia sẻ những tài liệu hay cũng như các kĩ năng lập trình.

Một lần nữa chúng em xin chân thành cảm ơn !

# DANH MỤC THAM KHẢO

[1] Tài liệu giảng dạy của cô giáo Nguyễn Thị Thanh Tân.

[2] Trang web: <https://nguyenvanhieu.vn/cay-tim-kiem-nhi-phan-binary-search-tree/>

[3] Trang web: <http://luanvan.net.vn/luan-van/de-tai-cay-nhi-phan-tim-kiem-76156/>

[4] Trang web: <https://tailieu.vn/tag/tai-lieu-cay-nhi-phan.html>