**BỘ GIÁO DỤC VÀ ĐÀO TẠO**

**TRƯỜNG ĐẠI HỌC NHA TRANG**

**KHOA CÔNG NGHÊ THÔNG TIN**

**BÁO CÁO THƯC TẬP CƠ SỞ**

Giáo viên hướng dẫn: Nhóm giảng viên số 3

Sinh viên thực hiện: Nguyễn Châu Thắng

Lớp: 58-CNTT2

MSSV: 58131410

Nơi thực tập: Tại trường

**Năm học: 2018 - 2019**

**BỘ GIÁO DỤC VÀ ĐÀO TẠO**

**TRƯỜNG ĐẠI HỌC NHA TRANG**

**KHOA CÔNG NGHÊ THÔNG TIN**

**BÁO CÁO THƯC TẬP CƠ SỞ**

Giáo viên hướng dẫn: Nhóm giảng viên số 3

Sinh viên thực hiện: Nguyễn Châu Thắng

Lớp: 58-CNTT2

MSSV: 58131410

Nơi thực tập: Tại trường

**Năm học: 2018 - 2019**

**NHẬN XÉT KẾT QUẢ THỰC TẬP**

Họ và tên sinh viên: Nguyễn Châu Thắng

Mã số sinh viên: 58131410 Lớp: 58-CNTT2 Hệ: Đại Học

Khoa Công nghệ Thông tin, Trường Đại học Nha Trang.

Nội dung thực tập:

Đánh giá kết quả thực tập:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| STT | Khả năng/kỹ năng | Tốt | Khá | TB | Yếu | Kém |
| 1 | Kiến thức 1 (lý thuyết, công nghệ) |  |  |  |  |  |
| 2 | Kiến thức 2 (thực hành) |  |  |  |  |  |
| 3 | Làm việc nhóm |  |  |  |  |  |
| 4 | Tổ chức công việc và làm việc độc lập |  |  |  |  |  |
| 5 | Giao tiếp và thái độ làm việc |  |  |  |  |  |

Nhận xét chung:

*…………………… Ngày ….. tháng ….. năm …..*

Giáo viên hướng dẫn

MỤC LỤC

[BÀI 1: DANH BẠ ĐIỆN THOẠI 5](#_Toc535743539)

[1.1. nghiên cứu ý tưởng và thiết kế danh bạ điện thoại: 5](#_Toc535743540)

[1.2. Cấu trúc dữ liệu: 5](#_Toc535743541)

[1.3. Phần chương trình: 5](#_Toc535743542)

[1.4. Giao diện chương trình: 7](#_Toc535743543)

[BÀI 2: Ngày Tháng 12](#_Toc535743544)

[2.1. nghiên cứu ý tưởng và thiết kế bài toán ngày tháng: 12](#_Toc535743545)

[2.2. Cấu trúc dữ liệu: 12](#_Toc535743546)

[2.3. Phần chương trình: 12](#_Toc535743547)

[2.4. Giao diện chương trình: 15](#_Toc535743548)

[BÀI 3: TRÒ CHƠI THI ĐỐ TRẮC NGHIỆM 17](#_Toc535743549)

[3.1. nghiên cứu ý tưởng và thiết kế bài trò chơi thi đố trắc nghiệm: 17](#_Toc535743550)

[3.2. Cấu trúc dữ liệu: 17](#_Toc535743551)

[3.3. Phần chương trình: 18](#_Toc535743552)

[3.4. Giao diện chương trình: 18](#_Toc535743553)

[BÀI 4: TÌM ĐƯỜNG ĐI NGẮN NHẤT 23](#_Toc535743554)

[4.1. nghiên cứu ý tưởng và thiết kế bài toán tìm đường đi ngắn nhất: 23](#_Toc535743555)

[4.2. Cấu trúc dữ liệu: 23](#_Toc535743556)

[4.3. Phần chương trình: 23](#_Toc535743557)

[4.4. Giao diện chương trình: 25](#_Toc535743558)

[BÀI 5: NGẮT 29](#_Toc535743559)

[5.1 Kiến thức tìm hiểu: 29](#_Toc535743560)

[Phụ lục: 30](#_Toc535743561)

# BÀI 1: DANH BẠ ĐIỆN THOẠI

## nghiên cứu ý tưởng và thiết kế danh bạ điện thoại:

* Đọc danh bạ từ file.
* Ghi danh bạ vào file
* Hiển thị danh bạ ra màn hình
* Thêm danh bạ mới
* Sửa danh bạ theo số điện thoai
* Xóa danh bạ theo số điện thoại
* Tìm kiếm danh bạ theo tên

## Cấu trúc dữ liệu:

struct ThongTin{

string Ten;

string GioiTinh;

string SoDienThoai;

string Email;

string DiaChi;

};

## Phần chương trình:

* Thực hiện các yêu cầu ở trên ta xây dựng các hàm tương ứng:

void Them(); // thêm số liên lạc vào danh bạ / ghi danh bạ vào file

void LietKe(); // liệt kê danh bạ / đọc danh bạ từ file và in ra màng hình

void ChinhSua(); // chỉnh sửa danh bạ theo tên

void Xoa(); // xóa danh bạ theo tên

void TimKiem(); // tìm kiếm danh bạ theo tên

* cuối cùng là hàm int main():

int main(){

Them(); // thêm danh bạ từ 1 file

LietKe(); // liệt kê danh bạ từ file

TimKiem(); // tìm kiếm danh ba theo tên và đưa ra màng hình

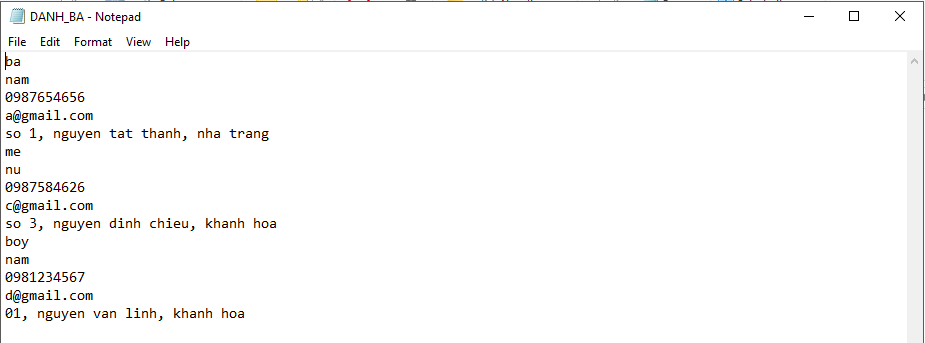
ChinhSua(); // chỉnh sửa thông tin danh bạ theo tên

Xoa(); // xóa danh bạ theo tên

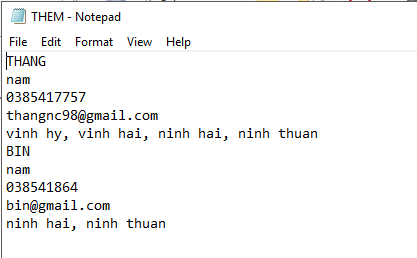
}

## Giao diện chương trình:

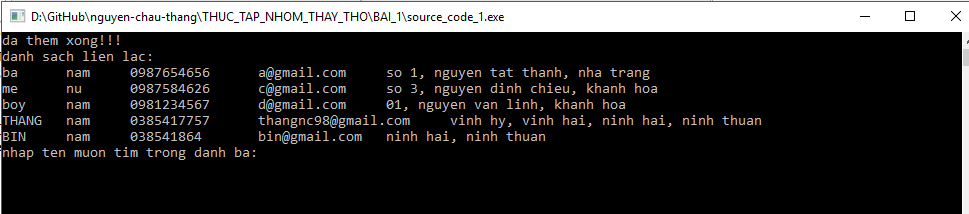
* File danh bạ ban đầu:



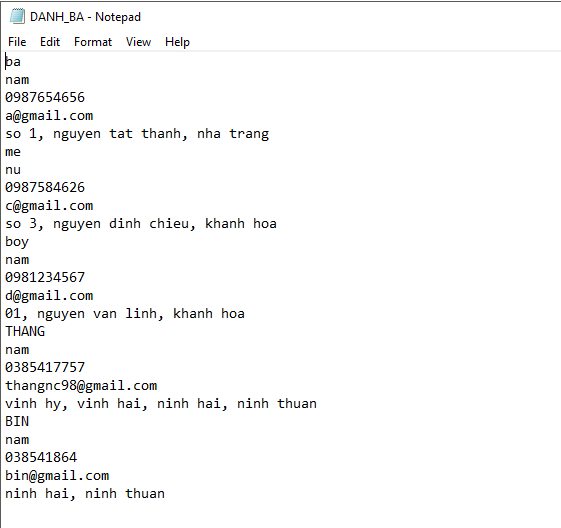
* File danh bạ cần thêm vào danh bạ:



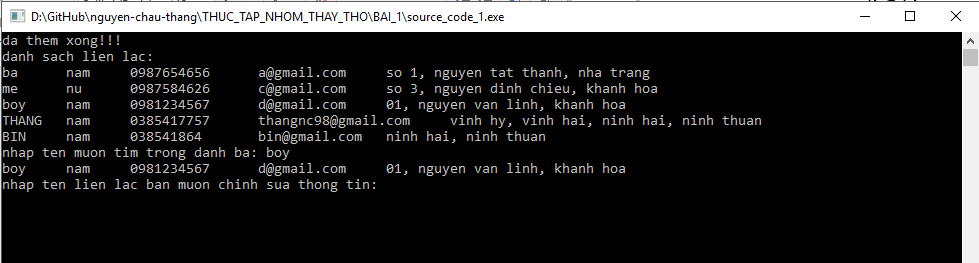
* Bắt đầu chạy chương trình:



Sau khi chạy chương trình các danh bạ cần thêm sẽ được thêm vào trong file danh bạ ban đầu:

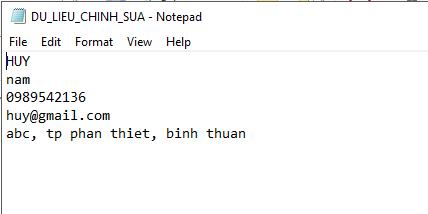


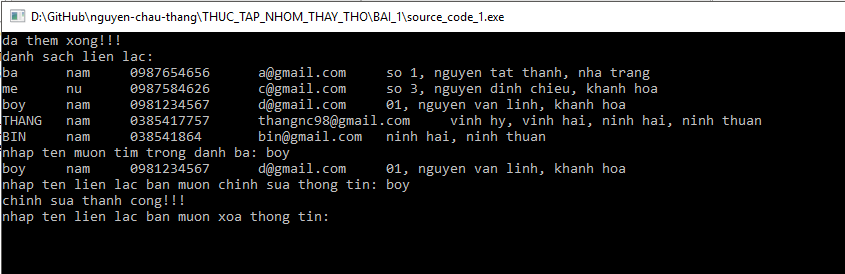
* Tiếp theo chương trình sẽ tìm theo tên và hiển thị lên màng hình:



sau do chương trình tiếp tục tìm tên thông tin cần chỉnh sửa và thay thế liên lạc đó và đưa ra 1 file mới:

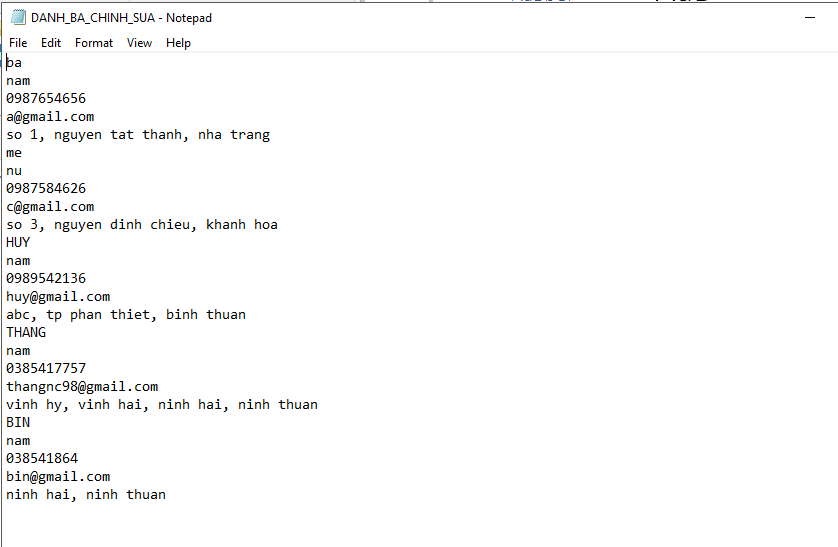
* đây là file dữ liệu sẽ chỉnh sữa:



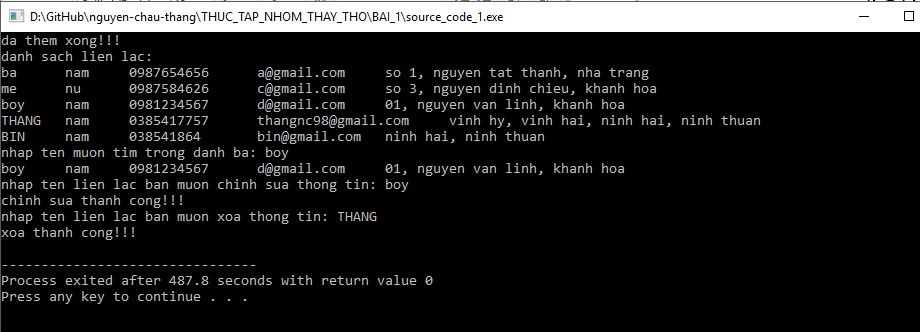


Sau khi chỉnh sửa xong chương trình sẽ thông báo

Đây là file mới sau khi đã chỉnh sửa:

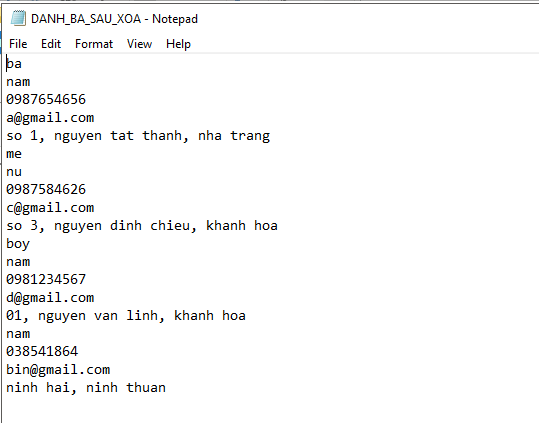


* Tiếp theo chương trình sẽ xóa thông tin theo tên:



Sau khi xóa thì nó sẽ hiển thị thông báo và kết thúc chương trình.

* Và đây là dữ liệu sau khi xóa:



# BÀI 2: Ngày Tháng

## nghiên cứu ý tưởng và thiết kế bài toán ngày tháng:

* Khai báo cấu trúc dữ liệu, NgayThang, để biểu diễn ngày tháng
* Tạo một biến dữ liệu NgayThang từ một chuỗi ngày tháng theo định dạng dd/MM/yyyy
* Viết hàm xác định một năm có phải là năm nhuận hay không?
* Viết hàm tính số thứ tự của một ngày trong một năm
* Viết hàm cộng một biến dữ liệu NgayThang và một số nguyên dương.
* Viết hàm trừ một biến dữ liệu NgayThang và một số nguyên dương.

## Cấu trúc dữ liệu:

struct NgayThang{

int D; // ngày

int M; // tháng

int Y; // năm

};

## Phần chương trình:

* Thực hiện các yêu cầu ở trên ta xây dựng các hàm tương ứng:

void Nhap(NgayThang &a); /\* nhập ngày tháng năm theo định dạng dd/MM/yyyy và

chuyển đổi định dạng \*/

bool NamNhuan(int a); // hàm xác định năm nhuận

int SoThuTu(NgayThang &a); // cho ra số thứ tự của ngày trong năm

void Cong(NgayThang a, int &x); // hàm cộng ngày cho số nguyên dương

void Tru(NgayThang a, int &x); // hàm trừ ngày cho số nguyên dương

* các hàm phụ để hổ trợ cho các hàm chính ở trên:

bool KT(int &d, int &m, int &y); /\* kiểm tra xem đã nhập đúng định dạng, đúng trình

tự, ngày tháng năm có hợp tự nhiên chưa \*/

void ChuyenDoi(string &s, NgayThang &a); /\* chuyển đổi định dạng dd/MM/yyyy

sang biến cấu trúc NgayThang ở trên \*/

int TinhNgay(int m, int y); // xem trong tháng xác định trong năm có mấy ngày

void SoSangNgay(NgayThang &b, int &n); /\* chuyển đổi từ số thứ tự sang ngày tháng

năm \*/

* hàm int main():

int main(){

NgayThang DMY;

Nhap(DMY); // nhập ngày tháng năm + chuyển đổi định dạng

if(NamNhuan(DMY.Y)) // xác định năm nhuận

cout << "nam " << DMY.Y << " la nam nhuan!" << endl;

else

cout << "nam " << DMY.Y << " khong la nam nhuan!" << endl;

// xuất ra ngày tháng năm vừa

cout << "ngay: " << DMY.D << endl;

cout << "thang: " << DMY.M << endl;

cout << "nam: " << DMY.Y << endl;

// cho ra số thứ tự ngày trong năm

cout << "so thu tu cua ngay " << DMY.D << "/" << DMY.M << "/" << DMY.Y << " la: " << SoThuTu(DMY) << endl;

int y;

Cong(DMY, y); // công ngày với số nguyên dương

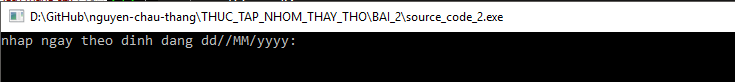
int z;

Tru(DMY, z); // trừ ngày với số nguyên dương

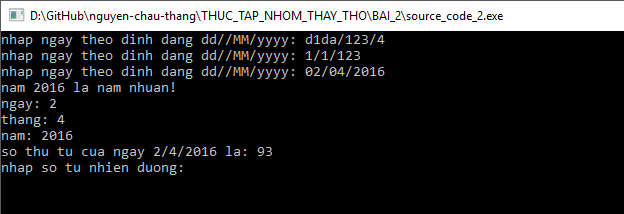
}

## Giao diện chương trình:

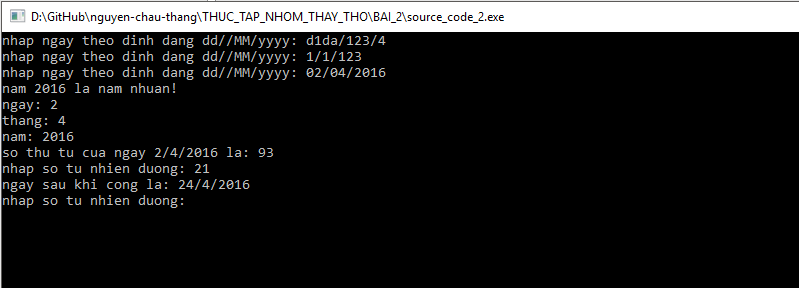
* Bắt đầu chạy chương trình:



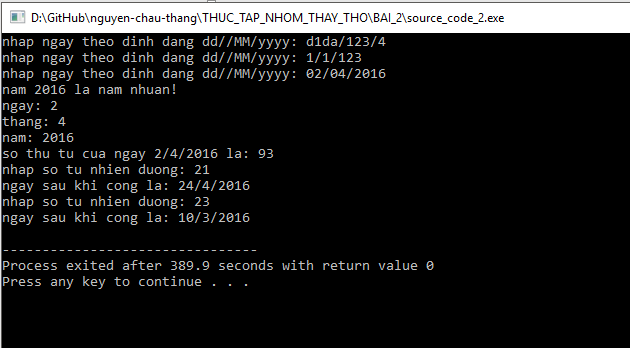
* Cần nhập đúng định dang để tiếp tục chương trình, nếu nhập sai bắt nhập lại đến khi đúng thì tiếp tục:



* Nhập xong nó sẽ chuyển đổi và đưa ra ngày tháng năm chính xác và số thứ tự của ngày trong năm, sau đó nó yêu cầu nhập số dương để bắt đầu cộng ngày tháng với số nguyên dương:



* Sau khi công nó cho ra kết quả và tiếp tục ta nhập 1 số nguyên dương nữa để trừ:



Sau khi trừ xong nó cho ra kết quả và kết thúc chương trình.

# BÀI 3: TRÒ CHƠI THI ĐỐ TRẮC NGHIỆM

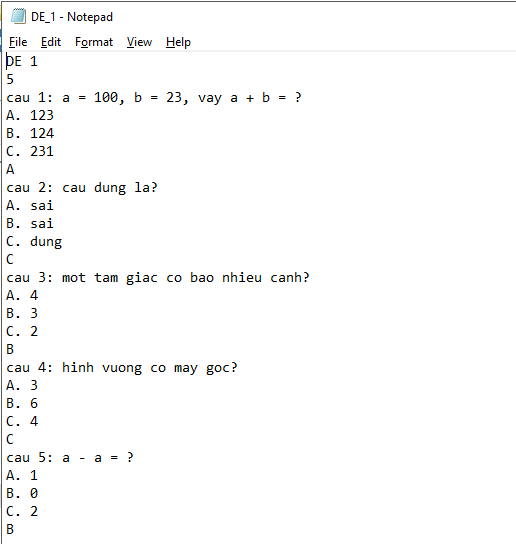
## nghiên cứu ý tưởng và thiết kế bài trò chơi thi đố trắc nghiệm:

Trò chơi gồm các bộ đề thi trắc nghiệm và đáp án được lưu trữ thành các file txt – mỗi bộ đề và đáp án được lưu trong một file. Người chơi có thể chọn bộ đề để trả lời các câu hỏi trắc nghiệm. Chương trình có các chức năng sau:

* Chọn bộ đề trắc nghiệm
* Trả lời các câu hỏi trắc nghiệm theo bộ đề đã chọn
* Lưu điểm của các người chơi
* Xem điểm của những người chơi.

## Cấu trúc dữ liệu:

Dữ liệu mỗi bộ đề thi được lưu trữ thành các file như sau: (ví dụ đề 1)



* Dòng 1 ghi tên đề
* Dòng 2 ghi số câu hỏi
* Lần lượt 5 dòng tiếp theo:
* 4 dòng đầu ghi câu hỏi và các đáp án lựu chọn
* Dòng 5 ghi đáp án của câu hỏi

## Phần chương trình:

* Thực hiện các yêu cầu ở trên ta xây dựng các hàm tương ứng:

void ChonDe(); // chọn 1 trong số các bộ đề để bắt đầu chơi

void MoDe(); /\* đọc file đề thi và cho ra câu hỏi và các đáp án cho người chơi chọn

và cho ra điểm đạt được \*/

void LuuDiem(int diem); // điểm được đưa vào và được lưu vào file sau khi nhập tên

void XemDiem(); // dùng đọc nội dung từ file lưu điểm và hiển thị ra màng hình

* các hàm phụ để hổ trợ cho các hàm chính ở trên:

bool KiemTraDA(string a, string b); /\* hàm này dùng kiểm tra đáp án người chơi lựu

chọn và đáp án đúng của câu hỏi \*/

* hàm int main():

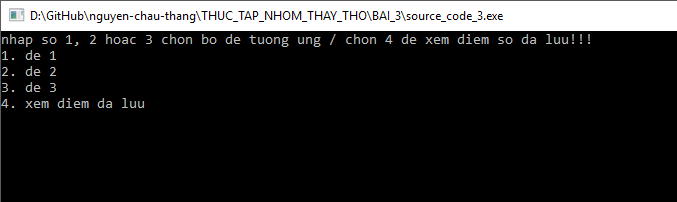
int main(){

ChonDe(); // toàn bộ chỉ cần gọi hàm chọn đề để thực hiên chương trình

}

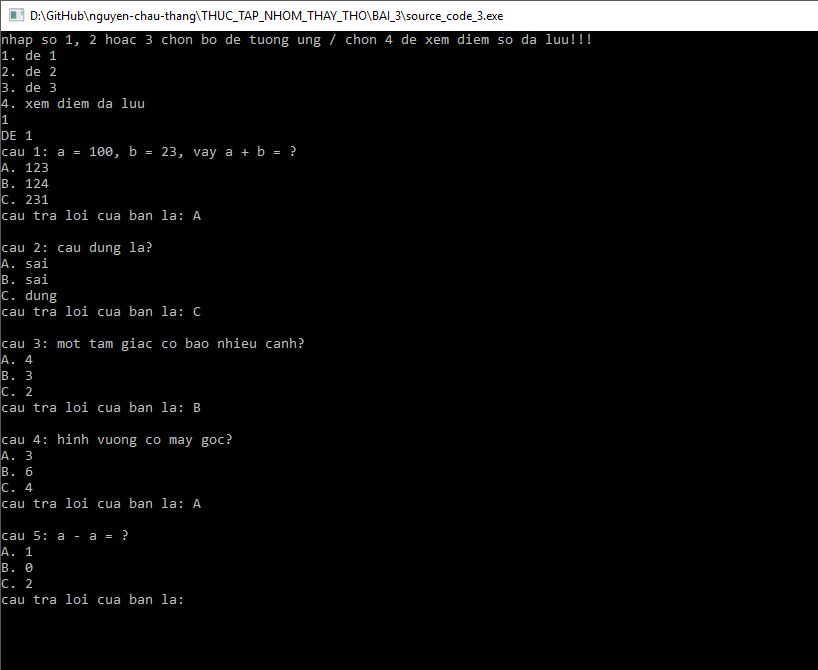
## Giao diện chương trình:

* Bắt đầu chạy chương trình:

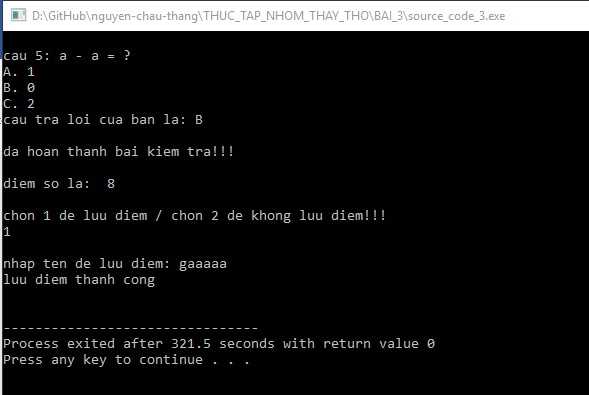


Người chơi chọn từ 1 đến 3 để chọn bộ đề thi và chọn 4 để xem điểm

* Khi chọn đề thì đề sẽ hiển thị từng câu và người chơi gõ A đến C để trả lời

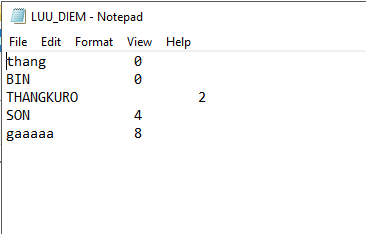


* Sau khi trả lời xong các câu hỏi người chơi sẽ biết được điểm số và có yêu cầu lưu điểm(người chơi có quyền lưu điểm hoặc không):

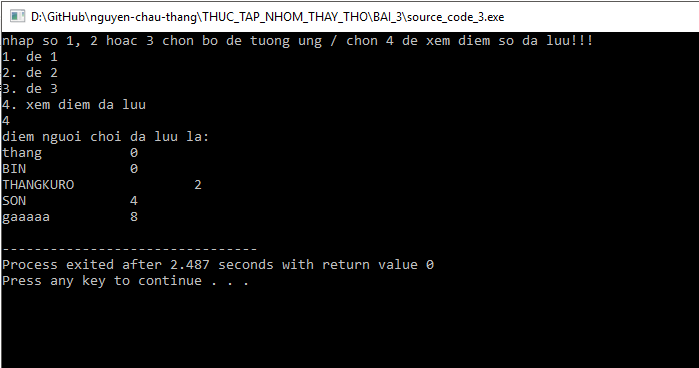


Nếu chọn lưu điểm cần nhập tên để lưu lại điểm số.

* File lưu điểm như sau:



* Ban đầu nếu người chơi chọn xem điểm thì điểm người chơi trước đó sẽ hiện lên:



# BÀI 4: TÌM ĐƯỜNG ĐI NGẮN NHẤT

## nghiên cứu ý tưởng và thiết kế bài toán tìm đường đi ngắn nhất:

Tổ chức dữ liệu và giải bài toán tìm đường đi ngắn nhất của một đồ thị vô hướng

* Tạo mới đồ thị và lưu vao file
* Mở đồ thị từ file dữ liệu đã có
* Tìm đường đi ngắn nhất, kết quả tìm được lưu vào một file
* Xuất lần lượt quá trình tìm ra đường đi ngắn nhất

## Cấu trúc dữ liệu:

struct BienLuu{

int ten;// --- ghi lại vị trí đã đi qua trước đó---

int tong;// --- tổng độ dài quảng đường đi được ---

int co;// --- 0: chưa xét / 1: đã xét ---

};

## Phần chương trình:

* Thực hiện các yêu cầu ở trên ta xây dựng các hàm tương ứng:

void NhapMaTran(int &n, int a[max][max]); // dùng tạo mớ đồ thị và lưu vào file

void MoDoThi(int &n, int a[max][max]); // dùng đọc đồ thị ra mảng 2 chiều

void XuatMaTran(int n, int a[max][max]); // dùng xuất ra màng hình cho dễ đối chiếu

void TimDuong(int n, int a[max][max]); // hàm chình để tìm đường đi ngắn nhất

* các hàm phụ để hổ trợ cho hàm chính tìm đường đi nhắn nhất ở trên:

void SoSanh(BienLuu &bc, BienLuu &bp); // dùng hổ trợ so sánh và gán giá trị biến chính

void Duyet(int n, BienLuu bc[max], int &vt); // đánh dấu vị trí có tổng nhỏ nhất

void LoTrinh(int n, BienLuu a[max], int x, int y); //cho ra lộ trình đường đi ngắn nhất

void LuuKQ(int n, BienLuu bc[max], int x, int y); // lưu các kết quả có được vào file

* đoạn chương trình sau trong hàm TimDuong(); sẽ cho ra lần lượt các bước tìm đường đi ngắn nhất:

for(int c = 0; c < n; c++)

cout << BC[c].ten << "," << BC[c].tong << "\t";

* cuối cùng là hàm int main():

int main(){

int n;

int a[max][max];

// --- tại đây do đồ thị đã được tao trực tiếp nên sẽ được vô hiệu hóa ---

// NhapMaTran(n, a);// --- tạo mới 1 đồ thị ---

MoDoThi(n, a);// --- đọc đồ thị từ file ---

XuatMaTran(n ,a);// --- xuất ma trận ra màng hình ---

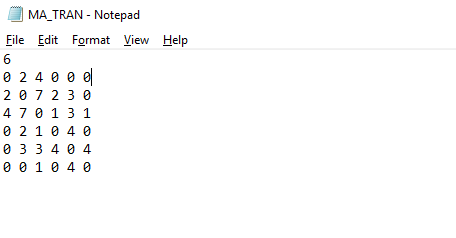
cout << endl;

TimDuong(n, a);// --- xữ lý tìm đường đi ---

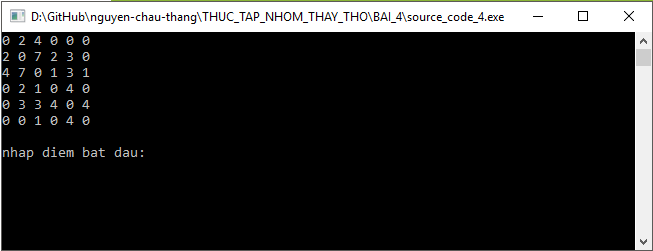
}

## Giao diện chương trình:

* File lưu ma trận(đồ thị) đã nhập: (có thể tự tạo bằng cách gõ trực tiếp vào file text hoặc nhập từ bàn phím sau khi chạy chương trình), ở đây tôi chọn nhập trực tiếp trong file:

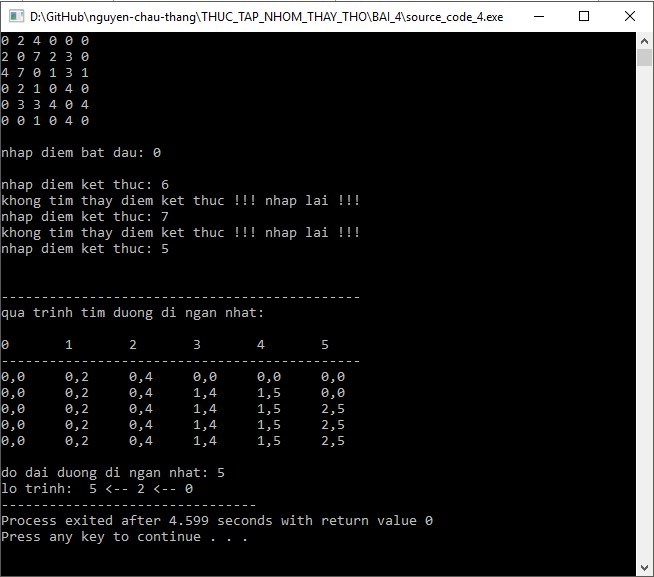


* Sau đó chạy chương trình:



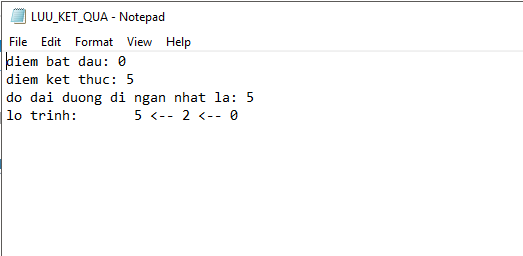
Ma trận sau khi đọc được in lên màng hình để dễ nhận biết

* Sau đó cần nhập điểm bắt đầu và điểm kết thúc để tìm đường (lưu ý điểm nằm trong khoảng từ điểm 0 đến điểm n-1), khi nhập sai sẽ có thông báo nhập lại đến khi nhập đúng:



Sau khi nhập xong, chương trình sẽ xữ lý và cho ra lần lượt các quá trình tìm đường đi ngắn nhất, các kết quả gồm: độ dài đường đi ngắn nhất và lộ trình.

Các kết quả sẽ được lưu vào file như sau:



# BÀI 5: NGẮT

## 5.1 Kiến thức tìm hiểu:

*(bài này em không làm được, nhưng cũng tìm hiểu được 1 chút kiến thức)*

đối với ngắt 10h: thì đây là ngắt để phục vụ cho các dịch vụ của BIOS liên qua đến việc xữ lý video hay màng hình.

Đối với ngắt 13h: đây là ngắt để phục vụ cho các dịch vụ của BIOS liên quan đến sử dụng ổ đỉa như đọc, ghi, viết …

Đới với ngắt 21h: là 1 ngắt phục vụ cho dịch vụ của DOS liên qua đến hầu hết các công việc khác nhau từ bàn phím, màng hình, ổ đĩa …

Cách xữ lý liên quan đến các thanh ghi gồm 14 thanh ghi có các chức năng phân theo từng nhóm, mỗi thanh ghi có 16 bit và chia ra làm 2 phần (phần \*L: low và \*H), \*H chứa lệnh ngắt, \*L chứa lệnh dịch vụ ngắt (mỗi lệnh ngắt \*H chứa nhiều dịch vụ ngắt khác nhau, nên cần đến \*L để gọi chúng ra), chúng cũng chứa dữ liệu nên ta có thể sử dụng các dữ liệu trên đó trong quá trình ta ngắt hoạt động của chúng.

# Phụ lục:

Bài 1: <https://github.com/thangsilver1/nguyen-chau-thang/tree/master/BAITAP/BAI_1>

Bài 2: <https://github.com/thangsilver1/nguyen-chau-thang/tree/master/BAITAP/BAI_2>

Bài 3: <https://github.com/thangsilver1/nguyen-chau-thang/tree/master/BAITAP/BAI_3>

Bài 4: <https://github.com/thangsilver1/nguyen-chau-thang/tree/master/BAITAP/BAI_4>

Bài 5: <https://github.com/thangsilver1/nguyen-chau-thang/tree/master/BAITAP/BAI_5>