**TRƯỜNG ĐẠI HỌC GIAO THÔNG VẬN TẢI PHÂN HIỆU TẠI TP. HỒ CHÍ MINH**



**BỘ MÔN CÔNG NGHỆ THÔNG TIN**

# **BÁO CÁO**

**MÔN: KỸ THUẬT LẬP TRÌNH**

**ĐỀ TÀI: XÂY DỰNG BÀI TOÁN ỨNG DỤNG QUẢN LÝ NHÂN VIÊN**

**Giảng viên hướng dẫn: ThS.Trần Phong Nhã**

**Sinh viên thực hiện:** **HÀ TIÊN TIẾN** **Lớp :** **CQ.65.CNTT**

**Khoá :** **65**

**Tp. Hồ Chí Minh, tháng 05 năm 2025**

**TRƯỜNG ĐẠI HỌC GIAO THÔNG VẬN TẢI PHÂN HIỆU TẠI TP. HỒ CHÍ MINH**



**BỘ MÔN CÔNG NGHỆ THÔNG TIN**

# **BÁO CÁO**

**MÔN: KỸ THUẬT LẬP TRÌNH**

**ĐỀ TÀI: XÂY DỰNG BÀI TOÁN ỨNG DỤNG QUẢN LÝ NHÂN VIÊN**

**Giảng viên hướng dẫn: ThS.Trần Phong Nhã**

**Sinh viên thực hiện: HÀ TIÊN TIẾN Lớp : CQ.65.CNTT**

**Khoá : 65**

**Tp. Hồ Chí Minh, tháng 05 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 “**XÂY DỰNG BÀI TOÁN ỨNG DỤNG QUẢN LÝ NHÂN VIÊN**”. Đặc biệt em xin cảm ơn thầy Trần Phong Nhã 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 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 và hơn hết là thầy Trần Phong Nhã có thật nhiều sức khỏe, có nhiều thành công trong công việc. Em xin chân thành cảm ơn!

# NHẬN XÉT CỦA GIÁO VIÊN HƯỚNG DẪN

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

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

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

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

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

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

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

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

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

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

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

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

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

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

|  |
| --- |
| *Tp. Hồ Chí Minh, ngày ….… tháng ….… năm ….…*  Giáo viên hướng dẫn  Trần Phong Nhã |

# **MỤC LỤC**

[**LỜI CẢM ƠN i**](#_Toc198986046)

[**NHẬN XÉT CỦA GIÁO VIÊN HƯỚNG DẪN ii**](#_Toc198986047)

[**MỤC LỤC iii**](#_Toc198986048)

[**DANH MỤC HÌNH ẢNH viii**](#_Toc198986049)

[**DANH MỤC CHỮ VIẾT TẮT ix**](#_Toc198986050)

[**CHƯƠNG 1: LÝ THUYẾT 1**](#_Toc198986051)

[**1.1 Hàm trong C 1**](#_Toc198986052)

[**1.2 Con trỏ. 1**](#_Toc198986053)

[**1.2.1 Địa chỉ 2**](#_Toc198986054)

[**1.2.2 Toán tử &: 2**](#_Toc198986055)

[**1.2.3 Toán tử \* 3**](#_Toc198986056)

[**1.3 Con trỏ mảng 4**](#_Toc198986057)

[**1.4 Mảng con trỏ 6**](#_Toc198986058)

[**1.4.1 Khái niệm, cú pháp, cách khai báo và sử dụng 6**](#_Toc198986059)

[**1.4.2 Sự khác nhau giữa mảng con trỏ và con trỏ mảng 7**](#_Toc198986060)

[**1.5 Con trỏ hàm 7**](#_Toc198986061)

[**1.6 Cấp phát động 8**](#_Toc198986062)

[**1.7 Xử lí file 11**](#_Toc198986063)

[**1.7.1 Khái quát về tệp trong C và cách khai báo, mở tệp 11**](#_Toc198986064)

[**1.7.2 Các mode khi mở file 11**](#_Toc198986065)

[**1.7.3 Ghi dữ liệu vào tệp và đọc dữ liệu trong tệp 13**](#_Toc198986066)

[**1.7.4 Đóng file 14**](#_Toc198986067)

[**1.8 Kiểu cấu trúc 14**](#_Toc198986068)

[**1.8.1 Khái niệm và cú pháp khai báo kiểu cấu trúc 14**](#_Toc198986069)

[**1.8.2 Khai báo và sử dụng kiểu cấu trúc 15**](#_Toc198986070)

[**1.8.3 Khai báo và sử dụng con trỏ cấu trúc 15**](#_Toc198986071)

[**1.8.4 Cấu trúc lồng nhau 16**](#_Toc198986072)

[**1.8.5 Cấu trúc và cấp phát động 17**](#_Toc198986073)

[**1.9 Danh sách liên kết 17**](#_Toc198986074)

[**CHƯƠNG 2: ỨNG DỤNG 20**](#_Toc198986075)

[**2.1 Giới thiệu đề tài 20**](#_Toc198986076)

[**2.2 Cấu trúc bài toán 43**](#_Toc198986077)

[**2.2.1 Cấu trúc dữ liệu 43**](#_Toc198986078)

[**2.2.2 Các hàm chính 43**](#_Toc198986079)

[**2.3 Các chức năng chính 44**](#_Toc198986080)

[**2.3.1 Thêm nhân viên mới 44**](#_Toc198986081)

[**2.3.2 Cập nhật thông tin nhân viên 44**](#_Toc198986082)

[**2.3.3 Xóa nhân viên 44**](#_Toc198986083)

[**2.3.4 Sắp xếp nhân viên theo độ tuổi 44**](#_Toc198986084)

[**2.3.5 Tách danh sách theo giới tính 44**](#_Toc198986085)

[**2.3.6 Tìm nhân viên sắp nghỉ hưu 44**](#_Toc198986086)

[**2.3.7 Hiển thị danh sách nhân viên 45**](#_Toc198986087)

[**2.3.8 Thống kê giờ làm và lương theo tháng 45**](#_Toc198986088)

[**2.3.9 Ghi thông tin tất cả nhân viên ra file 45**](#_Toc198986089)

[**2.3.10 Nhập lịch làm cố định cho nhân viên 45**](#_Toc198986090)

[**2.3.11 Thêm ca làm mới cho một nhân viên 45**](#_Toc198986091)

[**2.3.12 Cập nhật hoặc xóa ca làm cố định 45**](#_Toc198986092)

[**2.3.13 Đăng ký lịch tăng ca 45**](#_Toc198986093)

[**2.3.14 Đăng ký lịch nghỉ 45**](#_Toc198986094)

[**2.3.15 In lịch làm của tất cả nhân viên 46**](#_Toc198986095)

[**2.3.16 In lịch làm của một nhân viên cụ thể 46**](#_Toc198986096)

[**2.3.17 Ghi toàn bộ lịch làm ra file 46**](#_Toc198986097)

[**2.4 Một số hình ảnh khi chạy chương trình 46**](#_Toc198986098)

[**2.4.1 Menu khi chạy chương trình 46**](#_Toc198986099)

[**2.4.2 Danh sách thông tin tất cả nhân viên 47**](#_Toc198986100)

[**2.4.3 Hình ảnh khi thông tin nhân viên dược ghi trong file 47**](#_Toc198986101)

[**2.4.4 File khi lịch làm nhân viên được ghi 48**](#_Toc198986102)

[**2.4.5 Danh sách nhân viên được chia ra theo giới tính 48**](#_Toc198986103)

[**CHƯƠNG 3: KẾT LUẬN 49**](#_Toc198986104)

[**TÀI LIỆU THAM KHẢO 50**](#_Toc198986105)

# DANH MỤC HÌNH ẢNH

[Hình 2. 1 Hình ảnh menu khi chạy chương trình 46](#_Toc198987857)

[Hình 2. 2 Hình ảnh khi thông tin tất cả nhân viên được in ra danh sách 47](https://d.docs.live.net/e3e96261d899112e/tailieu/baitaplon/Báo%20cáo%20lỹ%20thuật%20lập%20trình.docx#_Toc198987858)

[Hình 2. 3 Hình ảnh khi ghi thông tin tất cả nhân viên vào file 47](#_Toc198987859)

[Hình 2. 4 Hình ảnh thông tin lịch làm khi được ghi vào file 48](#_Toc198987860)

[Hình 2. 5 Hình ảnh danh sách khi được tách theo giới tính 48](#_Toc198987861)

# DANH MỤC CHỮ VIẾT TẮT

|  |  |  |
| --- | --- | --- |
| **Mô tả** | **Ý nghĩa** | **Chú thích** |
| API | Application Programming Interface | Giao diện lập trình ứng dụng |
| GUI | Graphical User Interface | Giao diện người dùng đồ họa |
| OS | Operating System | Hệ điều hành |
| EOF | End Of File | Ký hiệu kết thúc tệp |

# CHƯƠNG 1: LÝ THUYẾT

## 1.1 Hàm trong C

- Một hàm là một nhóm các lệnh đi cùng nhau để thực hiện một nhiệm vụ. Mỗi chương trình C có ít nhất một hàm là hàm **main()**, và tất cả hầu hết các chương trình bình thường đều định nghĩa thêm các hàm.

- Bạn có thể chia đoạn code của bạn thành những hàm riêng biệt. Cách bạn chia đoạn code của bạn thành các hàm khác nhau phụ thuộc vào bạn, nhưng theo tính logic, một hàm thường có một nhiệm vụ nhất định.

- Một sự **khai báo** hàm thông báo với bộ biên dịch về tên của hàm, kiểu trả về và tham số. Một **định nghĩa** hàm cung cấp phần thân của một hàm.

- Các thư viện tiêu chuẩn của ngôn ngữ C cung cấp rất nhiều hàm có sẵn để chương trình của bạn có thể gọi. Ví dụ, hàm **strcat()** có thể nối hai đoạn chuỗi, hàm **memcpy()** dùng để copy một vùng nhớ đến một vùng nhớ khác và rất nhiều hàm khác nữa.

- Một hàm được biết đến với các tên khác nhau như một phương thức, một tuyến phụ hoặc một thủ tục.

- Cú pháp:

**<kiểu\_trả\_về> <tên\_hàm>(<danh\_sách\_tham\_số>) {**

// Các câu lệnh trong thân hàm

**}**

-Ví dụ:

// Hàm tính tổng có giá trị trả về, nhận 2 tham số a và b

int tinhTong1(int a, int b) {

return a + b; // Trả về tổng của a và b

}

// Hàm tính tổng không trả về giá trị và không nhận tham số

void tinhTong2() {

int a, b; // Khai báo hai biến nguyên a và b

printf("Nhap a: "); scanf("%d", &a);

printf("Nhap b: "); scanf("%d", &b);

printf("Tong hai so a va b la: %d", a + b);

}

## 1.2 Con trỏ.

### 1.2.1 Địa chỉ

* Trong ngôn ngữ C, bộ nhớ được chia thành các ô nhớ, mỗi ô lưu trữ một giá trị (có thể là một số nguyên, số thực, ký tự, hay các kiểu dữ liệu khác). Địa chỉ là vị trí của một ô nhớ trong bộ nhớ. Mỗi biến trong C có một địa chỉ bộ nhớ riêng biệt mà giá trị của biến đó được lưu trữ.
* Khi bạn khai báo một biến trong C, hệ thống sẽ cấp phát bộ nhớ cho biến đó, và bạn có thể lấy địa chỉ của biến bằng cách sử dụng toán tử &.

Ví dụ:

int a = 10; // Khai báo một biến int

printf("Giá trị của a: %d\n", a); // In giá trị của a

printf("Địa chỉ của a: %p\n", &a); // In địa chỉ của a (lưu ý: %p dùng để in địa chỉ)

->Trong ví dụ trên, a là tên của biến, và &a là **địa chỉ bộ nhớ** nơi giá trị của a được lưu trữ. Lưu ý rằng %p trong printf là định dạng dùng để in địa chỉ bộ nhớ.

### 1.2.2 **Toán tử &:**

*Toán tử & trong C có hai mục đích sử dụng*:

**- *Lấy địa chỉ của một biến***: Khi bạn đặt & trước tên một biến, bạn sẽ nhận được địa chỉ bộ nhớ của biến đó.

- Ví dụ:

int x = 5;

int \*ptr = &x; // ptr sẽ chứa địa chỉ của biến x

->Trong ví dụ trên, &x lấy địa chỉ của biến x, và gán nó cho con trỏ ptr.

**- *Lấy địa chỉ của đối tượng trong bộ nhớ***: Đây là cách để bạn làm việc với con trỏ, cho phép bạn gián tiếp truy cập và thay đổi giá trị của các biến thông qua địa chỉ của chúng.

- Ví dụ:

int a = 10;

int \*ptr = &a; // ptr trỏ tới địa chỉ của a

**->**Ở đây, &a trả về địa chỉ của biến a, và gán địa chỉ đó cho con trỏ ptr. Khi con trỏ ptr chứa địa chỉ của a, bạn có thể thay đổi giá trị của a thông qua con trỏ.

### **1.2.3 Toán tử \***

a) Khai báo con trỏ (Pointer Declaration):

- Khi khai báo một con trỏ, bạn sử dụng toán tử \* để chỉ ra rằng biến đó sẽ là một con trỏ, tức là nó sẽ lưu trữ địa chỉ của một biến thay vì giá trị trực tiếp.

- Ví dụ:

int \*ptr; // ptr là một con trỏ kiểu int

->Trong trường hợp này, \* cho biết rằng ptr không phải là một biến kiểu int, mà là **một con trỏ trỏ tới kiểu dữ liệu int.**

b) Dereferencing (Truy cập giá trị qua con trỏ):

- Khi con trỏ đã được gán một địa chỉ hợp lệ (chẳng hạn như địa chỉ của một biến), bạn có thể sử dụng toán tử \* để **truy cập giá trị** tại địa chỉ đó. Đây gọi là **dereferencing** con trỏ.

-Ví dụ:

int x = 20;

int \*ptr = &x; // ptr chứa địa chỉ của x

printf("Giá trị của x qua con trỏ ptr: %d\n", \*ptr); // In giá trị của x thông qua ptr

->Trong ví dụ này:

* ptr là con trỏ chứa địa chỉ của x.
* \*ptr lấy **giá trị tại địa chỉ** mà ptr trỏ tới, tức là giá trị của x.
* Nếu không sử dụng toán tử \*, con trỏ sẽ chỉ chứa địa chỉ của x, thay vì giá trị của x.

c) **Ví dụ kết hợp & và \***

- *Một ví dụ thực tế về việc sử dụng cả & và \* trong C là:*

#include <stdio.h>

void increment(int \*p) {

(\*p)++; // Tăng giá trị tại địa chỉ mà con trỏ p trỏ tới

}

int main() {

int a = 5;

printf("Trước khi gọi hàm: a = %d\n", a); // In giá trị của a trước khi thay đổi

increment(&a); // Gửi địa chỉ của a vào hàm increment

printf("Sau khi gọi hàm: a = %d\n", a); // In giá trị của a sau khi thay đổi

return 0;

}

**->**Trong ví dụ này:

* Trong hàm increment, p là con trỏ kiểu int \*. Khi gọi increment(&a), chúng ta truyền địa chỉ của a cho con trỏ p.
* Sau đó, trong hàm, toán tử \*p dùng để truy cập giá trị của a thông qua con trỏ p, và (\*p)++ làm tăng giá trị của a.

## 1.3 Con trỏ mảng

- *Khái niệm*: Con trỏ mảng là một con trỏ dùng để trỏ tới **toàn bộ mảng**, tức là nó giữ địa chỉ của **mảng** chứ không phải chỉ địa chỉ phần tử đầu tiên.

* Cú pháp:

**<Kiểu dữ liệu>** (\* **<tên biến con trỏ>** )[<**kích thước mảng**>];

-> Trong đó, p là con trỏ trỏ tới mảng có n phần tử kiểu int.

- *Cách khai báo và sử dụng:*

* Khai báo:

int a[5] = {1, 2, 3, 4, 5};

int (\*p)[5] = &a;

* Truy cập phần tử:

printf("%d", (\*p)[2]); // In ra 3

* Duyệt mảng bằng con trỏ:

for (int i = 0; i < 5; i++) {

printf("%d ", (\*p)[i]);

}

*- Công dụng*:

* Dùng để **truyền mảng 2 chiều vào hàm**:

void inMaTran(int (\*a)[3], int hang) {

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

for (int j = 0; j < 3; j++) {

printf("%d ", a[i][j]);

}

printf("\n");

}

}

* **Giữ đúng kích thước mảng** khi làm việc trong hàm, tránh lỗi vượt giới hạn.
* Quản lý dữ liệu mảng lớn rõ ràng, đặc biệt khi cần xử lý toàn bộ mảng thay vì từng phần tử.

*- Lưu ý:*

* Phải dùng dấu () trong khai báo: int (\*p)[n] (nếu không sẽ trở thành mảng con trỏ).
* Con trỏ mảng chỉ phù hợp khi kích thước mảng cố định, biết trước.

## 1.4 Mảng con trỏ

### 1.4.1 Khái niệm, cú pháp, cách khai báo và sử dụng

- ***Khái niệm*:**Mảng con trỏ là một mảng mà mỗi phần tử của nó là một con trỏ. Điều này cho phép mỗi phần tử trong mảng trỏ đến một vùng nhớ riêng biệt, thường được sử dụng để lưu trữ danh sách các chuỗi hoặc mảng có độ dài khác nhau.

*- Cú pháp:*

<kiểu dữ liệu> \*<tên\_mảng>[kích\_thước];

- *Khai báo:*

char \*dsTen[3]; // Mảng gồm 3 con trỏ kiểu char\*

*- Sử dụng:*

* Lưu trữ danh sách các chuỗi: Mỗi con trỏ trong mảng có thể trỏ đến một chuỗi khác nhau.
* **Truy cập phần tử:** Sử dụng chỉ số mảng để truy cập từng con trỏ và sau đó dereference để lấy giá trị.
* Ví dụ:

#include <stdio.h>

int main() {

char \*dsTen[] = {"Nam", "Lan", "Huy"};

for (int i = 0; i < 3; i++) {

printf("Tên %d: %s\n", i + 1, dsTen[i]);

}

return 0;

}

*- Lưu ý:*

* Cần đảm bảo mỗi con trỏ trong mảng được gán địa chỉ hợp lệ trước khi sử dụng.
* Mảng con trỏ rất hữu ích khi làm việc với danh sách các chuỗi có độ dài khác nhau, giúp tiết kiệm bộ nhớ và linh hoạt trong xử lý dữ liệu.

### 1.4.2 Sự khác nhau giữa mảng con trỏ và con trỏ mảng

*- Định nghĩa:*

* Con trỏ mảng: là con trỏ trỏ đến toàn bộ mảng.
* Mảng con trỏ: là mảng mà mỗi phần tử là một con trỏ.

*- Cú pháp:*

* Con trỏ mảng: <kiểu dữ liệu> (\*<tên biến>)[kích thước];
* Mảng con trỏ: <kiểu dữ liệu>\* <tên mảng>[kích thước];

*- Ví dụ khai báo:*

* Con trỏ mảng: int (\*p)[5];
* Mảng con trỏ: int \*a[5];

*- Gán giá trị:*

* Con trỏ mảng: p = &arr; // với arr là mảng int[5]
* Mảng con trỏ: a[0] = &x; a[1] = &y; ...

*- Công dụng:*

* Con trỏ mảng: dùng khi xử lý nguyên cả mảng cố định, truyền mảng vào hàm.
* Mảng con trỏ: dùng khi cần quản lý nhiều vùng nhớ riêng biệt, như danh sách chuỗi.

*- Truy cập phần tử:*

* Con trỏ mảng: (\*p)[i]
* Mảng con trỏ: \*a[i] hoặc a[i][j]

## 1.5 Con trỏ hàm

*- Khái niệm*: **Con trỏ hàm** là một loại con trỏ dùng để lưu trữ **địa chỉ của hàm**. Khi một hàm được gọi thông qua con trỏ hàm, chương trình sẽ thực thi hàm mà con trỏ trỏ tới. Con trỏ hàm cho phép chương trình linh hoạt hơn trong việc gọi các hàm khác nhau tại thời điểm chạy, điều này rất hữu ích trong các tình huống như callback, xử lý sự kiện, hoặc khi cần thay đổi hành vi của một phần mềm.

*- Cú pháp:*

<kiểu trả về> (\*<tên con trỏ>)(<danh sách tham số>);

*- Cách gán hàm cho con trỏ:*

<con trỏ hàm> = <tên hàm>;

*- Gọi hàm thông qua con trỏ:*

<con trỏ hàm>(<danh sách tham số>) || (\*<con trỏ hàm>)(<danh sách tham số>);

- Ví dụ:

#include <stdio.h>

int cong(int a, int b) {

return a + b;

}

int tru(int a, int b) {

return a - b;

}

int main() {

int (\*thaoTac)(int, int);

thaoTac = cong;

printf("Cộng: %d\n", thaoTac(5, 3));

thaoTac = tru;

printf("Trừ: %d\n", thaoTac(5, 3));

return 0;

}

->Trong ví dụ trên:

* int(\*ptrHam)(int,int);  
  → Khai báo con trỏ hàm ptrHam trỏ đến hàm có 2 tham số kiểu int, trả về int.
* ptrHam=cong;  
  → Gán địa chỉ hàm cong cho con trỏ ptrHam.
* ptrHam(5,3);  
  → Gọi hàm cong thông qua con trỏ.

=> **Ưu điểm:** Có thể thay đổi linh hoạt hàm thực thi bằng cách gán hàm khác vào con trỏ, không cần thay đổi mã gọi.

## 1.6 Cấp phát động

*- Khái niệm:* **Cấp phát động** là quá trình **xin cấp phát bộ nhớ trong lúc chương trình đang chạy** (runtime) thay vì xác định trước kích thước bộ nhớ tại thời điểm biên dịch (compile-time).

*- Khi nào cần cấp phát động:*

* Khi bạn không biết trước số lượng phần tử cần lưu trữ.
* Khi muốn tiết kiệm bộ nhớ bằng cách chỉ cấp phát đúng khi cần thiết.
* Khi cần cấu trúc dữ liệu linh hoạt như: danh sách liên kết, cây, đồ thị,...

- Bộ nhớ được cấp phát nằm ở **vùng Heap** – là vùng bộ nhớ dùng để quản lý dữ liệu có vòng đời linh hoạt hơn so với stack.

*- Cú pháp sử dụng:*

* **Malloc()**: Cấp phát size byte bộ nhớ, chưa khởi tạo giá trị.

Cú pháp: ptr = (type\*)malloc(size)

* **Calloc()**: Cấp phát bộ nhớ cho mảng n phần tử, mỗi phần tử size byte, **được khởi tạo về 0.**

Cú pháp: ptr = (type\*)calloc(n, size)

* **Realloc():** Thay đổi kích thước vùng nhớ đã cấp phát (có thể đổi địa chỉ).

Cú pháp: ptr = (type\*)realloc(ptr, new\_size)

* **free():** Giải phóng bộ nhớ đã cấp phát, tránh rò rỉ bộ nhớ.

Cú pháp: free(ptr)

*- Đối tượng áp dụng:*

* **Mảng động**: khi kích thước mảng được nhập từ người dùng.
* **Con trỏ đến kiểu dữ liệu**: int \*p = malloc(sizeof(int));
* **Cấu trúc (struct)**: tạo một struct động.

Ví dụ: struct SV \*s = malloc(sizeof(struct SV));

* **Cấu trúc dữ liệu động**: danh sách liên kết, cây, hàng đợi, ngăn xếp,...

-Ví dụ:

int \*a;

int n;

printf("Nhap so luong: ");

scanf("%d", &n);

a = (int\*)malloc(n \* sizeof(int)); // cấp phát mảng động

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

printf("a[%d] = ", i);

scanf("%d", &a[i]);

}

free(a); // giải phóng bộ nhớ

* Cấu trúc động:

- *Một số lưu ý:*

typedef struct {

char ten[30];

float diem;

} SV;

SV \*p = (SV\*)malloc(sizeof(SV));

strcpy(p->ten, "Nguyen Van A");

p->diem = 8.5;

free(p);

* Luôn kiểm tra con trỏ sau khi cấp phát:
* Nhớ giải phóng bộ nhớ bằng free() khi không còn dùng nữa → tránh memory leak.
* Không truy cập vào vùng nhớ đã free → gây lỗi nghiêm trọng.
* malloc() và realloc() không khởi tạo giá trị, còn calloc() khởi tạo toàn bộ bằng 0.
* realloc() có thể trả về địa chỉ khác → nên gán lại cho con trỏ.

## 1.7 Xử lí file

### 1.7.1 Khái quát về tệp trong C và cách khai báo, mở tệp

- Tệp (file) là nơi lưu trữ dữ liệu lâu dài trên đĩa cứng (HDD/SSD), không mất dữ liệu sau khi chương trình kết thúc.Trong C, để xử lý tệp, bạn sử dụng **con trỏ tệp** thuộc kiểu FILE \* và các hàm có sẵn trong thư viện <stdio.h>.

- Khai báo và mở tệp:

* Khai báo đồng thời mở tệp:

<FILE \*> <tên con trỏ> = fopen("<tên tệp>", "<chế độ>");

Ví dụ:

FILE \*f = fopen("data.txt", "r")

* Khai báo và mở tệp riêng:

<FILE \*> <tên\_con\_trỏ>;

<tên con trỏ> = = fopen("<tên tệp>", "<chế độ>");

->Kết luận:

* Khai báo đồng thời mở tệp: Nhanh gọn nhưng thiếu linh hoạt, dùng khi mở tệp đơn giản rõ ràng.
* Khai báo và mở tệp riêng: Linh hoạt, rõ ràng nhưng tốn dòng code hơn, dùng khi cần mở tệp theo điều kiện, dùng trong nhiều hàm.

### 1.7.2 Các mode khi mở file

*-Mode “ r ”:*

* Công dụng: Mở file để đọc.
* Lưu ý: Nếu file không tồn tại thì hàm fopen() trả về con trỏ NULL

*- Mode “ rb ”:*

* Công dụng: Mở file để đọc theo kiểu file nhị phân.
* Lưu ý: Nếu file không tồn tại thì hàm fopen() trả về con trỏ NULL

*- Mode “ w ”:*

* Công dụng: Mở file để ghi
* Lưu ý: Nếu file đã tồn tại thì sẽ làm việc với file đó, nếu file chưa tồn tại sẽ tạo 1 file mới

*- Mode “ wb ”:*

* Công dụng: Mở file để ghi theo kiểu file nhị phân
* Lưu ý: Nếu file đã tồn tại thì sẽ làm việc với file đó, nếu file chưa tồn tại sẽ tạo 1 file mới

*- Mode “ a ”:*

* Công dụng: Mở file text lên để ghi tiếp vào cuối file mà không xóa nội dung cũ trong file
* Lưu ý: Nếu file đã tồn tại thì sẽ làm việc với file đó, nếu file chưa tồn tại sẽ tạo 1 file mới

*- Mode “ ab ”:*

* Công dụng: Mở file nhị phân lên để ghi tiếp vào cuối file mà không xóa nội dung cũ trong file
* Lưu ý: Nếu file đã tồn tại thì sẽ làm việc với file đó, nếu file chưa tồn tại sẽ tạo 1 file mới

*- Mode “ r+ ”:*

* Công dụng: Mở file để vừa đọc vừa ghi
* Lưu ý: Nếu file không tồn tại thì hàm fopen() trả về con trỏ NULL

*- Mode “ rb+ ”:*

* Công dụng: Mở file để vừa đọc vừa ghi theo kiểu nhị phân
* Lưu ý: Nếu file không tồn tại thì hàm fopen() trả về con trỏ NULL

*- Mode “ w+ ”:*

* Công dụng: Mở file để vừa đọc vừa ghi
* Lưu ý: Nếu file đã tồn tại thì sẽ làm việc với file đó, nếu file chưa tồn tại sẽ tạo 1 file mới

*- Mode “ wb+ ”:*

* Công dụng: Mở file để vừa đọc vừa ghi theo kiểu nhị phân
* Lưu ý: Nếu file đã tồn tại thì sẽ làm việc với file đó, nếu file chưa tồn tại sẽ tạo 1 file mới

*- Mode “ a+ ”:*

* Công dụng: Mở file lên vừa để đọc và ghi vào cuối file
* Lưu ý: Nếu file đã tồn tại thì sẽ làm việc với file đó, nếu file chưa tồn tại sẽ tạo 1 file mới

*- Mode “ ab+ ”:*

* Công dụng: Mở file để vừa đọc vừa ghi vào cuối file theo kiểu nhị phân
* Lưu ý: Nếu file đã tồn tại thì sẽ làm việc với file đó, nếu file chưa tồn tại sẽ tạo 1 file mới

### 1.7.3 Ghi dữ liệu vào tệp và đọc dữ liệu trong tệp

- Ghi dữ liệu vào tệp:

fprintf(<tệp>, "<định dạng>", <giá trị>);

fputs("<chuỗi>", <tệp>);

fwrite(<con trỏ dữ liệu>, <kích thước phần tử>, <số phần tử>, <tệp>);

* Cú pháp:
* Ví dụ:

fprintf(f, "Diem: %.2f\n", diem);

fputs("Hello\n", f);

fwrite(&sv, sizeof(SinhVien), 1, f);

*- Đọc dữ liệu từ tệp đã có:*

* Cú pháp:

fscanf(<tệp>, "<định dạng>", &<biến>);

fgets(<chuỗi>, <kích thước>, <tệp>);

fread(<con trỏ lưu dữ liệu>, <kích thước phần tử>, <số phần tử>, <tệp>);

* Ví dụ:

fprintf(f, "Diem: %.2f\n", diem);

fputs("Hello\n", f);

fwrite(&sv, sizeof(SinhVien), 1, f);

### 1.7.4 Đóng file

fclose(<tên tệp>);

*- Cú pháp:*

fclose(f);

- Ví dụ:

## 1.8 Kiểu cấu trúc

### 1.8.1 Khái niệm và cú pháp khai báo kiểu cấu trúc

*- Khái niệm:* **Cấu trúc (**struct**)** trong C là một kiểu dữ liệu cho phép kết hợp nhiều kiểu dữ liệu khác nhau lại với nhau dưới một tên duy nhất. Các thành phần trong một cấu trúc được gọi là **trường (fields)** hoặc **thành viên (members).**

*- Cú pháp khai báo:*

//Không có từ khóa typedef

struct <tên cấu trúc> {

<kiểu dữ liệu thành viên1>;

<kiểu dữ liệu thành viên2>;

...

}<Danh sách các biến>;

Trong đó:

* **<tên cấu trúc>**: Tên của cấu trúc.
* **Các thành viên**: Các biến với kiểu dữ liệu khác nhau, ví dụ như int, float, char, ...
* **Lưu ý:** Khai báo như cách trên thì khi khai báo thêm biến thì cần từ khóa struct ở trước.

//Khai báo có từ khóa typedef

typedef struct <tên cấu trúc> {

<kiểu dữ liệu thành viên1>;

<kiểu dữ liệu thành viên2>;

...

}<Tên mới cho kiểu cấu trúc>;

Trong đó:

* **<tên cấu trúc>**: Tên của cấu trúc.
* **Các thành viên**: Các biến với kiểu dữ liệu khác nhau, ví dụ như int, float, char, ...
* **Lưu ý**: Khi khai báo có từ khóa typedef thì lúc khai báo thêm biến chỉ cần dùng tên mới của cấu trúc để khai báo mà không cần từ khóa struct.

### 1.8.2 Khai báo và sử dụng kiểu cấu trúc

*- Khai báo:*

* **Cấu trúc có thể được khai báo như một kiểu dữ liệu**: Sau khi khai báo cấu trúc, bạn có thể tạo nhiều biến kiểu cấu trúc đó.

typedef struct Student{

char name[50];

int age;

float gpa;

} St;

int main(){

St s1, s2, s3; //hoặc struct Student s1, s2, s3;

Return 0;

}

* **Truy cập các thành viên**: Sử dụng dấu chấm (.) để truy cập các thành viên của cấu trúc.

s1 = {"Nguyen Van A", 20, 8.5};

printf("Ho ten: %s\n", s1.name);

printf("Tuoi: %d\n", s1.age);

printf("Diem trung binh: %.2f\n", s1.gpa);

### 1.8.3 Khai báo và sử dụng con trỏ cấu trúc

*- Khai báo:*

Struct Student \*ptr; //hoặc St \*ptr;

*- Sử dụng:*

* Bạn có thể sử dụng con trỏ để tham chiếu tới cấu trúc. Khi đó, thay vì dấu chấm (.), bạn sẽ dùng dấu mũi tên (->) để truy cập thành viên

ptr = &s1; // Trỏ tới biến s1

// In thông tin thông qua con trỏ

printf("Ho ten: %s\n", ptr->name);

printf("Tuoi: %d\n", ptr->age);

printf("Diem TB: %.2f\n", ptr->gpa);

* **Lý do sử dụng con trỏ**: Con trỏ giúp bạn dễ dàng thao tác với các cấu trúc trong các hàm hoặc truyền cấu trúc qua các hàm mà không cần sao chép toàn bộ dữ liệu.

### 1.8.4 Cấu trúc lồng nhau

*-* Một cấu trúc có thể chứa cấu trúc khác bên trong nó, được gọi là **cấu trúc lồng nhau.**

**Ví dụ:**

#include <stdio.h>

typedef struct Date {

int day, year, month;

}date;

typedef struct Person {

char hoten[50];

date birthdate; // Cấu trúc lồng nhau

}Person;

int main() {

Person p1;

strcpy(p1.name, "Alice");

p1.birthdate.day = 15;

p1.birthdate.month = 7;

p1.birthdate.year = 1990;

return 0;

}

### 1.8.5 Cấu trúc và cấp phát động

- Bạn có thể sử dụng malloc() để cấp phát bộ nhớ cho một cấu trúc động.

Ví dụ:

struct Person \*ptr;

ptr = (struct Person\*) malloc(sizeof(struct Person));

ptr->age = 25;

strcpy(ptr->name, "Bob");

free(ptr);

## 1.9 Danh sách liên kết

*- Khái niệm:* **Danh sách liên kết đơn (**singly linked list**)** là một cấu trúc dữ liệu động bao gồm nhiều node liên kết nhau thông qua con trỏ, trong đó mỗi node chứa: dữ liệu (Data), con trỏ tới node kế tiếp (next).

*- Cú pháp khai báo và cấp phát:*

//Cấu trúc Node

typedef struct Node {

int data;

struct Node \*next;

} Node;

//Cách cấp phát một node mới

Node\* taoNode(int x) {

Node\* p = (Node\*)malloc(sizeof(Node));

if (p == NULL) {

printf("Khong du bo nho!\n");

exit(1);

}

p->data = x;

p->next = NULL;

return p;

}

*-Đặc điểm:*

* Không giới hạn số lượng phần tử như mảng.
* Thêm, xóa phần tử linh hoạt, không cần dịch chuyển phần tử như mảng.
* Được quản lý hoàn toàn bằng **cấp phát động** (heap memory).

- Các thao tác cơ bản:

* Chèn node vào cuối danh sách:

void chenCuoi(Node \*\*head, int x) {

Node \*newNode = taoNode(x);

if (\*head == NULL) {

\*head = newNode;

return;

}

Node \*p = \*head;

while (p->next != NULL) p = p->next;

p->next = newNode;

}

* Xóa toàn bộ danh sách:

void giaiPhong(Node \*head) {

Node \*tmp;

while (head != NULL) {

tmp = head;

head = head->next;

free(tmp);

}

}

* Chèn vào đầu danh sách:
* In danh sách:

void inDS(Node \*head) {

Node \*p = head;

while (p != NULL) {

printf("%d -> ", p->data);

p = p->next;

}

printf("NULL\n");

}

void chenDau(Node \*\*head, int x) {

Node\* newNode = taoNode(x);

newNode->next = \*head;

\*head = newNode;

}

# CHƯƠNG 2: ỨNG DỤNG

## 2.1 Giới thiệu đề tài

- Trong quá trình học môn lập trình C tại trường, em nhận thấy rằng việc kết hợp giữa kiến thức lý thuyết và thực hành thông qua một bài toán cụ thể là rất cần thiết. Việc áp dụng các khái niệm như **con trỏ, danh sách liên kết, cấp phát động và xử lý tệp tin** không chỉ giúp em hiểu rõ hơn mà còn nâng cao kỹ năng lập trình một cách thực tế.

- Đề tài **quản lý nhân viên** được em lựa chọn bởi đây là một bài toán có tính ứng dụng cao, thường gặp trong nhiều hệ thống phần mềm hiện đại, đặc biệt là trong lĩnh vực quản trị nhân sự. Việc tự động hóa quản lý thông tin nhân viên, lịch làm việc, tăng ca và nghỉ phép là một nhu cầu thiết yếu trong các tổ chức.

- Thông qua việc xây dựng chương trình này bằng ngôn ngữ C, em không chỉ rèn luyện tư duy lập trình mà còn học cách tổ chức chương trình một cách chặt chẽ, tối ưu và dễ mở rộng trong thực tế.

**Source code:**

#include <stdio.h>

#include <string.h>

#include <stdlib.h>

#include <math.h>

typedef struct tangCa{

char ngayTangCa[12];

char caTang[10];

char batDau[7];

char ketThuc[7];

struct tangCa \*next;

}tangCa;

typedef struct lichNghi{

char ngayNghi[12];

char caNghi[10];

char batDau[7];

char ketThuc[7];

struct lichNghi \*next;

}lichNghi;

typedef struct LichLam{

char caLam[10];

char batDau[7];

char ketThuc[7];

struct LichLam \*next;

}LichLam;

typedef struct NhanVien{

    char hoTen[50];

    char ngayLam[12];

    char gioiTinh[5];

    char ngaySinh[12];

    float luong;

    char maNV[11];

    LichLam \*LichLam;

    tangCa \*tangCa;

    lichNghi \*lichNghi;

}NhanVien;

//ham cap phat bo nho cho bien nhan vien

NhanVien \*capPhat(int n, NhanVien \*nv){

    NhanVien \*newNV=(NhanVien\*)realloc(nv, n\*sizeof(NhanVien));

    if(newNV==NULL){

        printf("Khong the cap phat bo nho !");

        return NULL;

    }

    return newNV;

}

//ham them nhan vien vao danh sach

void themNhanVien(NhanVien \*\*nv,int n, int m){

    \*nv=capPhat(n+m,\*nv);

    for(int i=m;i<n+m;i++){

        printf("Nhap ho ten: ");

        fgets((\*nv)[i].hoTen,sizeof((\*nv)[i].hoTen),stdin);

        (\*nv)[i].hoTen[strcspn((\*nv)[i].hoTen,"\n")]='\0';

         printf("Gioi tinh: ");

        fgets((\*nv)[i].gioiTinh,sizeof((\*nv)[i].gioiTinh),stdin);

        (\*nv)[i].gioiTinh[strcspn((\*nv)[i].gioiTinh,"\n")]='\0';

        printf("Nhap ngay sinh (vd: 09/11/2006): ");

        fgets((\*nv)[i].ngaySinh,sizeof((\*nv)[i].ngaySinh),stdin);

        (\*nv)[i].ngaySinh[strcspn((\*nv)[i].ngaySinh,"\n")]='\0';

         printf("Nhap ngay lam (vd: 09/11/2006): ");

        fgets((\*nv)[i].ngayLam,sizeof((\*nv)[i].ngayLam),stdin);

        (\*nv)[i].ngayLam[strcspn((\*nv)[i].ngayLam,"\n")]='\0';

         printf("Nhap ma nhan vien: ");

        fgets((\*nv)[i].maNV,sizeof((\*nv)[i].maNV),stdin);

        (\*nv)[i].maNV[strcspn((\*nv)[i].maNV,"\n")]='\0';

        printf("Luong: ");

        scanf("%f",&(\*nv)[i].luong);

        getchar();

        (\*nv)[i].LichLam=NULL;

        (\*nv)[i].tangCa=NULL;

        (\*nv)[i].lichNghi=NULL;

    }

}

//ham cap nhat thong tin sinh vien

void capNhatThongTinNhanVien(NhanVien \*nv, int m){

        if(nv==NULL){

                printf("Danh sach rong !");

                return;

        }

        char maNv[12];

        printf("Nhap ma nhan vien cua nhan vien ban muon thay doi thong tin: ");

        fgets(maNv,sizeof(maNv),stdin);

        maNv[strcspn(maNv,"\n")]='\0';

        for(int i=0;i<m;i++){

                 if(strcmp(maNv,nv[i].maNV)==0){

                        printf("Da tim thay nhan vien ban muon thay doi thong tin !\n");

                        printf("Nhap thong tin moi:\n");

                        printf("Nhap ho ten: ");

                        fgets(nv[i].hoTen,sizeof(nv[i].hoTen),stdin);

                        nv[i].hoTen[strcspn(nv[i].hoTen,"\n")]='\0';

                        printf("Gioi tinh: ");

                        fgets(nv[i].gioiTinh,sizeof(nv[i].gioiTinh),stdin);

                        nv[i].gioiTinh[strcspn(nv[i].gioiTinh,"\n")]='\0';

                        printf("Nhap ngay sinh (vd: 09/11/2006): ");

                        fgets(nv[i].ngaySinh,sizeof(nv[i].ngaySinh),stdin);

                        nv[i].ngaySinh[strcspn(nv[i].ngaySinh,"\n")]='\0';

                        printf("Nhap ngay lam (vd: 09/11/2006): ");

                        fgets(nv[i].ngayLam,sizeof(nv[i].ngayLam),stdin);

                       nv[i].ngayLam[strcspn(nv[i].ngayLam,"\n")]='\0';

                        printf("Nhap ma nhan vien: ");

                        fgets(nv[i].maNV,sizeof(nv[i].maNV),stdin);

                        nv[i].maNV[strcspn(nv[i].maNV,"\n")]='\0';

                        printf("Luong: ");

                        scanf("%f",&nv[i].luong);

                        getchar();

                        printf("Da cap nhat thanh cong !\n");

                        return ;

                        }

                }

        printf("Khong tim thay nhan vien co ma: %s\n",maNv);

}

//ham xoa nhan vien khoi danh sach

void xoaNhanVien(NhanVien \*nv,int \*m){

         if(nv==NULL){

                printf("Danh sach rong !");

                return;

        }

        char maNv[12];

        printf("Nhap ma nhan vien cua nhan vien ban muon xoa: ");

        fgets(maNv,sizeof(maNv),stdin);

        maNv[strcspn(maNv,"\n")]='\0';

        for(int i=0;i<\*m;i++){

                 if(strcmp(maNv,nv[i].maNV)==0){

                        LichLam \*ll =nv[i].LichLam;

                       while (ll!= NULL) {

                                LichLam \*llTemp = ll;

                                ll = ll->next;

                                free(llTemp);

                        }

                        tangCa \*tc = nv[i].tangCa;

                        while (tc != NULL) {

                                tangCa \*tcTemp = tc;

                                tc = tc->next;

                                free(tcTemp);

                                }

                        lichNghi \*ln = nv[i].lichNghi;

                        while (ln != NULL) {

                                lichNghi \*lnTemp = ln;

                                ln = ln->next;

                                free(lnTemp);

                                }

                        for(int j=i;j<\*m-1;j++){

                                nv[j]=nv[j+1];

                        }

                        (\*m)--;

                        printf("Da xoa nhan vien co ma so %s\n",maNv);

                        return;

                }

        }

        printf("Khong tim thay nhan vien ma ban muon\n");

}

//ham nhap lich lam co dinh cho cac nhan vien trong danh sach

void nhapLichLam(NhanVien \*nv, int m){

        if(nv==NULL){

                printf("Danh sach rong !\n");

        }

        for(int i=0;i<m;i++){

                printf("Nhap lich lam cho nhan vien: %s\n",nv[i].hoTen);

                LichLam \*newLichLam=(LichLam\*)malloc(sizeof(LichLam));

                if(newLichLam==NULL){

                        printf("Khong the nhap lich !");

                        continue;

                }

                newLichLam->next=NULL;

                printf("Nhap thong tin ca lam (co dinh):\n");

                printf("Ca lam (sang-chieu-toi): ");

                fgets(newLichLam->caLam,sizeof(newLichLam->caLam),stdin);

                newLichLam->caLam[strcspn(newLichLam->caLam,"\n")]='\0';

                printf("Gio bat dau ca (00:00): ");

                fgets(newLichLam->batDau,sizeof(newLichLam->batDau),stdin);

                newLichLam->batDau[strcspn(newLichLam->batDau,"\n")]='\0';

                printf("Gio ket thuc ca (00:00): ");

                fgets(newLichLam->ketThuc,sizeof(newLichLam->ketThuc),stdin);

                newLichLam->ketThuc[strcspn(newLichLam->ketThuc,"\n")]='\0';

                if(nv[i].LichLam==NULL){

                nv[i].LichLam=newLichLam;

                }

                else{

                        LichLam \*temp =nv[i].LichLam;

                        while(temp->next!=NULL){

                                temp=temp->next;

                        }

                        temp->next=newLichLam;

                }

        }

}

//ham them ca lam co dinh cho tung nhan vien

void themCaLamTungNhanVien(NhanVien \*nv, int m){

        if(nv==NULL){

                printf("Danh sach rong !\n");

                return;

        }

        char hoten[50];

        printf("Nhap ten nhan vien: ");

        fgets(hoten,sizeof(hoten),stdin);

        hoten[strcspn(hoten,"\n")]='\0';

        int test=0;

        for(int i=0;i<m;i++){

                if(strcasecmp(nv[i].hoTen,hoten)==0){

                        printf("Da tim thay nhan vien muon cap nhat.\n");

                        LichLam \*ll=(LichLam\*)malloc(sizeof(LichLam));

                        if(ll==NULL){

                                printf("Khong the cap phat !\n");

                                return;

                        }

                        ll->next=NULL;

                        printf("Nhap thong tin ca lam : \n");

                        printf("Ca lam (sang-chieu-toi): ");

                        fgets(ll->caLam,sizeof(ll->caLam),stdin);

                        ll->caLam[strcspn(ll->caLam,"\n")]='\0';

                       printf("Gio bat dau ca (00:00): ");

                        fgets(ll->batDau,sizeof(ll->batDau),stdin);

                        ll->batDau[strcspn(ll->batDau,"\n")]='\0';

                        printf("Gio ket thuc ca (00:00): ");

                        fgets(ll->ketThuc,sizeof(ll->ketThuc),stdin);

                        ll->ketThuc[strcspn(ll->ketThuc,"\n")]='\0';

                        LichLam \*temp = nv[i].LichLam;

                        if (temp == NULL) {

                        nv[i].LichLam = ll;

                        } else {

                        while (temp->next != NULL) {

                        temp = temp->next;

                        }

                        temp->next = ll;

                        }

                        printf("Cap nhat thanh cong.\n");

                        test=1;

                        return;

                }

        }

        if(!test){

        printf("Khong tim thay !\n");

        }

}

//ham nhap lich tang ca cho nhan vien

void nhapLichTangCa(NhanVien \*nv,int m){

        if(nv==NULL){

                printf("Danh sach rong !\n");

                return;

        }

        char hoten[50];

        printf("Nhap ten nhan vien dang ki tang ca: ");

        fgets(hoten,sizeof(hoten),stdin);

        hoten[strcspn(hoten,"\n")]='\0';

        int test=0;

        for(int i=0;i<m;i++){

                if(strcmpi(hoten,nv[i].hoTen)==0){

                                if(nv[i].LichLam==NULL){

                                        printf("Hay nhap lich lam co dinh truoc khi tang ca. \n");

                                        return;

                                }

                        printf("Da tim thay nhan vien: %s\n",nv[i].hoTen);

                        tangCa \*tc=(tangCa\*)malloc(sizeof(tangCa));

                        if(tc==NULL){

                                printf("Khong the cap phat bo nho !\n");

                                return;

                        }

                        tc->next=NULL;

                        printf("Nhap ngay tang ca (vd: 09/11/2006): ");

                       fgets(tc->ngayTangCa,sizeof(tc->ngayTangCa),stdin);

                       tc->ngayTangCa[strcspn(tc->ngayTangCa,"\n")]='\0';

                       printf("Nhap thoi gian tang ca:\n");

                       printf("Ca lam tang ca (sang-chieu-toi): ");

                       fgets(tc->caTang,sizeof(tc->caTang),stdin);

                       tc->caTang[strcspn(tc->caTang,"\n")]='\0';

                       printf("Gio bat dau ca (00:00): ");

                        fgets(tc->batDau,sizeof(tc->batDau),stdin);

                        tc->batDau[strcspn(tc->batDau,"\n")]='\0';

                        printf("Gio ket thuc ca (00:00): ");

                        fgets(tc->ketThuc,sizeof(tc->ketThuc),stdin);

                        tc->ketThuc[strcspn(tc->ketThuc,"\n")]='\0';

                        if(nv[i].tangCa==NULL){

                                nv[i].tangCa=tc;

                        }

                        else{

                                tangCa \*temp=nv[i].tangCa;

                                while(temp->next!=NULL){

                                        temp=temp->next;

                                }

                                temp->next=tc;

                        }

                        printf("Da dang ki tang ca thanh cong !\n");

                        test=1;

                       return;

                }

        }

        if(!test){

                printf("Khong tim thay !");

        }

}

//ham nhap lich xin nghi cho nhan vien

void nhapLichNghi(NhanVien \*nv,int m){

         if(nv==NULL){

                printf("Danh sach rong !\n");

                return;

        }

        char hoten[50];

        printf("Nhap ten nhan vien dang ki nghi: ");

        fgets(hoten,sizeof(hoten),stdin);

        hoten[strcspn(hoten,"\n")]='\0';

        int test=0;

        for(int i=0;i<m;i++){

                if(strcmpi(hoten,nv[i].hoTen)==0){

                        if(nv[i].LichLam==NULL){

                                printf("Hay nhap lich lich lam co dinh truoc khi nghi.\n");

                                return;

                        }

                        printf("Da tim thay nhan vien: %s\n",nv[i].hoTen);

                        lichNghi \*ln=(lichNghi\*)malloc(sizeof(lichNghi));

                        if(ln==NULL){

                                printf("Khong the cap phat bo nho !\n");

                                return;

                        }

                        ln->next=NULL;

                        printf("Nhap ngay nghi (vd: 09/11/2006): ");

                       fgets(ln->ngayNghi,sizeof(ln->ngayNghi),stdin);

                       ln->ngayNghi[strcspn(ln->ngayNghi,"\n")]='\0';

                       printf("Nhap thoi gian:\n");

                       printf("Nhap ca nghi (sang-chieu-toi): ");

                       fgets(ln->caNghi,sizeof(ln->caNghi),stdin);

                       printf("Gio bat dau nghi (00:00): ");

                        fgets(ln->batDau,sizeof(ln->batDau),stdin);

                        ln->batDau[strcspn(ln->batDau,"\n")]='\0';

                        printf("Gio ket thuc nghi (00:00): ");

                        fgets(ln->ketThuc,sizeof(ln->ketThuc),stdin);

                        ln->ketThuc[strcspn(ln->ketThuc,"\n")]='\0';

                        if(nv[i].lichNghi==NULL){

                                nv[i].lichNghi=ln;

                        }

                        else{

                                lichNghi \*temp=nv[i].lichNghi;

                                while(temp->next!=NULL){

                                        temp=temp->next;

                                }

                                temp->next=ln;

                        }

                        printf("Da dang ki nghi thanh cong !\n");

                        test=1;

                        return;

                 }

        }

        if(!test){

                printf("Khong tim thay !");

        }

}

//ham cap nhat lich lam co dinh

void capNhatLichLam(NhanVien \*nv, int m){

        if(nv==NULL){

                printf("Danh sach rong !\n");

                return;

        }

        char hoten[50];

        printf("Nhap  ho ten nhan vien ban muon thay doi lich lam: ");

        fgets(hoten,sizeof(hoten),stdin);

        hoten[strcspn(hoten,"\n")]='\0';

        int t =0;

        for(int i=0;i<m;i++){

                if(strcasecmp(hoten,nv[i].hoTen)==0){

                        printf("Da tim thay nhan vien: %s\n",nv[i].hoTen);

                        if(nv[i].LichLam==NULL){

                                printf("Nhan vien nay chua co lich lam !\n");

                                return;

                        }

                        int dem =1;

                        LichLam \*ll=nv[i].LichLam;

                        printf("Lich lam hien co la:\n");

                        while(ll!=NULL){

                                printf("[%d]: ca %s - (%s - %s)\n",dem,ll->caLam,ll>batDau,ll->ketThuc);

                                dem++;

                                ll=ll->next;

                        }

                        int test;

                        ll=nv[i].LichLam;

                        LichLam \*temp=NULL;

                        dem=1;

                        printf("Nhap so thu tu ca ma ban muon thay doi; ");

                        scanf("%d",&test);

                        getchar();

                        while(ll!=NULL&&dem<test){

                                temp=ll;

                                ll=ll->next;

                                dem++;

                        }

                        if(ll==NULL){

                                printf("Ca lam khong hop le !\n");

                                return;

                        }

                        int choice;

                        printf("1. Thay doi gio cua ca lam\n");

                        printf("2. Xoa ca lam\n");

                        printf("Lua chon cua ban la: ");

                        scanf("%d",&choice);

                        getchar();

                        if(choice ==1){

                                printf("Cap nhat thoi gian moi:\n");

                                 printf("Gio bat dau ca (00:00): ");

                                fgets(ll->batDau,sizeof(ll->batDau),stdin);

                                ll->batDau[strcspn(ll->batDau,"\n")]='\0';

                                printf("Gio ket thuc ca (00:00): ");

                                fgets(ll->ketThuc,sizeof(ll->ketThuc),stdin);

                                ll->ketThuc[strcspn(ll->ketThuc,"\n")]='\0';

                                printf("Da cap nhat lich thanh cong\n");

                                return;

                        }

                        else if(choice==2){

                                if (temp==NULL) {

                                nv[i].LichLam = ll->next;

                                } else {

                                temp->next = ll->next;

                                }

                                free(ll);

                                printf("Da xoa ca lam o vi tri %d.\n", test);

                                return;

                                }

                        else {

                printf("Lua chon khong hop le.\n");

            }

                t=1;

                }

        }

        if(!t){

                printf("Khong tim thay nhan vien !\n");

        }

}

//ham in danh sach thong tin nhan vien

void inDanhSachNhanVien(NhanVien \*nv,int n){

    int stt=0;

    printf("\n========================================Danhsach======================================\n");

    printf("---------------------------------------------------------------------------------------\n");

    printf("|stt|Ho va ten             |Ma nhan vien|Gioi tinh|Ngay sinh  |Ngay lam   |Luong      |\n");

    for(int i=0;i<n;i++){

       printf("---------------------------------------------------------------------------------------\n");

       printf("|%-3d|%-22s|%-12s|%-9s|%-11s|%-11s|%6.3f/gio|\n",++stt,nv[i].hoTen,nv[i].maNV,nv[i].gioiTinh,nv[i].ngaySinh,nv[i].ngayLam,nv[i].luong);

    }

 printf("---------------------------------------------------------------------------------------\n");

}

//ham tach nam trong chuoi thoi gian

int tachNam(char \*a){

    int x,y,z;

    sscanf(a,"%d/%d/%d",&x,&y,&z);

    return z;

}

//ham tach thang trong chuoi thoi gian

int tachThang(char \*a){

    int x,y,z;

    sscanf(a,"%d/%d/%d",&x,&y,&z);

    return y;

}

//ham tach ngay trong chuoi thoi gian

int tachNgay(char \*a){

    int x,y,z;

    sscanf(a,"%d/%d/%d",&x,&y,&z);

    return x;

}

//ham sap xep nhan vien theo do tuoi

void sapxeptheoDoTuoi(NhanVien \*nv, int m){

    for(int i=0;i<m-1;i++){

       for(int j=i+1;j<m;j++){

           if(tachNam(nv[i].ngaySinh)>=tachNam(nv[j].ngaySinh)){

               NhanVien temp=\*(nv+i);

               \*(nv+i)=\*(nv+j);

               \*(nv+j)=temp;

           }

       }

    }

}

//ham tim ra ca nhan vien da va toi han nghi huu

void timNhanVienSapNghiHuu(NhanVien \*nv,int m){

if(nv==NULL){

        printf("Danh sach rong !");

        return;

}

        int nam;

        printf("nhap nam ban muon chon: ");

        scanf("%d",&nam);

                getchar();

        printf("Cac nhan vien duoc nghi huu vao nam %d la: \n",nam);

        for(int i=0;i<m;i++){

                int namSinh=tachNam(nv[i].ngaySinh);

                if(nam-namSinh>=60){

                printf("%s\n",nv[i].hoTen);

                }

        }

}

//ham tach danh sach theo gioi tinh

void tachDanhSach(NhanVien \*nv ,NhanVien \*\*NvNam, NhanVien \*\*NvNu, int m, int \*dem1,int \*dem2){

        if(nv==NULL){

                printf("Danh sach rong !\n");

                return;

        }

        \*dem1=0;

        \*dem2=0;

        for(int i=0;i<m;i++){

              if(strcasecmp(nv[i].gioiTinh,"nam")==0){

                (\*NvNam)=capPhat(\*dem1+1,\*NvNam);

                if (\*NvNam == NULL) return;

                (\*NvNam)[\*dem1]=nv[i];

                (\*dem1)++;

              }

              else{

                \*NvNu=capPhat(\*dem2+1,\*NvNu);

                if (\*NvNu == NULL) return;

               (\*NvNu)[\*dem2]=nv[i];

                (\*dem2)++;

              }

        }

}

//tinh khoang cach thoi gian

float KhoangCachThoiGian(char \*a, char \*b) {

    int x1, x2, y1, y2;

    sscanf(a, "%d:%d",&x1,&x2);

    sscanf(b, "%d:%d",&y1,&y2);

    int start=x1\*60+x2;

    int end = y1\*60+y2;

    if (end<start) {

        end+=24\*60;

    }

    return (float)(end-start)/60;

}

//ham tinh tong gio lam thuc te hang thang

float tinhGioLam(NhanVien nv,int thang,int nam){

float tongGio=0;

LichLam \*ll=nv.LichLam;

if(tachNam(nv.ngayLam)>nam)

        return 0;

else if(tachThang(nv.ngayLam)>thang&&tachNam(nv.ngayLam)==nam)

        return 0;

if(ll==NULL){

        printf("Nhan vien chua co lich lam !\n");

        return 0;

}

        while(ll!=NULL){

       tongGio+=26\*KhoangCachThoiGian(ll->batDau,ll->ketThuc);

       ll=ll->next;

}

     tangCa \*tc=nv.tangCa;

                while(tc!=NULL){

                        if(tachThang(tc->ngayTangCa)==thang&&tachNam(tc->ngayTangCa)==nam){

                                tongGio+=KhoangCachThoiGian(tc->batDau,tc->ketThuc);

                }

                  tc=tc->next;

        }

        lichNghi \*ln=nv.lichNghi;

                while(ln!=NULL){

                        if(tachThang(ln->ngayNghi)==thang&&tachNam(ln->ngayNghi)==nam){

                                tongGio-=KhoangCachThoiGian(ln->batDau,ln->ketThuc);

                }

                 ln=ln->next;

        }

return tongGio;

}

//ham tinh tong gio lam co dinh hang thang

float tinhGioLamCoDinh(NhanVien nv) {

        if(nv.LichLam==NULL){

                return 0;

        }

    float tongGio = 0;

    LichLam \*ll = nv.LichLam;

    while (ll != NULL) {

        tongGio += 26 \* KhoangCachThoiGian(ll->batDau, ll->ketThuc);

        ll = ll->next;

    }

    return tongGio;

}

//ham tinh tong luong cua moi nhan vien

float tongLuong(NhanVien nv,int thang,int nam){

float tongluong=nv.luong\*tinhGioLam(nv,thang,nam);

return tongluong;

}

//ham in ra danh sach ca lam cho tat ca nhan vien

void InDanhSachCaLam(NhanVien \*nv, int m) {

    if (nv == NULL) {

        printf("Danh sach rong !\n");

        return;

    }

    for (int i = 0; i < m; i++) {

        printf("Ca lam cua nhan vien %s la:\n", nv[i].hoTen);

        if (nv[i].LichLam == NULL) {

            printf("Nhan vien nay chua co lich lam !\n");

            printf("-------------------------------------------------\n");

            continue;

        }

        int dem = 1;

        LichLam \*ll = nv[i].LichLam;

        printf("Lich lam co dinh la:\n");

        while (ll != NULL) {

            printf("[%d]. Ca %s - (%s - %s)\n", dem, ll->caLam, ll->batDau, ll->ketThuc);

            ll = ll->next;

            dem++;

        }

        dem = 1;

        printf("Lich tang ca:\n");

        if (nv[i].tangCa == NULL) {

            printf("Khong co lich tang ca !\n");

        } else {

            tangCa \*tc = nv[i].tangCa;

            while (tc != NULL) {

                printf("[%d]. Ngay %s - Ca %s - (%s - %s)\n", dem, tc->ngayTangCa, tc->caTang, tc->batDau, tc->ketThuc);

                tc = tc->next;

                dem++;

            }

        }

        dem = 1;

        printf("Lich nghi:\n");

        if (nv[i].lichNghi == NULL) {

            printf("Khong co lich nghi !\n");

        } else {

            lichNghi \*ln = nv[i].lichNghi;

            while (ln != NULL) {

                printf("[%d]. Ngay %s - Ca %s - (%s - %s)\n", dem, ln->ngayNghi, ln->caNghi, ln->batDau, ln->ketThuc);

                ln = ln->next;

                dem++;

            }

        }

        printf("-------------------------------------------------\n");

    }

}

//ham in danh sach thong ke hang thang

void inThongKeNhanVien(NhanVien \*ds, int soLuong, int thang, int nam) {

    if (ds==NULL) {

        printf("Danh sach rong !\n");

        return;

    }

    printf("===== DANH SACH THONG KE NHAN VIEN THANG %d =====\n", thang);

    printf("%-5s %-20s %-15s %-25s %-25s %-25s %-15s\n","STT", "Ho Ten", "Ma NV", "Lich Lam Co Dinh","Tang Ca (ngay, gio)", "Nghi (ngay, gio)", "Tong Luong");

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

        NhanVien nv = ds[i];

        printf("%-5d %-20s %-15s ", i+1,nv.hoTen,nv.maNV);

        LichLam \*ll=nv.LichLam;

        if (ll==NULL){

            printf("%-25s ","Chua co");

        }

        else{

            LichLam \*templl=ll;

            printf("[");

            while (templl!=NULL){

                printf("%s(%s-%s)",templl->caLam, templl->batDau, templl->ketThuc);

                if (templl->next != NULL) printf(", ");

                        templl = templl->next;

            }

            printf("]%-5s", "");

        }

        tangCa \*tc=nv.tangCa;

        if (tc==NULL) {

            printf("%-25s ", "Khong co");

        }

        else{

                tangCa \*tempTc = tc;

                printf("[");

                int in = 0;

                while (tempTc != NULL) {

                        if (tachThang(tempTc->ngayTangCa) == thang&&tachNam(tempTc->ngayTangCa)==nam) {

                        float gioTC = KhoangCachThoiGian(tempTc->batDau, tempTc->ketThuc);

                        printf("%s(%.1f gio)", tempTc->ngayTangCa, gioTC);

                        if (tempTc->next != NULL && tempTc->next->ngayTangCa[0] != '\0') printf(", ");

                        in = 1;

                        }

                        tempTc = tempTc->next;

                }

                if (!in) printf("Khong co");

                printf("]%-3s", "");

                }

        lichNghi \*ln = nv.lichNghi;

        if (ln==NULL) {

                printf("%-25s ","Khong co");

        }

        else{

                lichNghi \*templn = ln;

                printf("[");

                int test=0;

                while (templn!=NULL) {

                        if (tachThang(templn->ngayNghi)==thang){

                                float gioNghi = KhoangCachThoiGian(templn->batDau, templn->ketThuc);

                                printf("%s(%.1f gio)", templn->ngayNghi, gioNghi);

                                if (templn->next!=NULL && templn->next->ngayNghi[0] != '\0')

                                        printf(", ");

                                test=1;

                        }

                templn=templn->next;

            }

            if (!test)

                printf("Khong co");

                printf("]%-3s", "");

                }

        float luong=tongLuong(nv, thang,nam);

        printf("%-15.2ftrieu\n",luong);

    }

}

//ham in ra ca lam ma nhan vien ban muon

void timCaLamNhanVien(NhanVien \*nv, int m){

        if(nv==NULL){

                printf("Danh sach rong !\n");

                return;

        }

        char maNv[12];

          printf("Nhap ma nhan vien cua nhan vien ban muon tim: ");

        fgets(maNv,sizeof(maNv),stdin);

        maNv[strcspn(maNv,"\n")]='\0';

        for(int i=0;i<m;i++){

              if(strcmp(nv[i].maNV,maNv)==0){

                        if(nv[i].LichLam==NULL){

                                printf("Nhan vien chua co lich lam !\n");

                                return;

                        }

                int dem=1;

                 LichLam \*ll=nv[i].LichLam;

                 printf("Lich lam co dinh cua nhan vien %s la:\n",nv[i].hoTen);

                        while(ll!=NULL){

                                printf("[%d]. Ca %s - (%s - %s)\n",dem,ll->caLam,ll->batDau,ll->ketThuc);

                                ll=ll->next;

                                dem++;

                                }

                                dem=1;

                                tangCa \*tc=nv[i].tangCa;

                                printf("Lich tang ca:\n");

                        if(tc==NULL){

                                printf("Khong co lich tang ca !");

                                }

                        else{

                                while(tc!=NULL){

                                        printf("[%d]. Ngay %s - Ca %s - (%s - %s)\n",dem,tc->ngayTangCa,tc->caTang,tc->batDau,tc->ketThuc);

                                        tc=tc->next;

                                        dem++;

                                }

                        }

                        dem=1;

                        lichNghi \*ln=nv[i].lichNghi;

                        printf("Lich nghi:\n");

                        if(ln==NULL){

                                printf("Khong co lich nghi !");

                        }

                        else{

                                while(ln!=NULL){

                                        printf("[%d]. Ngay %s - Ca %s - (%s - %s)\n",dem,ln->ngayNghi,ln->caNghi,ln->batDau,ln->ketThuc);

                                        ln=ln->next;

                                        dem++;

                                }

                        }

                }

        }

}

void freedsGioiTinh(NhanVien \*\*nvNam, NhanVien \*\*nvNu) {

    if (\*nvNam != NULL) {

        free(\*nvNam);

        \*nvNam = NULL;

    }

    if (\*nvNu != NULL) {

        free(\*nvNu);

        \*nvNu = NULL;

    }

}

// Ghi thong tin tat ca nhan vien ra file van ban

void ghiThongTinTatCaNhanVien(NhanVien \*Nv, int m, const char \*filename) {

    if (Nv == NULL || m <= 0) {

        printf("Khong co nhan vien de ghi thong tin.\n");

        return;

    }

    FILE \*f = fopen(filename, "w");

    if (!f) {

        perror("Loi mo file");

        return;

    }

    fprintf(f, "===== DANH SACH TOAN BO NHAN VIEN =====\n\n");

    for (int i = 0; i < m; i++) {

        fprintf(f, "----- Nhan vien thu %d -----\n", i + 1);

        fprintf(f, "Ma NV    : %s\n", Nv[i].maNV);

        fprintf(f, "Ho va ten: %s\n", Nv[i].hoTen);

        fprintf(f, "Ngay sinh: %s\n", Nv[i].ngaySinh);

        fprintf(f, "Gioi tinh: %s\n", Nv[i].gioiTinh);

        fprintf(f, "Ngay lam : %s\n", Nv[i].ngayLam);

        fprintf(f, "Luong    : %.2f\n", Nv[i].luong);

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

    }

    fclose(f);

    printf("Da ghi thong tin tat ca %d nhan vien vao file %s\n", m, filename);

}

//Ghi lich lam tat ca nhan vien ra file van ban

void ghiLichLamNhanVien(NhanVien \*nv,int m){

        if(nv==NULL){

                printf("Danh sach nhan vien rong !\n");

                return;

        }

        FILE \*f=fopen("LichLam.txt","w");

        if(f==NULL){

                printf("Khong the mo file !\n");

                return;

        }

        for(int i=0;i<m;i++){

                fprintf(f,"Nhan vien: %s | Ma NV: %s\n", nv[i].hoTen, nv[i].maNV);

                if(nv[i].LichLam==NULL){

                        fprintf(f,"Nhan vien nay chua co lich lam !\n");

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

                        continue;

                }

                int dem=1;

                 LichLam \*ll=nv[i].LichLam;

                 fprintf(f,"Lich lam co dinh la:\n");

                while(ll!=NULL){

                      fprintf(f,"[%d]. Ca %s - (%s - %s)\n",dem,ll->caLam,ll->batDau,ll->ketThuc);

                      ll=ll->next;

                      dem++;

                }

                dem=1;

                tangCa \*tc=nv[i].tangCa;

                fprintf(f,"Lich tang ca:\n");

                if(tc==NULL){

                        fprintf(f,"Khong co lich tang ca !");

                }

                else{

                        while(tc!=NULL){

                                fprintf(f,"[%d]. Ngay %s - Ca %s - (%s -%s)\n",dem,tc->ngayTangCa,tc->caTang,tc->batDau,tc->ketThuc);

                                tc=tc->next;

                                dem++;

                        }

                }

                dem=1;

                lichNghi \*ln=nv[i].lichNghi;

                fprintf(f,"Lich nghi:\n");

                if(ln==NULL){

                       fprintf(f,"Khong co lich nghi !");

                }

                else{

                        while(ln!=NULL){

                                fprintf(f,"[%d]. Ngay %s - Ca %s - (%s -%s)\n",dem,ln->ngayNghi,ln->caNghi,ln->batDau,ln->ketThuc);

                                ln=ln->next;

                                dem++;

                        }

                }

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

        }

        fclose(f);

}

//ham giai phong bo nho cua con tro lich lam

void freeLichLam(NhanVien \*nv, int m) {

    for (int i = 0; i < m; i++) {

        LichLam \*ll = nv[i].LichLam;

        while (ll != NULL) {

            LichLam \*llTemp = ll;

            ll = ll->next;

            free(llTemp);

        }

    }

}

void freeLichNghi(NhanVien \*nv, int m) {

    if (nv == NULL) return;

    for (int i = 0; i < m; i++) {

        lichNghi \*ln = nv[i].lichNghi;

        while (ln != NULL) {

            lichNghi \*temp = ln;

            ln = ln->next;

            free(temp);

        }

        nv[i].lichNghi = NULL;

    }

}

void freeTangCa(NhanVien \*nv, int m) {

    if (nv == NULL) return;

    for (int i = 0; i < m; i++) {

        tangCa \*tc = nv[i].tangCa;

        while (tc != NULL) {

            tangCa \*temp = tc;

            tc = tc->next;

            free(temp);

        }

        nv[i].tangCa = NULL;

    }

}

void MENU(){

        printf("\n=================================================================\n");

        printf("|                     MENU QUAN LY NHAN VIEN                    |\n");

        printf("=================================================================\n");

        printf("|---------------------------NHAN SU-----------------------------|\n");

        printf("| 1. Them nhan vien moi                                         |\n");

        printf("| 2. Cap nhat thong tin nhan vien theo Ma NV                    |\n");

        printf("| 3. Xoa nhan vien theo Ma NV                                   |\n");

        printf("| 4. Sap xep nhan vien theo tuoi                                |\n");

        printf("| 5. Tach danh sach theo gioi tinh                              |\n");

        printf("| 6. Tim nhan vien sap nghi huu                                 |\n");

        printf("| 7. Hien thi danh sach nhan vien                               |\n");

        printf("| 8. Thong ke gio lam va luong theo thang                       |\n");

        printf("| 9. Ghi thong tin tat ca nhan vien ra file                     |\n");

        printf("|---------------------------LICH LAM----------------------------|\n");

        printf("| 10. Nhap lich lam co dinh cho nhan vien                       |\n");

        printf("| 11. Them ca lam co dinh moi cho mot nhan vien                 |\n");

        printf("| 12. Cap nhat/Xoa lich lam co dinh cho nhan vien               |\n");

        printf("| 13. Dang ky lich tang ca cho nhan vien                        |\n");

        printf("| 14. Dang ky lich nghi cho nhan vien                           |\n");

        printf("| 15. In chi tiet lich lam tat ca nhan vien                     |\n");

        printf("| 16. In chi tiet lich lam mot nhan vien                        |\n");

        printf("| 17. Ghi file lich lam cua tat ca nhan vien                    |\n");

        printf("|---------------------------------------------------------------|\n");

        printf("=================================================================\n");

}

int main() {

    NhanVien \*Nv = NULL;

    int m = 0;

    NhanVien \*NvNam = NULL, \*NvNu = NULL;

    int demNam = 0, demNu = 0;

    int choice;

    do {

        MENU();

        printf("Nhap lua chon: ");

        scanf("%d", &choice);

        getchar();

        switch (choice) {

            case 1: {

                int n;

                do {

                    printf("Nhap so luong NV muon them: ");

                    scanf("%d", &n);

                    getchar();

                } while (n <= 0);

                themNhanVien(&Nv, n, m);

                m += n;

                break;

            }

            case 2:

                capNhatThongTinNhanVien(Nv, m);

                break;

            case 3:

                xoaNhanVien(Nv, &m);

                if (m == 0) {

                free(Nv);

                Nv = NULL;

                 }

                break;

            case 4:

                sapxeptheoDoTuoi(Nv, m);

                inDanhSachNhanVien(Nv, m);

                break;

            case 5:

                freedsGioiTinh(&NvNam, &NvNu);

                demNam = demNu = 0;

                tachDanhSach(Nv, &NvNam, &NvNu, m, &demNam, &demNu);

                if (demNam)

                        inDanhSachNhanVien(NvNam, demNam);

                if (demNu)

                        inDanhSachNhanVien(NvNu, demNu);

                break;

            case 6:

                timNhanVienSapNghiHuu(Nv, m);

                break;

            case 7:

                inDanhSachNhanVien(Nv,m);

                break;

            case 8:{

                int thang;

                do {

                    printf("Nhap thang (1-12): ");

                    scanf("%d", &thang);

                    getchar();

                } while (thang < 1 || thang > 12);

                int nam;

                printf("Nhap nam: ");

                    scanf("%d", &nam);

                    getchar();

                inThongKeNhanVien(Nv, m, thang,nam);

                break;

            }

            case 9:

                ghiThongTinTatCaNhanVien(Nv, m, "tatca\_nv.txt");

                printf("Da ghi ra file tatca\_nv.txt\n");

                break;

            case 10:

                nhapLichLam(Nv, m);

                break;

            case 11:

                themCaLamTungNhanVien(Nv, m);

                break;

            case 12:

                capNhatLichLam(Nv, m);

                break;

            case 13:

                nhapLichTangCa(Nv, m);

                break;

            case 14:

                nhapLichNghi(Nv, m);

                break;

            case 15:

                InDanhSachCaLam(Nv, m);

                break;

            case 16:

                timCaLamNhanVien(Nv, m);

                break;

            case 17:

                ghiLichLamNhanVien(Nv, m);

                printf("Da ghi lich lam vao file LichLam.txt\n");

break;

            case 0:

                printf("Dang thoat chuong trinh...\n");

                break;

            default:

                printf("Lua chon khong hop le.\n");

        }

    } while (choice != 0);

    if (Nv){

        freeLichNghi(Nv,m);

        freeTangCa(Nv,m);

        freeLichLam(Nv, m);

        free(Nv);

    }

        freedsGioiTinh(&NvNam, &NvNu);

    printf("Chuong trinh ket thuc.\n");

    return 0;

}

## 2.2 Cấu trúc bài toán

### 2.2.1 Cấu trúc dữ liệu

#include <stdio.h>

#include <string.h>

#include <stdlib.h>

#include <math.h>

typedef struct tangCa{

char ngayTangCa[12];

char caTang[10];

char batDau[7];

char ketThuc[7];

struct tangCa \*next;

}tangCa;

typedef struct lichNghi{

char ngayNghi[12];

char caNghi[10];

char batDau[7];

char ketThuc[7];

struct lichNghi \*next;

}lichNghi;

typedef struct LichLam{

char caLam[10];

char batDau[7];

char ketThuc[7];

struct LichLam \*next;

}LichLam;

typedef struct NhanVien{

    char hoTen[50];

    char ngayLam[12];

    char gioiTinh[5];

    char ngaySinh[12];

    float luong;

    char maNV[11];

    LichLam \*LichLam;

    tangCa \*tangCa;

    lichNghi \*lichNghi;

}NhanVien;

### 2.2.2 Các hàm chính

- Các hàm liên quan đến thông tin cơ bản của nhân viên: sapXepTheoDoTuoi(), themNhanVien(), capNhatThongTinNhanVien(), xoaNhanVien(), inThongKeNhanVien(), tachDanhSach(), inDanhSachNhanVien(), ghiThongTinTatCaNhanVien().

-Các hàm liên quan đến lịch làm của nhân viên: themCaLamTungNhanVien(), nhapLichLam(), capNhatLichLam(),themCaLamTungNhanVien(), timCaLamNhanVien()

,inDanhSachCaLam(), nhapLichTangCa(), ghiLichLamNhanVien().

## 2.3 Các chức năng chính

### **2.3.1 Thêm nhân viên mới**

- Cho phép người dùng nhập thông tin cá nhân như họ tên, ngày sinh, giới tính, ngày bắt đầu làm việc, mã nhân viên và mức lương theo giờ.

- Bộ nhớ được cấp phát lại để mở rộng danh sách chứa nhân viên.

- Hệ thống đảm bảo mỗi nhân viên đều có cấu trúc rỗng ban đầu cho lịch làm, tăng ca và nghỉ.

### **2.3.2 Cập nhật thông tin nhân viên**

- Cho phép cập nhật toàn bộ thông tin nhân viên, xác định thông qua mã nhân viên duy nhất.

- Người dùng có thể thay đổi họ tên, ngày sinh, lương, giới tính, ngày làm,... một cách linh hoạt.

### **2.3.3 Xóa nhân viên**

- Tìm kiếm nhân viên dựa trên mã số và xóa khỏi danh sách hiện tại.

- Giải phóng toàn bộ vùng nhớ liên quan như lịch làm việc, lịch tăng ca và lịch nghỉ nếu có.

- Dồn danh sách về sau để giữ thứ tự liên tục.

### **2.3.4 Sắp xếp nhân viên theo độ tuổi**

- So sánh năm sinh của nhân viên để sắp xếp danh sách theo độ tuổi từ lớn đến bé.

- Giúp người dùng dễ dàng theo dõi hoặc phân loại nhân sự theo thâm niên.

### **2.3.5 Tách danh sách theo giới tính**

- Phân loại danh sách nhân viên thành hai nhóm: nam và nữ.

- Ghi nhận lại hai danh sách riêng biệt giúp dễ thống kê hoặc xử lý theo giới tính.

- Có thể in ra từng danh sách riêng để kiểm tra.

### **2.3.6 Tìm nhân viên sắp nghỉ hưu**

- Nhập năm kiểm tra và tính tuổi nhân viên dựa vào năm sinh.

- In ra danh sách những người đã hoặc sắp đến tuổi nghỉ hưu (>= 60 tuổi).

- Hữu ích cho công tác điều động, thay thế hoặc lên kế hoạch nhân sự.

### **2.3.7 Hiển thị danh sách nhân viên**

- In toàn bộ danh sách nhân viên dưới dạng bảng có định dạng rõ ràng.

- Bao gồm các trường thông tin như mã NV, họ tên, ngày sinh, giới tính, lương,…

### **2.3.8 Thống kê giờ làm và lương theo tháng**

- Tính tổng số giờ làm của nhân viên trong tháng, bao gồm giờ làm cố định, tăng ca và trừ thời gian nghỉ.

- Tính tổng lương theo giờ làm và đơn giá lương của từng người.

- Hiển thị báo cáo thống kê chi tiết cho từng nhân viên.

### **2.3.9 Ghi thông tin tất cả nhân viên ra file**

- Ghi toàn bộ thông tin cá nhân của nhân viên vào một file văn bản (tatca\_nv.txt).

- Dùng làm bản sao lưu, xuất báo cáo hoặc kiểm tra sau này.

### **2.3.10 Nhập lịch làm cố định cho nhân viên**

- Nhập ca làm chính (sáng, chiều, tối) với giờ bắt đầu và kết thúc.

- Mỗi nhân viên có thể có nhiều ca làm khác nhau.

- Ca làm được lưu bằng danh sách liên kết để dễ quản lý.

### **2.3.11 Thêm ca làm mới cho một nhân viên**

- Cho phép tìm theo tên và thêm ca làm mới cho người đó.

- Ca làm mới sẽ được nối vào cuối danh sách ca làm hiện có.

- Hữu ích khi thay đổi lịch làm hoặc thêm ca linh hoạt.

### **2.3.12 Cập nhật hoặc xóa ca làm cố định**

- Hiển thị danh sách ca làm hiện tại để chọn sửa hoặc xóa.

- Cho phép người dùng cập nhật thời gian ca hoặc xóa ca khỏi danh sách.

- Tránh trùng lặp và giúp cập nhật lịch làm nhanh chóng.

### **2.3.13 Đăng ký lịch tăng ca**

- Nhập ngày, ca, giờ bắt đầu và kết thúc cho ca làm ngoài giờ.

- Chỉ cho phép đăng ký nếu nhân viên đã có lịch làm cố định.

- Giúp theo dõi và cộng giờ tăng ca vào thống kê.

### **2.3.14 Đăng ký lịch nghỉ**

- Nhập ngày nghỉ và khoảng thời gian xin nghỉ (theo ca, giờ cụ thể).

- Giảm số giờ làm tương ứng trong thống kê lương tháng.

- Phục vụ cho mục đích quản lý phép năm hoặc nghỉ đột xuất.

### **2.3.15 In lịch làm của tất cả nhân viên**

- Hiển thị lịch làm, lịch tăng ca và nghỉ của tất cả nhân viên.

- Mỗi loại lịch được in riêng biệt để dễ theo dõi. Được chia rõ ràng theo từng người trong danh sách.

### **2.3.16 In lịch làm của một nhân viên cụ thể**

- Tìm nhân viên theo mã số và hiển thị toàn bộ lịch làm, tăng ca, nghỉ.

Giúp kiểm tra lịch trình làm việc chi tiết của từng cá nhân.

### **2.3.17 Ghi toàn bộ lịch làm ra file**

- Ghi tất cả lịch làm cố định, tăng ca và nghỉ của mỗi nhân viên ra file LichLam.txt.

- Phục vụ mục đích lưu trữ hoặc in báo cáo lịch làm.

## 2.4 Một số hình ảnh khi chạy chương trình

### 2.4.1 Menu khi chạy chương trình



Hình 2. 1 Hình ảnh menu khi chạy chương trình

### C:\Users\Lenovo\Pictures\Screenshots\Screenshot 2025-05-23 220341.png2.4.2 Danh sách thông tin tất cả nhân viên

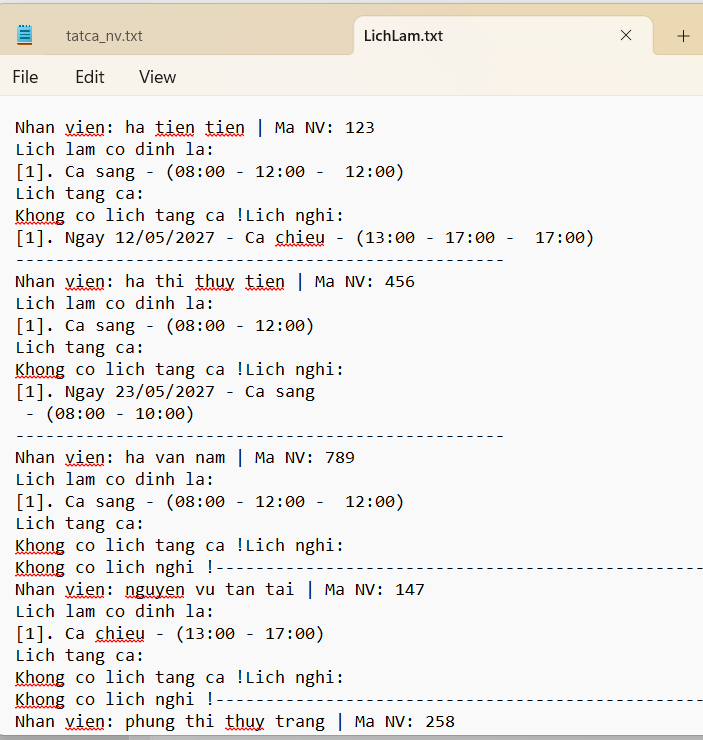
Hình 2. 2 Hình ảnh khi thông tin tất cả nhân viên được in ra danh sách

Hình 2

### 2.4.3 Hình ảnh khi thông tin nhân viên dược ghi trong fileC:\Users\Lenovo\Pictures\Screenshots\Screenshot 2025-05-24 083840.png

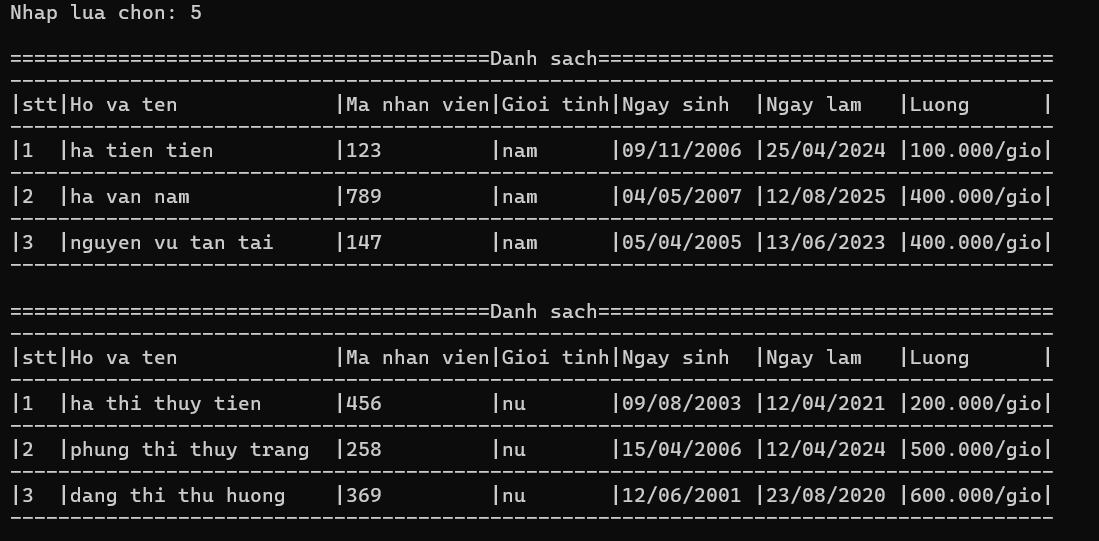
Hình 2. 3 Hình ảnh khi ghi thông tin tất cả nhân viên vào file

### 2.4.4 File khi lịch làm nhân viên được ghi



Hình 2. 4 Hình ảnh thông tin lịch làm khi được ghi vào file

### 2.4.5 Danh sách nhân viên được chia ra theo giới tính



Hình 2. 5 Hình ảnh danh sách khi được tách theo giới tính

# CHƯƠNG 3: KẾT LUẬN

Thông qua quá trình xây dựng và triển khai chương trình, em đã vận dụng linh hoạt giữa **mảng động** và **danh sách liên kết đơn** để quản lý thông tin nhân viên, lịch làm việc, tăng ca và nghỉ phép một cách linh hoạt. Việc tổ chức dữ liệu theo mô hình cấu trúc lồng nhau (struct lồng struct) kết hợp với các hàm xử lý nhập/xuất, tính toán giờ làm và lương đã giúp mô phỏng khá sát với thực tế công việc quản lý nhân sự trong một doanh nghiệp nhỏ.

Tuy nhiên, chương trình hiện tại vẫn còn là một phiên bản cơ bản, mới ở mức ý tưởng và đang trong quá trình cải thiện. Một số phần vẫn có thể được tối ưu thêm như: Bổ sung chức năng kiểm tra dữ liệu đầu vào (hợp lệ hóa ngày tháng, giờ làm…), tăng khả năng tái sử dụng mã lệnh, phân tách chương trình thành nhiều file để dễ bảo trì.

Bên cạnh đó, em rất mong nhận được những nhận xét, góp ý từ cô giáo để cải thiệnkỹ năng lập trình, tư duy giải quyết vấn đề và từng bước nâng cấp chương trình trở thành một ứng dụng quản lý nhân viên hiệu quả, tối ưu hơn.

# TÀI LIỆU THAM KHẢO

1. <https://200lab.io/blog/lap-trinh-c-co-ban>(05/11/2025)”Lập trình C cơ bản – 200lab”
2. <https://vietjack.com/lap_trinh_c/>(05/11/2025)”Lập trình C cơ bản, nâng cao”
3. <https://blog.28tech.com.vn/c%20>05/11/2025)”Lập trình C 28Tech”