# **Struct trong C**

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

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

Nó hoạt động như một template trong C++ và lớp trong Java. Bạn có thể khai báo các loại phần tử khác nhau trong đó.

Định nghĩa cấu trúc trong C

Từ khóa **struct** được sử dụng để xác định cấu trúc. Hãy xem cú pháp để định nghĩa cấu trúc trong C.

[?](https://viettuts.vn/lap-trinh-c/structure-trong-c)

|  |  |
| --- | --- |
| 1  2  3  4  5  6 | **struct** structure\_name {      data\_type member1;      data\_type member2;      ...      data\_type memeberN;  }; |

Ví dụ định nghĩa cấu trúc của Employee trong C.

[?](https://viettuts.vn/lap-trinh-c/structure-trong-c)

|  |  |
| --- | --- |
| 1  2  3  4  5 | **struct** employee {  **int** id;  **char** name[50];  **float** salary;  }; |

## Khai báo biến cấu trúc

Có hai cách để khai báo biến cấu trúc:

1. Sử dụng từ khóa struct trong hàm main().
2. Khai báo biến tại thời điểm định nghĩa cấu trúc.

### **1. Cách 1**

Khai báo biến cấu trúc bên trong hàm main(), ví dụ:

[?](https://viettuts.vn/lap-trinh-c/structure-trong-c)

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9 | **struct** employee {  **int** id;  **char** name[50];  **float** salary;  };    **int** main() {  **struct** employee e1, e2;  } |

### **1. Cách 2**

Khai báo biến cấu trúc tại thời điểm định nghĩa cấu trúc, ví dụ:

[?](https://viettuts.vn/lap-trinh-c/structure-trong-c)

|  |  |
| --- | --- |
| 1  2  3  4  5 | **struct** employee {  **int** id;  **char** name[50];  **float** salary;  } e1, e2; |

### **Cách tiếp cận nào tốt hơn?**

Khi số lượng biến không cố định, sử dụng phương pháp tiếp cận thứ nhất. Nó cung cấp cho bạn sự linh hoạt để khai báo biến cấu trúc nhiều lần. Khi số lượng biến được cố định, sử dụng phương pháp thứ 2.

## Truy cập các thành viên của cấu trúc

Có hai cách để truy cập vào các thành viên cấu trúc:

1. Bởi . (thành viên hoặc toán tử chấm).
2. Bởi -< (toán tử con trỏ cấu trúc).

Ví dụ:

[?](https://viettuts.vn/lap-trinh-c/structure-trong-c)

|  |  |
| --- | --- |
| 1  2  3 | e1.id = 17;  e1.name = "Vinh";  e1.salary = 1000; |

## Ví dụ cấu trúc (structure) trong C

**Ví dụ 1**: lưu trữ thông tin của một employee.

#include<stdio.h>

#include <string.h>

**struct** employee {

**int** id;

**char** name[50];

**float** salary;

} e1; // khai bao bien e1

**int** main() {

    // luu tru thong tin employee

    e1.id = 17;

**strcpy**(e1.name, "Vinh Tran"); // sao chep string thanh mang char

    e1.salary = 1000;

    // hien thi thong tin employee ra man hinh

**printf**("employee 1 id : %d\n", e1.id);

**printf**("employee 1 name : %s\n", e1.name);

**printf**("employee 1 salary : %f\n", e1.salary);

**return** 0;

}

Kết quả:

employee 1 id : 17

employee 1 name : Vinh Tran

employee 1 salary : 1000.000000

**Ví dụ 2**: lưu trữ thông tin của nhiều employee.

#include<stdio.h>

#include <string.h>

**struct** employee {

**int** id;

**char** name[50];

**float** salary;

};

**int** main() {

 // khai bao employee e1, e2

**struct** employee e1, e2;

    // luu tru thong tin employee e1

    e1.id = 17;

**strcpy**(e1.name, "Vinh Tran"); // sao chep string thanh mang char

    e1.salary = 1000;

    // luu tru thong tin employee e2

    e2.id = 37;

**strcpy**(e2.name, "Quy Pham"); // sao chep string thanh mang char

    e2.salary = 1700;

    // hien thi thong tin employee e1 ra man hinh

**printf**("employee 1 id : %d\n", e1.id);

**printf**("employee 1 name : %s\n", e1.name);

**printf**("employee 1 salary : %f\n", e1.salary);

    // hien thi thong tin employee e2 ra man hinh

**printf**("employee 2 id : %d\n", e2.id);

**printf**("employee 2 name : %s\n", e2.name);

**printf**("employee 2 salary : %f\n", e2.salary);

**return** 0;

}

Kết quả:

employee 1 id : 17

employee 1 name : Vinh Tran

employee 1 salary : 1000.000000

employee 2 id : 37

employee 2 name : Quy Pham

employee 2 salary : 1700.000000

## Bài tập C - Nhập và hiển thị sinh viên bởi sử dụng Struct trong C

#include<stdio.h>

#define SIZE 50

**struct** student {

**char** name[30];

**int** id;

**int** marks[3];

};

**int** main() {

**int** i, j, max, count, total, n, a[SIZE], ni;

**struct** student st[SIZE];

**printf**("Nhap so luong sinh vien: ");

**scanf**("%d", &n);

**printf**("----- Nhap thong tin sinh vien -----\n");

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

**printf**("Nhap ten cho sinh vien %d: ", i + 1);

**scanf**("%s", &st[i].name);

**printf**("Nhap ma sinh vien cho sinh vien %d: ", i + 1);

**scanf**("%d", &st[i].id);

    }

**printf**("----- Nhap diem thi cho sinh vien -----\n");

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

**for** (j = 0; j <= 2; j++) {

**printf**("Nhap diem thi %d cho sinh vien %d: ",

                    j + 1, st[i].id);

**scanf**("%d", &st[i].marks[j]);

        }

    }

**printf**("----- Tong diem cua sinh vien -----");

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

        total = 0;

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

            total = total + st[i].marks[j];

        }

**printf**("\nTong diem cua sinh vien %s la: %d",

                st[i].name, total);

        a[i] = total;

    }

**printf**("\n----- Liet ke mon hoc co diem cao nhat -----");

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

        max = 0;

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

**if** (max < st[i].marks[j]) {

                max = st[i].marks[j];

                ni = i;

            }

        }

**printf**("\nSinh vien %s dat diem cao nhat = %d voi mon hoc: %d",

                st[ni].name, max, j + 1);

    }

**printf**("\n----- Sinh vien co tong diem cao nhat -----");

    max = 0;

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

**if** (max < a[i]) {

            max = a[i];

            ni = i;

        }

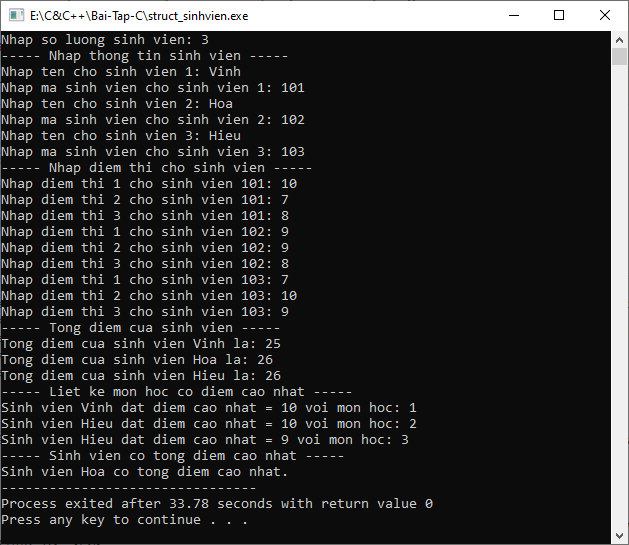
    }

**printf**("\nSinh vien %s co tong diem cao nhat.", st[ni].name);

**return**(0);

}

Chạy chương trình C trên cho kết quả như sau:



# **Structure lồng nhau trong C**

Như bạn đã biết structure trong C (cấu trúc trong C) là một **kiểu dữ liệu** người dùng tự định nghĩa (user defined datatype). Nên bạn hoàn toàn có thể sử dụng structure bên trong structure khác, nó được biết đến như **structure lồng nhau trong C**.

Có 2 cách để định nghĩa cấu trúc lồng nhau trong C:

1. Theo cấu trúc riêng biệt.
2. Theo cấu trúc nhúng.

## 1. Cấu trúc riêng biệt

Ví dụ, chúng ta tạo ra 2 cấu trúc và cấu trúc phụ thuộc được sử dụng bên trong cấu trúc chính như một thành viên.

**struct** Date {

**int** day;

**int** month;

**int** year;

};

**struct** Employee {

**int** id;

**char** name[20];

**struct** Date birthday;

} emp1;

## 2. Cấu trúc nhúng.

Cấu trúc nhúng là viêc định nghĩa cấu trúc bên trong một cấu trúc khác và nó cũng là một trường hợp của cấu trúc (structure) lồng nhau trong C.

Ví dụ:

**struct** Employee {

**int** id;

**char** name[20];

**struct** Date {

**int** day;

**int** month;

**int** year;

    } birthday;

} emp1;

## Truy cập cấu trúc lồng nhau

Bạn có thể truy cập các thành viên của cấu trúc lồng nhau bởi Outer\_Structure.Nested\_Structure.member như dưới đây:

emp1.birthday.day

emp1.birthday.month

emp1.birthday.year

## Ví dụ về cấu trúc lồng nhau trong C

#include <stdio.h>

#include <string.h>

/\*\*

 \* dinh nghia cau truc Employee

 \*/

**struct** Employee {

**int** id;

**char** name[20];

**struct** Date {

**int** day;

**int** month;

**int** year;

    } birthday;

} e1;

/\*\*

 \* ham main

 \*/

**int** main() {

    // luu tru thong tin employee

    e1.id = 101;

**strcpy**(e1.name, "Vinh Tran"); // chuyen doi chuoi thanh mang char

    e1.birthday.day = 10;

    e1.birthday.month = 11;

    e1.birthday.year = 2001;

    // hien thi thong tin sinh vien ra man hinh

**printf**("employee id: %d\n", e1.id);

**printf**("employee name: %s\n", e1.name);

**printf**("employee birthday (dd/mm/yyyy): %d/%d/%d\n", e1.birthday.day,

            e1.birthday.month, e1.birthday.year);

**return** 0;

}

Kết quả:

employee id: 101

employee name: Vinh Tran

employee birthday (dd/mm/yyyy): 10/11/2001

## Bài tập C - Tính kích cỡ của Struct trong C

Để tìm kích cỡ của Struct chúng ta sử dụng toán tử **sizeof()** trong C. **sizeof** là một toán tử chứ KHÔNG phải một hàm trong C. Toán tử **sizeof** này nhận một tham số là bất kỳ biến nào và trả về kích cỡ của biến đó.

Giả sử một Struct sinh viên gồm các thuộc tính: name, id, marks

**Kích cỡ của Struct = sizeof(name) + sizeof(id) + sizeof(marks)**

Dưới đây là chương trình C để giải bài tính kích cỡ của Struct trong C:

#include<stdio.h>

**struct** student {

**char** name[30];

**int** id;

**int** marks[3];

};

**int** main() {

**int** size;

**struct** student s;

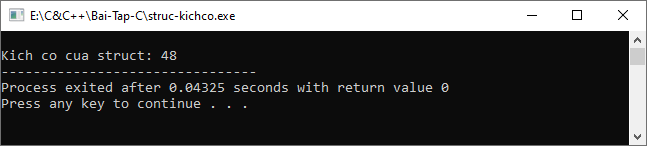
    size = **sizeof**(s);

**printf**("\nKich co cua struct: %d", size);

**return**(0);

}

Chạy chương trình C trên cho kết quả như sau:



Giả sử một giải đấu thể thao sinh viên đấu vòng tròn có 10 sinh viên tham gia. Mỗi sinh viên thuộc một đội khác nhau và sinh viên có các thông tin: tên đội (tname), tên sinh viên (pname), điểm số (avg). Khai báo một mảng các sinh viên. Yêu cầu là hãy sắp xếp các sinh viên này dựa trên bất kỳ tiêu chí nào ở trên (ví dụ sắp xếp theo tên đội, tên sinh viên hoặc điểm số đạt được).

#include<stdio.h>

#include<string.h>

struct tennis {

char pname[20];

char tname[20];

int avg;

} player[3], temp;

int main() {

int i, j, n;

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

printf("\nNhap ten nguoi choi: ");

scanf("%s", &player[i].pname);

printf("\nNhap ten truong: ");

scanf("%s", &player[i].tname);

printf("\nNhap diem so: ");

scanf("%d", &player[i].avg);

printf("\n");

}

n = 3;

for (i = 1; i < n; i++)

for (j = 0; j < n - i; j++) {

if (strcmp(player[j].tname, player[j + 1].tname) > 0) {

temp = player[j];

player[j] = player[j + 1];

player[j + 1] = temp;

}

}

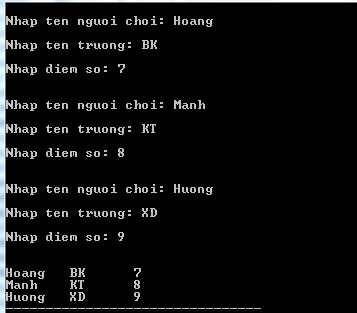
for (i = 0; i < n; i++) {

printf("\n%s\t%s\t%d",player[i].pname,player[i].tname,player[i].avg);

}

return(0);

}



Yêu cầu bài tập C là sử dụng Struct để nhập thông tin các thành phố khác nhau. Mỗi thành phố (state) có: tên thành phố (name), tổng thu nhập (income), dân số (population), tỉ lệ biết đọc viết (literacyRate). Sau đó hiển thị thành phố có tổng thu nhập cao nhất, có tỉ lệ biết đọc viết cao nhất.

#include<stdio.h>

#define M 50

struct state {

char name[50];

long int population;

float literacyRate;

float income;

} st[M]; /\* khai bao mang cac struct \*/

int main() {

int i, n, ml, mi, maximumLiteracyRate, maximumIncome;

float rate;

ml = mi = -1;

maximumLiteracyRate = maximumIncome = 0;

printf("Nhap so thanh pho:");

scanf("%d", &n);

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

printf("\nNhap chi tiet thanh pho thu %d: ", i);

printf("\nNhap ten thanh pho: ");

scanf("%s", &st[i].name);

printf("\nNhap tong so dan: ");

scanf("%ld", &st[i].population);

printf("\nNhap ti le biet doc viet: ");

scanf("%f", &rate);

st[i].literacyRate = rate;

printf("\nNhap tong thu nhap: ");

scanf("%f", &st[i].income);

}

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

if (st[i].literacyRate >= maximumLiteracyRate) {

maximumLiteracyRate = st[i].literacyRate;

ml++;

}

if (st[i].income > maximumIncome) {

maximumIncome = st[i].income;

mi++;

}

}

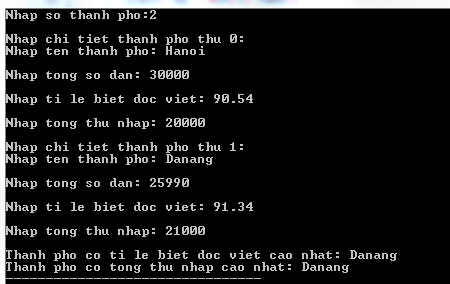
printf("\nThanh pho co ti le biet doc viet cao nhat: %s", st[ml].name);

printf("\nThanh pho co tong thu nhap cao nhat: %s", st[mi].name);

return (0);

}

Biên dịch chương trình C trên sẽ cho kết quả:



Giải thích chương trình C trên

Đầu tiên, bạn sử dụng vòng lặp để nhập thông tin cho từng thành phố:

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

printf("\nNhap chi tiet thanh pho thu %d: ", i);

printf("\nNhap ten thanh pho: ");

scanf("%s", &st[i].name);

printf("\nNhap tong so dan: ");

scanf("%ld", &st[i].population);

printf("\nNhap ti le biet doc viet: ");

scanf("%f", &rate);

st[i].literacyRate = rate;

printf("\nNhap tong thu nhap: ");

scanf("%f", &st[i].income);

}

Sau đó sử dụng vòng lặp kết hợp với lệnh **IF** trong C để tìm các giá trị lớn nhất.

if (st[i].literacyRate >= maximumLiteracyRate) {

maximumLiteracyRate = st[i].literacyRate;

ml++;

}

if (st[i].income > maximumIncome) {

maximumIncome = st[i].income;

mi++;

}

Đây là chương trình C cơ bản minh họa cách đọc ghi file bởi sử dụng các hàm trong C.

* Sử dụng hàm **fopen()** để mở file trong các chế độ khác nhau
* Sử dụng hàm **fwrite()** để ghi file
* Sử dụng hàm **fread()** để đọc file
* Sử dụng hàm **fclose()** để đóng file

#include<stdio.h>

struct Student {

int roll;

char name[12];

int mark;

} s1 = { 101, "Nguyen Hoang Nam", 9 };

int main() {

FILE \*fp;

struct Student s2;

//ghi thong tin sinh vien s1 vao file

fp = fopen("baitapc.txt", "w");

fwrite(&s1, sizeof(s1), 1, fp);

fclose(fp);

fp = fopen("baitapc.txt", "r");

fread(&s2, sizeof(s2), 1, fp);

fclose(fp);

printf("Ma sv: %d", s2.roll);

printf("\nHo ten: %s", s2.name);

printf("\nDiem thi: %d", s2.mark);

return (0);

}

Biên dịch và chạy chương trình C trên sẽ cho kết quả sau:

Đọc ghi file trong C

Đây là bài tập C cơ bản giúp bạn hiểu cách mở file, đọc ghi file bởi sử dụng các hàm **fopen(), fgetc()** và **fputc()** trong C.

Dưới đây là chương trình C để giải bài tập sao chép nội dung từ file này sang file khác trong C:

#include<stdio.h>

#include<process.h>

int main() {

FILE \*fp1, \*fp2;

char a;

fp1 = fopen("baitapc.txt", "r");

if (fp1 == NULL) {

puts("Khong the mo file nay");

exit(1);

}

fp2 = fopen("baitapc1.txt", "w");

if (fp2 == NULL) {

puts("Khong the mo file nay");

fclose(fp1);

exit(1);

}

do {

a = fgetc(fp1);

fputc(a, fp2);

} while (a != EOF);

return(0);

}

## Giải thích chương trình C trên

Đầu tiên chúng ta mở một file trong chế độ đọc:

* fp1=fopen("baitapc.txt","r");

và mở một file trong chế độ ghi:

* fp2=fopen("baitapc1.txt","w");

Bạn nên kiểm tra xem file đã được mở thành công hay không bởi sử dụng lệnh kiểm tra **NULL**.

* if(fp2==NULL)
* {
* //file khong duoc mo thanh cong
* }
* Nếu mọi thứ đã sẵn sàng: chúng ta bắt đầu đọc các ký tự từ file đã mở bởi sử dụng hàm **fgetc(file)**và sau đó ghi dữ liệu vào trong file khác bởi sử dụng hàm **fputc(file)**.
* Ký tự EOF, là viết tắt của End of File, xác định vị trí cuối cùng của file. Kiểm tra nếu gặp ký tự này thì tiến trình ghi dữ liệu vào file của chúng ta sẽ kết thúc.
* do {
* a = fgetc(fp1);
* fputc(a,fp2);
* }while(a!=EOF);

Kết quả của ví dụ C trên

Giả sử trong **baitapc.txt** chúng ta có nội dung sau:

Xin Chao PhongDepTrai.

Sau khi biên dịch chương trình C trên, bạn sẽ thấy trong **baitapc1.txt** có nội dung sau:

Xin Chao PhongDepTrai.

## Bài tập C: Đọc n ký tự cuối của file

Bài tập này giúp bạn làm quen với các hàm để mở file, đóng file, đọc ghi file trong C.

## Chương trình C

Dưới đây là chương trình C để giải bài tập đọc n ký tự cuối của file trong C:

#include<stdio.h>

#include<stdlib.h>

int main() {

FILE \*fp;

char ch;

int num;

long length;

printf("Nhap gia tri cua num: ");

scanf("%d", &num);

fp = fopen("baitapc.txt", "r");

if (fp == NULL) {

puts("Khong the mo file nay");

exit(1);

}

fseek(fp, 0, SEEK\_END);

length = ftell(fp);

fseek(fp, (length - num), SEEK\_SET);

do {

ch = fgetc(fp);

putchar(ch);

} while (ch != EOF);

fclose(fp);

return(0);

}

## Giải thích chương trình C trên

* Đầu tiên mở file trong trong chế độ đọc:
* fp=fopen("baitapc.txt","r");
* Chọn một giá trị tại đó chúng ta bắt đầu đọc. Chúng ta đang di chuyển con trỏ file tới vị trí cuối cùng bởi sử dụng hàm **fseek()** và đang truyền hằng số **SEEK\_END**.
* fseek(fp,0,SEEK\_END);
* Bây giờ chúng ta ước lượng vị trí hiện tại của con trỏ file:
* length = ftell(fp);
* Hàm ftell() sẽ nói cho chúng ta biết vị trí của con trỏ file.
* Chúng ta cần đọc n ký tự cuối cùng từ file vì thế chúng ta cần di chuyển con trỏ tới ký tự thứ (length-n) bắt đầu từ cuối file. Chúng ta sẽ đọc nội dung file từ vị trí này.

fseek(fp, (length - num), SEEK\_SET);

## Kết quả của ví dụ C trên

Giả sử trong **baitapc.txt** chúng ta có nội dung sau:

Xin Chao PhongDepTrai

Biên dịch chương trình C trên sẽ cho kết quả:

Num: 11

PhongDepTrai

## Bài tập C: Chuyển đổi nội dung file thành chữ hoa và ghi nội dung vào file

Bài tập này giúp bạn làm quen với các hàm để mở file, đóng file, đọc ghi file trong C. Ngoài ra, để chuyển đổi thành chữ hoa, bạn sử dụng hàm **toupper()** trong C.

## Chương trình C

Dưới đây là chương trình C để giải bài tập chuyển đổi nội dung file thành chữ hoa và ghi nội dung vào file trong C:

#include<stdio.h>

#include<process.h>

#include<ctype.h>

int main() {

FILE \*fp1, \*fp2;

char a;

fp1 = fopen("baitapc.txt", "r");

if (fp1 == NULL) {

puts("Khong the mo file nay");

exit(1);

}

fp2 = fopen("baitapc1.txt", "w");

if (fp2 == NULL) {

puts("Khong the mo file nay");

fclose(fp1);

exit(1);

}

do {

a = fgetc(fp1);

a = toupper(a);

fputc(a, fp2);

} while (a != EOF);

return(0);

}

## Giải thích chương trình C trên

* Đầu tiên chúng ta mở một file trong chế độ đọc và mở file khác trong chế độ ghi:
* fp1=fopen("baitapc.txt","r");
* fp2=fopen("baitapc1.txt","w");
* Bây giờ đọc từng ký tự, và sử dụng hàm **toupper()** để chuyển đổi sang chữ hoa.
* do {
* a=fgetc(fp1);
* a=toupper(a);
* fputc(a,fp2);
* }while(a!=EOF);
* Sau khi chuyển đổi sang chữ hoa, chúng ta ghi dữ liệu vào file khác. Bất cứ khi nào gặp ký tự **EOF** (viết tắt của End of File) thì chúng ta kết thúc tiến trình đọc và ghi file.

## Kết quả của ví dụ C trên

Giả sử trong **baitapc.txt** chúng ta có nội dung sau:

Xin Chao PhongDepTrai

Sau khi biên dịch chương trình C trên, bạn sẽ thấy trong **baitapc1.txt** có nội dung sau:

XIN CHAO PHONGDEPTRAI

## Bài tập C: So sánh nội dung của hai file

Bài tập này minh họa cách so sánh nội dung của hai file để kiểm tra xem hai file có đồng nhất hay là không.

Đây cũng là một ví dụ minh họa giúp bạn hiểu cách đọc ghi file trong C. Chúng ta thực hiện theo các bước:

* Đầu tiên, chúng ta khai báo hai con trỏ file cho hai file.
* Mở hai file trong chế độ đọc.
* Bên trong vòng lặp **while** chúng ta đọc từng ký tự của hai file và kiểm tra xem ký tự đó có phải là **EOF** (viết tắt của End of File) hay không
* Kiểm tra xem từng ký tự của hai file có bằng nhau hay không
* Nếu tất cả là bằng nhau thì kết luận nội dung hai file là đồng nhất. Nếu không thì kết luận rằng nội dung hai file là khác nhau.

## Chương trình C

Dưới đây là chương trình C để giải bài tập so sánh nội dung của hai file trong C:

#include<stdio.h>

#include<stdlib.h>

int main() {

FILE \*fp1, \*fp2;

int ch1, ch2;

char fname1[40], fname2[40];

printf("Nhap ten cua file thu nhat: ");

gets(fname1);

printf("Nhap ten cua file thu hai: ");

gets(fname2);

fp1 = fopen(fname1, "r");

fp2 = fopen(fname2, "r");

if (fp1 == NULL) {

printf("Khong the mo %s de doc", fname1);

exit(1);

} else if (fp2 == NULL) {

printf("Khong the mo %s de doc", fname2);

exit(1);

} else {

ch1 = getc(fp1);

ch2 = getc(fp2);

while ((ch1 != EOF) && (ch2 != EOF) && (ch1 == ch2)) {

ch1 = getc(fp1);

ch2 = getc(fp2);

}

if (ch1 == ch2)

printf("\nHai file la dong nhat");

else if (ch1 != ch2)

printf("\nHai file la khong dong nhat");

fclose(fp1);

fclose(fp2);

}

return (0);

}

## Kết quả của ví dụ C trên

Giả sử chúng ta có hai file là **baitapc.txt** và **baitapc1.txt** có nội dung sau:

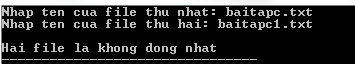
**baitapc.txt**

Xin Chao PhongDepTrai

**baitapc1.txt**

Xin Chao PhongHandSome

Biên dịch và chạy chương trình C trên sẽ cho kết quả:



**Đề bài**: Viết chương trình C sắp xếp mảng Struct trong C.

Yêu cầu: sử dụng Struct để nhập thông tin các thành phố khác nhau. Mỗi thành phố (city) có: tên thành phố (name), tổng thu nhập (income), dân số (population). Sau đó hiển thị thành phố có tổng thu nhập cao nhất và thành phố đông dân nhất.

#include<stdio.h>

#define MAX 50

**struct** city {

**char** name[50];

**long** **int** population;

**float** income;

} ct[MAX]; /\* khai bao mang cac ctruct \*/

**int** main() {

**int** i, n, mp, mi, maxPopulation, maxIncome;

**float** rate;

    mp = mi = -1;

    maxPopulation = maxIncome = 0;

**printf**("Nhap so thanh pho: ");

**scanf**("%d", &n);

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

**printf**("\nNhap chi tiet thanh pho thu %d: \n", i + 1);

**printf**("Nhap ten thanh pho: ");

**scanf**("%s", &ct[i].name);

**printf**("Nhap tong so dan: ");

**scanf**("%ld", &ct[i].population);

**printf**("Nhap tong thu nhap: ");

**scanf**("%f", &ct[i].income);

    }

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

        // tim thanh pho co dan so dong nhat

**if** (ct[i].population >= maxPopulation) {

            maxPopulation = ct[i].population;

            mp++;

        }

        // tim thanh pho co thong thu nhap cao nhat

**if** (ct[i].income > maxIncome) {

            maxIncome = ct[i].income;

            mi++;

        }

    }

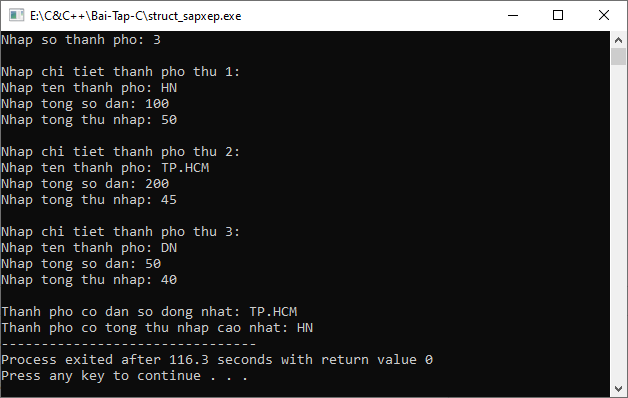
**printf**("\nThanh pho co dan so dong nhat: %s", ct[mp].name);

**printf**("\nThanh pho co tong thu nhap cao nhat: %s", ct[mi].name);

**return** (0);

}

Chạy chương trình C trên cho kết quả như sau:



# **Hàm Math cơ bản trong C**

Lập trình C cho phép chúng ta thực hiện các phép toán thông qua các hàm được định nghĩa trong tập tin tiêu đề <math.h>. File tiêu đề <math.h> chứa các phương thức khác nhau để thực hiện các phép toán như sqrt(), pow(), ceil(), floor(), vv

Hàm Math cơ bản trong C

Có nhiều phương thức trong tập tin tiêu đề math.h. Các phương thức cơ bả và thường được sử dụng của tập tin tiêu đề math.h được đưa ra dưới đây.

|  |  |  |
| --- | --- | --- |
| **No.** | **Phương thức** | **Mô tả** |
| 1. | ceil(number) | làm tròn lên một số đã cho. Nó trả về giá trị số nguyên lớn hơn hoặc bằng với số đã cho. |
| 2. | floor(number) | làm tròn xuống một số đã cho. Nó trả về giá trị số nguyên nhỏ hơn hoặc bằng số đã cho. |
| 3. | sqrt(number) | trả về căn bậc hai của số đã cho. |
| 4. | pow(base, exponent) | trả về số mũ của số đã cho. |
| 5. | abs(number) | trả về giá trị tuyệt đối của số đã cho. |

## Ví dụ hàm Math cơ bản trong C

#include<stdio.h>

#include <math.h>

/\*\*

 \* ham main

 \*/

**int** main() {

**printf**("%f", **ceil**(3.6));

**printf**("\n%f", **ceil**(3.3));

**printf**("\n%f", **floor**(3.6));

**printf**("\n%f", **floor**(3.2));

**printf**("\n%f", **sqrt**(16));

**printf**("\n%f", **sqrt**(7));

**printf**("\n%f", **pow**(2, 4));

**printf**("\n%f", **pow**(3, 3));

**printf**("\n%d", **abs**(-12));

**return** 0;

}

Kết quả:

4.000000

4.000000

3.000000

3.000000

4.000000

2.645751

16.000000

27.000000

12

## Ví dụ hàm trong C

Ví dụ dưới đây là mã nguồn cho một hàm được gọi là max(). Hàm này truyền vào hai tham số num1 và num2 và trả về giá trị lớn nhất giữa hai tham số:

/\* hàm trả về giá trị lớn nhất giữa 2 số \*/

**int** max(**int** num1, **int** num2) {

   /\* khai báo biến local \*/

**int** result;

**if** (num1 > num2)

      result = num1;

**else**

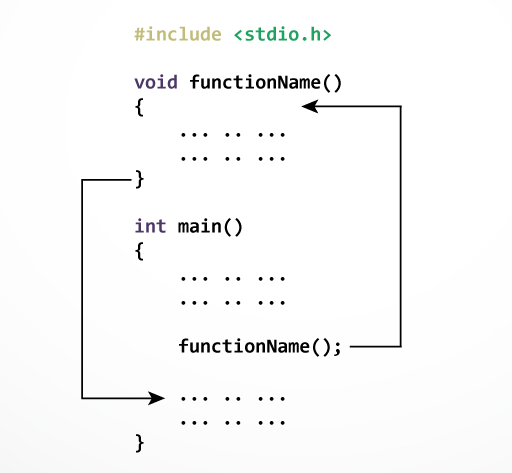
      result = num2;

**return** result;

}

Gọi một hàm trong C

Hàm trong lập trình C hoạt động như thế nào? Hình ảnh sau đây mô tả gọi [một hàm do người dùng định nghĩa](https://viettuts.vn/lap-trinh-c/ham-do-nguoi-dung-dinh-nghia) bên trong hàm **main**():



#include <stdio.h>

/\* khai bao ham \*/

**int** max(**int** num1, **int** num2);

**int** main () {

   /\* dinh nghia bien local \*/

**int** a = 100;

**int** b = 200;

**int** ret;

   /\* goi mot ham de lay gia tri lon nhat \*/

   ret = max(a, b);

**printf**( "Max value is : %d\n", ret );

**return** 0;

}

/\* ham tra ve gia tri lon nhat giua hai so \*/

**int** max(**int** num1, **int** num2) {

   /\* dinh nghia bien local \*/

**int** result;

**if** (num1 > num2)

      result = num1;

**else**

      result = num2;

**return** result;

}

Kết quả:

Max value is : 200

Đề bài

Viết chương trình **giải phương trình bậc 2** trong C. Phương trình bậc 2 có dạng:

## giải phương trình bậc 2 trong C

## Lời giải

Lời giải của chúng tôi sử dụng phương thức **sqrt()** để tính căn bậc 2 của một số trong C. Sử dụng phương thức **scanf("%f", &a)** để đọc số thực từ bàn phím và gán cho biến a.

#include <stdio.h>

/\*\*

 \* giai phuong trinh bac 2: ax2 + bx + c = 0

 \*

 \* @param a: he so bac 2

 \* @param b: he so bac 1

 \* @param c: so hang tu do

 \*/

**void** giaiPTBac2(**float** a, **float** b, **float** c) {

    // kiem tra cac he so

**if** (a == 0) {

**if** (b == 0) {

**printf**("Phuong trinh vo nghiem!");

        } **else** {

**printf**("Phuong trinh co mot nghiem: x = %f", (-c / b));

        }

**return**;

    }

    // tinh delta

**float** delta = b\*b - 4\*a\*c;

**float** x1;

**float** x2;

    // tinh nghiem

**if** (delta > 0) {

        x1 = (**float**) ((-b + **sqrt**(delta)) / (2\*a));

        x2 = (**float**) ((-b - **sqrt**(delta)) / (2\*a));

**printf**("Phuong trinh co 2 nghiem la: x1 = %f va x2 = %f", x1, x2);

    } **else** **if** (delta == 0) {

        x1 = (-b / (2 \* a));

**printf**("Phong trinh co nghiem kep: x1 = x2 = %f", x1);

    } **else** {

**printf**("Phuong trinh vo nghiem!");

    }

}

/\*\*

 \* ham main

 \*/

**int** main() {

**float** a, b, c;

**printf**("Nhap he so bac 2, a = ");

**scanf**("%f", &a);

**printf**("Nhap he so bac 1, b = ");

**scanf**("%f", &b);

**printf**("Nhap so hang tu do, c = ");

**scanf**("%f", &c);

    giaiPTBac2(a, b, c);

**return** 1;

}

Kết quả:

Nhap he so bac 2, a = 2

Nhap he so bac 1, b = 3

Nhap so hang tu do, c = 1

Phuong trinh co 2 nghiem la: x1 = -0.500000 va x2 = -1.000000

# **Bài tập quản lý sinh viên trong C**

**Nội dung chính**

* Bài tập quản lý sinh viên trong C
* [Lời giải](https://viettuts.vn/bai-tap-cpp/bai-tap-quan-ly-sinh-vien-trong-cpp#goto-h2-2)
* [Tạo cấu trúc sinh vien](https://viettuts.vn/bai-tap-cpp/bai-tap-quan-ly-sinh-vien-trong-cpp#goto-h2-3)
* [Định nghĩa các hàm](https://viettuts.vn/bai-tap-cpp/bai-tap-quan-ly-sinh-vien-trong-cpp#goto-h2-4)
* [1. Hàm printLine(int n)](https://viettuts.vn/bai-tap-cpp/bai-tap-quan-ly-sinh-vien-trong-cpp#goto-h2-5)
* [2. Hàm idLonNhat(SV a[], int n)](https://viettuts.vn/bai-tap-cpp/bai-tap-quan-ly-sinh-vien-trong-cpp#goto-h2-6)
* [3. Hàm nhập thông tin sinh viên](https://viettuts.vn/bai-tap-cpp/bai-tap-quan-ly-sinh-vien-trong-cpp#goto-h2-7)
* [4. Hàm cập nhật thông tin sinh viên](https://viettuts.vn/bai-tap-cpp/bai-tap-quan-ly-sinh-vien-trong-cpp#goto-h2-8)
* [5. Hàm xóa sinh viên theo ID](https://viettuts.vn/bai-tap-cpp/bai-tap-quan-ly-sinh-vien-trong-cpp#goto-h2-9)
* [6. Hàm tìm kiếm sinh viên theo tên](https://viettuts.vn/bai-tap-cpp/bai-tap-quan-ly-sinh-vien-trong-cpp#goto-h2-10)
* [7. Hàm tính điểm trung bình và xếp loại học lực](https://viettuts.vn/bai-tap-cpp/bai-tap-quan-ly-sinh-vien-trong-cpp#goto-h2-11)
* [8. Hàm sắp xếp danh sách sinh viên theo điểm trung bình](https://viettuts.vn/bai-tap-cpp/bai-tap-quan-ly-sinh-vien-trong-cpp#goto-h2-12)
* [9. Hàm sắp xếp danh sách sinh viên theo tên](https://viettuts.vn/bai-tap-cpp/bai-tap-quan-ly-sinh-vien-trong-cpp#goto-h2-13)
* [10. Hàm hiện thị danh sách sinh viên ra màn hình](https://viettuts.vn/bai-tap-cpp/bai-tap-quan-ly-sinh-vien-trong-cpp#goto-h2-14)
* [11. Hàm đọc danh sách sinh viên từ file](https://viettuts.vn/bai-tap-cpp/bai-tap-quan-ly-sinh-vien-trong-cpp#goto-h2-15)
* [12. Hàm ghi danh sách sinh viên từ file](https://viettuts.vn/bai-tap-cpp/bai-tap-quan-ly-sinh-vien-trong-cpp#goto-h2-16)
* [13. Hàm main của chương trình quản lý sinh vien trong C/](https://viettuts.vn/bai-tap-cpp/bai-tap-quan-ly-sinh-vien-trong-cpp#goto-h2-17)
* [Cài đặt chương trình quản lý sinh viên trong C](https://viettuts.vn/bai-tap-cpp/bai-tap-quan-ly-sinh-vien-trong-cpp#goto-h2-18)
* [Run chương trình quản lý sinh viên trong C](https://viettuts.vn/bai-tap-cpp/bai-tap-quan-ly-sinh-vien-trong-cpp#goto-h2-19)
* [Download chương trình quản lý sinh viên trong C](https://viettuts.vn/bai-tap-cpp/bai-tap-quan-ly-sinh-vien-trong-cpp#goto-h2-20)

**Đề bài**: Viết chương trình quản lý sinh viên trong C. Mỗi đối tượng sinh viên có các thuộc tính sau: id, tên, giới tính, tuổi, điểm toán, điểm lý, điểm hóa, điểm trung bình và học lực.

Id là mã sinh viên tự động tăng.

Khi khởi tạo, chương trình sẽ đọc file **"student.txt"**để lấy ra danh sách sinh viên đã được lưu trước đó.

Điểm trung bình là giá trị trung bình của 3 môn toán, lý và hóa.

Học lực được tính như sau:

* Giỏi: nếu điểm trung bình lớn hơn hoặc bằng 8.
* Khá: nếu điểm trung bình nhỏ hơn 8 và lớn hơn hoặc bằng 6.5.
* Trung Bình: nếu điểm trung bình nhỏ hơn 6.5 và lớn hơn hoặc bằng 5.
* Yếu: nếu điểm trung bình nhỏ hơn 5.

Yêu cầu: tạo ra một menu với các chức năng sau:

* 1. Thêm sinh viên.
* 2. Cập nhật thông tin sinh viên bởi ID.
* 3. Xóa sinh viên bởi ID.
* 4. Tìm kiếm sinh viên theo tên.
* 5. Sắp xếp sinh viên theo điểm trung bình (GPA).
* 6. Sắp xếp sinh viên theo tên.
* 7. Hiển thị danh sách sinh vien.
* 8. Ghi danh sách sinh viên vào file **"student.txt"**.

Lời giải

Chúng ta sẽ sử dụng struct để định nghĩa một cấu trúc sinh viên, và một mảng struct để lưu trữ danh sách sinh viên. Sau đây là các bước cụ thể để cài đặt chương trình quản lý sinh viên trong C/C++ bằng struct.

Tạo cấu trúc sinh vien

Chúng ta sẽ tạo một cấu trúc sinh viên như sau, **diemTB** sẽ được tính bằng trung bình của các môn toán, lý và hóa. **hocluc** sẽ được tính toán dựa vào **diemTB** như yêu cầu của bài toán.

**struct** SinhVien {

**int** id;

**char** ten[30];

**char** gioiTinh[5];

**int** tuoi;

**float** diemToan;

**float** diemLy;

**float** diemHoa;

**float** diemTB = 0;

**char** hocluc[10] = "-";

};

Bây giờ chúng ta có thể sử dụng struct SinhVien với đơn giản hơn bằng SV với từ khóa typedef, như sau:

**typedef** SinhVien SV;

## Định nghĩa các hàm

Sau đây là danh sách các hàm mà chúng ta sẽ sử dụng trong chương trình quản lý sinh viên, các hàm này sẽ được cung cấp thân hàm sau đó:

**void** printLine(**int** n);

**int** idLonNhat(SV a[], **int** n);

**void** nhapThongTinSV(SV &sv, **int** id);

**void** nhapSV(SV a[], **int** id, **int** n);

**void** capNhatThongTinSV(SV &sv);

**void** capNhatSV(SV a[], **int** id, **int** n);

**int** xoaTheoID(SV a[], **int** id, **int** n);

**void** timKiemTheoTen(SV a[], **char** ten[], **int** n);

**void** tinhDTB(SV &sv);

**void** xeploai(SV &sv);

**void** sapxepTheoDTB(SV a[], **int** n);

**void** sapXepTheoTen(SV a[], **int** n);

**void** showStudent(SV a[], **int** n);

**int** docFile(SV a[], **char** fileName[]);

**void** ghiFile(SV a[], **int** n, **char** fileName[]);

**void** pressAnyKey();