Chào mừng các bạn trở lại với Series Lập Trình Căn Bản C. Hôm nay chúng ta sẽ tiếp tục bài học về Mảng trong ngôn ngữ C.

Trước khi vào bài học, chúng ta sẽ đến với một vài tình huống mở đầu.

**Tình huống 1:**

Cô giáo bảo bạn làm một bài tập, yêu cầu chương trình như sau:

* Viết một chương trình nhập vào 3 số rồi sau đó tính tổng 3 số đó.
  + Vấn đề vẫn còn đơn giản, bạn hoàn toàn có thể làm được.
* Viết một chương trình nhập vào 6 số rồi tính tổng 6 số đó.
  + Thêm vài biến nữa, dài hơn tí nhưng không sao.
* Viết một chương trình nhập vào 100 số rồi tính tổng 100 số đó.
  + 100 số? dài thế rồi sao làm đây….

**Tình huống 2:**

Lần này cô giáo bảo bạn viết giùm cô danh sách học sinh trong lớp, lớp khoảng 10 hay 20 người gì đó thôi. Giải quyết vấn đề.

* Tạo khoảng 10 hoặc 20 biến gì đó, rồi lưu thông tin của học sinh vào đó.
* Vài tháng sau, 2 học sinh nghỉ học => bạn phải xóa 2 biến.
* Vài tháng sau nữa, có 1 học sinh chuyển vào lớp bạn => Tạo thêm 1 biến mới.

Ta thấy ở 2 tình huống, mỗi tình huống có một số vấn đề mà sử dụng biến chưa được tốt cho lắm:

* Tạo ra 100 biến rồi thực hiện tính tổng lại ở tình huống 1.
* Làm sao để biết có bao nhiều người trong lớp khi mà cứ phải thêm rồi bớt ở tình huống 2.

Cần phải có một cách khắc phục cho tình huống này.

Và để khắc phục nó, chúng ta cần phải sử dụng một cơ chế mới mang tên **MẢNG**.

Bản chất của mảng giống như một danh sách vậy, danh sách này sẽ chứa nhiều biến có cùng kiểu dữ liệu, giúp cho chúng ta thao tác nó nhanh gọn và bao quát hơn.

Ví dụ: tạo 5 biến rồi tìm giá trị lớn nhất.

#include "stdio.h"

int main(){

int a, b, c, d, e;

printf("Nhap a: ");

scanf("%d", &a);

printf("Nhap b: ");

scanf("%d", &b);

printf("Nhap c: ");

scanf("%d", &c);

printf("Nhap d: ");

scanf("%d", &d);

printf("Nhap e: ");

scanf("%d", &e);

int Max = a;

if(b > Max)

Max = b;

if(c > Max)

Max = c;

if(d > Max)

Max = d;

if(e > Max)

Max = e;

printf("Max la: %d", Max);

return 0;

}

Các bạn thấy chương trình khá dài phải không, giờ thử sử dụng mảng nhé.

#include "stdio.h"

int main(){

int a[5];

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

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

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

}

int Max = a[0];

for(int i = 1 ; i < 5 ; i++)

if(a[i] > Max)

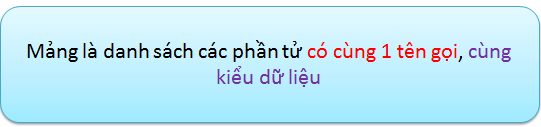
Max = a[i];

printf("Max la: %d", Max);

return 0;

}

Chương trình đã ngắn lại rồi. Giờ thì chúng ta sẽ bắt đầu học về Mảng.



1. Khai báo Mảng

Cách khai báo mảng tương tự như khai báo một biến. Sự khác biệt duy nhất là dấu mở đóng ngoặc vuông và bên trong là số lượng phần tử khai báo.

**Cú pháp: kiểu\_dữ\_liệu tên\_mảng [ số\_lượng\_phần\_tử\_tối\_đa ]**



Hay có một số cách khai báo khác như

**int a[] = {1, 2, 3, 4, 5} *// mảng kiểu int, tên a và chứa 5 phần tử.***

1. Cách lưu trữ dữ liệu và thao tác trên mảng

**Cách lưu trữ dữ liệu**

Bắt đầu phần này bằng một ví dụ dưới đây:

“Giả sử ai đó đưa bạn một đống kẹo, và bảo bạn đếm xem có bao nhiêu cây kẹo

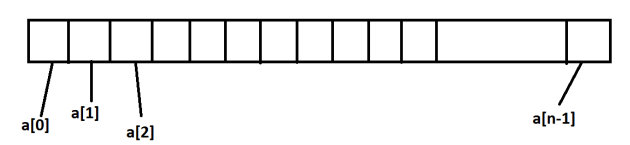
ở trong đó.

Khi bạn đếm chúng, bạn sẽ bắt đầu bằng 1 và đếm: 1, 2, 3,..., n.”

Phải vậy không?

Vậy đó là cách bạn đếm kẹo. Hay bất cứ thứ gì bình thường khác. Vì nó luôn bắt đầu bằng 1.

Nhưng với mảng. Nó không bắt đầu bằng 1. Mà nó bắt đầu bằng 0. Và kết thúc là phần tử n - 1.



Đây là cách lưu trữ dữ liệu của mảng.

Ví dụ: Bạn có mảng

int a[5];

mảng trên sẽ chứa các số nguyên: a[0], a[1], a[2], a[3], a[4]. Và các bạn đếm xem có phải có 5 phần tử rồi không.

* Đó chính là MẢNG.

Giờ chúng ta đến với ví dụ lúc nãy: int a[] = {1, 2, 3, 4, 5}

Nhìn vào mảng này ai cũng biết nó gồm 5 phần tử.

Kết hợp với cách lưu trữ mảng ta có thể suy ra.

a[0] = 1;

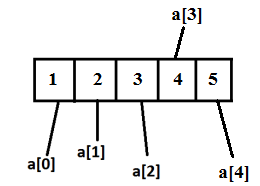
a[1] = 2;

a[2] = 3;

a[3] = 4;

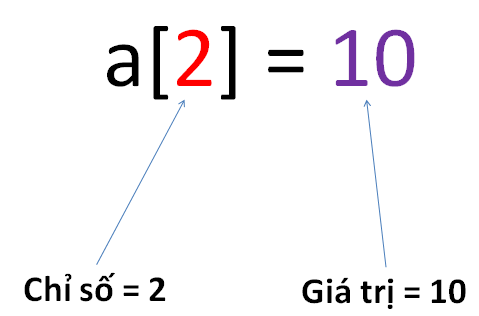
a[4] = 5;

Mảng vẫn bắt đầu bằng 0. Và lần này thì mỗi phần tử đã có giá trị của riêng nó, được biểu diễn như sau



Khi nhìn vào MẢNG, chúng ta sẽ biết được 2 tính chất:

1. Chỉ số: Số thứ tự của mảng ( bắt đầu từ 0)
2. Giá trị: giá trị của phần tử.



Từ đây các bạn cũng sẽ suy ra được cách truy xuất giá trị của mảng thông qua chỉ số.

1. Thao tác với MẢNG.

Khi thao tác với biến, các bạn sẽ nhập nó vào hoặc gán một cách thủ công, nhưng qua mảng thì ta nên sử dụng một cách khác thông minh hơn dựa vào vòng lặp.

Ví dụ về việc khởi tạo giá trị của mảng:

int a[10];

a[0] = 1;

a[1] = 2;

a[2] = 3;

a[3] = 4;

a[4] = 5;

a[5] = 6;

a[6] = 7;

...

a[9] = 10;

Cách này khá dài dòng, chúng ta cũng có thể khởi tạo kiểu như sau:

int a[] = {1, 2, 3, 4, 5, 6, 7, 8, 9, 10}.

Giờ chúng ta thử sài vòng lặp.

for ( i = 0 ; i < n ; i++) // mảng bắt đầu bằng 0

{

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

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

}

Code đã ngắn gọn và dễ nhìn hơn rất nhiều rồi phải không các bạn.

|  |  |
| --- | --- |
| Thủ công | Cải tiến |
| Int a[10];  a[0] = 1;  a[1] = 2;  a[2] = 3;  a[3] = 4;  a[4] = 5;  a[5] = 7;  a[6] = 10;  a[7] = 33;  a[8] = 29;  a[9] = 0; | for ( i = 0 ; i < n ; i++) // mảng bắt đầu bằng 0  {  printf("nhap a[%d]: ",i);  scanf("%d", &a[i]);  }  Các bạn chú ý chạy đến < n hoặc <= n – 1 ( vì mảng bắt đầu từ 0 đến n – 1 ). |

Tương tự vậy, chúng ta có thể viết code xuất mảng.

for ( i = 0 ; i < n ; i++) // mảng bắt đầu bằng 0

{

// thao tác xuất mảng

}

Sau khi học đến đây. Chúng ta đã biết cơ bản về Mảng, giờ chúng ta sẽ học một kỹ thuật liên quan tới mảng có tên là **KỸ THUẬT DUYỆT MẢNG.**

**KỸ THUẬT DUYỆT MẢNG**

Kỹ thuật này giúp ta lần lượt đi qua các phần tử trong mảng, nó có thể làm khá nhiều việc.

Kỹ thuật này sử dụng vòng lặp for để duyệt, tương tự như nhập mảng và xuất mảng, tôi sẽ demo bằng code.

Ví dụ 1: Nhập mảng

Int a[5];

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

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

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

}

Dựa vào vòng lặp for, chương trình chạy lần lượt từ 0 đến 4, mỗi lần đi qua nó sẽ cho phép chúng ta nhập vào các giá trị của mảng.

Ví dụ 2: Xuất mảng

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

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

}

Nếu bạn đã có một mảng hoàn chỉnh thì bạn có thể xuất mảng ra dựa trên đoạn code này.

Ví dụ 3: Đếm các số nguyên dương ( > 0 ) trong mảng

int dem = 0;

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

if(a[i] > 0)

dem++;

}

Code đầy đủ cho chương trình đếm số nguyên dương trong mảng có độ dài N phần tử.

#include "stdio.h"

int main(){

int a[100];

int n;

printf("Nhap n: ");

scanf("%d", &n);

// nhap mang

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

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

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

}

// xuat mang

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

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

}

int dem = 0;

// dem so nguyen duong

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

if(a[i] > 0)

dem++;

}

printf("Dem = %d", dem);

return 0;

}

Tương tự, với kỹ thuật này ta cũng có thể làm một số việc khác như tính tổng của mảng, xóa phần tử, tìm kiếm trong mảng, sắp xếp trong mảng.

Bài học của chúng ta đã kết thúc, các bạn có thể xem phần tiếp theo ở đây.

Bài 1. Nhập, xuất mảng, tính tổng các phần tử của mảng.

Bài 2. Nhập, xuất mảng, tìm phần tử lớn nhất trong mảng.

Bài 3: Nhập mảng, xuất mảng, xóa phần tử đầu tiên ra khỏi mảng.