Chào mừng các bạn quay trở lại với Series Lập Trình C căn bản.

Với phần trước, các bạn đã biết thế nào là lập trình, đã có thể sử dụng câu lệnh in ra màn hình.

Ở phần này, chúng ta sẽ chính thức đến với bài học đầu tiên về lập trình C.

**KIỂU DỮ LIỆU TRONG LẬP TRÌNH C**

1. Biến trong C

***Biến là gì?***

Hiểu đơn giản thì nó là một cái gì đó có thể giữ đồ cho mình. Khi mình cần thì chỉ lấy nó ra và sử dụng.

Giả sử như bạn có một cái thẻ ngân hàng, trong đó có tiền của bạn. Khi cần thì chúng ta tới cây ATM hay ngân hàng nào đó, rút tiền ra và sử dụng.

Như các bước ở trước, để học thứ gì đó thì phải có quy trình, và bước đầu tiên sẽ là

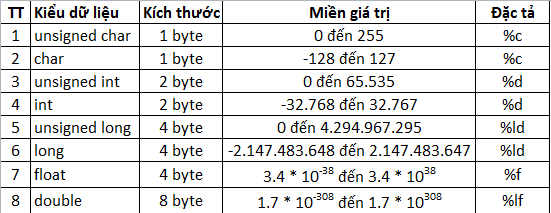
**Chuẩn bị nguyên liệu**

Chúng ta sẽ tìm hiểu về các kiểu dữ liệu trong C trước khi sử dụng biến.

Trong lập trình C cũng như các ngôn ngữ lập trình khác, nó có 4 kiểu dữ liệu chính:

1. Kiểu dữ liệu cơ bản : Chúng là các loại số học và được phân loại thành: các kiểu số nguyên và các kiểu số thực.
2. Kiểu Enumeration : Chúng lại là các loại số học và chúng được sử dụng để xác định các biến chỉ có thể chỉ định một số giá trị số nguyên rời rạc trong suốt chương trình.
3. Kiểu dữ liệu Derived : Chúng bao gồm các kiểu con trỏ (Pointer), mảng (Array), cấu trúc (Structure), Union và Function.
4. Kiểu Void : Kiểu Void (vô nghĩa) chỉ định rằng không có giá trị nào có sẵn.

Tại đây tôi sẽ giới thiệu các bạn về kiểu dữ liệu cơ bản. Mời các bạn theo dõi bảng sau



Bảng trên đã giải thích đầy đủ về một số thứ cần chú ý khi sử dụng kiểu dữ liệu trong lập trình C.

Khi sử dụng thì bạn khai báo theo cú pháp sau:

Kieu\_du\_lieu ten\_bien [= gia\_tri];

Ví dụ: int A = 10; // khai báo A có kiểu dữ liệu int với giá trị là 10

int A; // khai báo kiểu dữ liệu A không có giá trị sẵn.

Để lấy dữ liệu, các bạn thao tác thong qua tên biến và đặc tả tương ứng.

|  |
| --- |
| *#include <stdio.h>*  *int main(){*  *int A = 10;*  *printf("A = %d", A); // A = 10*  *return 0;*  *}* |

**Sau khi đã qua phần biến, các bạn sẽ học về Phạm vi sử dụng biến.**

Ngôn ngữ C có định nghĩa về sử dụng biến thong qua vị trí của nó, khi sử dụng biến, các bạn nên biết về Vị trí khai báo và phạm vi tương ứng. Có 2 kiểu khai báo biến:

1. Khai báo toàn cục:Một biến khai báo ngoài các hàm, cấu trúc,… thì được gọi là biến Cục Bộ. Nó sẽ có thể được sử dụng trong toàn bộ chương trình ở dưới nó.
2. Khai báo cục bộ: Một biến khai báo bên trong các hàm, cấu trúc,.. thì được gọi là Biến Cục Bộ. Nó sẽ chỉ sử dụng được bên trong hàm.

Để hiểu rõ hơn, chúng ta sẽ đến với demo về nó.

|  |
| --- |
| *#include <stdio.h>*  *int a = 5, b = 7; // biến toàn cục, được sử dụng ở bất kì đâu*  *int main(){*  *printf("Bien toan cuc: a = %d, b = %d\n", a, b);*  *int c = 1, d = 10; // biến cục bộ*  *printf("Bien cuc bo: c = %d, d = %d", c, d);*  *return 0;*  *}* |

Ở ví dụ trên, các bạn thấy mình in biến **a, b** mặc dù mình không khai báo trong hàm **main**. Nghĩa là các biến **a, b** chính là biến toàn cục.

Còn biến **c,d** là các biến cục bộ và chỉ có thể dụng ở dưới nó ( khai báo rồi mới được sử dụng).

**Hằng trong C**

Hằng là một kiểu dữ liệu đặc biệt, có thể khai báo ở bất kì đâu trong chương trình.

1. **Biến const**: đây là một kiểu biến hằng, khi sử dụng nó thì dữ liệu của các bạn không thể sửa được.

Ví dụ nhé:

|  |
| --- |
| *#include <stdio.h>*  *int main(){*  *int a = 10; // khai báo a = 10*  *const int b = 10; // b = 10*  *printf("ban dau a = %d, b = %d", a, b); // ban dau a = 10, b = 10*  *a = 5; // lúc này a thay đổi, a = 5*  *b = 5; // do b là dữ liệu const nên không thể thay đổi, ở đây sẽ bị lỗi*  *printf("sau khi thay doi a = %d, b = %d", a, b); // chương trình lỗi*  *return 0;*  *}* |

1. Define: đây cũng được coi là kiểu dữ liệu hằng, bạn sẽ sử dụng nó khi muốn định nghĩa một cái gì đó.

Ví dụ: Trong toán học có số pi ( Π ) thì giá trị tương ứng là 3.141593. Ta sẽ định nghĩa nó lại sau đây

|  |
| --- |
| *#include <stdio.h>*  *#define pi 3.141593 // định nghĩa giá trị pi*  *int main(){*  *printf("pi = %lf",pi); // pi = 3.141593*  *return 0;*  *}* |

**Lưu ý: Dữ liệu kiểu hằng (const) và define không thể thay đổi giá trị.**

Ngoài ra, các bạn còn có thể sử dụng kiểu define với những giá trị khác, dưới đây là một vài ví dụ rất hay, các bạn có thể tham khảo.

|  |
| --- |
| *#include <stdio.h>*  *#define in(a) printf("a = %d",a);*  *int main(){*  *in(3); // in ra màn hình : a = 3.*  *return 0;*  *}* |

Bài học này của chúng ta đã kết thúc, bài tiếp theo chúng ta sẽ đến với nhập xuất trong Lập Trình C.