



28TECH
Become A Better Developer



PROGRAM IN C++





28TECH

Become A Better Developer

Khuôn mẫu chung của một chương trình

INPUT



OUTPUT





Chương trình C++ đầu tiên

```
#include <iostream>
```

Khai báo thư viện cần thiết

```
using namespace std;
```

Thêm namespace

```
int main () {
```

Hàm main: là nơi bắt đầu thực thi của chương trình

```
    cout << "Hello world !" << endl;  
    return 0;
```

```
}
```



Những phần chính của một chương trình C++:

/01 Thư viện mà chương trình sử dụng.

/02 Namespace.

/03 Chương trình chính với mã nguồn.





Những chú ý khi viết chương trình C++

▶▶▶ Các câu lệnh kết thúc bằng dấu ";"



Không hợp lệ

```
cout << "Hello 28tech" <<
```



Hợp lệ

```
cout << "Hello 28tech" << ;
```



Những chú ý khi viết chương trình C++

- ▶▶▶ Các câu lệnh kết thúc bằng dấu ";"
- ▶▶▶ Luôn thụt lề các câu lệnh so với hàm main



Không chấp nhận

```
int main () {  
cout << "Hello world !" << endl;  
return 0;  
}
```



Chấp nhận

```
int main () {  
    cout << "Hello world !" << endl;  
    return 0;  
}
```





28TECH

Become A Better Developer

Kiểu dữ liệu (DATA TYPE)





KIỂU DỮ LIỆU SỐ NGUYÊN

1 byte = 8 bit

👉 Đối với số nguyên ta chia làm số nguyên không dấu và số nguyên có dấu, từ số byte lưu trữ ta có thể suy ra số bit cần để biểu diễn số nguyên đó, quy tắc xác định giá trị của 1 số nguyên.

Giả sử số nguyên có K bit

-Số nguyên có dấu : -2^{K-1} tới $2^{K-1} - 1$

-Số nguyên không dấu : 0 tới $2^K - 1$





Kiểu dữ liệu số nguyên

Kiểu dữ liệu	Ý nghĩa	Số Byte	Giải giá trị có thể lưu
short	Số nguyên có dấu	2 byte	-32768 đến 32767
unsigned short	Số nguyên không dấu	2 byte	0 đến 65535
int	Số nguyên có dấu	4 byte	-2147483648 đến 2147483647
unsigned int	Số nguyên không dấu	4 byte	0 đến 4294967295
long long	Số nguyên	8 byte	-9223372036854775808 đến 9223372036854775807
unsigned long long	Số nguyên không dấu	8 byte	0 đến $2^{64} - 1$



KIỂU DỮ LIỆU SỐ THỰC

Kiểu dữ liệu	Ý nghĩa	Số Byte	Giải giá trị có thể lưu
float	Số thực độ chính xác đơn	4 byte	1.17549e-038 đến 3.40282e+038 Độ chính xác : 7 chữ số sau dấu phẩy
double	Số thực độ chính xác kép	8 byte	2.22507e-308 đến 1.79769e+308 Độ chính xác : 15 chữ số sau dấu phẩy



Ưu tiên sử dụng double vì double có độ chính xác cao hơn float





Kiểu dữ liệu ĐÚNG SAI

Kiểu dữ liệu	Ý nghĩa	Số Byte	Giải giá trị có thể lưu
bool	Kiểu dữ liệu luận lý	1 byte	true hoặc false true : Đúng false : Sai





Kiểu dữ liệu ký tự

Kiểu dữ liệu	Ý nghĩa	Số Byte	Giải giá trị có thể lưu
char	Kiểu dữ liệu ký tự	1 byte	-128 tới 127





Tổng quan về kiểu dữ liệu:

/01 Kiểu số nguyên có **int** và **long long**.

>> Sử dụng long long cho các kết quả là số lớn

/02 Kiểu số thực có **float** và **double**.

>> Sử dụng double vì có độ chính xác cao

/03 Kiểu đúng sai có **bool**.

/04 Kiểu ký tự có **char**.





BIẾN (VARIABLE)





BIẾN (VARIABLE)

Khái niệm

- ▶▶▶ Biến được sử dụng để lưu các giá trị trong quá trình tính toán của chương trình. Tùy theo kiểu dữ liệu của biến, một ô trong bộ nhớ sẽ được cấp phát để lưu trữ giá trị của biến này.

Cú pháp

- ▶▶▶ [Kiểu dữ liệu] [Tên biến];

Ví dụ: `int x; long long b; char ki_tu; bool check; double dienTich; ...`





BIẾN (VARIABLE)

Quy tắc đặt tên biến:

Ví dụ cách đặt sai:

01	Không đặt tên biến bắt đầu bằng chữ số	1dientich, 2chuvi, 222bankinh, 9a,...
02	Tên biến không được chứa dấu cách và các kí tự đặc biệt	ban kinh, dien#tich, chu@vi,...
03	Tên biến không được trùng với tên từ khóa trong C++	int, main, for, while, ...
04	Tên biến trong C++ là phân biệt hoa thường	banKinh và BanKinh là 2 biến khác nhau
05	Không được đặt 2 biến có cùng tên trong cùng một phạm vi	int a; float a;





CHÚ THÍCH TRONG C++

//

/* */





CHÚ THÍCH

Khái niệm

Chú thích (Comment) là một giải pháp bổ sung thông tin vào code của bạn, nhằm làm rõ nội dung, giải thích câu lệnh, mục đích của code...

Giúp người đọc code có thể dễ nắm bắt nội dung code và thuận lợi trong việc bảo trì code

Các chú thích sẽ không được coi là câu lệnh và sẽ được loại bỏ khi chương trình thực thi



CHÚ THÍCH

Cách chú thích

Để chú thích trên 1 dòng ta dùng //

Ví dụ :

```
//Đây là chú thích
```

```
//Chú thích giúp code rõ ràng hơn
```

Để chú thích trên nhiều dòng ta dùng /* ND cần chú thích */

```
/*
```

Đây là

chú thích trên nhiều dòng

```
*/
```



Khai báo biến

MỘT SỐ VÍ DỤ

```
#include <iostream>
using namespace std;

int main(){
    int a; // biến a có kiểu int
    int x, y, z; // khai báo 3 biến cùng kiểu int
    long long dien_tich; // biến dien_tich có kiểu ll
    bool check; // biến check kiểu đúng sai
    char kitu; // biến kitu kiểu char
}
```



MỘT SỐ VÍ DỤ

Hiển thị giá trị của biến ra màn hình

👉 Cú pháp: `cout << var ;`

Hiển thị giá trị của biến số thực `var` với độ chính xác `k` số sau dấu phẩy:

👉 Cú pháp:

`cout << fixed << setprecision(k) << var;`

Phải khai báo thư viện `iomanip` !

```
#include <iostream>
#include <iomanip>
using namespace std;
```

```
int main(){
    int a = 28; // biến a có kiểu int và có giá trị là 28
    cout << a << endl; //xuất biến a ra màn hình
    int x = 100, int y = 200; //khởi tạo nhiều biến
    cout << x << ' ' << y << ' ' << endl; //xuất cùng lúc
    double res = 3.14;
    cout << fixed << setprecision(5) << res << endl;
    //output: 3.14000
}
```



MỘT SỐ VÍ DỤ

Nhập giá trị cho biến từ bàn phím

 Cú pháp: `cin >> var ;`

Chú ý : Khi nhập xong giá trị các bạn ấn enter để cin bắt đầu nhập giá trị cho biến, trong trường hợp bạn nhập giá trị cho nhiều số, có thể nhập từng số rồi enter hoặc nhập 1 lúc nhiều số rồi ấn enter 1 lần.

```
#include <iostream>
using namespace std;
```

```
int main(){
    int a; // biến a có kiểu int
    cin >> a; //nhập giá trị cho a
    cout << a << endl; //xuất ra giá trị vừa nhập
    int x, y, z; // khai báo 3 biến cùng kiểu int
    cin >> x >> y >> z;
    cout << x << " " << y << " " << z << " " << endl;
}
```





28TECH

Become A Better Developer



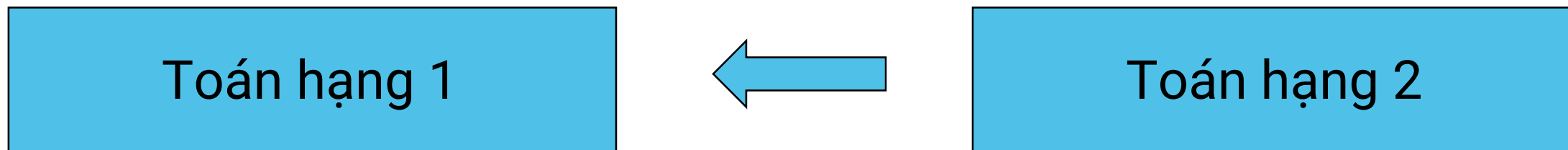
TOÁN TỬ (OPERATOR)





TOÁN TỬ GÁN: (ASSIGNMENT OPERATOR)

👉 Cú pháp: [Toán hạng 1] = [Toán hạng 2]



👉 Ý nghĩa: Gán giá trị của toán hạng 2 cho toán hạng 1.

VÍ DỤ: `ban_kinh = 100; // Gán giá trị 100 cho biến ban_kinh`
`Chu_vi = 1000; // Gán giá trị của chu vi là 1000`





TOÁN TỬ TOÁN HỌC: (ARITHMETIC OPERATOR)

Toán tử	Ý nghĩa	Ví dụ
+	Cộng 2 toán hạng	$X = Y + Z$ $X = 100 + 200; // 300$
-	Trừ 2 toán hạng	$X = Y - Z$ $X = 200 - 100; // 100$
*	Nhân 2 toán hạng	$X = Y * Z$ $X = 10 * 50; // 500$
/	Chia 2 toán hạng	$X = Y / Z$ $X = 300 / 200; // 1$
%	Chia dư 2 toán hạng	$X = Y \% Z$ $X = 300 \% 200; // 100$





TOÁN TỬ TOÁN HỌC: (ARITHMETIC OPERATOR)

Chú ý 1: Nếu bạn chia 2 số nguyên (int, long long) cho nhau thì phép chia ở trên sẽ là phép chia nguyên, tức là nó chỉ lấy phần nguyên và bỏ phần thập phân ở thương

- Nếu muốn kết quả ở số thập phân thì ít nhất 1 trong 2 số phải ở kiểu số thực và biến thương phải ở dạng số thực

+) Ép kiểu

+) Nhân thêm 1 số thực khi tính toán

```
int a = 100, b = 30;  
float thuong = (float) a / b;  
cout << fixed << setprecision(2) <<  
thuong << endl; //3.33
```





TOÁN TỬ TOÁN HỌC: (ARITHMETIC OPERATOR)

Chú ý 1: Nếu bạn chia 2 số nguyên (int, long long) cho nhau thì phép chia ở trên sẽ là phép chia nguyên, tức là nó chỉ lấy phần nguyên và bỏ phần thập phân ở thương

- Nếu muốn kết quả ở số thập phân thì ít nhất 1 trong 2 số phải ở kiểu số thực và biến thương phải ở dạng số thực

+) Ép kiểu

+) Nhân thêm 1 số thực khi tính toán

```
int a = 100, b = 30;  
float thuong = 1.0 * a / b;  
cout << fixed << setprecision(2) <<  
thuong << endl; //3.33
```



TOÁN TỬ TOÁN HỌC: (ARITHMETIC OPERATOR)

Chú ý 2: Nếu bạn nhân 2 số nguyên int với nhau mà kết quả của tích vượt giới hạn lưu của số int thì kết quả sẽ bị tràn, ngay cả khi bạn sử dụng biến long long để lưu biến tích. Việc cần xử lý ở đây là can thiệp vào phép nhân

Cách xử lí sai

```
int a = 1000000, b = 1000000;  
long long tích = a * b; //sai  
long long tích = (long long) (a * b);  
// vẫn sai  
cout << tích << endl; //tràn số
```

Cách xử đúng

```
int a = 1000000, b = 1000000;  
// ép 1 trong 2 số thành ll  
long long tích = (long long) a * b;  
long long tích = 1ll * a * b; //1ll ?  
cout << tích << endl; //10^12
```



TOÁN TỬ SO SÁNH: (COMPARISON OPERATOR)



Khi bạn sử dụng các toán tử so sánh để so sánh 2 toán hạng thì kết quả của phép so sánh sẽ trả về đúng hoặc sai.

Toán tử	Ý nghĩa	Ví dụ
>	Lớn hơn	$10 > 50$: false
>=	Lớn hơn hoặc bằng	$20 \geq 10$: true
<	Nhỏ hơn	$10 < 50$: true
<=	Nhỏ hơn hoặc bằng	$20 \leq 20$: true
!=	So sánh khác	$10 \neq 20$: true
==	So sánh bằng	$10 == 10$: true





TOÁN TỬ SO SÁNH: (COMPARISION OPERATOR)

VÍ DỤ:

Ví dụ	Kết quả
$10 == 20$	Sai
$10 <= 20$	Đúng
$10 == 10$	Đúng
$50 != 50$	Sai
$100 <= 100$	Đúng



TOÁN TỬ LOGIC: (LOGICAL OPERATOR)



Trong trường hợp bạn muốn kết hợp nhiều phép so sánh lại với nhau, ta sử dụng 3 cổng logic cơ bản của máy tính là **AND, OR, NOT**

Toán tử	Ký hiệu	Ví dụ
AND	&&	$(x \geq 10) \&\& (x \leq 50)$
OR		$(a \geq 10) (a \% 2 == 0)$
NOT	!	$!(a \leq 10)$



Để tính toán giá trị cuối cùng của biểu thức các bạn xác định giá trị của từng mệnh đề thành phần, sau đó áp dụng bảng chân lý của các cổng logic để tìm giá trị của biểu thức ban đầu.





TOÁN TỬ LOGIC: (LOGICAL OPERATOR)

Bảng chân lý của cổng AND

A	B	A && B
0	0	0
0	1	0
1	0	0
1	1	1

Chú ý: Cổng AND chỉ cho kết quả đúng khi mọi mệnh đề thành phần đều có giá trị đúng, sai trong các trường hợp còn lại.





TOÁN TỬ LOGIC: (LOGICAL OPERATOR)

Bảng chân lý của cổng OR

A	B	A B
0	0	0
0	1	1
1	0	1
1	1	1

Chú ý: Cổng OR chỉ cho kết quả sai khi mọi mệnh đề thành phần đều có giá trị sai, đúng khi chỉ cần ít nhất 1 mệnh đề thành phần có giá trị đúng





TOÁN TỬ LOGIC: (LOGICAL OPERATOR)

Bảng chân lý của cổng NOT

A	NOT A
0	1
1	0



TOÁN TỬ TĂNG GIẢM: (INCREMENT, DECREMENT OPERATOR)



Để tăng hoặc giảm giá trị của một biến đi 1 đơn vị ta có thể sử dụng toán tử tăng, giảm này sẽ thuận tiện hơn.

Toán tử	Ý nghĩa	Ví dụ
++	Tăng trước 1 đơn vị	++a
++	Tăng sau 1 đơn vị	a++
--	Giảm trước 1 đơn vị	--a
--	Giảm sau 1 đơn vị	a--





TOÁN TỬ TĂNG GIẢM: (INCREMENT, DECREMENT OPERATOR)

VÍ DỤ:

Ví dụ	Kết quả	Giải thích
<pre>int a = 100; int b = a++; cout << a << " " << b << endl;</pre>	101 100	Đây là tăng sau , tức ban đầu câu lệnh sẽ gán giá trị của a là 100 cho b, sau đó mới tăng giá trị của a lên 101
<pre>int a = 100; int b = ++a; cout << a << " " << b << endl;</pre>	101 101	Đây là tăng trước , giá trị của a sẽ được tăng ngay lập tức lên 101, sau đó mới lấy giá trị đó và gán cho b





TOÁN TỬ 3 NGÔI: (CONDITIONAL OPERATOR)

Cú pháp

▶▶▶ [Biểu thức so sánh] ? [Giá trị trả về khi biểu thức đúng] : [Giá trị trả về khi biểu thức sai]

Ví dụ	Kết quả	Giải thích
<code>int x = 10 < 20 ? 10 : 20;</code>	<code>x = 20</code>	Nếu <code>10 < 20</code> thì vế phải sẽ trả về 10, ngược lại sẽ trả về 20. Sau đó giá trị này được gán cho x
<code>int y = (10 < 20) && (20 > 20) ? 5 : 10;</code>	<code>y = 10</code>	Nếu <code>10 < 20</code> và <code>20 > 20</code> thì sẽ gán 5 cho y, ngược lại gán 10 cho y



CHÚ Ý Ở PHẦN TOÁN TỬ



Các toán tử sẽ có **thứ tự ưu tiên nhất định**, ví dụ như nhân chia trước, cộng trừ sau hoặc cùng mức độ ưu tiên thì thực thi từ trái qua phải



Nhưng **dấu đóng mở ngoặc tròn** luôn có độ ưu tiên cao nhất, vì thế khi viết biểu thức thì các bạn nên sử dụng dấu ngoặc để biểu thức được thực thi theo đúng mong muốn của mình.





CẤU TRÚC RẺ NHÁNH

IF...ELSE

SWITCH CASE



1. CÂU LỆNH IF

- ▶▶▶ Câu lệnh if được sử dụng trong trường hợp bạn muốn chương trình của mình thực hiện 1 hoặc 1 nhóm câu lệnh khi một điều kiện nào đó thỏa mãn.
- ▶▶▶ VD: Nếu chỉ số máu của nhân vật bằng 0 thì nhân vật sẽ chết, vậy điều kiện ở đây là “chỉ số máu của nhân vật bằng 0”, và hành động được thực hiện ở đây là sẽ có một câu lệnh thực thi làm nhân vật bị chết.

SYNTAX

```
if (condition) {  
    // code  
}
```





1. CÂU LỆNH IF

VÍ DỤ:

Ví dụ	Kết quả	Ý nghĩa
<pre>int a = 100, b = 200; if (a < b){ cout << "OK"; }</pre>	OK	Vì $a < b$ có giá trị là đúng nên câu lệnh bên trong if được thực thi
<pre>int a = 100, b = 200; if ((a % 2) == 0){ cout << "Even"; }</pre>	Even	Nếu a chia dư cho 2 bằng 0, thì a là số chẵn

Chú ý: Các bạn nhớ thật lề các câu lệnh bên trong if so với if nhé, trong trường hợp bên trong if chỉ có 1 câu lệnh, các bạn có thể bỏ dấu đóng mở ngoặc nhọn.



2. CÂU LỆNH IF ELSE

- If được sử dụng khi bạn muốn thực thi code với điều kiện nào đó đúng, trong trường hợp điều kiện đó sai bạn muốn thực thi một đoạn code khác thì cấu trúc if else sẽ được sử dụng



SYNTAX

```
if (condition) {  
    // code if  
}  
else {  
    // code else  
}
```



2. CÂU LỆNH IF ELSE

VÍ DỤ:

Ví dụ	Kết quả	Ý nghĩa
<pre>int a = 100; if((a % 2) == 0){ cout << "Chan"; } else{ cout << "Le"; }</pre>	Chan	Nếu a chia 2 dư 0, tức điều kiện trong if đúng thì câu lệnh in ra "Chan", ngược lại nếu điều kiện đó sai thì in ra "Le"



3. CÂU LỆNH IF VÀ ELSE IF

- ▶▶▶ Nếu bạn muốn kiểm tra nhiều điều kiện khác nhau thì sử dụng cấu trúc else if sẽ hiệu quả hơn so với sử dụng nhiều câu lệnh if else lồng nhau.



SYNTAX

```
if (condition1) {  
    // code  
}  
else if (condition2){  
    // code  
}  
....  
else if(conditionN){  
    //code  
}  
else{  
    // code  
}
```





3. CÂU LỆNH IF VÀ ELSE IF

Chú ý: Nếu một điều kiện nào trong N điều kiện trong cấu trúc trên đúng và câu lệnh bên trong nhánh đó được thực hiện thì khối lệnh if else if này sẽ kết thúc ngay lập tức. Ví dụ, nếu condition2 đúng thì khối lệnh bên trong nhánh đó được thực thi, sau đó cấu trúc này sẽ kết thúc ngay mà không kiểm tra các điều kiện còn lại cũng như trong else.

Good	Bad
<pre>int day; cin >> day; if(day == 1) cout << "Chu nhat" << endl; else if(day == 2) cout << "Thu hai" << endl; else if(day == 3) cout << "Thu ba" << endl; else if(day == 4) cout << "Thu tu" << endl; else if(day == 5) cout << "Thu nam" << endl; else if(day == 6) cout << "Thu sau" << endl; else if(day == 7) cout << "Thu bay" << endl;</pre>	<pre>int day; cin >> day; if(day == 1) cout << "Chu nhat" << endl; if(day == 2) cout << "Thu hai" << endl; if(day == 3) cout << "Thu ba" << endl; if(day == 4) cout << "Thu tu" << endl; if(day == 5) cout << "Thu nam" << endl; if(day == 6) cout << "Thu sau" << endl; if(day == 7) cout << "Thu bay" << endl;</pre>



4. CÂU LỆNH SWITCH CASE

- ▶▶▶ Switch case cũng giúp bạn kiểm tra nhiều điều kiện khác nhau.
- ▶▶▶ **Ý nghĩa** : Giá trị của val sẽ được so sánh lần lượt với các giá trị trong các case, nếu giá trị của val bằng giá trị tại 1 case nào đó thì câu lệnh bên trong case đó được thực thi. Nếu val không giống bất cứ một giá trị trong các case nào thì câu lệnh bên trong default được thực thi.
- ▶▶▶ **Chú ý** : Giá trị của val có thể là số, kí tự, xâu kí tự (sẽ học sau) .Các khối lệnh bên trong các case sẽ được kết thúc bằng câu lệnh break.

SYNTAX

```
switch (val) {  
    case 1:  
        // code  
        break;  
    case 2:  
        //code  
        break;  
    .....  
    case n:  
        // code  
        break;  
    default :  
        // code  
}
```



28TECH

Become A Better Developer

BẢNG MÃ ASCII VÀ CÁC LỆNH LIÊN QUAN



BẢNG MÃ ASCII

0		20	¶	40	<	60	<	80	P	100	d	120	x	140	î	160	á	180		200	£	220		240	≡
1	©	21	§	41	>	61	=	81	Q	101	e	121	y	141	ï	161	â	181		201	¤	221	¡	241	±
2	®	22	¶	42	*	62	>	82	R	102	f	122	z	142	ä	162	ä	182		202	¥	222	¢	242	²
3	™	23	±	43	+	63	?	83	S	103	g	123	{	143	å	163	å	183		203	¦	223	£	243	³
4	◆	24	↑	44	,	64	@	84	T	104	h	124	}	144	æ	164	æ	184		204	§	224	¤	244	´
5	♣	25	↓	45	.	65	A	85	U	105	i	125	~	145	ç	165	ç	185		205	¨	225	¥	245	µ
6	♠	26	→	46	-	66	B	86	V	106	j	126	Δ	146	œ	166	œ	186		206	©	226	¦	246	¶
7		27	←	47	/	67	C	87	W	107	k	127	∇	147	ø	167	ø	187		207	ª	227	§	247	÷
8		28	⊥	48	0	68	D	88	X	108	l	128	∆	148	õ	168	õ	188		208	µ	228	¨	248	·
9		29	⊕	49	1	69	E	89	Y	109	m	129	ç	149	ö	169	ö	189		209	¶	229	©	249	ˆ
10		30	▲	50	2	70	F	90	Z	110	n	130	é	150	ù	170	ù	190		210	⌘	230	µ	250	˜
11	♂	31	▼	51	3	71	G	91	[111	o	131	â	151	û	171	û	191		211	⌘	231	τ	251	¸
12	♀	32		52	4	72	H	92	\	112	p	132	ä	152	ü	172	ü	192		212	⌘	232	θ	252	˙
13		33	!"	53	5	73	I	93]	113	q	133	å	153	Û	173	Û	193		213	⌘	233	Ω	253	˚
14	♫	34	"	54	6	74	J	94	^	114	r	134	æ	154	Ü	174	Ü	194		214	⌘	234	∞	254	¸
15	✳	35	#	55	7	75	K	95	_	115	s	135	ç	155	ē	175	ē	195		215	⌘	235	δ	255	˘
16	▲	36	\$	56	8	76	L	96	`	116	t	136	œ	156	ë	176	ë	196		216	⌘	236	ε		
17	▼	37	%	57	9	77	M	97	a	117	u	137	œ	157	ë	177	ë	197		217	⌘	237	ø		
18	↑	38	&	58	:	78	N	98	b	118	v	138	ë	158	ë	178	ë	198		218	⌘	238	€		
19	::	39	'	59	;	79	O	99	c	119	w	139	ï	159	ë	179	ë	199		219	⌘	239	∞		



BẢNG MÃ ASCII

Bảng mã này có 256 kí tự, mỗi kí tự được gán với 1 mã nhất định gọi là mã ASCII.

Bạn có thể coi kiểu dữ liệu char như số hoặc kí tự đều được, tức là bạn hoàn toàn có thể sử dụng nó để cộng, trừ, nhân, chia

Chú ý: Hãy luôn nhớ khi cộng, trừ, nhân, chia một kí tự nào đó thì mã ASCII của nó sẽ được sử dụng.

Một vài dải kí tự cần lưu ý:

Dải kí tự	Dải mã ASCII
A-Z	65-90
a-z	97-122
0-9	48-57



Một vài câu lệnh kiểm tra kiểu ký tự cần nắm vững:

Câu lệnh	Ý nghĩa
<pre>char c; if ((c >= 'a') && (c <= 'z'))</pre>	Kiểm tra ký tự in thường
<pre>char c; if ((c >= 97) && (c <= 122))</pre>	Kiểm tra ký tự in thường
<pre>char c; if ((c >= 'A') && (c <= 'Z'))</pre>	Kiểm tra ký tự in hoa
<pre>char c; if ((c >= 65) && (c <= 90))</pre>	Kiểm tra ký tự in hoa
<pre>char c; if ((c >= '0') && (c <= '9'))</pre>	Kiểm tra ký tự là chữ số
<pre>char c; if ((c >= 48) && (c <= 57))</pre>	Kiểm tra ký tự là chữ số
<pre>char c = 'A'; c += 32;</pre>	Chuyển ký tự c thành dạng in hoa tương ứng
<pre>char c = 'a'; c -=32;</pre>	Chuyển ký tự c thành dạng in thường tương ứng





28TECH

Become A Better Developer

KẾT THÚC BUỔI HỌC



FINISH

