



Giới thiệu về ngôn ngữ lập trình C

Pay It Forward

C20

1. Giới thiệu về C

- Một chương trình bao gồm:
 - Data: Dữ liệu
 - Statement: Câu lệnh để điều khiển hoặc xử lý data



1.1. Data types

Kiểu char

`char` là kiểu nguyên một byte, kiểu này có thể được sử dụng để khai báo biến, biến đó sẽ chiếm kích thước trong bộ nhớ là 1 byte và có thể giữ một kí tự hoặc một giá trị 8 bit.

Ví dụ

```
char c;  
c = 'a';
```

```
char c;  
c = 89;
```



1.1. Data types

Kiểu int

`int` là một kiểu số nguyên, kiểu này có thể được sử dụng để khai báo biến, biến đó có kích thước trong bộ nhớ là kích thước của số nguyên mà trình biên dịch quy định, có thể là 16 bit hoặc 32 bit.

Ví dụ

```
int i;  
i = 1234;  
i = i + 123;
```



1.1. Data types

Kiểu float và double

float là kiểu số thực dấu chấm động, có độ chính xác đơn (7 kí tự sau dấu chấm thập phân).

double là kiểu số thực dấu chấm động, có độ chính xác kép (15 kí tự sau dấu chấm thập phân).

Ví dụ

```
float f;  
f = 2.045;
```

```
double d;  
d = 3.14;
```



1.1. Data types

Kiểu signed, unsigned, short và long

	signed	unsigned	short	long
char	signed char = char	unsigned char		
int	signed int = int	unsigned int = unsigned	short int = short/int	long int = long
float				long float = double
double				long double

1.1. Data types

Thư viện stdint

```
#include <stdint.h>
```

- int8_t, uint8_t
- int16_t, uint16_t
- int32_t, uint32_t
- int64_t, uint64_t



1.2. Variable (biến):

Tất cả các biến được sử dụng trong một chương trình C đều phải được khai báo trước.

Cú pháp khai báo:

<kiểu_dữ_liệu> <tên_biến>;

Ex:

```
int numOfReceivedChar;  
bool isDeviceReady;
```



1.2.1 Variable (biến):

Tên biến phải tuân thủ các nguyên tắc sau:

- Không trùng với từ khóa.
- Không trùng với danh hiệu chuẩn.
- Có thể bắt đầu bằng các kí tự $a \rightarrow z$, $A \rightarrow Z$ hoặc dấu “_”; tên biến có thể chứa chữ số $0 \rightarrow 9$.

Ex:

```
int lap, count, max, lf, _hoten;  
double he_so_1, delta;  
char ki_tu = 'A';
```



1.2.2 Array (mảng):

Khai báo mảng

Cú pháp khai báo mảng một chiều:

```
kieu_du_lieu ten_mang [kich_thuoc] ;
```

Cú pháp khai báo mảng nhiều chiều:

```
kieu_du_lieu ten_mang  
[kich_thuoc_chieu1][kich_thuoc_chieu2]  
[...] ;
```



1.2.2 Array (mảng):

Khai báo mảng

Ví dụ

```
int a[4] = {3, 6, 7, 8};  
// Giá trị a[0] = 3  
char b[2][3] = {4, 5, 6}, {6, 0, 9};  
// Giá trị b[2][1] = 6  
char s[5] = "Hello";  
// Giá trị s[1] = 'e'  
char s[] = "Hello, world!";
```



1.2.2 Array (mảng):

Mảng là đối số hàm, mảng là biến toàn cục

Khi khai báo đối số của hàm là mảng, kích thước của chiều đầu tiên của mảng không cần xác định cụ thể. Tuy nhiên, cần phải xác định kích thước của chiều thứ hai trở đi.

Mảng có thể được khai báo như biến toàn cục hay biến cục bộ giống như một biến thông thường.

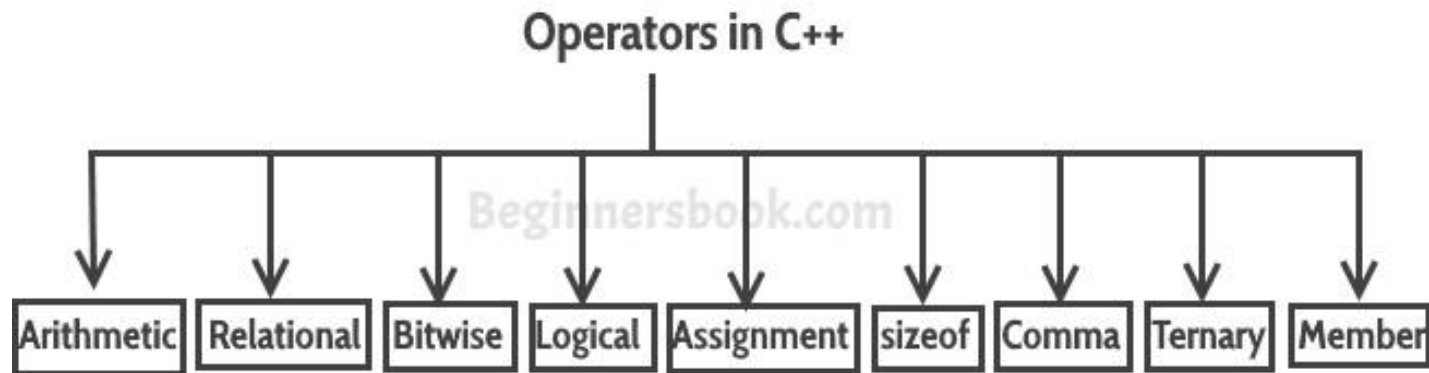
Ví dụ

```
int a[100];  
void sort(int b[][3], int n);  
main() {  
    ...  
    char c[] = "Done!";  
}
```



1.3. Operator – Toán tử

- Arithmetic operators - Toán tử số học
- Relational operators - Toán tử quan hệ
- Logical operators - Toán tử logic
- Bitwise operators – Toán tử trên bit



1.3.1. Arithmetic operators

- Toán tử cộng (+)
- Toán tử trừ (-)
- Toán tử nhân (*)
- Toán tử chia (/)
- Toán tử modulo (%)

Ex:

$$5 + 10 = ?$$

$$7 - 1 = ?$$

$$5 * 3 = ?$$

$$7 \% 3 = ?$$

$$7 / 3 = ?$$



1.3.2. Relational operators

- Toán tử bằng (==)
- Toán tử khác (!=)
- Toán lớn hơn (>)
- Toán nhỏ hơn (<)
- Toán tử lớn hơn hoặc bằng (>=)
- Toán tử nhỏ hơn hoặc bằng (<=)

Ex:

$3 == 3$ $6 < 7$

$3 != 3$ $6 <= 6$

$3 > 2$ $5 >= 4$



1.3.3. Logical operators

- Toán tử not (!)
- Toán tử and (&&)
- Toán tử or (||)

Ex:

2 != 1

0x01 && 0x02

0x00 || 0x02



1.3.4. Logical operators

- Toán tử NOT (\sim)
- Toán tử AND ($\&$)
- Toán tử OR ($|$)
- Toán tử XOR (\wedge)
- Toán tử dịch trái (\ll)
- Toán tử dịch phải (\gg)

Ex:

$\sim 0x21 = ?$

$0x01 \& 0x02 = ?$

$0x02 \gg 1 = ?$

$0x01 | 0x02 = ?$

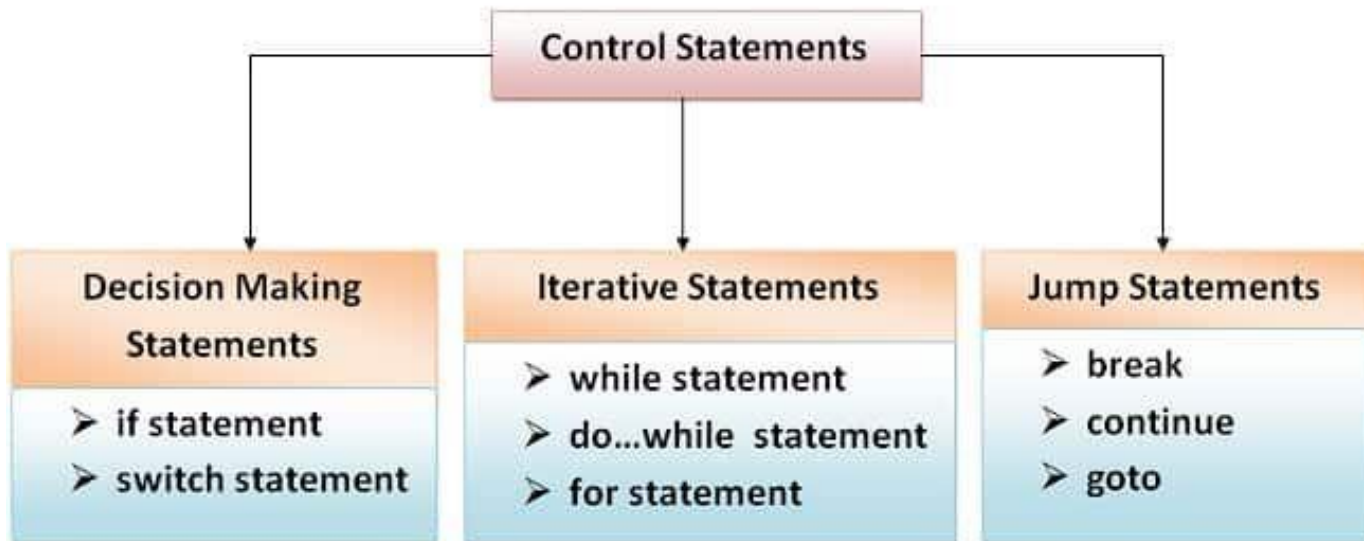
$0x01 \wedge 0x02 = ?$

$0x02 \ll 1 = ?$



1.4. Control statements – Lệnh điều khiển

- Selection statements – Lệnh chọn
- Iteration statements – Lệnh lặp
- Jump statements – Lệnh nhảy



1.4. Selection statements

- Lệnh if:
- Cú pháp lệnh if có 2 dạng:
- Dạng 1:

```
if (<bieu_thuc>)  
    <lệnh>;
```

- Dạng 2:

```
if (<bieu_thuc>){  
    <lệnh_1>;  
}  
else{  
    <lệnh_2>;  
}
```



1.4. Selection statements

- Lệnh if:
- Cú pháp lệnh if có 2 dạng:
- Dạng 1:

```
if (<bieu_thuc>)  
    <lệnh>;
```


- Dạng 2:

```
if (<bieu_thuc>){  
    <lệnh_1>;  
}  
else{  
    <lệnh_2>;  
}
```

- Ví dụ

```
if (x==2)  
    y = 5;
```

```
if (x){  
    y++;  
}  
else{  
    y--;  
}
```

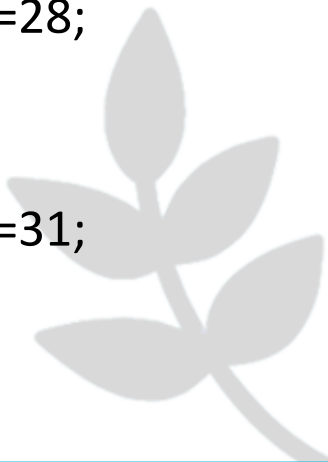


1.4. Selection statements

- Lệnh switch-case
- Cú pháp lệnh:

```
switch (<bieu_thuc>){  
    case <hang_1>:  
        <lenh_1>;  
        break;  
    case <hang_n>:  
        <lenh_n>;  
        break;  
    default:  
        <lenh>;  
        break;  
}
```

```
switch (thang){  
    case 4:  
    case 6:  
    case 9:  
    case 11:  
        so_ngay=30;  
        break;  
    case 2:  
        so_ngay=28;  
        break;  
    default:  
        so_ngay=31;  
        break;  
}
```



1.4. Iteration statements

Lệnh while

Cú pháp lệnh:

```
while(<bieu_thuc>){  
    do_somthing();  
}
```

```
while (x != 0){  
    x--;  
    do_somthing();  
}
```

```
while (1){  
    do_somthing();  
}
```



1.4. Iteration statements

Lệnh do-while

Cú pháp lệnh:

```
do{  
    <lenh>;  
}  
while (<bieu_thuc>);
```

```
do{  
    x--;  
}  
while (x != 0);
```

```
do x++;  
while (1);
```



1.4. Iteration statements

Lệnh for

Cú pháp lệnh:

```
for(<bieu_thuc1>;<bieu_thuc2>;<bieu_thuc3>
){
    <lệnh>;
}
```

Ex:

```
int x;
for (x = 0; x < 10; x++)
    sum += x;
```



1.5. Function

Khai báo hàm

Kieu_du_lieu

```
ten_ham(danh_sach_doi_so){  
    khai_bao_bien_cuc_bo;  
    lenh;  
}
```

Ex:

```
char tong (char a, char b){  
    return (a+b);  
}
```



1.6. C – Preprocessors – Tiền xử lý

Lệnh `#include`

Cho phép đưa một tập tin khác lồng vào tập tin đang làm việc dưới dạng nguồn. Cú pháp:

```
#include "ten_tap_tin"
```

```
#include <ten_tap_tin>
```

Ex:

```
#include <msp430g2553.h>
```

```
#include "LED.h"
```



1.6. C – Preprocessors – Tiền xử lý

Lệnh `#define`

Lệnh `#define` cho phép định nghĩa một chuỗi hay biểu thức bằng một tên cụ thể (tên macro).

Cú pháp:

`#define ten_macro bieu_thuc_thay_the`

Ex:

```
#define MAX 100
```

```
#define LED 0x01
```



 payitforward.edu.vn