# CÁC THÀNH PHẦN TRONG NGÔN NGỮ C

## Comment trong C

Hướng dẫn cách **comment trong C**. Comment trong C được dùng để chú thích nội dung của dòng code viết trong chương trình. Comment trong C sẽ không ảnh hưởng tới kết quả xử lý chương trình, do đó chúng ta có thể tự do viết chúng. Có hai cách **comment trong C** và bạn sẽ học chúng hoàn hảo sau bài viết này.

### Comment trong C là gì

**Comment trong C** hay còn gọi là cách **chú thích trong C** hoặc là **ghi chú trong C** là các dòng code được bỏ qua khi chạy chương trình, nhằm giúp bạn lưu lại các thông tin khi viết chương trình như biến số này dùng làm gì, ai là người tạo và tạo ra lúc nào chẳng hạn. Chỉ cần viết **comment trong C** theo những luật xác định thì dòng comment trong C sẽ không ảnh hưởng tới kết quả chương trình.

Các thông tin cần thiết được lưu giữ bởi **comment trong C** sẽ giúp bạn bảo trì chương trình dễ hơn, cũng như dễ chuyển giao lại dự án cho người khác hoặc là chia sẽ dự án cho nhiều người cùng làm.

### Cách comment trong C

#### Cách comment trên một dòng trong C

Chúng ta sử dụng // để Để comment một dòng trong C. Comment trong C sẽ được tính từ vị trí bắt đầu // cho tới hết dòng.

//dòng comment

Chúng ta có thể bắt đầu comment bằng cách ghi // từ vị trí đầu dòng, hay ở giữa dòng đều được.

|  |
| --- |
| #include <stdio.h>  int main(void){  // người tạo: Xuân Thức  int num;  num = 10;  num = num + 5; // cộng thêm 5 đơn vị  printf("num = %d\n",num);// in ra màn hình  } |

Trong ví dụ trên hai dòng dưới đây chính là comment trong C

|  |
| --- |
| // người tạo: Xuân Thức // cộng thêm 5 đơn vị // in ra màn hình |

#### Cách comment trên nhiều dòng trong C

Để comment trên nhiều dòng trong C, ngoài cách ghi vào đầu từng dòng dấu // như ở cách trên, thì chúng ta có thể dùng một cách đơn giản hơn nữa, đó là viết các dòng chú thích ở giữa cặp dấu /\* và \*/ với cú pháp sau đây:

/\*  
dòng comment 1  
dòng comment 2  
…  
\*/

Chúng ta sử dụng cặp dấu /\* và \*/ để **comment trên nhiều dòng trong C** như ví dụ sau đây:

|  |
| --- |
| #include <stdio.h>  int main(void){  /\*  Ngày viết : 210304  Tác giả : Xuân Thức  \*/   int num = 10;  num = num + 5; /\* cộng thêm 5 đơn vị\*/  printf("num = %d\n",num);// in ra màn hình  } |

Kết quả:

|  |
| --- |
| 15 |

### Sử dụng comment để tạm thời ngừng chạy câu lệnh

Ngoài cách sử dụng **comment trong C** để lưu lại các thông tin cần thiết, bạn cũng có thể dùng comment để ngừng chạy một hoặc nhiều câu lệnh. Cách làm này gọi là **comment out** trong tiếng Anh

Cách sử dụng comment để tạm thời ngừng chạy câu lệnh giống như ví dụ sau đây:

|  |
| --- |
| #include <stdio.h>  int main(void){  int price = 500;  price = price \* 2;  printf("price = %d\n",price); } |

Nếu chạy chương trình trên, chúng ta sẽ có kết quả là:

|  |
| --- |
| price = 1000 |

Nếu chúng ta không muốn chạy dòng lệnh thứ 2 nữa, chúng ta có thể xoá nó. Tuy nhiên nếu chúng ta chỉ tạm thời không muốn chạy dòng lệnh này, hãy comment dòng lệnh thứ hai như sau:

|  |
| --- |
| #include <stdio.h>  int main(void){  int price = 500;  //price = price \* 2;  printf("price = %d\n",price); } |

Khi đó dòng lệnh thứ hai bị ngưng chạy, do đó chỉ có dòng lệnh 1 và 3 được thực thi và kết quả trả về như sau:

|  |
| --- |
| price = 500 |

Nếu sau đó chúng ta muốn tiếp tục chạy dòng code này, chỉ cần đơn giản bỏ đi dấu // trước nó là xong, mà không cần phải viết lại nó nữa.

### Phím tắt comment trong c

Tùy thuộc vào phần mềm viết code C mà bạn dùng thì chúng ta sẽ có các cách dùng phím tắt khác nhau như dưới đây:

#### Phím tắt comment trong Dev C++

Sau khi chọn các dòng code, chúng ta có thể sử dụng các tổ hợp phím sau để comment trong Dev C++:

* Ctrl +. Chuyển vùng code được chọn thành comment
* Ctrl +, Bỏ commet của vùng code được chọn

#### Phím tắt comment trong Notepad ++

Sau khi chọn các dòng code, chúng ta có thể sử dụng các tổ hợp phím sau để comment trong Notepad ++:

* **Ctrl + K**: Chuyển vùng code được chọn thành comment
* **Ctrl + Q**: Bỏ commet của vùng code được chọn

#### Phím tắt comment trong Visual Studio

Sau khi chọn các dòng code, chúng ta có thể sử dụng các tổ hợp phím sau để comment trong Visual Studio:

* **Ctrl + K + C**: Chuyển vùng code được chọn thành comment
* **Ctrl + K + U**: Bỏ commet của vùng code được chọn

#### Phím tắt comment trong Sublime text 3

Sau khi chọn các dòng code, chúng ta có thể sử dụng các tổ hợp phím sau để comment trong Sublime text 3:

* **Ctrl + /**: Chuyển vùng code được chọn thành comment
* **Ctrl + Shift + /**: Bỏ commet của vùng code được chọn

## BIẾN

### Định danh

Định danh là tên được đặt cho các thực thể như biến, hàm, cấu trúc, … Định danh phải là duy nhất. Chúng được tạo ra để đặt một tên duy nhất cho một thực thể để xác định nó trong quá trình thực thi chương trình. Định danh phải khác với từ khóa.

|  |
| --- |
| **VD:** Một số định danh |
| int money;  double accountBalance; |

#### Bảng chữ cái

Mọi ngôn ngữ lập trình đều được xây dựng từ một bộ kí tự nào đó và các quy tắc trên đó để xây dựng các từ, các câu lệnh và cấu trúc chương trình. Ngôn ngữ lập trình C sử dụng bộ ký tự ASCII (American Standard Code for Informations Interchange). Theo chuẩn này, bộ kí tự gồm có 256 kí tự đó là:

* Các chữ cái: A … Z, a .. z
* Các chữ số: 0 1 2 3 4 5 6 7 8 9
* Các dấu phép toán số học: +,-,\*,/,...
* Các ký tự đặc biệt trong lập trình C

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Các ký tự đặc biệt trong lập trình C** | | | | |
| , | < | > | . | ~ |
| ( | ) | ; | $ | : |
| % | [ | ] | # | ? |
| ' | & | { | } | " |
| ^ | ! | \* | / | | |

Mỗi kí tự có tương ứng 1 số duy nhất gọi là mã, trong đó có 128 kí tự đầu (có mã từ 0 tới 127) là kí tự cố định và 128 kí tự còn lại (có mã từ 128 tới 255) là các kí tự mở rộng, tức là nó có thể thay đổi tuỳ theo ngôn ngữ mỗi quốc gia sử dụng.

#### Khoảng trắng

Khoảng trắng là thuật ngữ được sử dụng trong C để mô tả khoảng trống, tab, ký tự dòng mới và nhận xét. Khoảng trắng phân tách một phần của một câu lệnh với phần khác và cho phép trình biên dịch xác định vị trí của một phần tử trong một câu lệnh, chẳng hạn như int, kết thúc và phần tử tiếp theo bắt đầu.

|  |
| --- |
| **VD:** Trong câu lệnh sau, có ít nhất một ký tự khoảng trắng giữa int và age để trình biên dịch có thể phân biệt chúng. |
| int age; |
| **VD:** Trong câu lệnh sau, không có ký tự khoảng trắng nào giữa fruit và =, hoặc giữa = và apples, mặc dù bạn có thể thêm một số ký tự nếu muốn tăng khả năng đọc. |
| fruit=apples + oranges; |

#### Quy tắc đặt định danh

Định danh rất quan trọng trong quá trình lập trình, nó không những thể hiện rõ ý nghĩa trong chương trình mà còn dùng để xác định các đối tượng khác nhau trong chương trình.

* Chiều dài tối đa của định danh là 32 ký tự.
* Định danh hợp lệ là một chuỗi ký tự liên tục gồm:
* Ký tự chữ, số và dấu gạch dưới.
* Ký tự đầu phải là chữ hoặc dấu gạch dưới.
* Không được đặt trùng với các từ khóa.
* Định danh nên được đặt tên theo quy tắc Hungarian Notation, CameCase.

**Lưu ý:** Trong ngôn ngữ C có phân biệt chữ hoa, chữ thường.

|  |  |
| --- | --- |
| **VD:** Một số định danh đúng trong ngôn ngữ C. | |
| **Mã chương trình** | **Ý nghĩa** |
| int hocSinh; | Biến hocSinh có kiểu dữ liệu int |
| int Tuoi; | Biến Tuoi có kiểu dữ liệu int |

|  |  |  |
| --- | --- | --- |
| **VD:** Tên biến phân biệt chữ hoa và chữ thường. | | |
| **Mã chương trình** | | **Ý nghĩa** |
| number | Number | Biến number khác biến Number |
| case | Case | case là từ khóa, Case vẫn đúng |

***1. Quy tắc Hungarian Notation***

**VD**: Ta có biến Tuoi có kiểu dữ liệu int, khi sử dụng quy tắc Hungarian Notation bạn thêm chữ i (kí tự đầu của kiểu dữ liệu) vào đầu tên biến Tuoi để trong quá trình lập trình hoặc sau này xem lại, sửa chữa… bạn dễ dàng nhận ra biến Tuoi có kiểu int mà không cần phải di chuyển đến phần khai báo mới biết kiểu của biến này.

|  |
| --- |
| **VD:** Khai báo biến theo quy tắc ***Hungarian Notation*** |
| int ihocSinh;  double dlaiSuat; |

***2. Quy tắc CameCase***

Quy tắc CameCase nghĩa là nếu tên biến có nhiều hơn hai từ thì bắt đầu từ thứ hai trở đi của tên biến sẽ lên hoa.

|  |
| --- |
| **VD:** Khai báo biến theo quy tắc ***CameCase.*** |
| int ihocSinhGioi;  double dlaiSuat; |

### Từ khóa

Từ khóa là các từ dành riêng được sử dụng trong lập trình có ý nghĩa đặc biệt đối với trình biên dịch đã được xác định trước. Từ khóa là một phần của cú pháp và chúng không thể được sử dụng làm hằng số hoặc biến hoặc bất kỳ tên định danh nào khác.

**Lưu ý**: Ngôn ngữ C phân biệt chữ hoa chữ thường, nên tất cả các từ khóa phải được viết bằng chữ thường.

|  |  |
| --- | --- |
| **Mã chương trình** | **Ý nghĩa** |
| int money; | int là một từ khóa cho biết biến money là một biến kiểu int. |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Danh sách từ khóa** | | | | | | | |
| auto | continue | double | for | int | signed | struct | void |
| break | do | else | if | long | static | switch | while |
| case | default | enum | goto | register | sizeof | typedef | volatile |
| char | const | extern | float | return | short | union | unsigned |

## KIỂU DỮ LIỆU

### Kiểu dữ liệu

Kiểu dữ liệu của một biến còn thể hiện dung lượng không gian mà nó chiếm trong bộ nhớ và cách diễn giải mẫu bit được lưu trữ.

### Phân loại kiểu dữ liệu

Có 3 loại kiểu dữ liệu khác nhau trong ngôn ngữ C.

|  |  |  |
| --- | --- | --- |
| **STT** | **Loại kiểu dữ liệu** | **Mô tả** |
| 1 | Loại Cơ bản | Số nguyên, số thực, ký tự |
| 2 | Loại bool | Đúng hay sai |
| 3 | Loại void | Giá trị trống |

#### Kiểu dữ liệu cơ bản

Ngôn ngữ C có 4 kiểu dữ liệu cơ bản là: char, interger, float, double

|  |  |
| --- | --- |
| **Kiểu dữ liệu cơ bản** | |
| **Tên** | **Kích thước (Byte)** |
| char | 1 |
| int | 4 |
| float | 4 |
| double | 8 |

Kích thước trong bộ nhớ và miền giá trị của các kiểu dữ liệu còn phụ thuộc vào hệ thống và chương trình dịch tương ứng. Giá trị được đưa ra ở đây là trên hệ thống Windows 64 bit và trình dịch GCC MinGW.

##### Kiểu số nguyên

Trong ngôn ngữ C, có 4 công cụ sửa đổi kiểu dữ liệu. các bổ ngữ này kết hợp với các kiểu dữ liệu chính để phân loại phạm vi sử dụng Tối ưu bộ nhớ lưu trữ.

|  |  |  |
| --- | --- | --- |
| **Các bổ ngữ** | **Mục đích** | **Mô tả** |
| signed | Thay đổi khoảng giá trị của byte. | Có dấu |
| unsigned | Không dấu |
| long | Thay đổi giá trị byte. | 4 byte |
| short | 2 byte |

**Bảng:** Giá trị kiểu dữ liệu số nguyên.

|  |  |  |  |
| --- | --- | --- | --- |
| **Tên** | **Mô tả** | **Kích thước** | **Khoảng giá trị** |
| char | Kí tự hoặc số nguyên bé | 1 byte | signed: -  unsigned: 0 |
| short | Số nguyên ngắn | 2 byte | signed: -  unsigned: 0 |
| int | Số nguyên | 4 byte | signed: -  unsigned: 0 |
| long | Số nguyên dài | 4 byte | signed: -  unsigned: 0 |
| long long | Số nguyên cực dài | 8 byte | signed: -  unsigned: 0 |

Ngoài ra, một số kiểu dữ liệu khác như:

|  |  |  |  |
| --- | --- | --- | --- |
| **Tên** | **Tên** | **Kích thước** | **Khoảng giá trị** |
| uint8\_t | unsigned char | 1 | 0 |
| uint16\_t | unsigned short  unsigned short int | 2 | 0 |
| uint32\_t | unsigned int  unsigned long | 4 | 0 |

###### 2.1.2 Kiểu số thực

Kiểu số thực được dùng để chứa những số có dấu phẩy động.

|  |  |  |  |
| --- | --- | --- | --- |
| **Tên** | **Kích thước** | **Khoảng giá trị** | **Độ chính xác** |
| float | 4 | 1.2E-38 to 3.4E+38 | Độ chính xác 6 con số |
| double | 8 | 2.3E-308 to 1.7E+308 | Độ chính xác 15 con số |
| long double | 16 | 3.4E-4932 to 1.1E+4932 | Độ chính xác 19 con số |

###### 2.1.3 Kiểu ký tự

Kiểu ký tự dùng để chứa những ký tự.

|  |  |  |  |
| --- | --- | --- | --- |
| **Tên** | **Mô tả** | **Kích thước** | **Khoảng giá trị** |
| char | Kí tự hoặc số nguyên bé | 1 byte | - hay 0 |
| signedchar | - |
| unsigned char | 0 |

##### 2.3 Kiểu bool

Giá trị 1: là đúng ( true )

Giá trị 0: là sai (false)

##### 2.4 Kiểu void

Kiểu void chỉ định rằng không có giá trị nào. Nó được sử dụng trong ba loại tình huống:

|  |  |
| --- | --- |
| **STT** | **Loại & Mô tả** |
| 1 | ***Hàm trả về dạng void***  Có nhiều hàm khác nhau trong C không trả về bất kỳ giá trị nào hoặc bạn có thể nói rằng chúng trả về giá trị vô hiệu. Một hàm không có giá trị trả về có kiểu trả về là void.  **VD:** void exit (int status); |
| 2 | ***Đối số hàm là void***  Có nhiều hàm khác nhau trong C không chấp nhận bất kỳ tham số nào. Một hàm không có tham số có thể chấp nhận một khoảng trống.  **VD:** int rand (void); |
| 3 | ***Con trỏ void***  Một con trỏ kiểu void \* đại diện cho địa chỉ của một đối tượng, nhưng không phải là kiểu của nó.  **VD:** một hàm cấp phát bộ nhớ void \* malloc (size\_t size); trả về một con trỏ đến void có thể được truyền đến bất kỳ kiểu dữ liệu nào. |

#### 3. Xác định kích thước kiểu dữ liệu

##### 3.1.Toán tử sizeof là gì?

Toán tử sizeof nhận một tham số là bất kỳ kiểu dữ liệu nào và trả về kích thước của kiểu dữ liệu đó.

**Cú pháp:**

|  |
| --- |
| sizeof (Kiểu dữ liệu) |

##### 3.2. Kiểm tra kích thước các kiểu dữ liệu

|  |
| --- |
| **VD:** Xác định kích thước của các kiểu dữ liệu cơ bản trong C. |
| #include<stdio.h>  int main() {      printf("Kich thuoc kieu char: %i\n", sizeof(char));      printf("Kich thuoc kieu int: %i\n", sizeof(int));      printf("Kich thuoc kieu float: %i\n", sizeof(float));      printf("Kich thuoc kieu double: %i\n", sizeof(double));      return 0;  } |

**Kết quả:**

|  |
| --- |
| Kich thuoc kieu char: 1  Kich thuoc kieu int: 4  Kich thuoc kieu float: 4  Kich thuoc kieu double: 8 |

**VD:** Xác định kích thước của các kiểu dữ liệu

|  |
| --- |
| //Xác định kích thước của các kiểu dữ liệu  #include <stdio.h>      int main() {      char c;      short s;      int i;      unsigned int ui;      float f;      double d;      long l;      long long ll;      unsigned long long ull;      // Lấy kích thước từng kiểu dữ liệu//      printf("Size cua kieu char: %i\n",sizeof(c));      printf("Size cua kieu short: %i\n",sizeof(s));      printf("Size cua kieu int: %i\n",sizeof (i));      printf("Size cua kieu unsigned int: %i\n",sizeof (ui));      printf("Size cua kieu float: %i\n",sizeof (f));      printf("Size cua kieu double: %i\n",sizeof (d));      printf("Size cua kieu long: %i\n",sizeof (l));      printf("Size cua kieu long long: %i\n",sizeof (ll));      printf("Size cua kieu unsigned long long: %i\n",sizeof (ull));      return 0;  } |

**Kết quả:**

|  |
| --- |
| Size cua kieu char: 1  Size cua kieu short: 2  Size cua kieu int: 4  Size cua kieu unsigned int: 4  Size cua kieu float: 4  Size cua kieu double: 8  Size cua kieu long: 4  Size cua kieu long long: 8  Size cua kieu unsigned long long: 8 |

**VD:** Xác định kích thước max, min của kiểu dữ liệu số nguyên

|  |
| --- |
| #include <stdio.h>  #include <stdlib.h>  #include <limits.h>  #include <float.h>  int main() {      printf("\n------------Signed -------------\n");      printf("char      - Bit: %-20lli\n", CHAR\_BIT);      printf("char      - max: %-20lli - min: %-20lli\n",CHAR\_MAX, CHAR\_MIN);      printf("short     - max: %-20lli - min: %-20lli\n",SHRT\_MAX, SHRT\_MIN);      printf("int       - max: %-20lli - min: %-20lli\n",INT\_MAX, INT\_MIN);      printf("long      - max: %-20lli - min: %-20lli\n",LONG\_MAX, LONG\_MIN);      printf("long long - max: %-20lli - min: %-20lli\n",LLONG\_MAX, LLONG\_MIN);      printf("\n------------Unsigned -------------\n");      printf("unsigned char      - max: %-20llu\n",UCHAR\_MAX);      printf("unsigned short     - max: %-20llu\n",USHRT\_MAX);      printf("unsigned int       - max: %-20llu\n",UINT\_MAX);      printf("unsigned long      - max: %-20llu\n",ULONG\_MAX);      printf("unsigned long long - max: %-20llu\n",ULLONG\_MAX);      return 0;  } |

**Kết quả:**

|  |
| --- |
| ------------signed -------------  char - Bit: 8  char - max: 127 - min: 4294967168  short - max: 32767 - min: 4294934528  int - max: 2147483647 - min: 2147483648  long - max: 2147483647 - min: 2147483648  long long - max: 9223372036854775807 - min: -9223372036854775808  ------------unsigned -------------  unsigned char - max: 255  unsigned short - max: 65535  unsigned int - max: 4294967295  unsigned long - max: 4294967295  unsigned long long - max: 18446744073709551615 |

Tệp tiêu đề float.h xác định các macro cho phép bạn sử dụng các giá trị này và các chi tiết khác về biểu diễn nhị phân của số thực trong chương trình của bạn. Ví dụ sau sẽ in không gian lưu trữ được thực hiện bởi một kiểu float và các giá trị phạm vi của nó:

**VD:** Xác định kích thước max, min của kiểu dữ liệu số thực

|  |
| --- |
| #include <stdio.h>  #include <stdlib.h>  #include <limits.h>  #include <float.h>  int main() {      printf("float       - max: %-20g - min: %-20g\n",FLT\_MAX, FLT\_MIN);      printf("double      - max: %-20g - min: %-20g\n",DBL\_MAX, DBL\_MIN);      printf("long double - max: %-20g - min: %-20g\n",LDBL\_MAX, LDBL\_MIN);      return 0;  } |

**Kết quả:**

|  |
| --- |
| float - max: 3.40282e+038 - min: 1.17549e-038  double - max: 1.79769e+308 - min: 2.22507e-308  long double - max: 3.20528e-317 - min: 3.20528e-317 |