

4. 資料型態

C++ 語言提供多種資料型態，包含基本型態(fundamental type)與複合資料型態(compound type)兩類。本章僅就基本型態做一說明，複合型態請參閱後續章節。

4.0.1 整數型態

C++ 語言一共有以下8種整數型態：

- Standard signed integer types (標準符號整數型態)
 - short int
 - int
 - long int
 - long long int
- Standard unsigned integer types (標準無符號整數型態)
 - unsigned short int
 - unsigned int
 - unsigned long int
 - unsigned long long int

<note important> ++++ 型態也可以縮寫！我們在宣告整數型態的變數時，可以將int省略，例如：可以short來代表short int，以unsigned代表unsigned int，以long代表long int，long long代表long long int，以及unsigned short代表unsigned short int，unsigned long代表unsigned long int，unsigned long long代表unsigned long long int。++++ </note> 在整數型態的數值範圍方面，都是取決於其所使用的記憶體空間。由於不同平台上可能有差異，C++ 語言僅提供規範，實際情形由各平台上的實作決定。因此，在一個平台上撰寫程式時，我們通常會使用以下的程式，先行瞭解各型態所佔的空間：

```
using namespace std;
#include <iostream>
#include <climits> // use limits.h

int main()
{

    int n_int = INT_MAX;          // initialize n_int to max int value
    short int n_short = SHRT_MAX; // symbols defined in climits file
    long int n_long = LONG_MAX;
    long long int n_llong = LLONG_MAX;

    // sizeof operator yields size of type or of variable
    cout << "int is " << sizeof (int) << " bytes." << endl;
    cout << "short is " << sizeof n_short << " bytes." << endl;
    cout << "long is " << sizeof n_long << " bytes." << endl;
```

```

cout << "long long is " << sizeof n_llong << " bytes." << endl;
cout << endl;

cout << "Maximum values:" << endl;
cout << "int: " << n_int << endl;
cout << "short: " << n_short << endl;
cout << "long: " << n_long << endl;
cout << "long long: " << n_llong << endl << endl;

cout << "Minimum int value = " << INT_MIN << endl;
cout << "Bits per byte = " << CHAR_BIT << endl;

return 0;
}

```

以我們的ws.csie.nptu.edu.tw工作站為例limits.cpp的執行結果如下：

```

[09:52 junwu@ws ch4]$ g++ limits.cpp
[09:52 junwu@ws ch4]$ ./a.out
int is 4 bytes.
short is 2 bytes.
long is 8 bytes.
long long is 8 bytes.

Maximum values:
int: 2147483647
short: 32767
long: 9223372036854775807
long long: 9223372036854775807

Minimum int value = -2147483648
Bits per byte = 8
[09:52 junwu@ws ch4]$

```

上述程式，主要使用了sizeof()函式以及位在climits標頭檔中的定義，有關climits標頭檔，可至/usr/include/c++目錄下查詢。下表為climits中所定義的部份常數：

Symbolic Constant	Represents
CHAR_BIT	Number of bits in a char
CHAR_MAX	Maximum char value
CHAR_MIN	Minimum char value
SCHAR_MAX	Maximum signed char value
SCHAR_MIN	Minimum signed char value
UCHAR_MAX	Maximum unsigned char value
SHRT_MAX	Maximum short value
SHRT_MIN	Minimum short value
USHRT_MAX	Maximum unsigned short value
INT_MAX	Maximum int value

Symbolic Constant	Represents
INT_MIN	Minimum int value
UINT_MAX	Maximum unsigned int value
LONG_MAX	Maximum long value
LONG_MIN	Minimum long value
ULONG_MAX	Maximum unsigned long value
LLONG_MAX	Maximum long long value
LLONG_MIN	Minimum long long value
ULLONG_MAX	Maximum unsigned long long value

Tab. 1: 在climits中定義的常數

4.0.2 浮點數

C++語言中有3種浮點數的型態：float, double與long double分別實作了IEEE 754當中的單精確度、倍精確度與擴充精確度：

- float: 單精確度浮點數(single-precision floating-point)
- double: 倍精確度浮點數(double-precision floating-point)
- long double: 擴充精確度浮點數(extended-precision floating-point)

4.0.3 字元型態

所謂的字元型態就是用以表示文字、符號等資料，在C/C++語言中只有一種字元型態：

- char

在不同的系統中，字元的數值可能會代表不同意義，視其所採用的字元集(character set)而定。現行最常見的字元集為ASCII(American Standard Code for Information Interchange)請參考[Wikipedia關於ASCII的說明](#)

既然char型態就是整數，那可不可以再配合unsigned使用呢？因為char型態的整數數值是用以對應特定的字元集(如ASCII)而每個字元集都有其可表達的字元個數要求C++語言會自動將char定義為signed或unsigned以符合字元集的需求。因此我們通常不會特別在char前加上unsigned但是，如果您有某些較小的整數資料要處理，就可以考慮使用char來代替int因為int為32 bits甚至short int也要使用到16 bits若您只需要處理一些介於-128到127之間的數值，那您就可以考慮改用char來代替int或是宣告為unsigned char來處理那些介於0到255的正整數資料。

4.0.4 布林型態

布林型態為C++所新增的資料型態，其名稱為bool一個bool型態的資料只可能有true或false兩種可能的數值。與傳統的C語言一樣，若你要以整數來表達bool型態的值，則以0表示false其它非0的值皆視為true

```
bool isQuit = false;

int continueProcess = true; // 將true轉換為1
```

4.0.5 資料型態轉換

如果在程式碼中，我們想要把某個數值之型態加以轉換，可以使用顯式型態轉換(explicit conversion)來對數值進行強制的轉型(casting)[]使用的方法很簡單，只要在想要轉型的數值前加上一組()其中指定欲轉換的型態即可，例如：

```
int x;  
long int y;  
  
y=(long)x;  
y = (long)(x+837);  
x = (int)sizeof(int);
```

From:

<https://junwu.nptu.edu.tw/dokuwiki/> - Jun Wu的教學網頁

國立屏東大學資訊工程學系

CSIE, NPTU

Total: 122236

Permanent link:

<https://junwu.nptu.edu.tw/dokuwiki/doku.php?id=cpp:object>

Last update: **2022/02/24 11:37**

