

ACPS C++ 教學講義 02

C++ 基本的輸入輸出

C++ 標準庫提供了一組豐富的輸入/輸出功能，我們將討論 C++ 程式中最基本和最常見的 I/O 操作。

C++ 的 I/O 發生在流中，流是字節序列。

如果字節流是從設備（如鍵盤、磁碟驅動器、網路連接等）流向**記憶體**，這叫做**輸入操作**。

如果字節流是從記憶體**流向設備**（如顯示器、印表機、磁碟驅動器、網路連接等），這叫做**輸出操作**。

I/O 頭文件

下列的頭文件在 C++ 程式中很重要。

頭文件	函數和描述
<code><iostream></code>	該文件定義了 <code>cin</code> 、 <code>cout</code> 、 <code>cerr</code> 和 <code>clog</code> 對象，分別對應於 標準輸入流 、 標準輸出流 、 非緩衝標準錯誤流 和 緩衝標準錯誤流 。
<code><fstream></code>	該文件為用戶控制的文件處理宣告服務。

標準輸出流（cout）

預定義的對象 `cout` 是 `ostream` 類別的一個實例。

`cout` 對象"連接"到標準輸出設備，通常是顯示器。`cout` 是與流插入運算符號 `<<` 結合使用的，如下所示：

```
1  #include <stdio.h>
2  #include <stdlib.h>
3  #include <iostream>
4  using namespace std;
5
6  int main()
7  {
8      char str[] = "Hello C++";
9      cout << "Value of str is : " << str << endl;
10
11     cout<<endl;
12     system("pause");
13     return 0;
14 }
15
```

當上面的程式碼被編譯和執行時，它會產生下列結果：

```
Value of str is : Hello C++
```

C++ 編譯器根據要輸出變數的資料類型，選擇合適的流插入運算符號來顯示值。<< 運算符號被重載來輸出內建類型（整數型、浮點型、double 型、字串和指標）的資料項。

流插入運算符號 << 在一個語句中可以多次使用，endl 用於在行末添加一個換行符號。

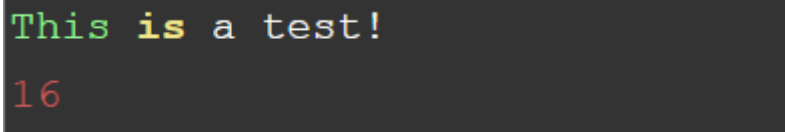
printf()

C 藉由 printf() 將訊息輸出至主控台，這個標準 C 的輸出函數，在 C++ 中仍可正常使用。在 C 中標準輸入輸出是由 stdio.h 提供，所以在程式的一開頭要加上：

```
#include <stdio.h>
```

```
1  #include <stdio.h>
2  #include <stdlib.h>
3  #include <iostream>
4  using namespace std;
5
6  int main()
7  {
8      int count = printf("This is a test!\n");
9      printf("%d\n", count);
10
11     cout<<endl;
12     system("pause");
13     return 0;
14 }
15
```

printf()可以傳回顯示的字元數量，"This is a test!\n" 當中包括換行字元，共有 16 個字元，因此 count 的值會是 16，顯示結果如下：



```
This is a test!
16
```

如果在使用 printf 時要指定整數、浮點數、字元等進行顯示，要配合格式指定字 (format specifier)，以下列出幾個可用的格式指定碼：

%c：以字元方式輸出

%d：10 進位整數輸出

%f：浮點數輸出

%s：字串輸出

%o：以 8 進位整數方式輸出

%u：無號整數輸出

%x、%X：將整數以 16 進位方式輸出

%e、%E：使用科學記號顯示浮點數

%%：顯示 %

%p：指標型態

- 要顯示什麼樣的資料型態，就必須搭配對應資料型態的格式指定字

```
1  #include <stdio.h>
2  #include <stdlib.h>
3  #include <iostream>
4  using namespace std;
5
6  int main()
7  {
8      printf("顯示字元 %c\n", 'A');
9      printf("顯示字元編碼 %d\n", 'A');
10     printf("顯示字元編碼 %c\n", 65);
11     printf("顯示十進位整數 %d\n", 15);
12     printf("顯示八進位整數 %o\n", 15);
13     printf("顯示十六進位整數 %X\n", 15);
14     printf("顯示十六進位整數 %x\n", 15);
15     printf("顯示科學記號 %E\n", 0.001234);
16     printf("顯示科學記號 %e\n", 0.001234);
17
18     cout<<endl;
19     system("pause");
20     return 0;
21 }
22
```

顯示結果如下：

```
顯示字元 A
顯示字元編碼 65
顯示字元編碼 A
顯示十進位整數 15
顯示八進位整數 17
顯示十六進位整數 F
顯示十六進位整數 f
顯示科學記號 1.234000E-03
顯示科學記號 1.234000e-03
```

- 可以在輸出浮點數時指定精度，例如若為浮點數：

```
1  #include <stdio.h>
2  #include <stdlib.h>
3  #include <iostream>
4  using namespace std;
5
6  int main()
7  {
8      printf("example:%6.2f\n", 19.234);
9      printf("example:%-6.2f\n", 19.234);
10
11     cout<<endl;
12     system("pause");
13     return 0;
14 }
15
```

整數 6 表示預留 6 個字元寬度，由於預留了 6 個字元寬度，不足的部份要由空白字元補上

執行結果 19.23 只佔五個字元，所以補上一個空白在前端

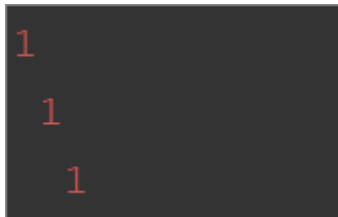
若在 % 之後指定負號，例如 %-6.2f，表示靠左對齊，沒有指定則靠右對齊

```
example: 19.23
example:19.23
```

- 若事先無法決定字元寬度，則可以使用 *，例如：

```
1  #include <stdio.h>
2  #include <stdlib.h>
3  #include <iostream>
4  using namespace std;
5
6  int main()
7  {
8      printf("%*d\n", 1, 1);
9      printf("%*d\n", 2, 1);
10     printf("%*d\n", 3, 1);
11
12     cout<<endl;
13     system("pause");
14     return 0;
15 }
16
```

printf() 的 * 將被之後的第一個引數所取代，所以第一個 printf() 將預留一個字元寬度，第二個預留兩個字元寬度，第三個預留三個，顯示結果如下：

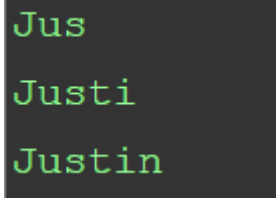


```
1
 1
  1
```

- 若是字串的話，也可以使用 `%.*s`，這表示要顯示字串中 0 到多個字元，實際的字元數可以在第二個參數指定，例如：

```
1  #include <stdio.h>
2  #include <stdlib.h>
3  #include <iostream>
4  using namespace std;
5
6  int main()
7  {
8      printf("%.3s\n", "Justin");
9      printf("%.5s\n", "Justin");
10     printf("%.7s\n", "Justin");
11
12     cout<<endl;
13     system("pause");
14     return 0;
15 }
16
```

執行結果如下：



```
Jus
Justi
Justin
```

cout 控制輸出的寬度與小數點位數

必須引用 `<iomanip>` 標頭檔

```
1  #include <stdio.h>
2  #include <stdlib.h>
3  #include <iostream>
4  #include <iomanip>
5  using namespace std;
6
7  int main()
8  {
9      cout << setprecision(3) << 3.123 << endl;
10     cout << fixed << setprecision(3) << 3.1234 << endl;
11     cout << fixed << setprecision(3) << setw(3) << 3.1234 << endl;
12     cout << fixed << setprecision(3) << setw(10) << 3.1234 << endl;
13     cout << fixed << setprecision(3) << setfill('*') << setw(10) << 3.1234 << endl;
14
15     cout << endl;
16     system("pause");
17     return 0;
18 }
19
```

標準輸入流 (cin)

預定義的對象 cin 是 istream 類別的一個實例。

cin 對象附屬到標準輸入設備，通常是鍵盤。

cin 是與流提取運算符號 >> 結合使用的，如下所示：

```
1  #include <stdio.h>
2  #include <stdlib.h>
3  #include <iostream>
4  using namespace std;
5
6  int main()
7  {
8      char name[50];
9
10     cout << "請輸入您的名字： ";
11     cin >> name;
12     cout << "您的名字是： " << name << endl;
13
14     cout << endl;
15     system("pause");
16     return 0;
17 }
18
```


當上面的程式碼被編譯和執行時，它會提示用戶輸入名稱。當用戶輸入一個值，並按 Enter 鍵，就會看到下列結果：

```
請輸入您的名字： HuaHero
您的名字是： HuaHero
```

C++ 編譯器根據要輸入值的資料類型，選擇合適的流提取運算符號來提取值，並把它儲存在給定的變數中。

流提取運算符號 >> 在一個語句中可以多次使用，如果要求輸入多個資料，可以使用如下語句：

```
cin >> name >> age;
```

這相當於下面兩個語句：

```
cin >> name;
cin >> age;
```

scanf()

也可以使用 C 標準輸入的 scanf 函式，並搭配格式指定字與 & 取址運算子指定給變數，例如：

```
1  #include <stdio.h>
2  #include <stdlib.h>
3  #include <iostream>
4  using namespace std;
5
6  int main()
7  {
8      int input;
9
10     printf("請輸入數字：");
11     scanf("%d", &input);
12     printf("你輸入的數字：%d\n", input);
13
14     cout<<endl;
15     system("pause");
16     return 0;
17 }
```

執行結果：

```
請輸入數字：10
你輸入的數字：10
```

scanf 在接受輸入時，可以接受多個值，也可以指定輸入的格式，例如：

```
1  #include <stdio.h>
2  #include <stdlib.h>
3  #include <iostream>
4  using namespace std;
5
6  int main()
7  {
8      int number1, number2;
9
10     printf("請輸入兩個數字，中間使用空白區隔)：");
11     scanf("%d %d", &number1, &number2);
12     printf("你輸入的數字：%d %d\n", number1, number2);
13
14     printf("請再輸入兩個數字，中間使用-號區隔)：");
15     scanf("%d-%d", &number1, &number2);
16     printf("你輸入的數字：%d-%d\n", number1, number2);
17
18     cout<<endl;
19     system("pause");
20     return 0;
21 }
22
```

執行結果：

```
請輸入兩個數字，中間使用空白區隔)：10 20
你輸入的數字：10 20
請再輸入兩個數字，中間使用-號區隔)：30-40
你輸入的數字：30-40
```

scanf 還可以指定可接受的字元集合，例如若只想接受 1 到 5 的字元：

```
1  #include <stdio.h>
2  #include <stdlib.h>
3  #include <iostream>
4  using namespace std;
5
6  int main()
7  {
8      char buf[50];
9
10     printf("請輸入 1 到 5 的字元：");
11     scanf("%[1-5]", buf);
12     printf("輸入的字元為 %s\n", buf);
13
14     cout<<endl;
15     system("pause");
16     return 0;
17 }
18
```

執行結果：

```
請輸入 1 到 5 的字元：146731
輸入的字元為 14
```