國立屏東大學 資訊工程學系 物件導向程式設計

12. 字串

字串(string)也是一種常見的複合型態，它是一些字元的集合C++有兩種處理字串的方法，其一為傳統的C語言所採用的方式：用char陣列或是指標，其二為新的string類別。

12.1 C++語言的字串陣列或字串指標

關於C語言原本就支援的字串處理方法，我們在此不做贅述，同學可以自行參考C語言的字串。我們僅就C++對於字串陣列或字串指標的處理加以討論。

C++語言可以使用cout輸出字串，若有兩個以上的字串常值(String Literals)要輸出時可以直接串接。以下程式碼是等價的：

```
cout << "Hello " << "C++";
cout << "Hello C++";
cout << "Hello " "C++";
cout << "Hello "
"C++";
```

我們可以cin取得使用者輸入的字串，請參考下例：

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

int main()
{
  char name[20];

  cout << "Please input your name: ";

  cin >> name;
  cout << "Hello, " << name << endl;
  return 0;
}
```

此程式的執行結果如下：

```
[14:32 user@ws home] ./a.out
Please input your name: Jack
```

```
Hello, Jack
[14:32 user@ws home]
```

看起來沒問題，但如果我們輸入的名字中含有空白字元，其結果將不正確，例如：

```
[14:32 user@ws home] ./a.out
Please input your name: Jack Lin
Hello, Jack
[14:32 user@ws home]
```

注意到了嗎？那個空白字元及其後的Lin都不見了。其實cin是C++的一個物件，你應該使用它的方法來取得字串，例如get()或getline()□關於物件及其方法的概述，我們將在以後的章節再行說明。

使用cin.get()或cin.getline()可以取得使用者的輸入直到遇到換行為止，我們以cin.getline()為例，改寫上述程式如下：

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

int main()
{
  char name[20];

  cout << "Please input your name:";

  cin.getline(name, 20);

  cout << "Hello, " << name <<  endl;
  return 0;
}
```

使用getline()方法需要兩個引數，第一個是存放輸入的字串，第二個則為字串的長度(最後一個字元存放「\0」)。要注意getline()會將最後的換行字元從標準輸入讀入，但不會放在輸入結果內□get()則會將該換行字元留在標準輸入內，因此下一次的輸入將有可能遇到問題。請參考下面的程式：

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

int main()
{
  char firstname[20];
  char lastname[20];

  cout << "Please input your first name: ";
  cin.get(firstname, 20);

  cout << "Please input your last name: ";
```

```
  cin.get(lastname, 20);

  cout << "Hello, " << firstname << " " << lastname <<  endl;
  return 0;
}
```

測試看看，上面這個程式會遇到什麼問題？ 要解決這個問題，可以在第一個cin.get()後再加上一個cin.get()將換行字元清除：

```
cin.get(firstname, 20);
cin.get();
```

或是

```
cin.get(firstname, 20).get();
```

下面這個程式解決了上述問題：

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

int main()
{
  char firstname[20];
  char lastname[20];

  cout << "Please input your first name: ";
  cin.get(firstname, 20).get();

  cout << "Please input your last name: ";
  cin.get(lastname, 20);

  cout << "Hello, " << firstname << " " << lastname <<  endl;
  return 0;
}
```

再考慮下一個程式：

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

int main()
{
```

```
  char firstname[20];
  char lastname[20];
  int lucky;

  cout << "Please input a lucky number: ";
  cin >> lucky;

  cout << "Please input your first name: ";
  cin.get(firstname, 20).get();

  cout << "Please input your last name: ";
  cin.get(lastname, 20);

  cout << "Hello, " << firstname << " " << lastname <<  endl;
  return 0;
}
```

執行看看，會發生什麼結果？為解決此問題，我們可以在取得字串前，先以一個cin.get()來將前一個輸入數字時所遺留下來的換行清除掉，請參考下例:

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

int main()
{
  char firstname[20];
  char lastname[20];
  int lucky;

  cout << "Please input a lucky number: ";
  (cin >> lucky).get();
//  cin >> lucky;
//  cin.get();

  cout << "Please input your first name: ";
  cin.get(firstname, 20).get();

  cout << "Please input your last name: ";
  cin.get(lastname, 20);

  cout << "Hello, " << firstname << " " << lastname <<  endl;
  return 0;
}
```

最後，如果你要使用原本C語言相關的字串處理函式時，請記得載入□cstring□□請參考下面的程式：

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

```
#include <cstring>

int main()
{
  char *str1="Hello";
  char *str2=" World";
  char str3[20];

  cout << "str1=" << str1 << endl;
  cout << "str2=" << str2 << endl;

  strcat(str3, str1);
  strcat(str3, str2);

  cout << "The length of str1 is " << strlen(str1) << endl;
  cout << "str1+str2= " << str3 << endl;
  return 0;
}
```

12.2 動態字串

你也可以使用 new 及「delete 來動態地建立與回收字串所需的記憶體空間，請參考下面這個程式:

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

char * getUserName(void);

int main()
{
  char *name;
  name=getUserName();
  cout << "Hello, " << name << "!\n";
  delete [] name;

  return 0;
}

char * getUserName()
{
  char temp[80];
  cout << "Enter your name: ";
  cin.getline(temp, 80);
//  cin >> temp;
```

```
  char *pName = new char[ strlen(temp) + 1];
  strcpy(pName, temp);

  return pName;
}
```

12.3 string類別

不同於C語言，C++提供了一個string類別，我們可以利用此類別進行字串的操作。以string類別進行字串處理，必須先宣告產生string類別的物件，然後才可以使用。使用string類別須載入string(#include <string>)下面的程式顯示一個簡單的例子：

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

int main()
{
  string str;

  cout << "Enter your name: ";
  getline(cin, str);

  cout << "Hi, " << str << "!" << endl;
  cout << "The third letter in your name is " << str[2] << "." << endl;

  return 0;
}
```

要注意的是，我們使用的是在string中的getline()函式，而非cin.getline()，這兩者是不同的，getline()函式的第一個引數為所要取得輸入的資料流物件(也就是透過cin來取得)，第二個引數為所要儲存的字串(string類別的物件)，string類別的物件，也可以被當成是字串陣列，直接以陣列的索引值加以存取。以下我們將詳細說明string類別的字串物件：

12.3.1 字串初始化

其實，在還沒開始學習何謂物件、何謂類別之前，我們可以暫時把string類別視為一種資料型態(當然是比較特別的資料型態)，其字串的宣告與初始化就如同一般的變數宣告一樣：

```
string strName = "Hello";
string strName ("Hello");
string strName = {"Hello"};
string strName {"Hello"};
```

12.3.2 字串操作

string類別的字串物件可以進行各式的操作，包含assignment、concatenation、appending等，請參考以下的程式碼：

```
string str1;
string str2 {"Hello"};
string str3;

str1 = str2; //assignment
str3 = str1 + str2; //assign str3 the joined strings
str1 += str2; //ass str2 to the end of str1
```

另外、string類別的物件還提供許多操作的方法，包含：

- size - 傳回字串的長度
- length - 傳回字串的長度
- clear - 清除字串內容
- empty - 查詢字串內容是否空白
- find - 查詢特定子字串出現在字串中的位置
- substr - 依條件產生子字串
- compare - 比較字串內容
- c_str - 產生傳統C語言的字串

假如要使用上述的方法，則以物件名稱加上「.」方法名稱即可，請參考下面的範例：

```
string str1;
getline(cin, str1);
cout << str1.size() << endl;
```

關於string類別更完整的資料，可以至C++ Reference查詢。