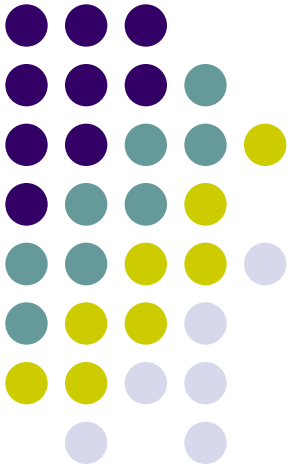
第三章
變數與資料處理

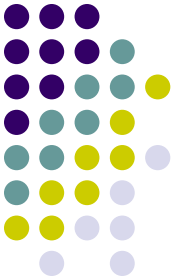
認識常數與變數

學習C++所提供的各種基本資料型態

瞭解溢位的發生

學習認識資料型態之間的轉換



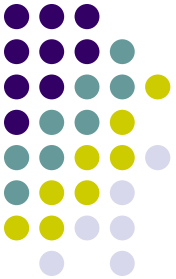


簡單的實例

- 下面是一個簡單的變數使用範例

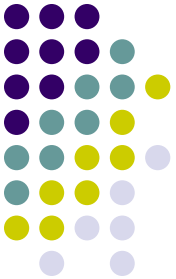
```
01  // prog3_1, 簡單的實例
02  #include <iostream>
03  #include <cstdlib>
04  using namespace std;
05  int main(void)
06  {
07     char ch='w';          // 宣告 ch 為字元，並設值為 w
08     int num=6;            // 宣告 num 為整數，並設值為 6
09
10     cout << ch << " is a character\n";
11     cout << num << " is an integer\n";
12     system("pause");
13     return 0;
14  }
```

```
/* prog3_1 OUTPUT---

w is a character
6 is an integer

-------------------*/
```



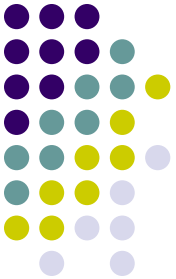
變數(1/3)

- 變數配置
  - 變數具有四個形成要素
    - 名稱：變數在程式中使用的名字，最好取個有意義的名稱，並且需符合變數命名規則。
    - 值：程式中所代表的值，變數值可隨時改變。
    - 參考位址：變數在記憶體中的儲存位址。
    - 資料型別：例如整數、字串、浮點數等。



變數(2/3)

- 變數名稱
  - 變數命名必須遵守下列規則，否則在程式編譯時會產生錯誤
    - 變數名稱的第一個字母必須是大小寫字母、_、中文
    - 變數名稱不能與 C++ 的保留字相同。
    - 變數名稱不能包含空白字元及特殊字元，例如：~、\、@、?、%、&、#等。
    - 變數名稱中英文字母的大小寫是有區別的，例如 APPLE、apple 與 AppLE 皆為不同的變數。



變數(3/3)

- 變數名稱
  - 錯誤變數名稱的範例

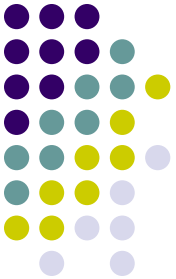
錯誤變數名稱	錯誤原因
8Cake	第一個字元不能是數字
R&D	包含特殊字元「&」
Dr Epson	包含空白字元
short	C++的保留字
_(king)	「_」後必須為數字或字母

C++ 的基本資料型態

- 下表列出基本資料型態所佔的記憶體空間及範圍

型別名稱	資料種類	記憶體大小	範圍
char	具符號位元組	1位元組	-128 到 127
unsigned char	無符號位元組	1位元組	0 到 255
short	具符號短整數	2位元組	-32768到 32767
unsigned short	無符號短整數	2位元組	0 到65535
int	具符號整數	4位元組	-2,147,483,648到 2,147,483,647
unsigned int	無符號整數	4位元組	0 到 4,294,967,295
long	具符號長整數	4位元組	-2,147,483,648到 2,147,483,647
unsigned long	無符號長整數	4位元組	0 到 4,294,967,295
float	單精度浮點數	4位元組	負數：-3.402×10^{38}到 -1.401×10^{-45}

float 資料型別（單精度）的有效位數為 7 位數，如果數值資料大於 7 位數將以科學記號方式處理。
double 資料型別（雙精度）的有效位數為 15 位數，如果數值資料大於 15 位數將以科學記號方式處理。



整數型態 (1/2)

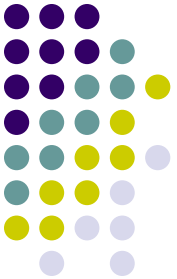
- 若要宣告變數sum為短整數，可利用下面的語法：

```
short int sum;                    // 宣告 sum 為短整數
```

- 若要宣告變數num為無號整數，可利用下面的語法：

```
unsigned int num;                 // 宣告 num 為無號整數
```

整數型態 (2/2)



```
01  // prog3_2, 印出各種資料型態的長度
02  #include <iostream>
03  #include <cstdlib>
04  using namespace std;
05  int main(void)
06  {
07     //定義各種資料型態的變數
08     unsigned int i=0;
09     unsigned short int j=0;
10     char ch=' ';
11     float f=0.0f;
12     double d=0.0;
13
14     //印出各種資料型態的長度
15     cout << "sizeof(int)=" << sizeof(int) << endl;
16     cout << "sizeof(long int)=" << sizeof(long int) << endl;
17     cout << "sizeof(unsigned int)=" << sizeof(i) << endl;
18     cout << "sizeof(short int)=" << sizeof(short int) << endl;
19     cout << "sizeof(unsigned short int)=" << sizeof(j) << endl;
20     cout << "sizeof(char)=" << sizeof(ch) << endl;
21     cout << "sizeof(float)=" << sizeof(f) << endl;
22     cout << "sizeof(double)=" << sizeof(d) << endl;
23     cout << "sizeof(bool)=" << sizeof(bool) << endl;
24     system("pause");
25     return 0;
26  }
```

- 印出各種型態所佔用的位元組長度

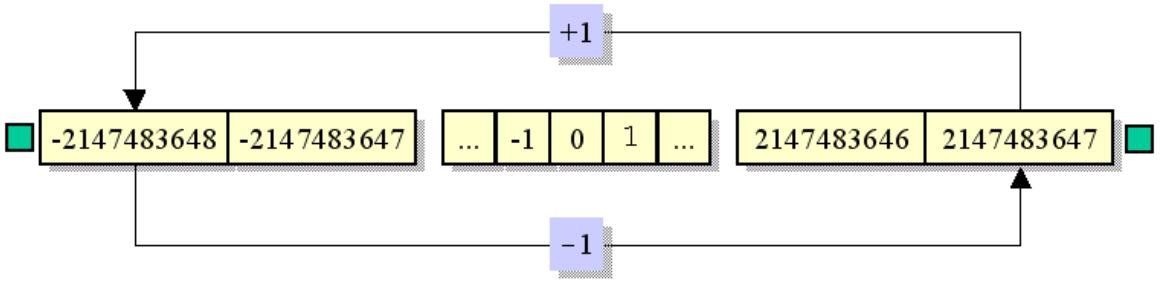
```
/* prog3_2 OUTPUT----------

sizeof(int)=4
sizeof(long int)=4
sizeof(unsigned int)=4
sizeof(short int)=2
sizeof(unsigned short int)=2
sizeof(char)=1
sizeof(float)=4
sizeof(double)=8
sizeof(bool)=1

--------------------------*/
```

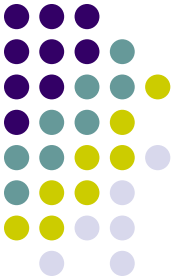
整數資料型態的溢位

- 下面的程式範例可用來瞭解溢位的發生情形

```
01  // prog3_3, 整數資料型態的溢位
02  #include <iostream>
03  #include <cstdlib>
04  using namespace std;
05  int main(void)
06  {
07     int i=2147483647;              //宣告 i 為整數，並設值為 2147483647
08
09     cout << "i=" << i << endl;                          // 印出 i 的值
10     cout << "i+1=" << i+1 << endl;                      // 印出 i+1 的值
11     cout << "i+2=" << i+2 << endl;                      // 印出 i+2 的值
12     system("pause");
13     return 0;
14  }
```

```
/* prog3_3 OUTPUT---

i=2147483647
i+1=-2147483648
i+2=-2147483647

--------------------*/
```

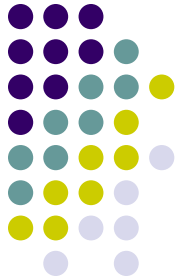




字元型態 (1/5)

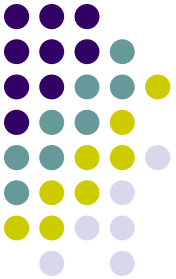
- 字元型別代表使用一個位元組 (Byte) 來儲存字元，其實它儲存的是字元的 ASCII 數值，換句話說，char 儲存的是 -128~127 的整數。
- 字元常使用單引號「'」括住，且只能有一個字元，也可以使用數值直接設定。例如：

- char chrA; //每個字元佔一個位元組
- chrA = 'a'; //以 ASCII=97 值儲存，chrA的值為「a」
- chrA = 97; //字元變數也可設定數值，chrA的值為「a」
- chrA = 0x61; //也可使用十六進位，chrA的值為「a」
- chrA = '7'; //以 字元7 值儲存



字元型態 (2/5)

- 如果字元型別中包含多個字元，則變數值為最後一個字元。
  - chrA = 'abcdef'; // 包含多個字元，chrA的值為「f」

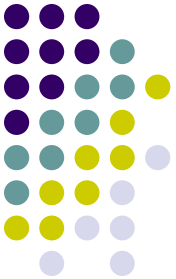


字元型態 (3/5)

- 下面的程式，分別以不同的格式來列印字元h：

```
01  // prog3_4, 字元型態的列印
02  #include <iostream>
03  #include <cstdlib>
04  using namespace std;
05  int main(void)
06  {
07     char ch='h';                          // 定義一個名為 ch 的字元，其值為 h
08     int i=ch;
09     cout << "ch=" << ch << endl;                    // 印出 ch 的值
10     cout << "The ASCII code is " << i << endl;     // 印出 ASCII 值
11     system("pause");
12     return 0;
13  }
```

```
/* prog3_4 OUTPUT-----

ch=h
The ASCII code is 104

----------------------*/
```



字元型態 (4/5)

- 把字元變數以相對應的ASCII碼列印出來

```
01  // prog3_5, 字元的列印
02  #include <iostream>
03  #include <cstdlib>
04  using namespace std;
05  int main(void)
06  {
07     char ch='8';                          // 將'8'設給字元變數 ch
08     int i=ch;
09     cout << "ch=" << ch << endl;           // 印出 ch 的值
10     cout << "The ASCII code is " << i << endl;
11     system("pause");
12     return 0;
13  }
```

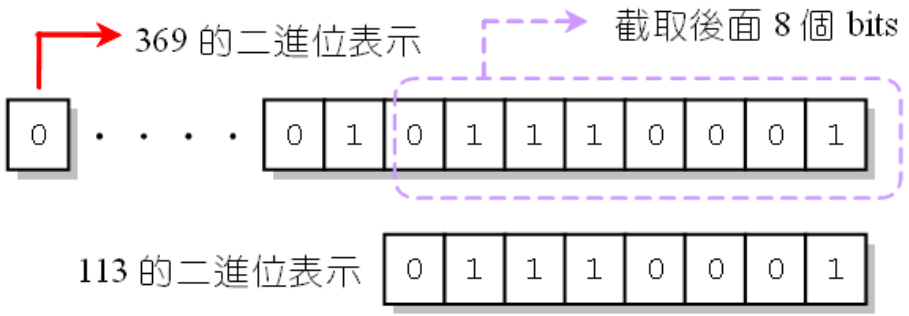
```
/* prog3_5 OUTPUT----

ch=8
The ASCII code is 56

--------------------*/
```

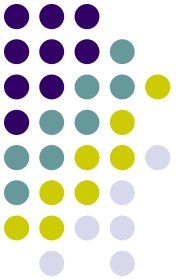
字元型態 (5/5)

- 列印超過字元型態可表示範圍的例子：

```
01  // prog3_6, 字元型態的列印
02  #include <iostream>
03  #include <cstdlib>
04  using namespace std;
05  int main(void)
06  {
07     int i=369;                          // 宣告整數變數 i,其值為 369
08     char ch=i;                          // 將 i 的值設給字元變數 ch
09     cout << "ch=" << ch << endl;         // 印出 ch 的值
10     system("pause");
11     return 0;
12  }
```

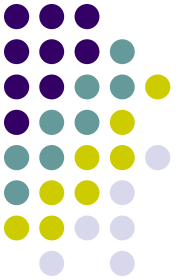
```
/* prog3_6 OUTPUT---
ch=q
--------------------*/
```





基本資料型態

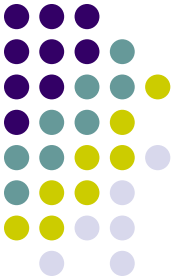
- 數值型別
  - 在 C++ 除了常用的十進位數值外，也可以使用八進位或十六進位，八進位表示法是在數字 0 後面加上八進位數字 (0~7) ，十六進位是使用 0x 或 0X (數字0) 後面加上十六進位數字 (0~9，A~F) 表示十六進位。例如：
    - 054;　// 八進位表示相當於十進位數 44
    - 0x54;　// 十六進位表示相當於十進位數 84



跳脫字元與跳脫序列 (1/5)

- 反斜線「\」稱為跳脫字元（escape character）
- 「\」加上控制碼，稱為跳脫序列（escape sequence）

跳脫序列	所代表的意義	ASCII 十進位值	ASCII 十六進位值
\a	警告音(Alert)	7	0x7
\b	倒退一格(Backspace)	8	0x8
\n	換行(New line)	10	0xA
\r	歸位(Carriage return)	13	0xD
\t	跳格(Tab)	9	0x9
\0	字串結束字元(Null character)	0	0x0
\\	反斜線(Backslash)	92	0x5C
\'	單引號(Single quote)	39	0x27
\"	雙引號(Double quote)	34	0x22



跳脫字元與跳脫序列 (2/5)

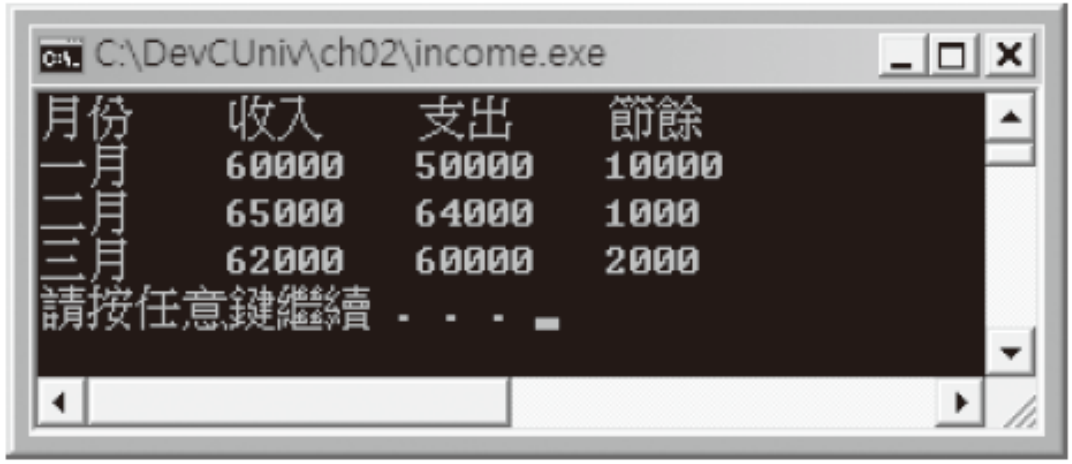
- 範例時間：家庭收支表

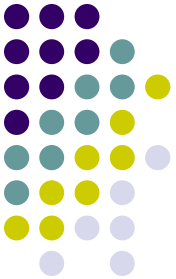
程式碼

```
1   #include <iostream>
2   using namespace std;
3   int main()
4   {
5      cout << "月份\t收入\t支出\t節餘\n";
6      cout << "一月\t60000\t50000\t10000\n";
7      cout << "二月\t65000\t64000\t1000\n";
8      cout << "三月\t62000\t60000\t2000\n";
9      system("pause");
10     return 0;
11  }
```

跳脫字元與跳脫序列 (3/5)

- 範例時間：家庭收支表
- 執行結果

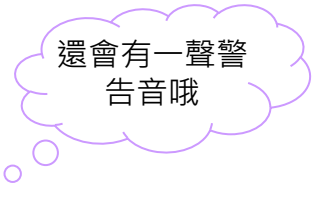


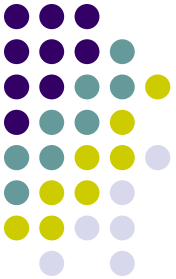


跳脫字元與跳脫序列 (4/5)

- 下面的程式是列印跳脫字元：

```
01  // prog3_7, 跳脫序列的列印
02  #include <iostream>
03  #include <cstdlib>
04  using namespace std;
05  int main(void)
06  {
07     char beep='\a';
08     int i=beep;                          // 將 beep 的值設給 i
09     cout << "beep=" << beep;             // 印出 beep 的值
10     cout << i << endl;
11     system("pause");
12     return 0;              /* prog3_7 OUTPUT---
13  }                         beep=7

                              ----------------------*/
```





跳脫字元與跳脫序列 (5/5)

- 再舉一個例子來說明跳脫序列的應用：

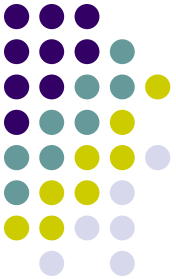
```
01  // prog3_8, 跳脫字元的列印
02  #include <iostream>
03  #include <cstdlib>
04  using namespace std;
05  int main(void)
06  {
07     char ch='\\';
08     cout << ch << "Live and learn!" << ch << endl;    // 印出字串
09     system("pause");
10     return 0;
11  }
```

```
/* prog3_8 OUTPUT---
\Live and learn!\
-------------------*/
```

浮點數與倍精度浮點數型態 (1/2)

- 浮點數佔4個位元組，有效範圍為1.2e-38到3.4e38
- 倍精度浮點數佔8個位元組，範圍從2.2e-308到1.8e308
- 下面是浮點數宣告的範例：

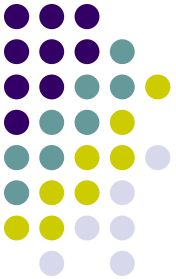
```
double num;              // 宣告 num 為倍精度浮點數變數
float sum=6.28f;         // 宣告 sum 為浮點數變數，其初值為 6.28
double num1=-5.6e64;     // 宣告 num1 為 double，其值為 -5.6×10^64
double num2=-6.32E16;    // e 也可以用大寫的 E 來取代
float num3=5.674f;       // 宣告 num3 為 float，並設初值為 5.674
float num4=2.63e64;      // 錯誤，因已超過 float 可表示的範圍
```



浮點數與倍精度浮點數型態 (2/2)

- 下面的範例是將浮點數列印到螢幕上：

```
01  // prog3_9, 浮點數的使用
02  #include <iostream>
03  #include <cstdlib>
04  using namespace std;
05  int main(void)
06  {
07     float num=2.3F;                    // 宣告 num 為浮點數，並設值為 2.3
08     cout << num << "*" << num;         // 印出 num*num 的值
09     cout << "=" << num*num << endl;
10     system("pause");
11     return 0;
12  }
                                  /* prog3_9 OUTPUT---

                                  2.3*2.3=5.29

                                  --------------------*/
```



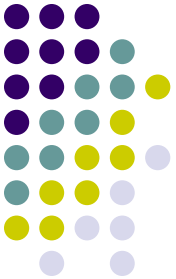
布林型態 (1/2)

- 布林型態的變數，只有true（真）和false（假）兩種
- 布林型態的變數，其值只能是1（true）或0（false）
- 宣告布林變數status，並設值為false，可寫出如下的敘述：

```
bool status=false;          //  宣告布林變數 status，並設值為 false
```

或者是

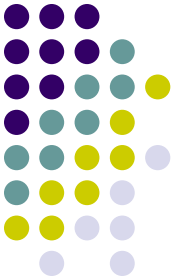
```
bool status=0;
```



布林型態 (2/2)

- 下面的程式印出布林型態變數的值

```
01  // prog3_10, 印出布林值
02  #include <iostream>
03  #include <cstdlib>
04  using namespace std;
05  int main(void)
06  {
07     bool status=false;          // 宣告布林變數 status, 設值為 false
08     cout << "status=" << status << endl;
09     status=1;                   // 設定 status 的值為 1
10     cout << "status=" << status << endl;
11
12     system("pause");            /* prog3_10 OUTPUT---
13     return 0;
14  }                              status=0
                                   status=1
                                   ----------------------*/
```



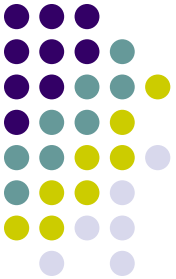
輸入整數 (1/3)

- 由鍵盤中讀取一整數值，並指定給變數num存放：

```
cin >> num;        // 由鍵盤中讀取一整數值，並指定給變數 num 存放
```

- 使用cin前，可利用cout輸出一個提示訊息：

```
cout << "Input an integer:";  // 提示訊息，請使用者輸入資料
cin >> num;                   // 由鍵盤中讀取一整數值，並指定給變數 num 存放
```

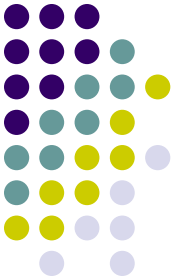


輸入整數 (2/3)

- 下面的程式是由鍵盤輸入及輸出一個數值

```
01  // prog3_11, 資料的輸入
02  #include <iostream>
03  #include <cstdlib>
04  using namespace std;
05  int main(void)
06  {
07     double n;
08     cout << "Input a number:";        // 輸入一個數
09     cin >> d;                        // 由鍵盤讀取數值，指定給變數 d 存放
10     cout << "num=" << d << endl;      // 輸出 d
11     system("pause");
12     return 0;
13  }
```

```
/* prog3_11 OUTPUT----

Input a number:2.63
num=2.63

----------------------*/
```



輸入整數 (3/3)

- 下面的程式是由鍵盤輸入2個整數，再將它們相加

```
01  // prog3_12, 資料的輸入
02  #include <iostream>
03  #include <cstdlib>
04  using namespace std;
05  int main(void)
06  {
07     int x,y;
08     cout << "Input first integer:";      // 輸入第一個整數
09     cin >> x;                  // 由鍵盤中讀取一整數值，並指定給變數 x 存放
10     cout << "Input second integer:";     // 輸入第二個整數
11     cin >> y;                  // 由鍵盤中讀取一整數值，並指定給變數 y 存放
12     cout << x << "+" << y << "=" << x+y << endl;   // 計算並輸出 x+y
13     system("pause");
14     return 0;
15  }
```

```
/* prog3_12 OUTPUT----

Input first integer:3
Input second integer:6
3+6=9

----------------------*/
```


-The End-