

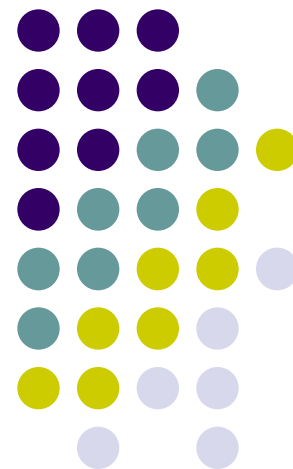
# 第二十章

## 大型程式的發展

使用名稱空間

熟悉大型程式的開發方式

條件式編譯指令的撰寫





# 名稱空間

- 名稱空間就像是一個管理介面，可以將定義的識別字放在名稱空間之下，而不會有相互衝突的發生
- 名稱空間的語法如下

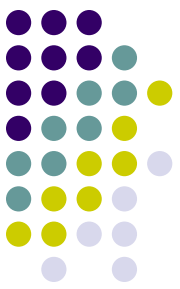
```
namespace 名稱空間名稱  
{  
    程式主體  
}
```

- 要存取使用名稱空間name1的變數var

```
name1::var;
```

- 把變數var放在名稱空間name1內

```
namespace name1  
{  
    int var;    // 在名稱空間 name1 內宣告整數變數 var  
}
```

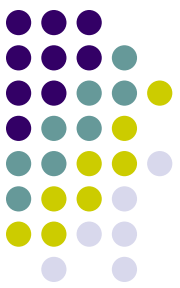


# 簡單的範例 (1/2)

- namespace的使用範例

```
01 //prog20_1, 使用 namespace
02 #include <iostream>
03 #include <cstdlib>
04
05 namespace name1 // 設定名稱空間 name1
06 {
07     int var=5; // 在名稱空間 name1 內宣告變數 var
08 }
09 using namespace std;
10 int main(void)
11 {
12     int var=10; // 宣告區域變數 var
13
14     cout << "in name1, var= " << name1::var << endl;
15     cout << "var= " << var << endl;
16
17     system("pause");
18     return 0;
19 }
```

**/\* prog20\_1 OUTPUT---**  
in name1, var= 5  
var= 10  
-----\*/



## 簡單的範例 (2/2)

- 使用兩個名稱空間的例子

```
01 //prog20_2, 使用數個名稱空間
02 #include <iostream>
03 #include <cstdlib>
04
```

```
05 namespace name1 // 設定名稱空間 name1
06 {
07     int var=5;
08 }
```

```
09
10 namespace name2 // 設定名稱空間 name2
11 {
12     int var=10;
13 }
```

```
14 using namespace std;
15 int main(void)
16 {
```

```
17     cout << "in name1, var= " << name1::var << endl;
18     cout << "in name2, var= " << name2::var << endl;
```

```
19
20     system("pause");
21     return 0;
22 }
```

**/\* prog20\_2 OUTPUT---**

in name1, var= 5  
in name2, var= 10

**-----\*/**



# 使用using關鍵字

- 下面的程式，是名稱空間使用using關鍵字的範例

```
01 //prog20_3, 使用 using 關鍵字
02 #include <iostream>
03 #include <cstdlib>
04
05 namespace name1          // 設定名稱空間 name1
06 {
07     int var=5;
08 }
09
10 using namespace name1;    // 設定此行以下的程式碼均使用 name1 名稱空間
11 using namespace std;      // 設定此行以下的程式碼也使用 std 名稱空間
12 int main(void)
13 {
14     cout << "var= " << var << endl;    // 印出 name1 名稱空間內 var 的值
15
16     int var=10;            // 設定區域變數 var
17     cout << "main()裡的變數 var= " << var << endl; // 印出區域變數 var 的值
18
19     cout << "name1::var= " << name1::var << endl; // 印出 name1 內 var 的值
20
21     system("pause");
22     return 0;
23 }
```



# 使用不同的名稱空間 (1/2)

01 //prog20\_4, 在區塊內使用不同的名稱空間

02 #include <iostream>

03 #include <cstdlib>

04

05 namespace name1

// 設定名稱空間 name1

06 {

07 int var=5;

08 }

09

10 namespace name2

// 設定名稱空間 name2

11 {

12 int var=10;

13 }

14 using namespace std;

15 int main(void)

16 {

17 {

18 using namespace name1;

// 使用名稱空間 name1

19 cout << "in namespace name1: ";

20 cout << "var= " << var << endl;

21 }

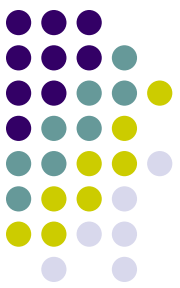
- 下面的範例，是在區塊內使用不同的名稱空間



# 使用不同的名稱空間 (2/2)

```
22
23     {
24         using namespace name2;           // 使用名稱空間 name2
25         cout << "in namespace name2: ";
26         cout << "var= " << var << endl;
27     }
28
29     system("pause");
30     return 0;
31 }
```

```
/* prog20_4 OUTPUT-----
in namespace name1: var= 5
in namespace name2: var= 10
-----*/
```



# 名稱空間std (1/3)

- 根據ANSI C++，標準函數庫裡所包含的函數、類別與物件等等，均是全部定義在std這個名稱空間內
- 使用標準函數庫裡所提供的物件時，必須加上「std::」

```
01 //prog20_5, 使用 ANSI C++的標準語法來撰寫
02 #include <iostream>
03 #include <cstdlib>
04
05 int main(void)
06 {
07     std::cout << "Hello ANSI C++ " << std::endl;
08
09     system("pause");
10     return 0;
11 }
```

**/\* prog20\_5 OUTPUT---**  
Hello ANSI C++  
-----\*/





## 名稱空間std (2/3)

- ANSI C++以iostream來取代舊有的標頭檔iostream.h，以cstdlib取代原來的stdlib.h標頭檔
- cout與endl物件前面加上「std::」，用以表示cout物件是在std名稱空間內所定義的
- 您也可以在主程式main() 的前面加上

```
using namespace std;    // 使用std名稱空間
```

如此便可省去撰寫「std::」



# 名稱空間std (3/3)

- 在C++ Bulider中，system() 函數是定義在std名稱空間中，若是在編譯時得到

Call to undefined function 'system'

- 只要在system("pause"); 敘述前加上「std::」，即可正常編譯執行



# 舊式的寫法 (1/2)

- 下面的程式碼是使用舊有的C++語法來撰寫

```
01  //prog20_6, 使用舊有的 C++語法來撰寫
02  #include <iostream.h>
03  #include <stdlib.h>
04
05  int main(void)
06  {
07      cout << "Hello C++ " << endl;
08
09      system("pause");
10      return 0;
11  }
```

**/\* prog20\_6 OUTPUT---**

Hello ANSI C++

**-----\*/**



## 舊式的寫法 (2/2)

- Dev C++在編譯時會出現下列的警告訊息

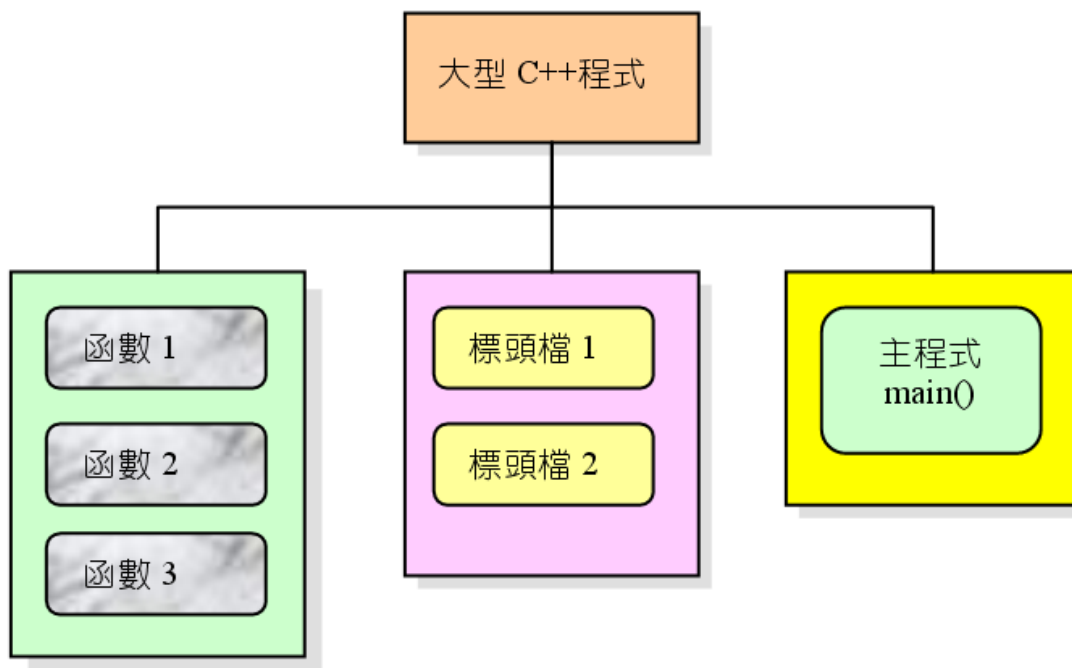
```
#warning This file includes at least one deprecated or antiquated header. Please consider using one of the 32 headers found in section 17.4.1.2 of the C++ standard. Examples include substituting the <X> header for the <X.h> header for C++ includes, or <iostream> instead of the deprecated header <iostream.h>. To disable this warning use -Wno-deprecated.
```

這個訊息是說，標頭檔裡使用較舊的語法



# 程式的模組化

- 模組化有利於程式的管理與維護，如下圖所示





# 各別編譯的實作 (1/11)

- 下面為各別編譯實作的範例

```
01 //prog20_7, 檔案分割的練習 - 完整的程式碼
02 #include <iostream>
03 #include <cstdlib>
04 using namespace std;
05
06 class CWin // 定義視窗類別 CWin
07 {
08     protected:
09         char id;
10         int width;
11         int height;
12     public:
13         CWin(char ch, int w, int h):id(ch),width(w),height(h){}
14         void show(void); // 成員函數 area() 的原型
15 };
16
```



## 各別編譯的實作 (2/11)

```
17 void CWin::show(void)                // 定義 show() 函數
18 {
19     cout << "Window " << id << ":" << endl;
20     cout << "Area = " << width*height << endl;
21 }
```

```
22
23 int main(void)                        // 主程式 main()
24 {
25     CWin win1('A',50,60);
26     win1.show();
27
28     system("pause");
29     return 0;
30 }
```

**/\* prog20\_7 OUTPUT---**

Window A:  
Area = 3000

**-----\*/**



# 各別編譯的實作 (3/11)

- 標頭檔cwin.h，用來儲存CWin類別的定義

```
01 // 標頭檔 cwin.h，用來儲存 CWin 類別的定義
02 class CWin                                // 定義視窗類別 CWin
03 {
04     protected:
05         char id;
06         int width;
07         int height;
08     public:
09         CWin(char ch, int w, int h):id(ch),width(w),height(h){}
10         void show(void);                  // 成員函數 area() 的原型
11 };
```

將此程式儲  
存成 cwin.h





# 各別編譯的實作 (4/11)

- show() 函數的定義儲存成檔案show.cpp

```
01 // show.cpp, 用來顯示資料成員
02 #include "cwin.h"           // 載入 cwin.h 標頭檔
03 #include <iostream>
04 using namespace std;
05
06 void CWin::show(void)       // 定義 show() 函數
07 {
08     cout << "Window " << id << ":" << endl;
09     cout << "Area = " << width*height << endl;
10 }
```

將此程式存  
成 show.cpp

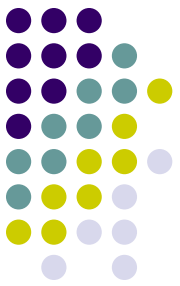


# 各別編譯的實作 (5/11)

- 主程式main() 存成prog20\_8.cpp

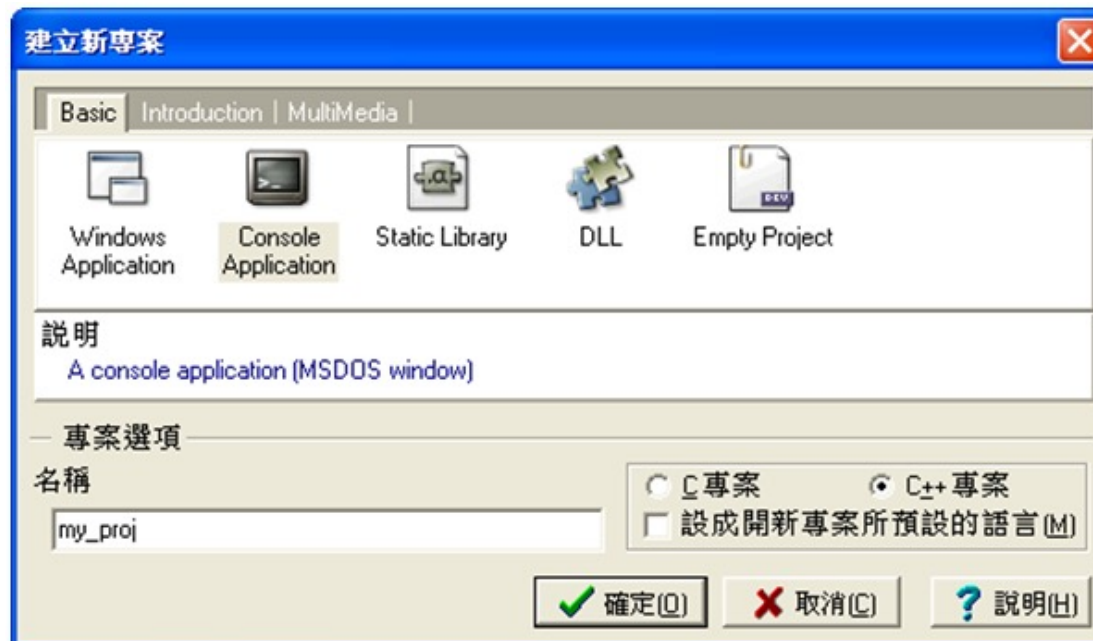
```
01 // prog20_8.cpp 主程式的部分
02 #include <iostream>
03 #include <cstdlib>
04 #include "cwin.h"           // 載入 cwin.h 標頭檔
05 using namespace std;
06
07 int main(void)
08 {
09     CWin win1('A', 50, 60);
10     win1.show();
11
12     system("pause");
13     return 0;
14 }
```

將此程式存成  
prog20\_8.cpp



# 各別編譯的實作 (6/11)

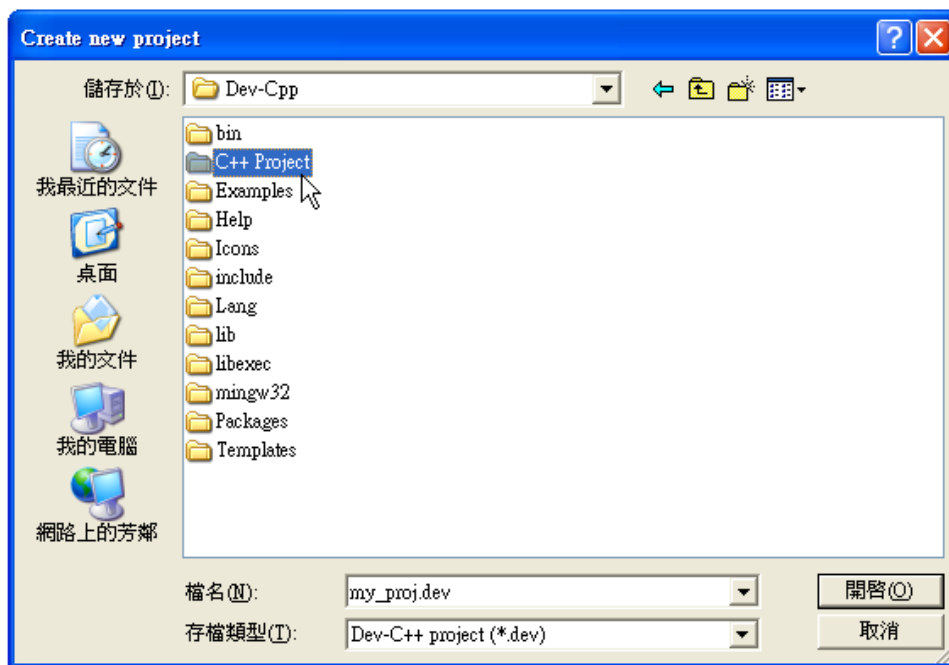
- 下面的步驟介紹如何於Dev C++裡分別建立主程式、函數模組，以及標頭檔
- 步驟1** 首先建立一個全新的專案





# 各別編譯的實作 (7/11)

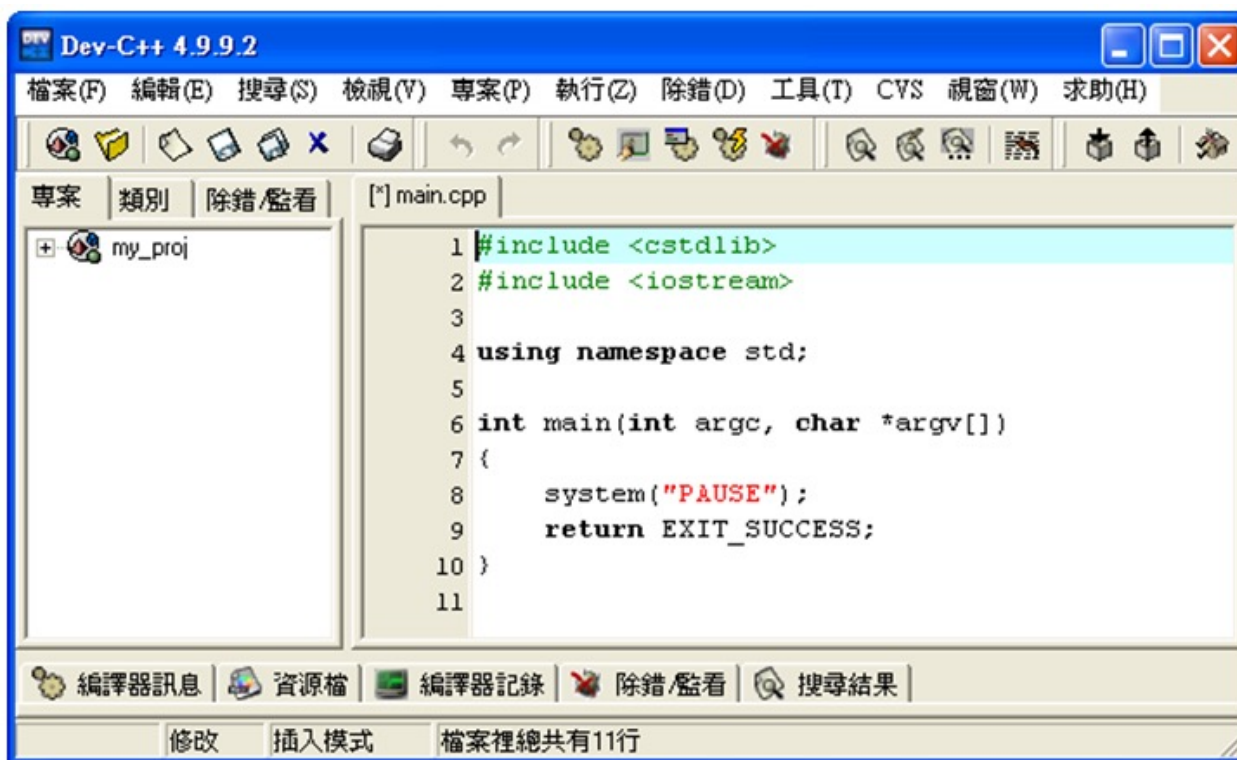
- 步驟2 選擇所要存放的資料夾，如下圖所示





# 各別編譯的實作 (8/11)

- 步驟3 按下「儲存」鈕後，進入Dev C++的專案開發環境





# 各別編譯的實作 (9/11)

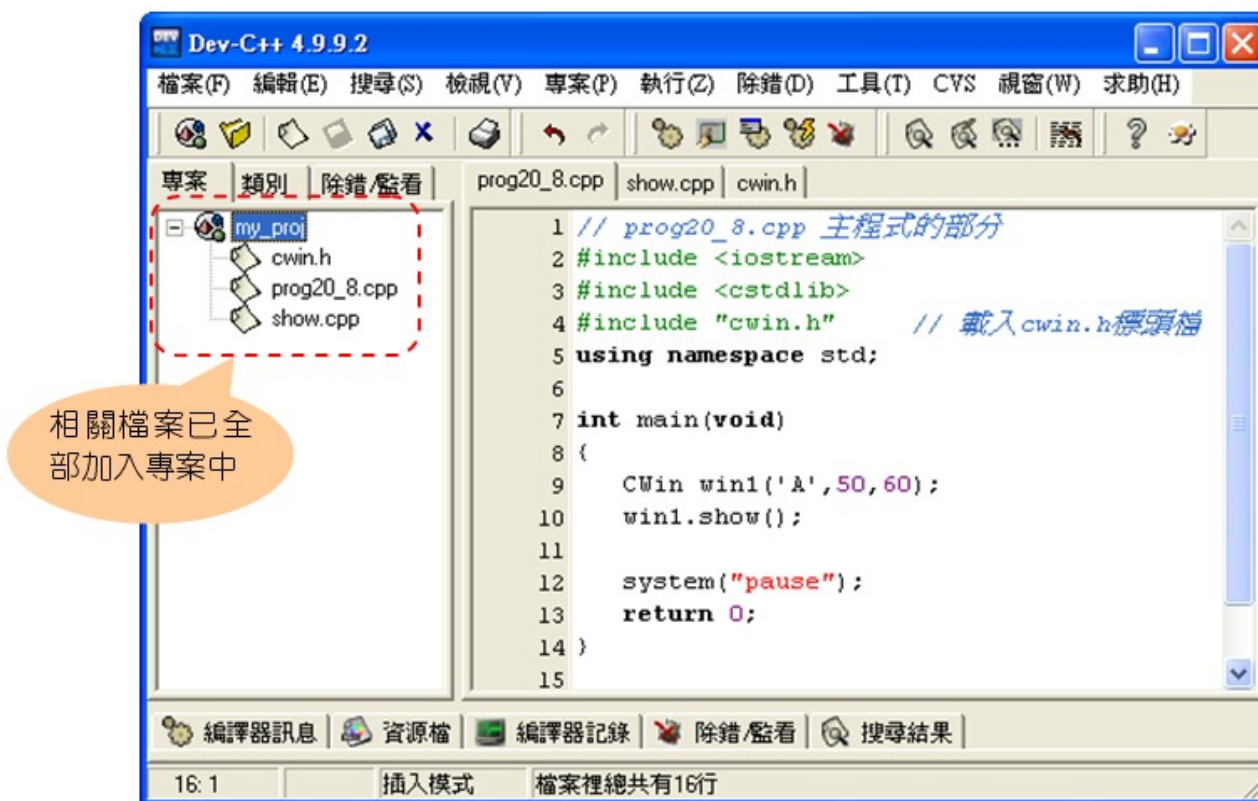
- 步驟4 輸入主程式prog20\_8.cpp的內容

```
1 // prog20_8.cpp 主程式的部分
2 #include <iostream>
3 #include <cstdlib>
4 #include "cwin.h" // 載入 cwin.h 標頭檔
5 using namespace std;
6
7 int main(void)
8 {
9     CWin win1('A', 50, 60);
10    win1.show();
11
12    system("pause");
13    return 0;
14 }
15
```



# 各別編譯的實作 (10/11)

- **步驟5** 重複步驟4，將show.cpp與cwin.h加入my\_proj中，最後應該會得到如下的視窗：



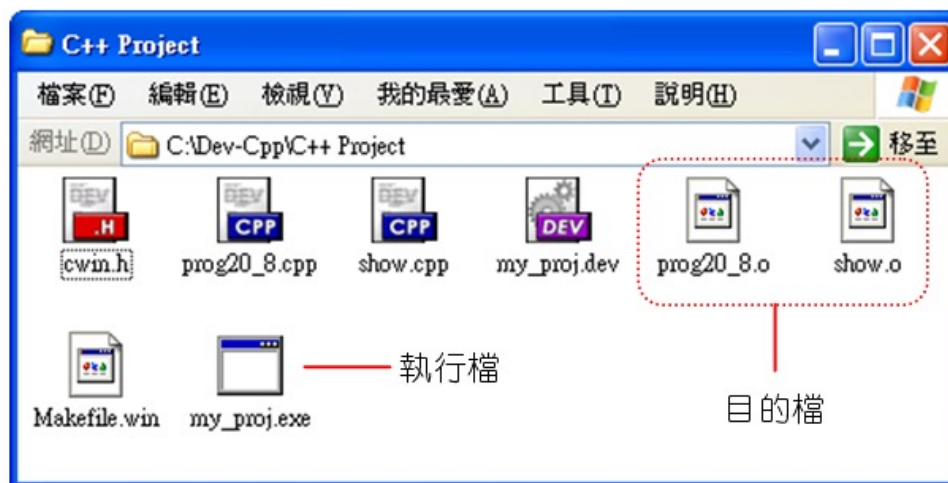


# 各別編譯的實作 (11/11)

- **步驟6** 按下F9鍵，將程式一起編譯。程式執行的結果如下所示：

```
/* prog 20_8 OUTPUT---  
Window A:  
Area = 3000  
-----*/
```

- 編譯後的目的檔與執行檔如下圖所示







# 條件式編譯

- #ifdef、#else與 #endif指令

```
#ifdef 識別字
    // 如果識別字有被定義過，即執行此部分的程式碼
#else
    // 否則執行此部分的程式碼
#endif
```



# 使用條件式編譯指令 (1/2)

- 接下來舉一個簡單的範例來說明如何使用這些指令

```
01 // prog20_9, 使用#ifdef、#else 與#endif 指令
02 #include <iostream>
03 #include <cstdlib>
04 #define STR "Hello C++"           // 定義 STR 為"Hello C++"字串
05
06 using namespace std;
07 int main(void)
08 {
09     #ifdef STR                     // 如果 STR 有被定義
10         cout << STR << endl;
11     #else                         // STR 沒有被定義
12         cout << "STR not defined" << endl;
13     #endif
14     system("pause");
15     return 0;
16 }
```

**/\* prog20\_9 OUTPUT---**  
Hello C++  
**-----\*/**



## 使用條件式編譯指令 (2/2)

- 實際送至編譯器裡編譯的主程式只剩下面的程式碼

```
int main(void)
{
    cout << STR << endl;
    system("pause");
    return 0;
}
```



## #if、#else、#elif 與 #endif 指令

- #if、#else、#elif 與 #endif 指令格式如下

```
#if 運算式 1
    // 若運算式 1 的結果成立，則執行此區段的敘述
#elif 運算式 2
    // 若運算式 2 的結果成立，則執行此區段的敘述
#elif 運算式 3
    :
#endif
```



## 前置處理指令的練習

```
01 // prog20_10, 使用#if、#else 與#endif 指令
02 #include <iostream>
03 #include <cstdlib>
04 #define SIZE 15
05 using namespace std;
06
07 int main(void)
08 {
09     #ifdef SIZE
10         #if SIZE>20
11             char str[SIZE]="Hello C++";
12         #else
13             char *str="SIZE too small";
14         #endif
15     #else
16         char *str="SIZE not defined";
17     #endif
18
19     cout << str << endl;
20
21     system("pause");
22     return 0;
23 }
```

- 下面的程式是前置處理指令的綜合練習

```
/* prog20_10 OUTPUT---
SIZE too small
-----*/
```



# #undef 指令

- #undef指令格式如下

```
#undef 識別字
```

- 若識別字並沒有被定義過，則不受#undef指令的影響



# 條件式編譯與大型程式的發展(1/5)

- 接續my\_proj專案，想加入一個繼承自CWin類別的子類別CMiniWin，可以撰寫出如下的程式碼

```
01 // cminiwin.h, 此檔案定義子類別 CMiniWin
02 #include <iostream>
03 #include "cwin.h"
04 using namespace std;
05
06 class CMiniWin: public CWin // 定義子類別 CMiniWin
07 {
08     public:
09         CMiniWin(char ch, int w, int h):CWin(ch,w,h){}
10         void show(void)
11         {
12             cout << "Mini window " << id << ":" << endl;
13             cout << "Area = " << 0.8*width*height << endl;
14         }
15 };
```

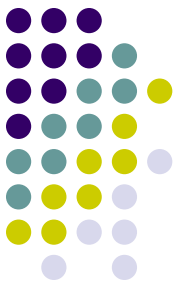


# 條件式編譯與大型程式的發展(2/5)

- 測試cminiwin.h標頭檔是否撰寫正確

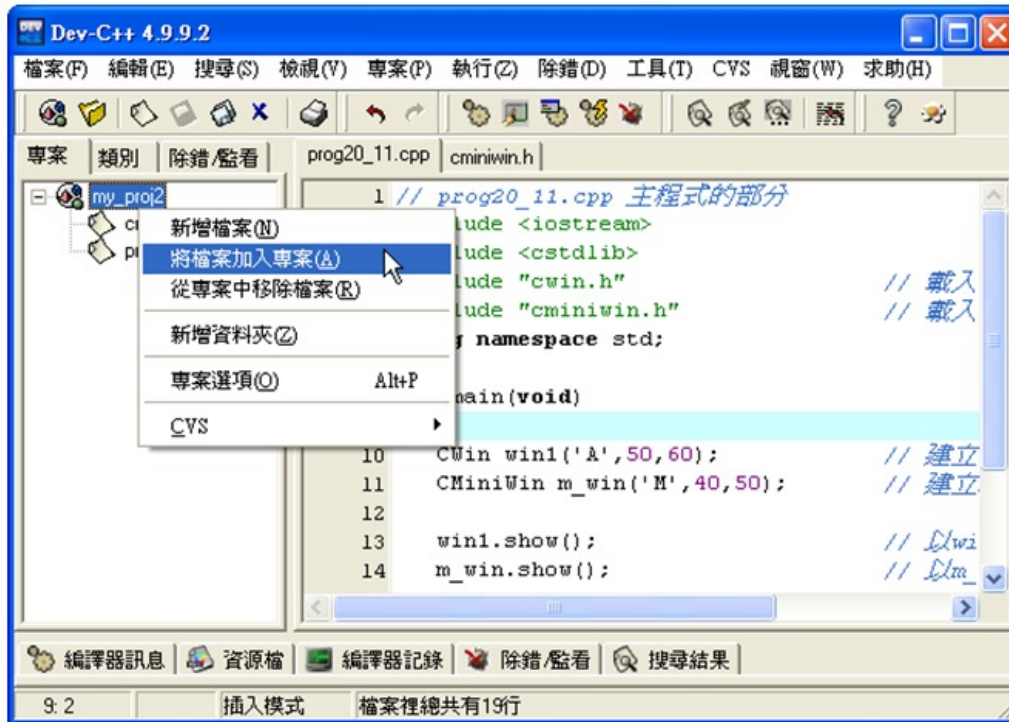
```
01 // prog20_11.cpp 主程式的部分
02 #include <iostream>
03 #include <cstdlib>
04 #include "cwin.h" // 載入 cwin.h 標頭檔
05 #include "cminiwin.h" // 載入 cminiwin.h 標頭檔
06 using namespace std;
07
08 int main(void)
09 {
10     CWin win1('A',50,60); // 建立 win1 物件
11     CMiniWin m win('M',40,50); // 建立 m win 物件
12
13     win1.show(); // 以 win1 物件呼叫 show()
14     m win.show(); // 以 m win 物件呼叫 show()
15
16     system("pause");
17     return 0;
18 }
```





# 條件式編譯與大型程式的發展(3/5)

- 將檔案cminiwin.h及prog20\_11加入專案中



- 接著請分別將cwin.h與show.cpp加入專案裡



# 條件式編譯與大型程式的發展(4/5)

- 於Dev C++裡編譯後得到下列的錯誤訊息：

```
In file included from cminiwin.h
from prog20_11.cpp
redefinition of 'class CWin'
```

這錯誤訊息告訴我們CWin類別被重複定義



# 條件式編譯與大型程式的發展(5/5)

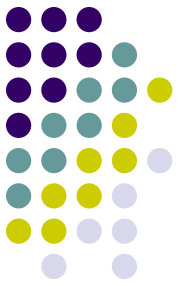
- 重新建立一個專案，把cwin.h修改成如下的程式碼

```

01 // 修正後的標頭檔 new_cwin.h，可判別此標頭檔是否有被載入過
02
03 #ifndef CWIN H // 如果 CWIN H 沒有被定義過
04 #define CWIN H // 則定義 CWIN H
05
06 class CWin // 定義視窗類別 CWin
07 {
08     protected:
09         char id;
10         int width;
11         int height;
12     public:
13         CWin(char ch, int w, int h):id(ch),width(w),height(h){}
14         void show(void); // 成員函數 area() 的原型
15 };
16
17 #endif // #ifndef 到此結束

```

**/\* prog 20\_12 OUTPUT---**  
 Window A:  
 Area = 3000  
 Mini window M:  
 Area = 1600  
**-----\*/**



The End-