Chapter 3： 迴圈(Loop)

迴圈為程式中常用的功能，使電腦進行重複性的計算工作。

C/C++提供了以下三種迴圈結構：

1. while
2. do … while
3. for

3.1 while 迴圈結構

格式：

```
while(布林運算式)
{
    程式命令；
}
```

說明：

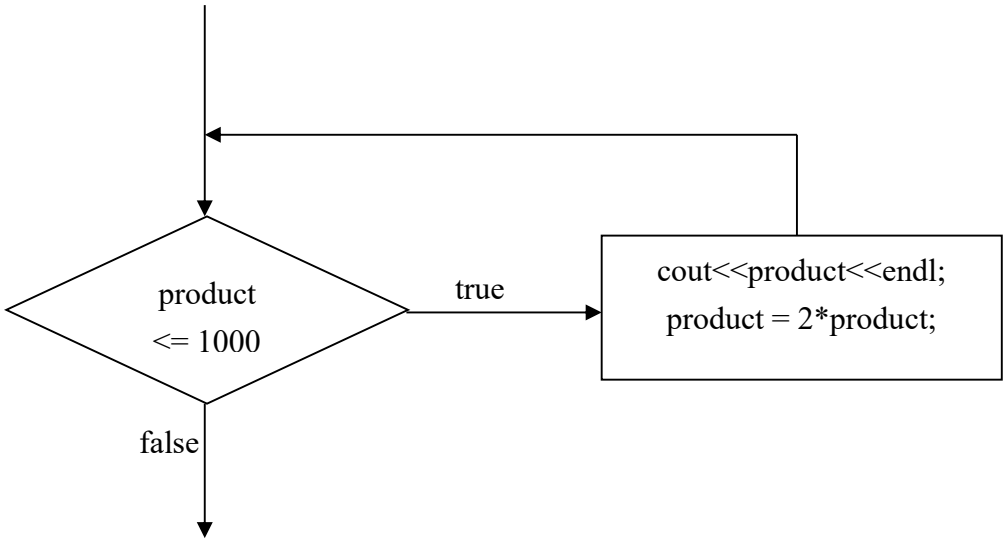
當 while 中布林運算式的結果為「成立」(true)，則執行{ }中的程式命令，之後再跳回 while 中布林運算式，再次進行判斷，如仍為成立，則再次執行{ }中的程式命令，如此重複，直到布林運算式的結果為「不成立」(false)，即跳出迴圈。

while 迴圈常用於迴圈次數不確定的情況，迴圈次數視是否達到某一條件，而非由次數所控制。通常{ }中的程式命令會變更( )中布林運算式某一或某些變數的值，如此迴圈才有終結的可能。

例：

```
int product = 2;

while(product <= 1000)
{
    cout << product <<endl;
    product = 2 * product;
}
```



程式範例：cpp_ex13.cpp

注意要點：{ }中的程式命令，亦可包含有(1)迴圈結構，此稱為巢式迴圈、(2)條件式結構。

3.2 do….while 迴圈結構

格式：

```
do
{
    程式命令；
} while(布林運算式)；
```

說明：

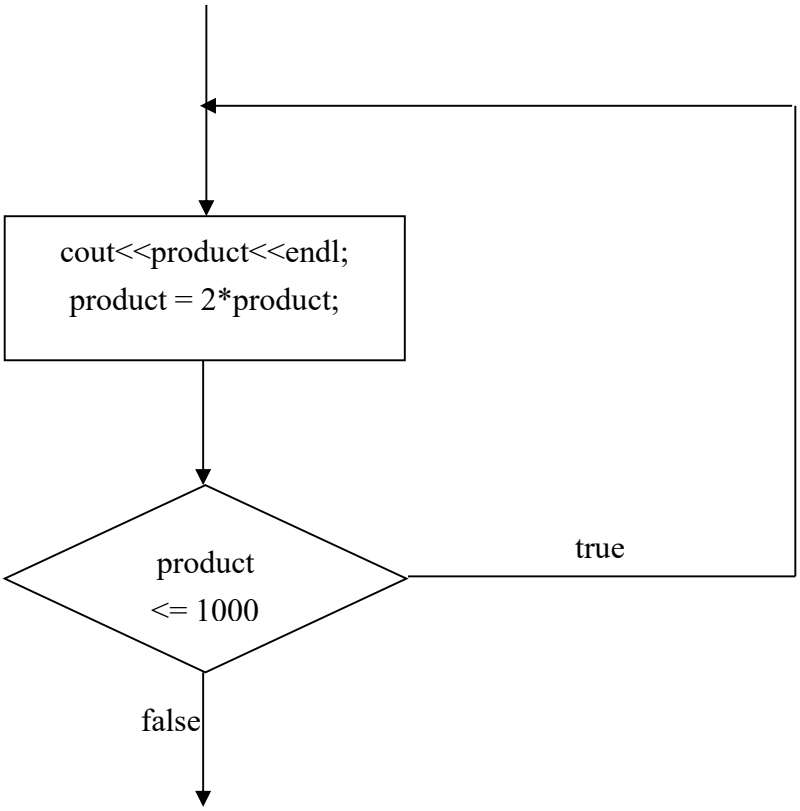
先執行{ }中的程式命令後，再對 while 後( )中的布林運算式進行判斷，如結果為「成立」(true)，則再一次執行{ }中的程式命令之後進行判斷，如此重複，直到布林運算式的結果為「不成立」(false)，即跳出迴圈。

比較 do….while 迴圈與 while 迴圈的不同，while 是先判斷後執行，而 do…..while 是先執行後判斷。

例：

```
int product = 2;

do
{
      cout << product <<endl;
      product = 2 * product;
} while(product <= 1000);
```



程式範例：cpp_ex14.cpp

注意要點：使用 do…..while 迴圈，至少會執行{　}中的程式命令一次，而使用 while 迴圈，{　}中的程式命令可能不被執行。

3.3 for 迴圈結構

格式：

```
for( 初值運算式(0) ； 布林條件式(1) ； 變量運算式(3) )
{
    程式命令(2)；
}
```

說明：

首先執行初值運算式，對控制變數(下例中之 i)做初值設定，隨後執行布林條件式並以其結果判斷迴圈是否繼續，如果布林條件式的結果為「成立」(true)，則執行{ }中的程式命令，之後執行變量運算式，再跳回由布林條件式判斷，迴圈重複直到布林條件式的結果為「不成立」(false)。

For 迴圈結構之執行程序：(0)->(1)->(2)->(3)->(1)->(2)->(3)->……..->(1)

比較 for 迴圈與 while 迴圈，for 迴圈主要用於迴圈次數已知的情況。

例：

```
for( int i =1; i<=10; i++)
{
    cout<< i << endl;
}
```

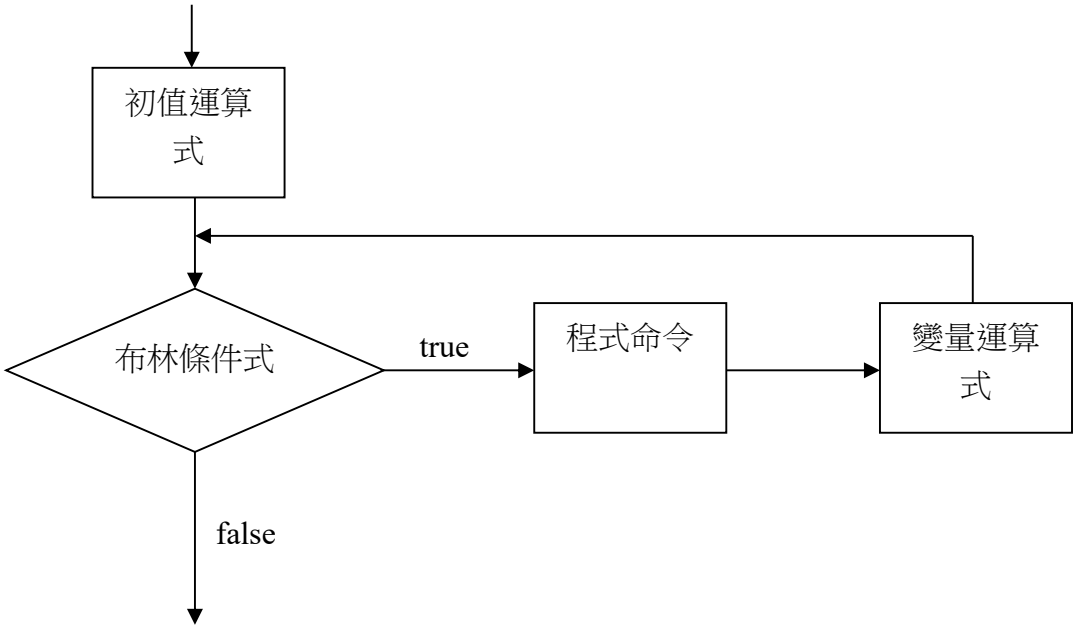
其結果與以下相同。

```
int i = 1;

while(i<=10)
{
    cout << i << endl;
    i++;
}
```

```
for(int j=10; j>=1; j--)
        cout << j << endl;

for(int k=1; k<=10; k+=2)
        cout<< k <<endl;
```



程式範例：cpp_ex15.cpp

注意要點：

1. for 之後的括弧內，敘述的間隔是使用分號。
2. 控制變數的起始值，可為任意整數(如配合陣列時，多為陣列之 index，起始值常為 0)。
3. { }中可使用控制變數，但通常不改變其值，其值之增減由變量式負責。
4. 變量式可視演算需要對控制變數增加(++)、減少(--)，增減之量亦可隨意(+=2)，不一定要是 1。

3.4 巢狀迴圈(Nested Loop)

以某種迴圈結構使之重複執行的程式命令，其內容可包含另一個(相同或不同種的)迴圈結構，如此的結構稱為**巢狀迴圈**，其結構分為外層迴圈(outer loop)與內層迴圈(inner loop)。以兩層的 for 迴圈結構為例，其**格式**與**例**如下：

其中外層迴圈與內層迴圈，分別有一個各自的**控制變數**(如例中的 i、j)，分別用於各自的一組**初值運算式、 布林條件式、 變量運算式**(於下格式標示為 A、B)，來控制各自{ }內程式的重複，如此一來，內層迴圈即會受外層迴圈的控制，

使其控制變數(如例中的 j)重複地始初化而重複執行多次。

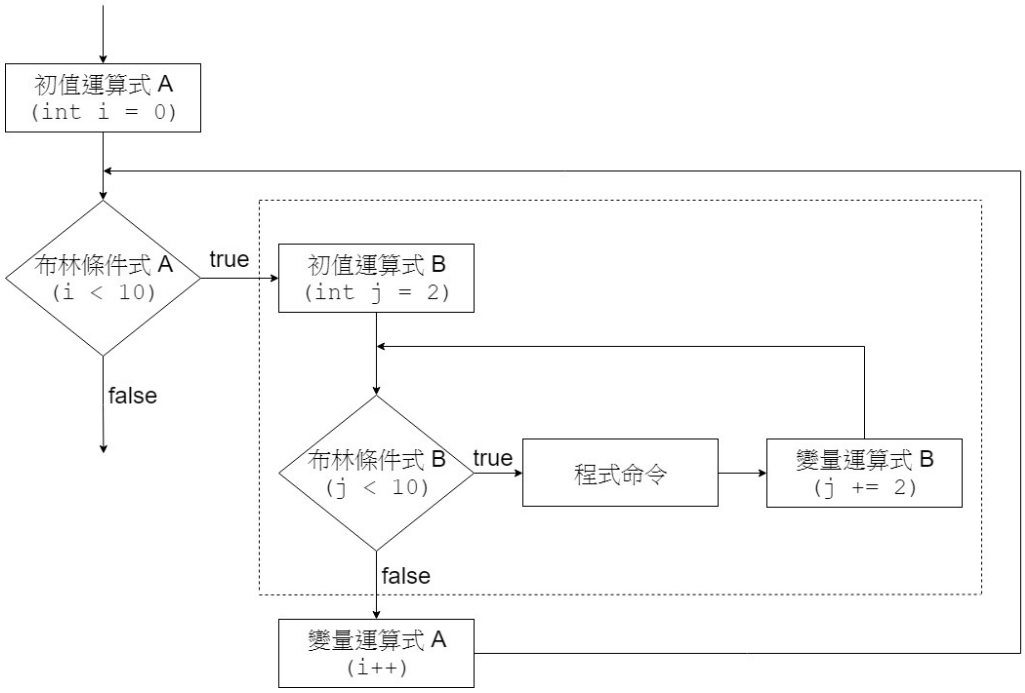
請注意於巢狀迴圈結構中，外層迴圈與內層迴圈各自的**控制變數**(如例中的 i、j)**其值變化的規律性**，以及基於此規律性下所做的程式範例應用。

格式：

```
for( 初值運算式 A； 布林條件式 A； 變量運算式 A )
{
    程式命令； //視情況可有可無(optional)
    for( 初值運算式 B； 布林條件式 B； 變量運算式 B )
    {
        程式命令；
    }
    程式命令； //視情況可有可無(optional)
}
```

例：

```
for(int i=0；i<10；i++)   // i : A 組式子的控制變數
{
    for(int j=2；j<10；j+=2 )   // j : B 組式子的控制變數
        cout<< i<<”  “<<j<<endl;
    cout<<endl;
}
```



程式範例：cpp_ex16.cpp

3.5 break 與 continue 指令

break 與 continue 是用於迴圈結構的指令。

3.5.1 break:

當 break 指令在某一 while, do…while, for, 或 switch 結構中被執行，其將使程式自此迴圈結構中立即跳出。

例：

```
for(int x = 1; x <= 10 ; x++)
{
    if( x = = 5)
        break;

    cout<< x <<" ";
}
```

3.5.2 continue:

當 continue 指令在某一 while, do/while 或 for 迴圈結構中被執行，其將使本輪迴圈所剩餘的程式命令被跳過省略，然後繼續進行下一輪的迴圈。

例：

```
for(int y = 1; y <= 10 ; y++)
{
    if( y = = 5)
        continue;

    cout<< y <<" ";
}
```

程式範例：cpp_ex17.cpp　oop_ex18.cpp

3.6 goto 指令

使用 goto 指令可將程式之執行無條件地跳到某一指定點。

格式：

```
goto   標籤；

……

標籤：
    程式命令；
```

例：

```
while(true)     //for( ; ; )
{
    cout << i++;

    if( i >= 10)
        goto END;
}

END：
    exit(0);
```

程式範例：cpp_ex19.cpp

注意要點：

1. 標籤位於 goto 指令之前之後均可，惟需注意程式之流程。
2. stdlib.h 中的 exit( ) 函式，可使程式強制結束。