迴圈練習

1104資訊社

C++中的迴圈

在C++中，迴圈有3種：

- **while** 迴圈 (**while** loop)
- **do-while** 迴圈 (**do-while** loop)
- **for** 迴圈 (**for** loop)

while 迴圈

```
while(條件式(布林值)){
    程式區塊;
}
```

當條件式成立時，就執行程式區塊

-> 當條件式成立時，就執行程式區塊

-> 當條件式成立時，就執行程式區塊

-> 當條件式成立時，就執行程式區塊

-> 當條件式成立時，就執行程式區塊

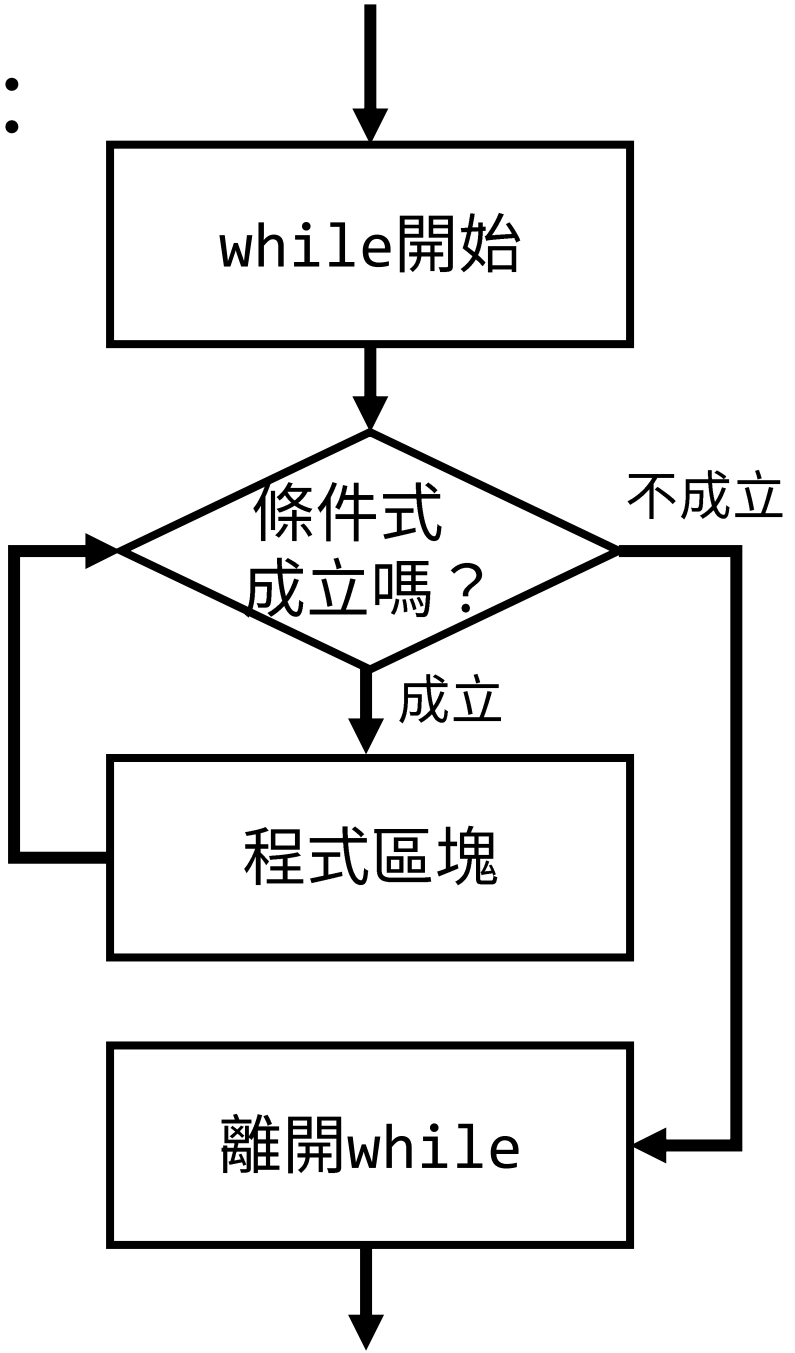
...

-> 當條件式不成立時，離開while迴圈

while 迴圈

```
while(條件式(布林值)){
    程式區塊;
}
```

流程圖：

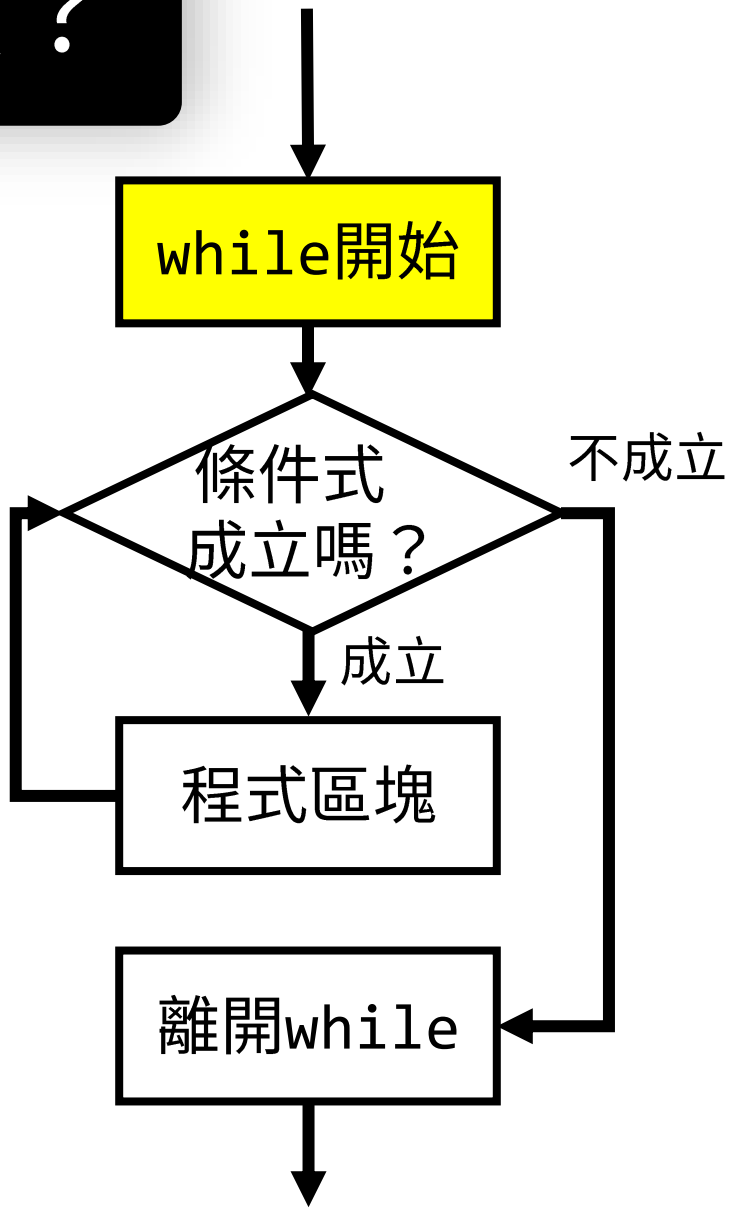


如果今天想要列出1~5間的整數？

```
1   #include <iostream>
2   using namespace std;
3
4   int main(){
5       int i = 1;
6       while(i <= 5){
7           cout << i << endl;
8           i++;
9       }
10      return 0;
11  }
```

i = 1

輸出：

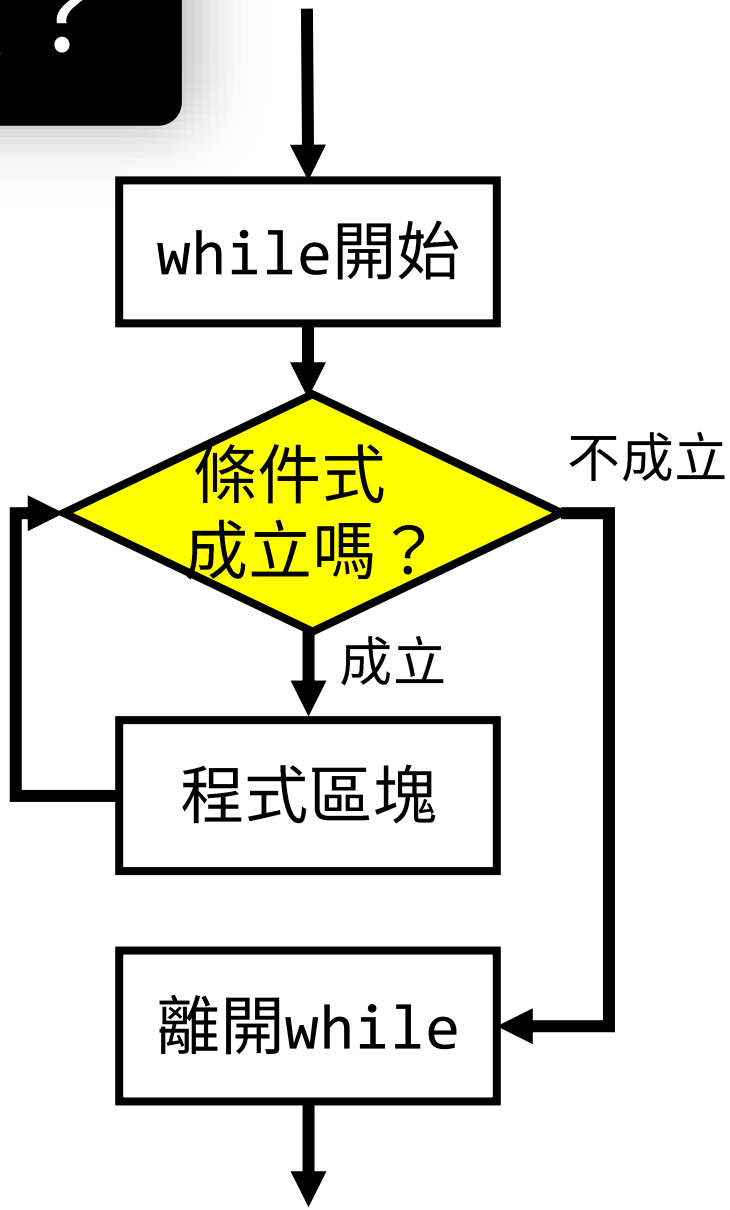


如果今天想要列出1~5間的整數？

```
1   #include <iostream>
2   using namespace std;
3
4   int main(){
5       int i = 1;
6       while(i <= 5){
7           cout << i << endl;
8           i++;
9       }
10      return 0;
11  }
```

i = 1

輸出：



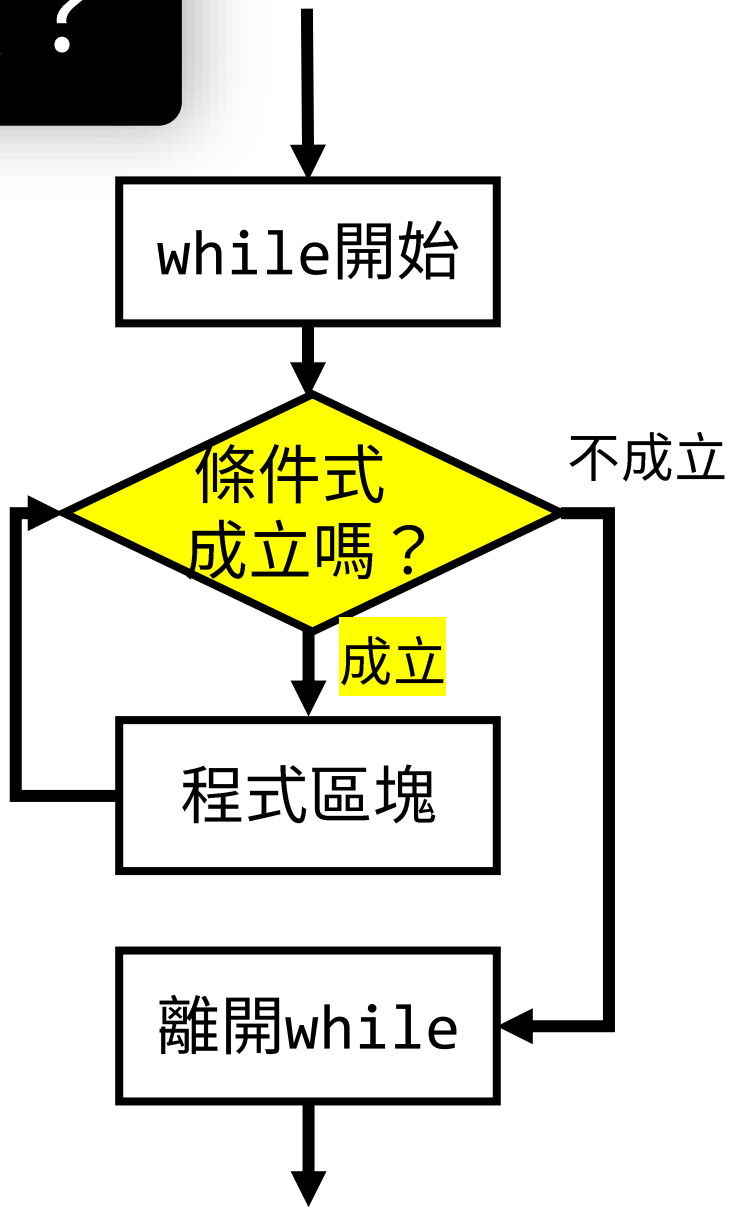
如果今天想要列出1~5間的整數？

```
1   #include <iostream>
2   using namespace std;
3
4   int main(){
5       int i = 1;
6       while(i <= 5){
7           cout << i << endl;
8           i++;
9       }
10      return 0;
11  }
```

i = 1

i <= 5 為真

輸出：



如果今天想要列出1~5間的整數？

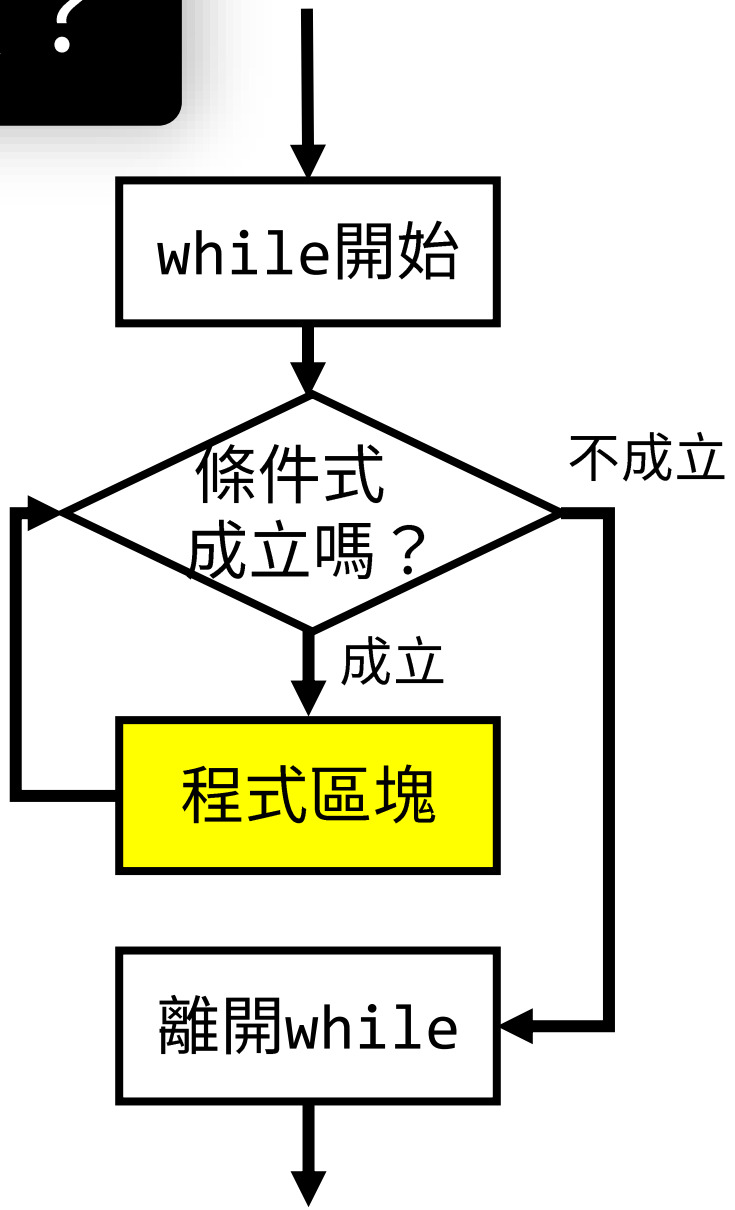
```
1   #include <iostream>
2   using namespace std;
3
4   int main(){
5       int i = 1;
6       while(i <= 5){
7           cout << i << endl;
8           i++;
9       }
10      return 0;
11  }
```

i = 1

i <= 5 為真

輸出 i

```
輸出：
1
```



如果今天想要列出1~5間的整數？

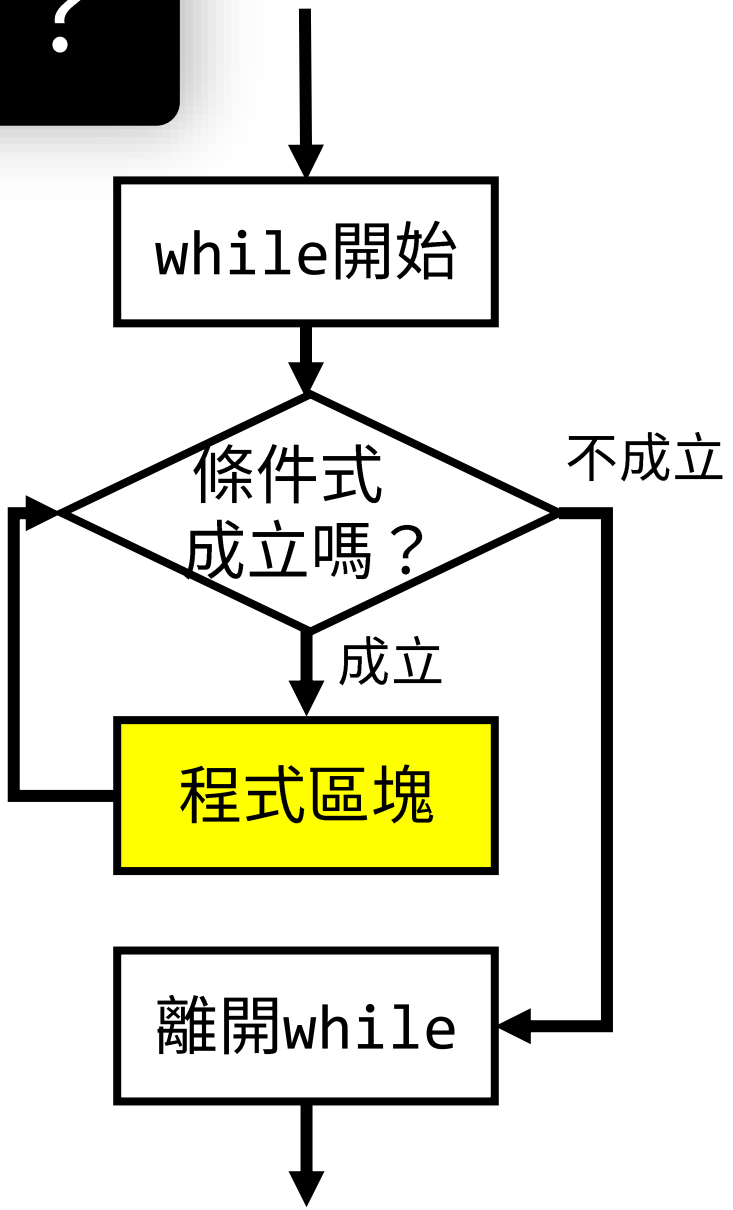
```
1   #include <iostream>
2   using namespace std;
3
4   int main(){
5       int i = 1;
6       while(i <= 5){
7           cout << i << endl;
8           i++;
9       }
10      return 0;
11  }
```

i = 2

i <= 5 為真

i = i + 1

```
輸出：
1
```

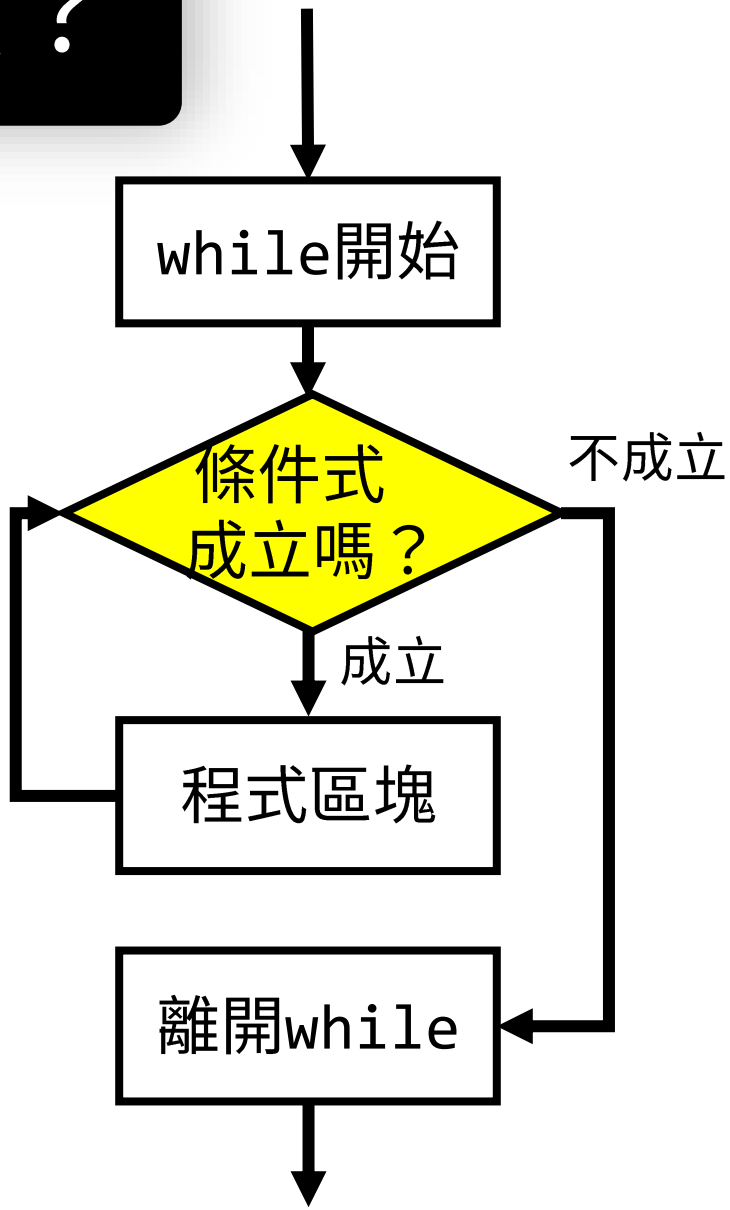


如果今天想要列出1~5間的整數？

```
1   #include <iostream>
2   using namespace std;
3
4   int main(){
5       int i = 1;
6       while(i <= 5){
7           cout << i << endl;
8           i++;
9       }
10      return 0;
11  }
```

i = 2

```
輸出：
1
```



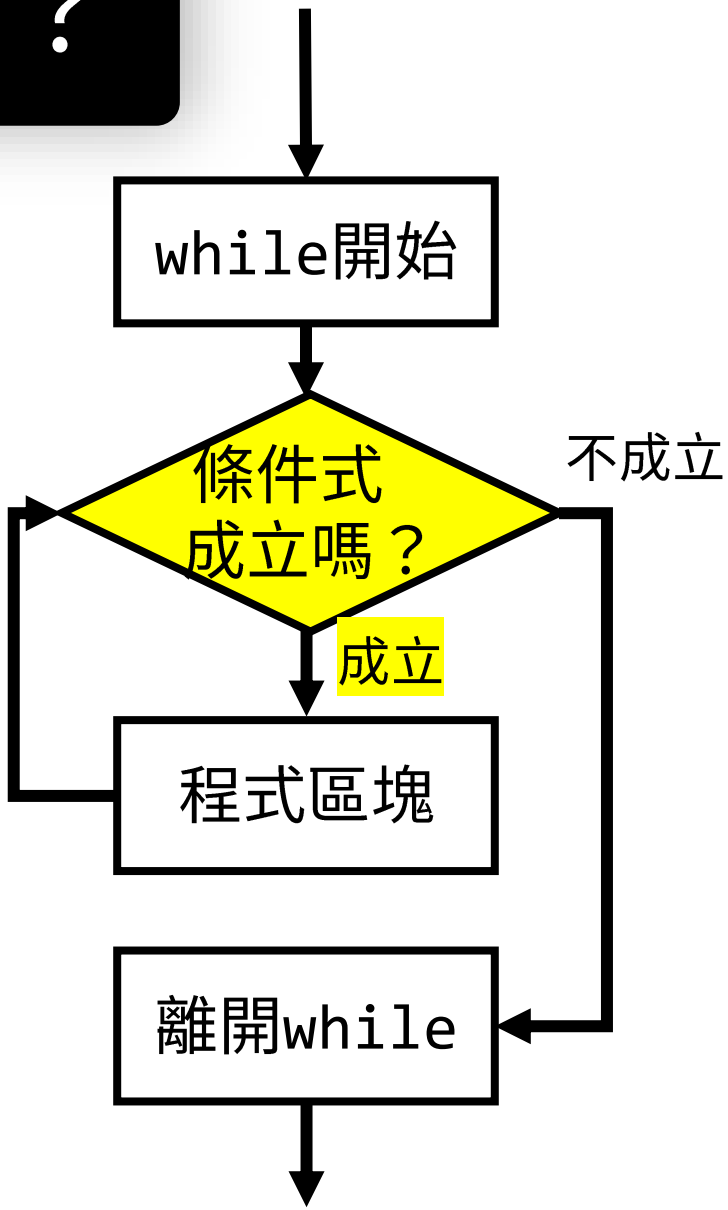
如果今天想要列出1~5間的整數？

```
1   #include <iostream>
2   using namespace std;
3
4   int main(){
5       int i = 1;
6       while(i <= 5){
7           cout << i << endl;
8           i++;
9       }
10      return 0;
11  }
```

i = 2

i <= 5 為真

```
輸出：
1
```



如果今天想要列出1~5間的整數？

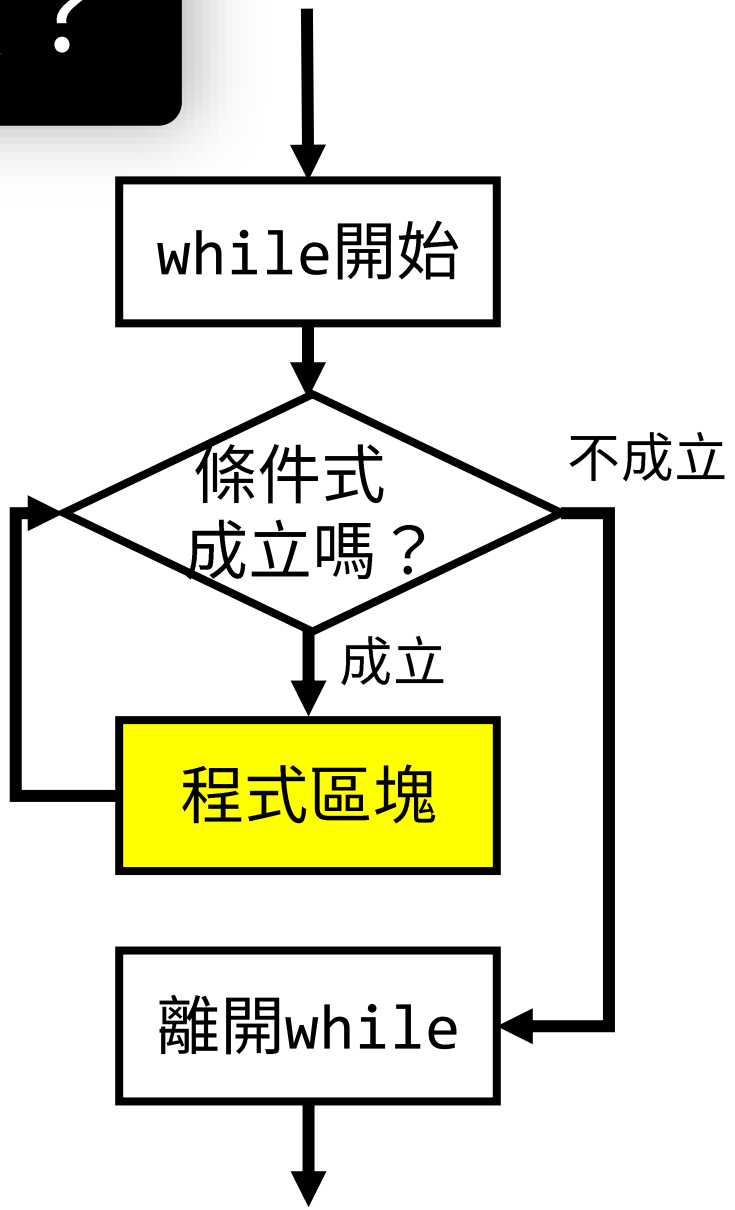
```
1   #include <iostream>
2   using namespace std;
3
4   int main(){
5       int i = 1;
6       while(i <= 5){
7           cout << i << endl;
8           i++;
9       }
10      return 0;
11  }
```

i = 2

i <= 5 為真

輸出 i

```
輸出：
1
2
```



如果今天想要列出1~5間的整數？

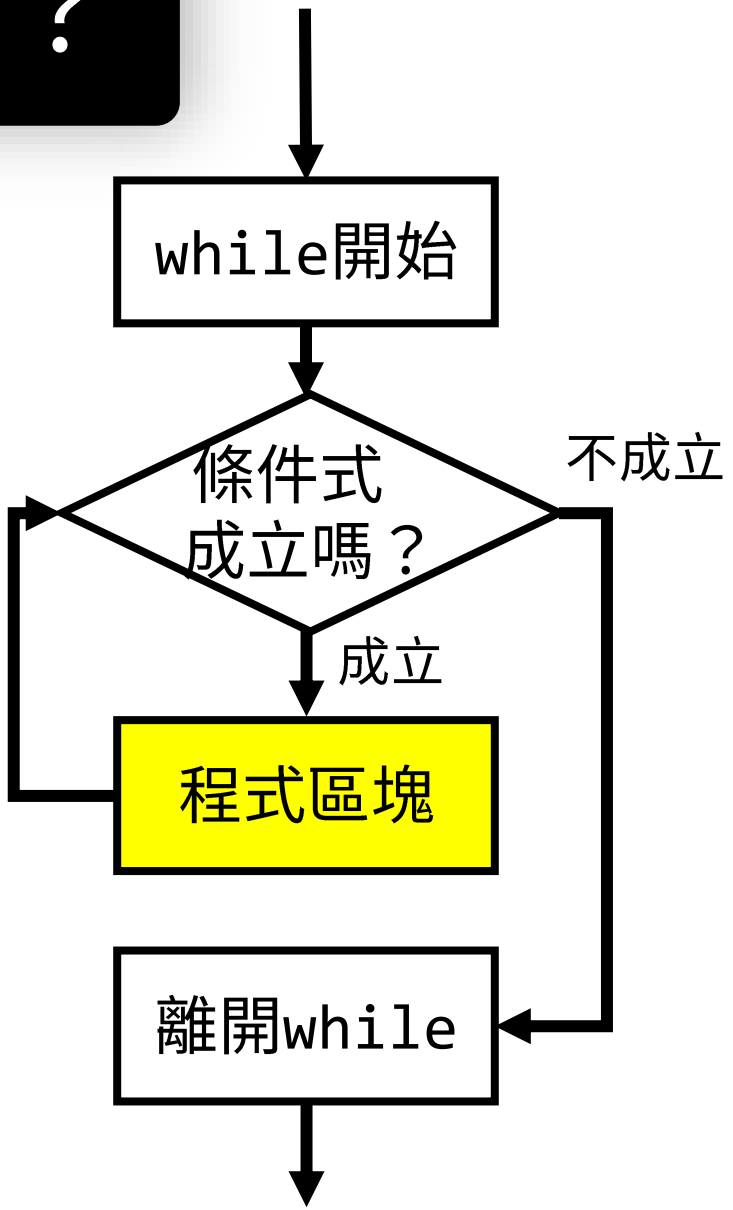
```
1   #include <iostream>
2   using namespace std;
3
4   int main(){
5       int i = 1;
6       while(i <= 5){
7           cout << i << endl;
8           i++;
9       }
10      return 0;
11  }
```

i = 3

i <= 5 為真

i = i + 1

```
輸出：
1
2
```

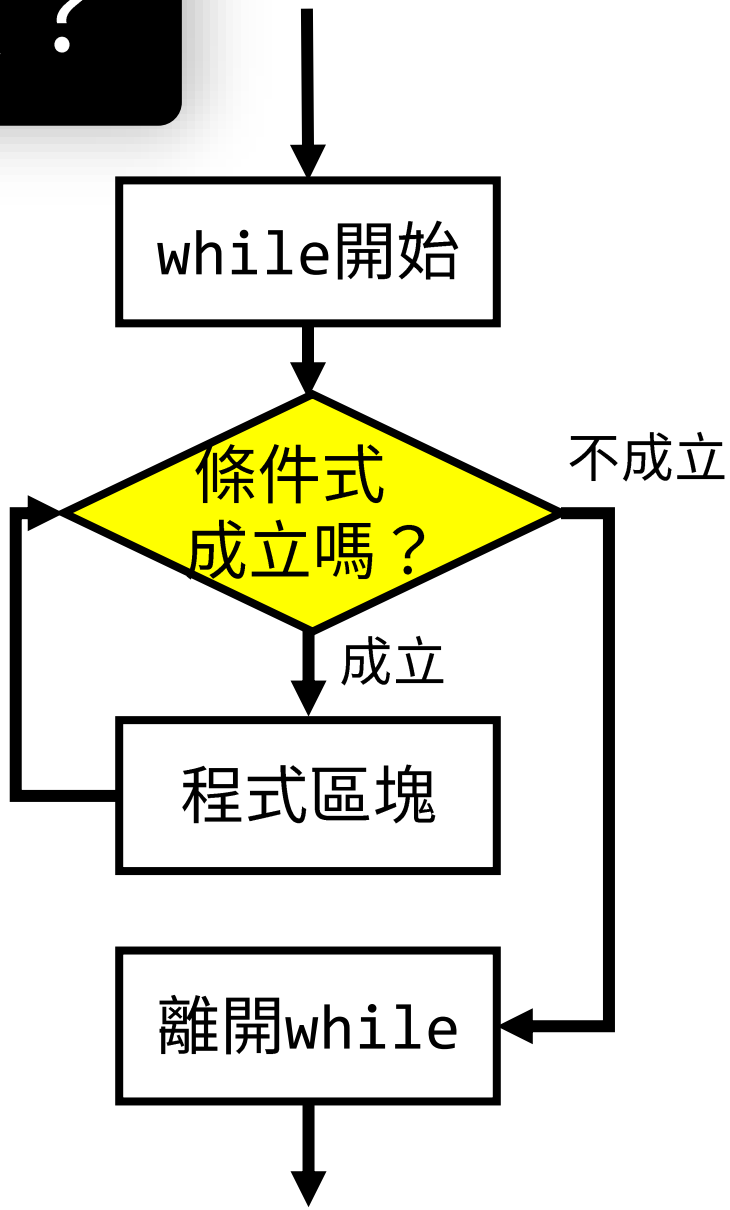


如果今天想要列出1~5間的整數？

```
1   #include <iostream>
2   using namespace std;
3
4   int main(){
5       int i = 1;
6       while(i <= 5){
7           cout << i << endl;
8           i++;
9       }
10      return 0;
11  }
```

i = 3

```
輸出：
1
2
```



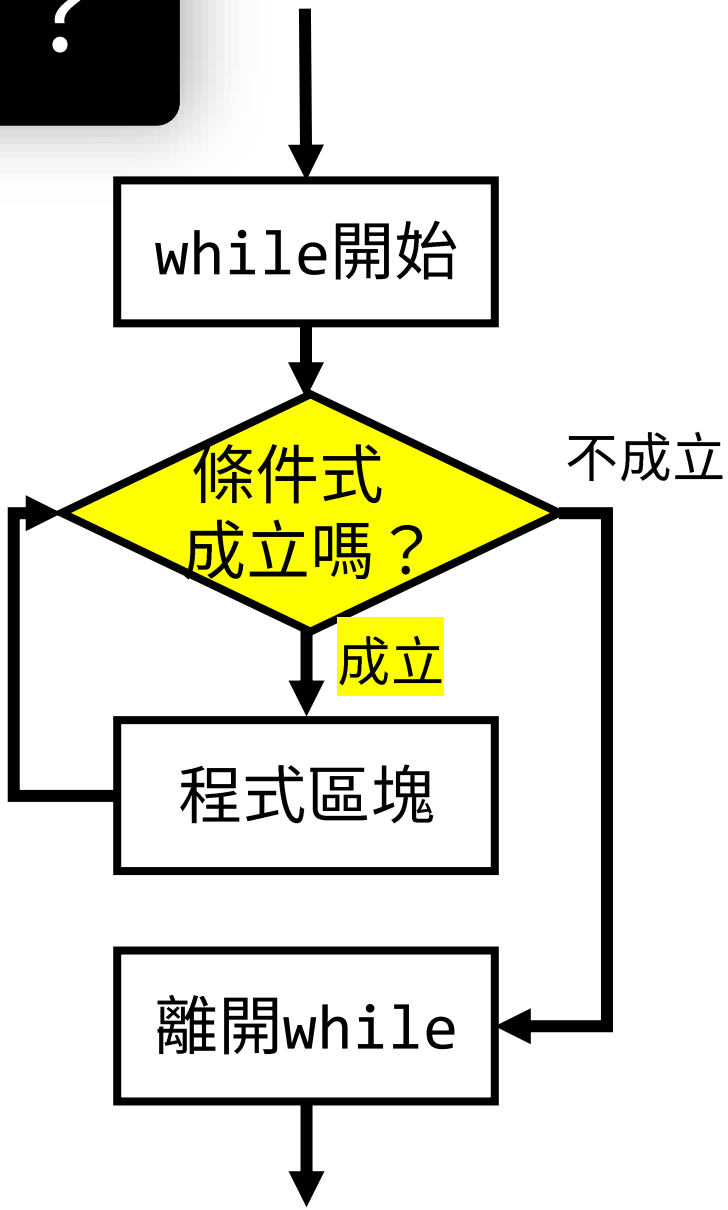
如果今天想要列出1~5間的整數？

```
1   #include <iostream>
2   using namespace std;
3
4   int main(){
5       int i = 1;
6       while(i <= 5){
7           cout << i << endl;
8           i++;
9       }
10      return 0;
11  }
```

i = 3

i <= 5 為真

```
輸出：
1
2
```



如果今天想要列出1~5間的整數？

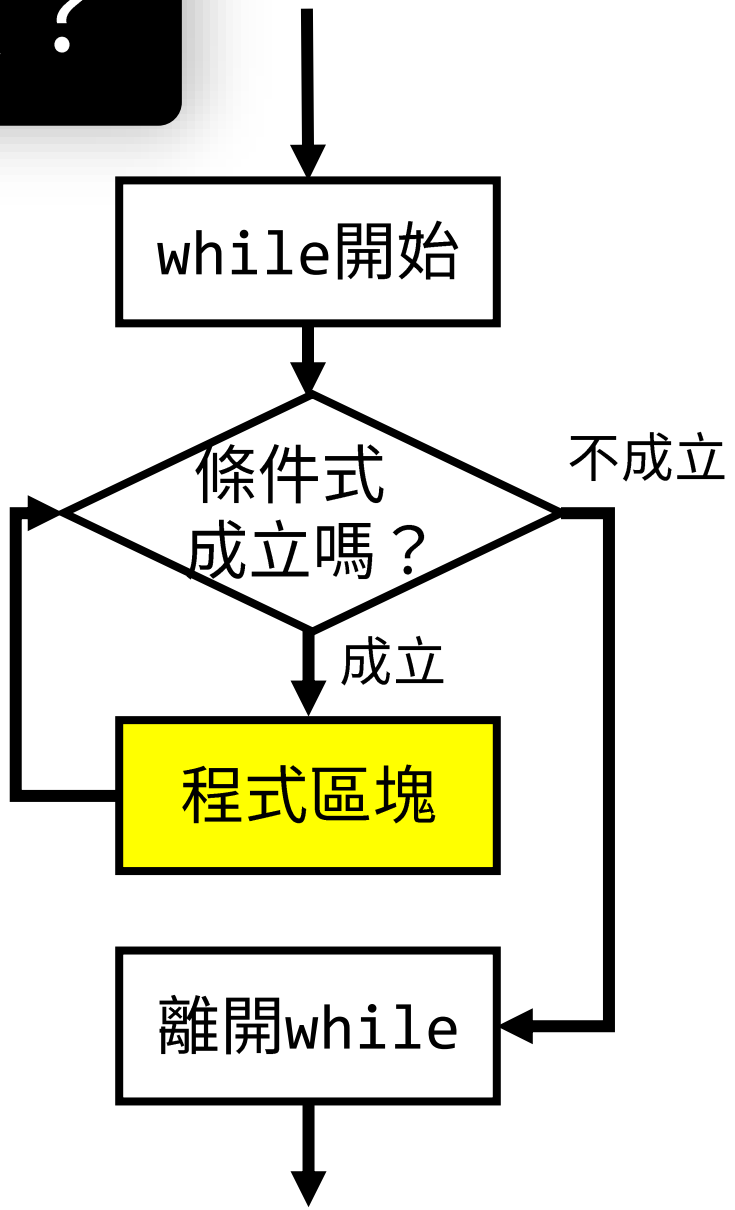
```
1   #include <iostream>
2   using namespace std;
3
4   int main(){
5       int i = 1;
6       while(i <= 5){
7           cout << i << endl;
8           i++;
9       }
10      return 0;
11  }
```

i = 3

i <= 5 為真

輸出 i

```
輸出：
1
2
3
```



如果今天想要列出1~5間的整數？

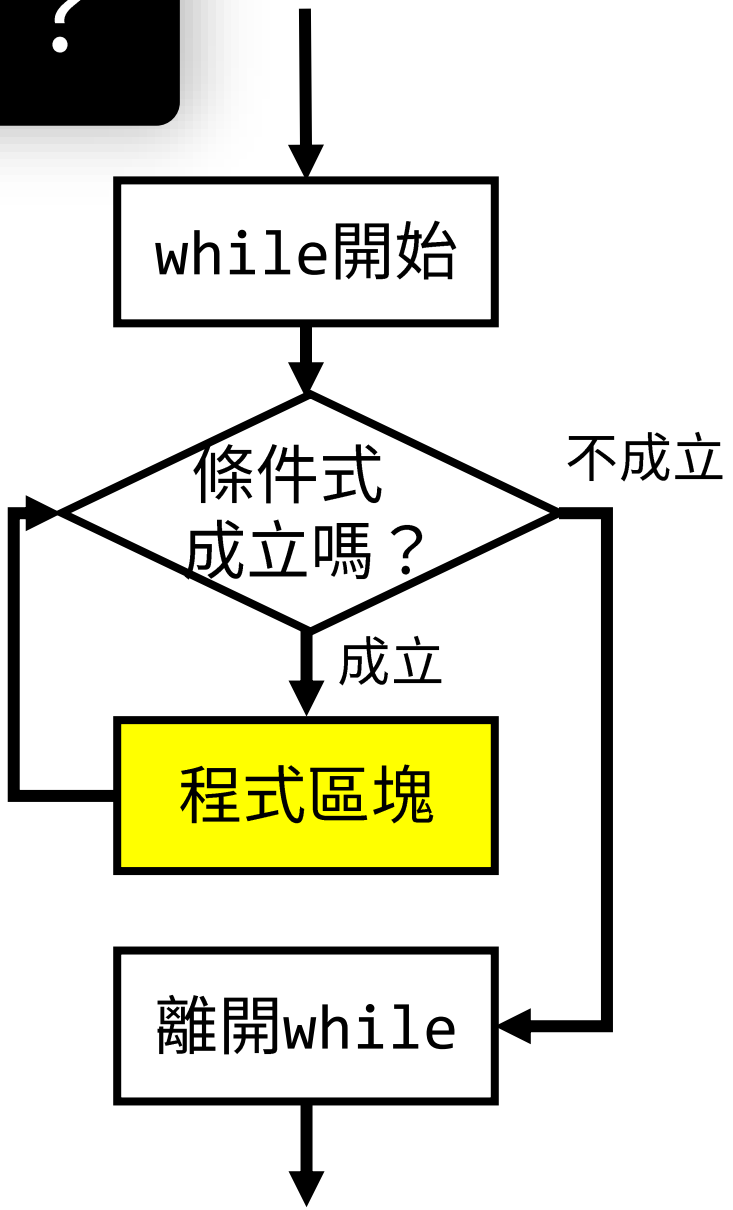
```
1   #include <iostream>
2   using namespace std;
3
4   int main(){
5       int i = 1;
6       while(i <= 5){
7           cout << i << endl;
8           i++;
9       }
10      return 0;
11  }
```

i = 4

i <= 5 為真

i = i + 1

```
輸出：
1
2
3
```

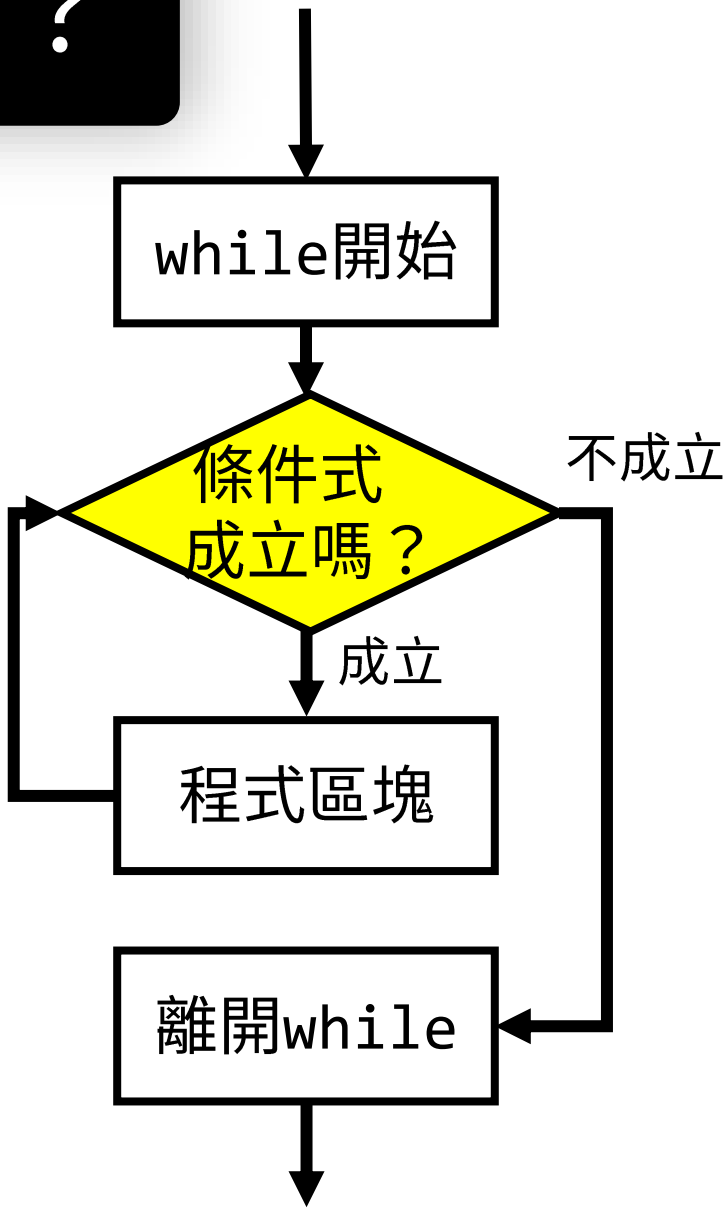


如果今天想要列出1~5間的整數？

```
1   #include <iostream>
2   using namespace std;
3
4   int main(){
5       int i = 1;
6       while(i <= 5){
7           cout << i << endl;
8           i++;
9       }
10      return 0;
11  }
```

i = 4

```
輸出：
1
2
3
```



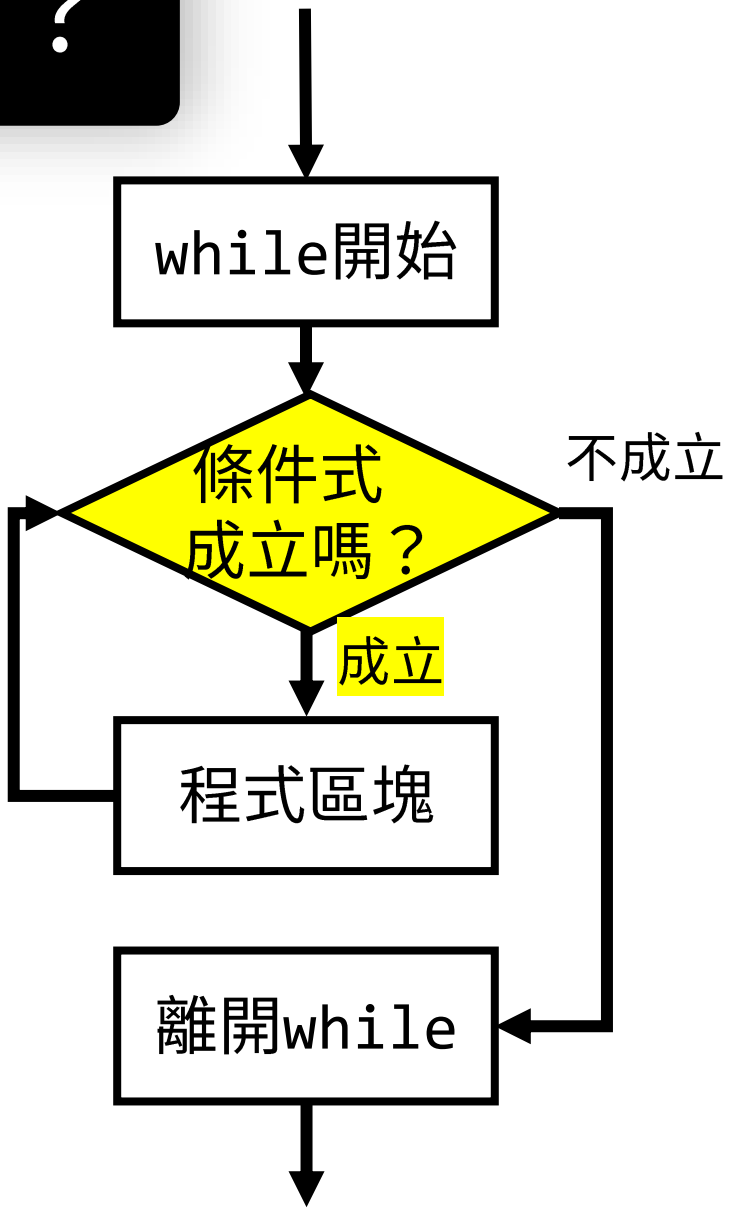
如果今天想要列出1~5間的整數？

```
1   #include <iostream>
2   using namespace std;
3
4   int main(){
5       int i = 1;
6       while(i <= 5){
7           cout << i << endl;
8           i++;
9       }
10      return 0;
11  }
```

i = 4

i <= 5 為真

```
輸出：
1
2
3
```



如果今天想要列出1~5間的整數？

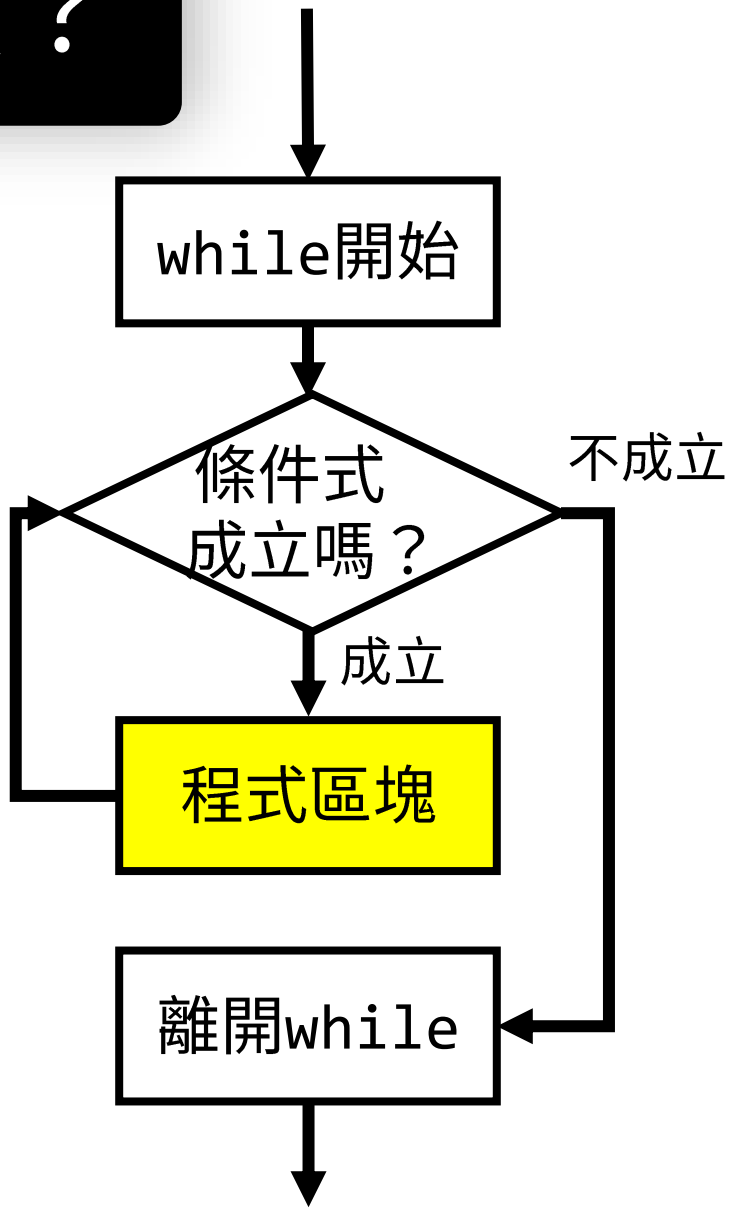
```
1   #include <iostream>
2   using namespace std;
3
4   int main(){
5       int i = 1;
6       while(i <= 5){
7           cout << i << endl;
8           i++;
9       }
10      return 0;
11  }
```

i = 4

i <= 5 為真

輸出 i

```
輸出：
1
2
3
4
```



如果今天想要列出1~5間的整數？

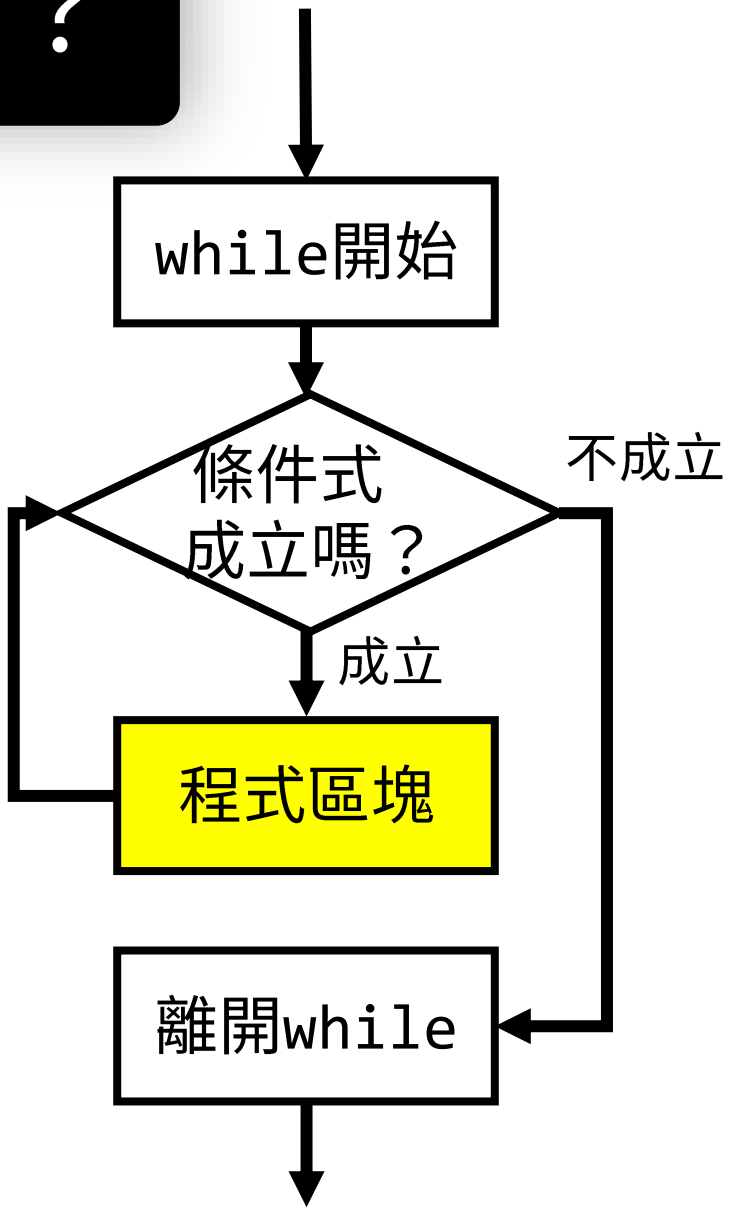
```
1   #include <iostream>
2   using namespace std;
3
4   int main(){
5       int i = 1;
6       while(i <= 5){
7           cout << i << endl;
8           i++;
9       }
10      return 0;
11  }
```

i = 5

i <= 5 為真

i = i + 1

```
輸出：
1
2
3
4
```

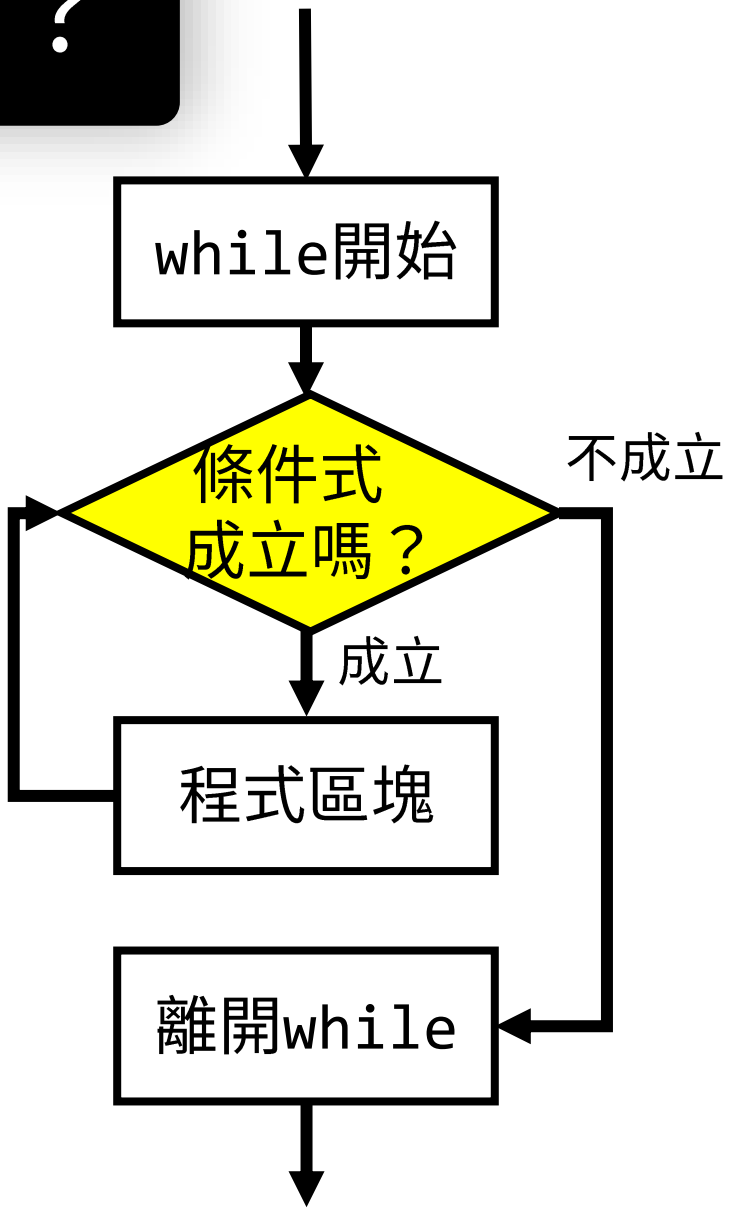


如果今天想要列出1~5間的整數？

```
1   #include <iostream>
2   using namespace std;
3
4   int main(){
5       int i = 1;
6       while(i <= 5){
7           cout << i << endl;
8           i++;
9       }
10      return 0;
11  }
```

i = 5

```
輸出：
1
2
3
4
```



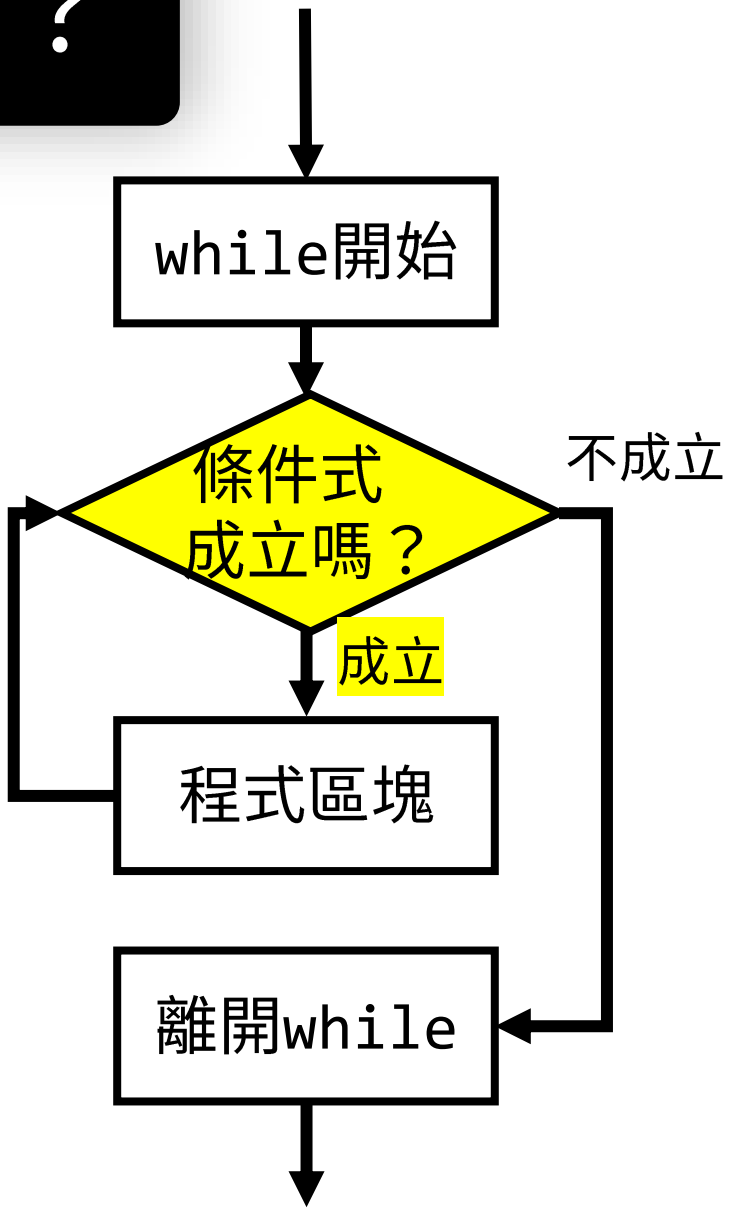
如果今天想要列出1~5間的整數？

```
1   #include <iostream>
2   using namespace std;
3
4   int main(){
5       int i = 1;
6       while(i <= 5){
7           cout << i << endl;
8           i++;
9       }
10      return 0;
11  }
```

i = 5

i <= 5 為真

```
輸出：
1
2
3
4
```



如果今天想要列出1~5間的整數？

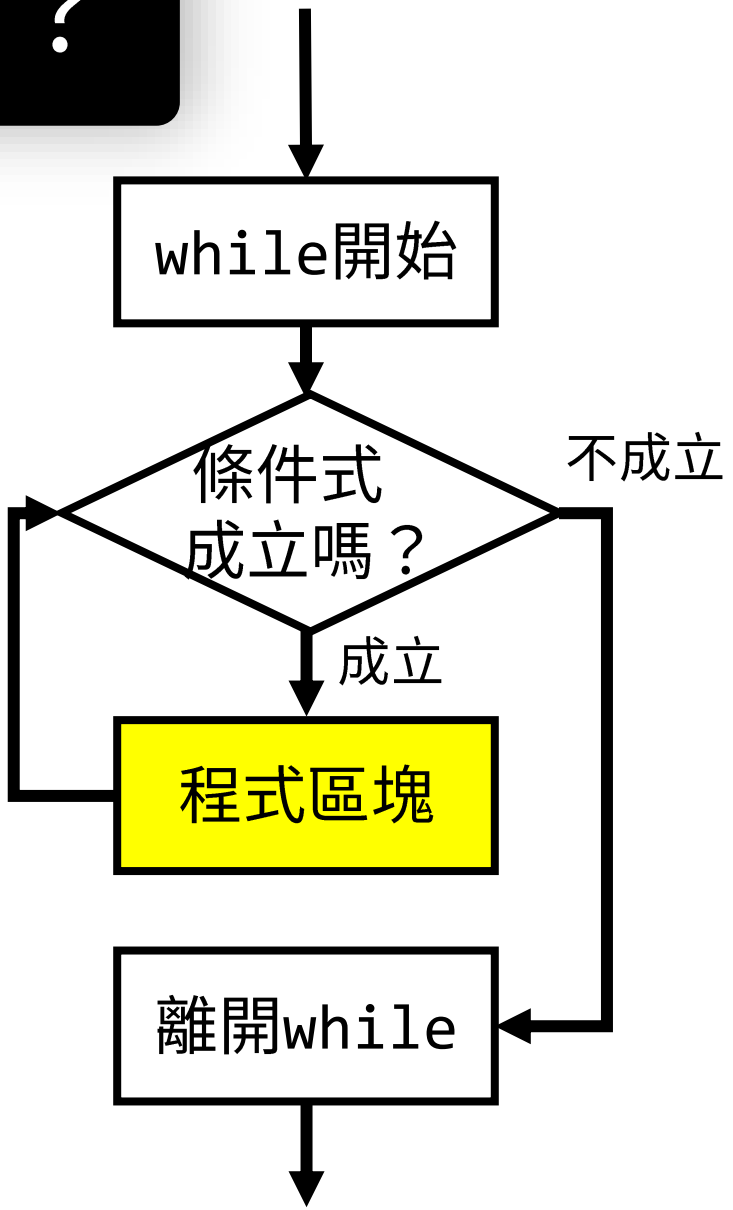
```
1   #include <iostream>
2   using namespace std;
3
4   int main(){
5       int i = 1;
6       while(i <= 5){
7           cout << i << endl;
8           i++;
9       }
10      return 0;
11  }
```

i = 5

i <= 5 為真

輸出 i

```
輸出：
1
2
3
4
5
```



如果今天想要列出1~5間的整數？

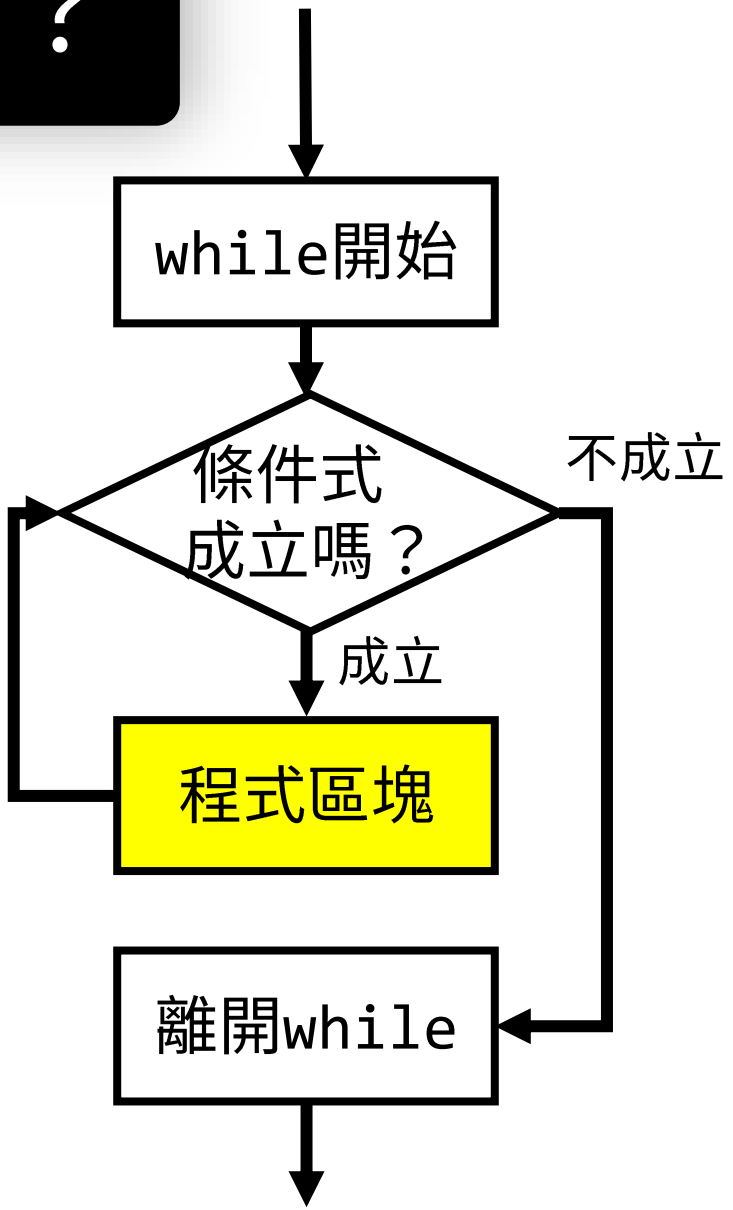
```
1   #include <iostream>
2   using namespace std;
3
4   int main(){
5       int i = 1;
6       while(i <= 5){
7           cout << i << endl;
8           i++;
9       }
10      return 0;
11  }
```

i = 6

i <= 5 為真

i = i + 1

```
輸出：
1
2
3
4
5
```

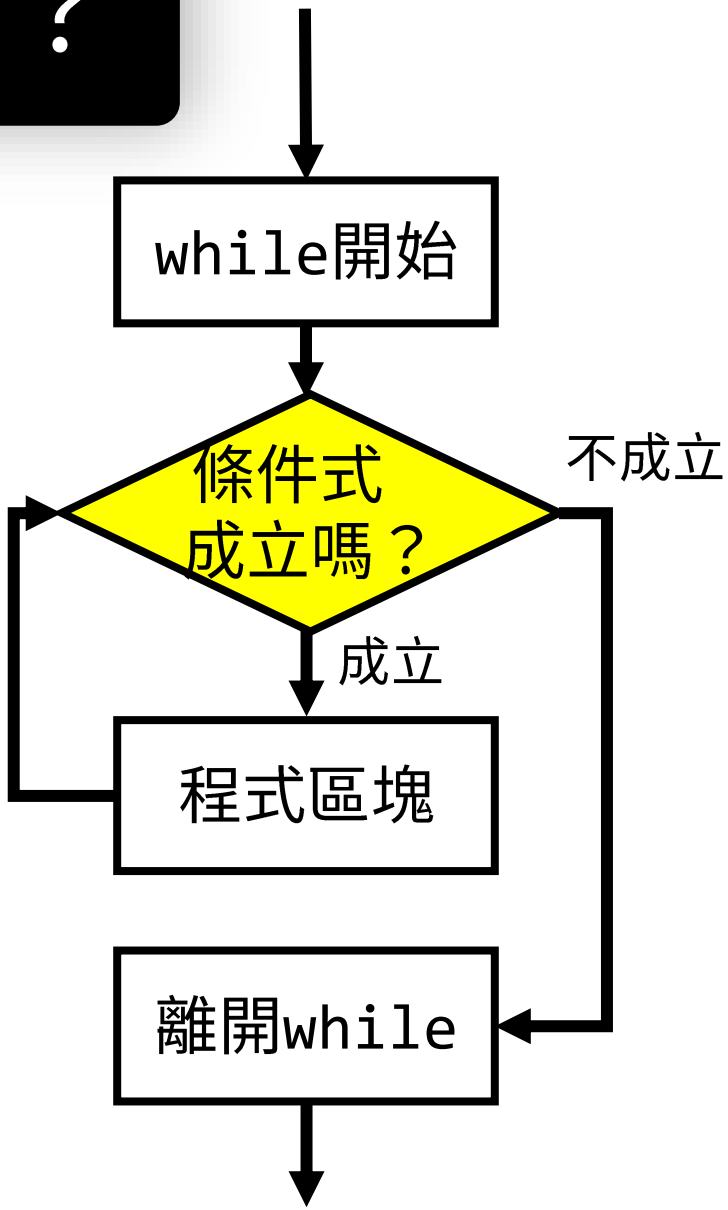


如果今天想要列出1~5間的整數？

```
1   #include <iostream>
2   using namespace std;
3
4   int main(){
5       int i = 1;
6       while(i <= 5){
7           cout << i << endl;
8           i++;
9       }
10      return 0;
11  }
```

i = 6

```
輸出：
1
2
3
4
5
```



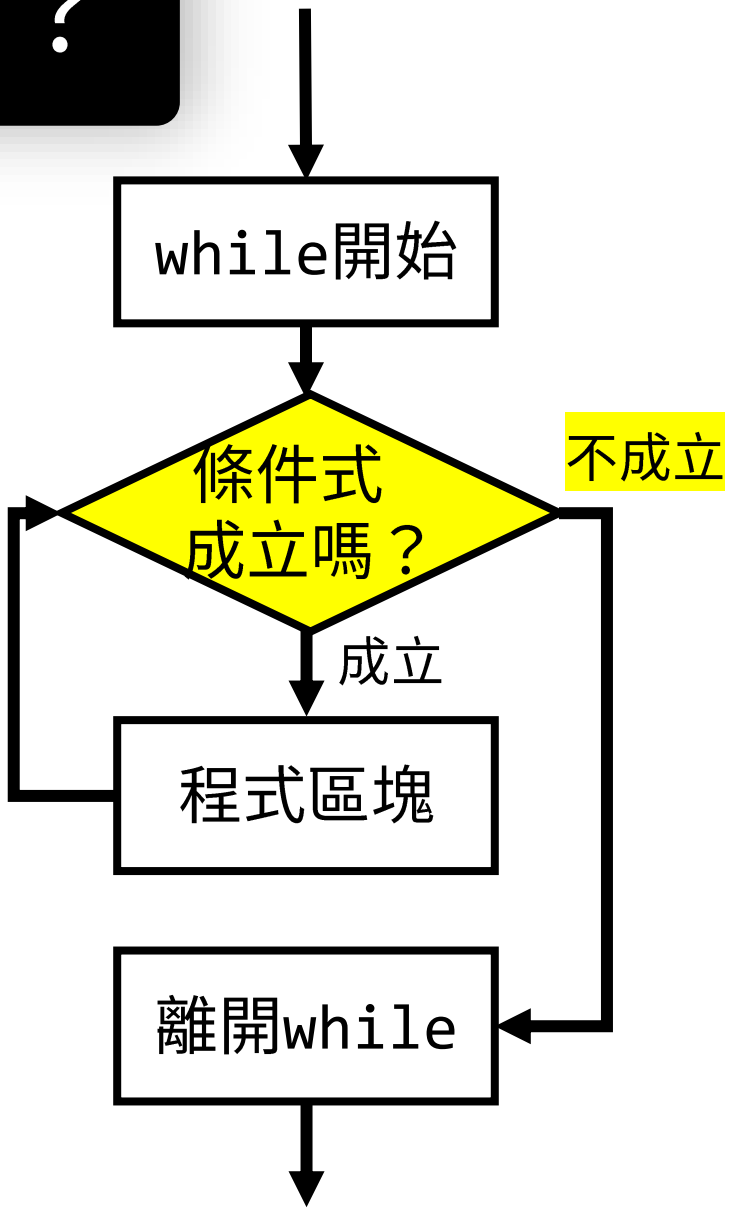
如果今天想要列出1~5間的整數？

```
1   #include <iostream>
2   using namespace std;
3
4   int main(){
5       int i = 1;
6       while(i <= 5){
7           cout << i << endl;
8           i++;
9       }
10      return 0;
11  }
```

i = 6

i <= 5 為非

```
輸出：
1
2
3
4
5
```



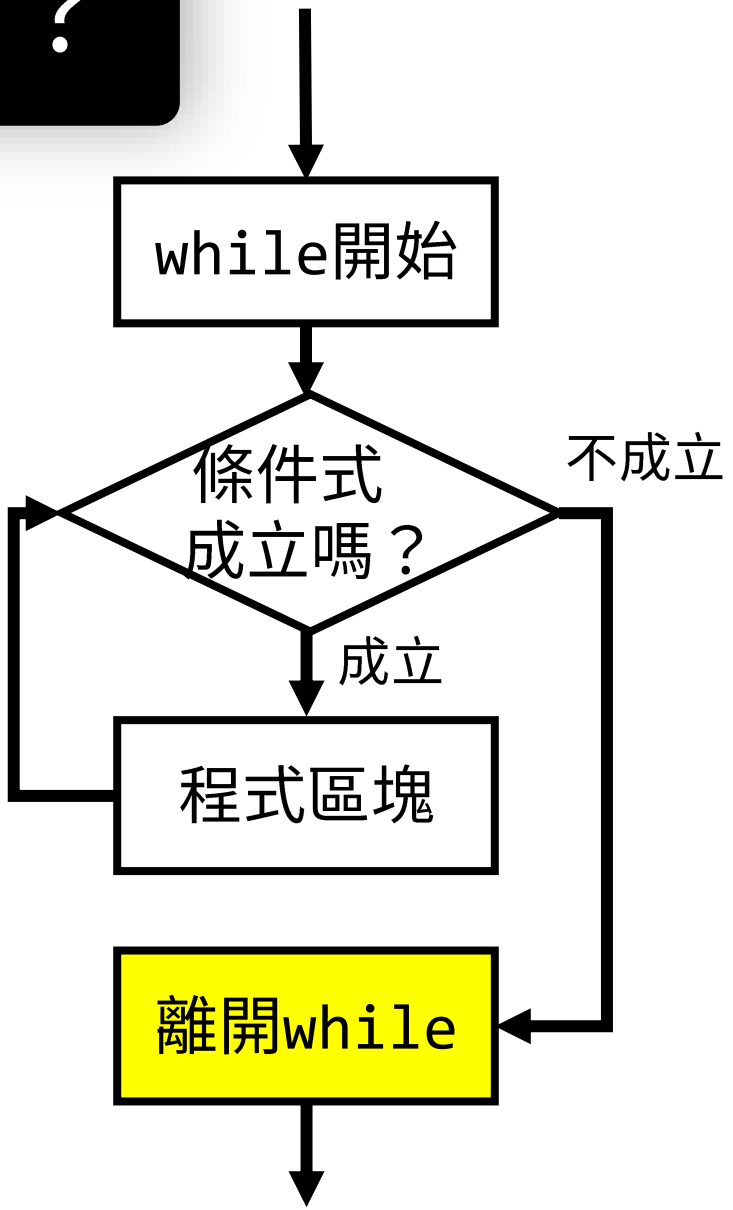
如果今天想要列出1~5間的整數？

```
1   #include <iostream>
2   using namespace std;
3
4   int main(){
5       int i = 1;
6       while(i <= 5){
7           cout << i << endl;
8           i++;
9       }
10      return 0;
11  }
```

i = 6

i <= 5 為非

```
輸出：
1
2
3
4
5
```



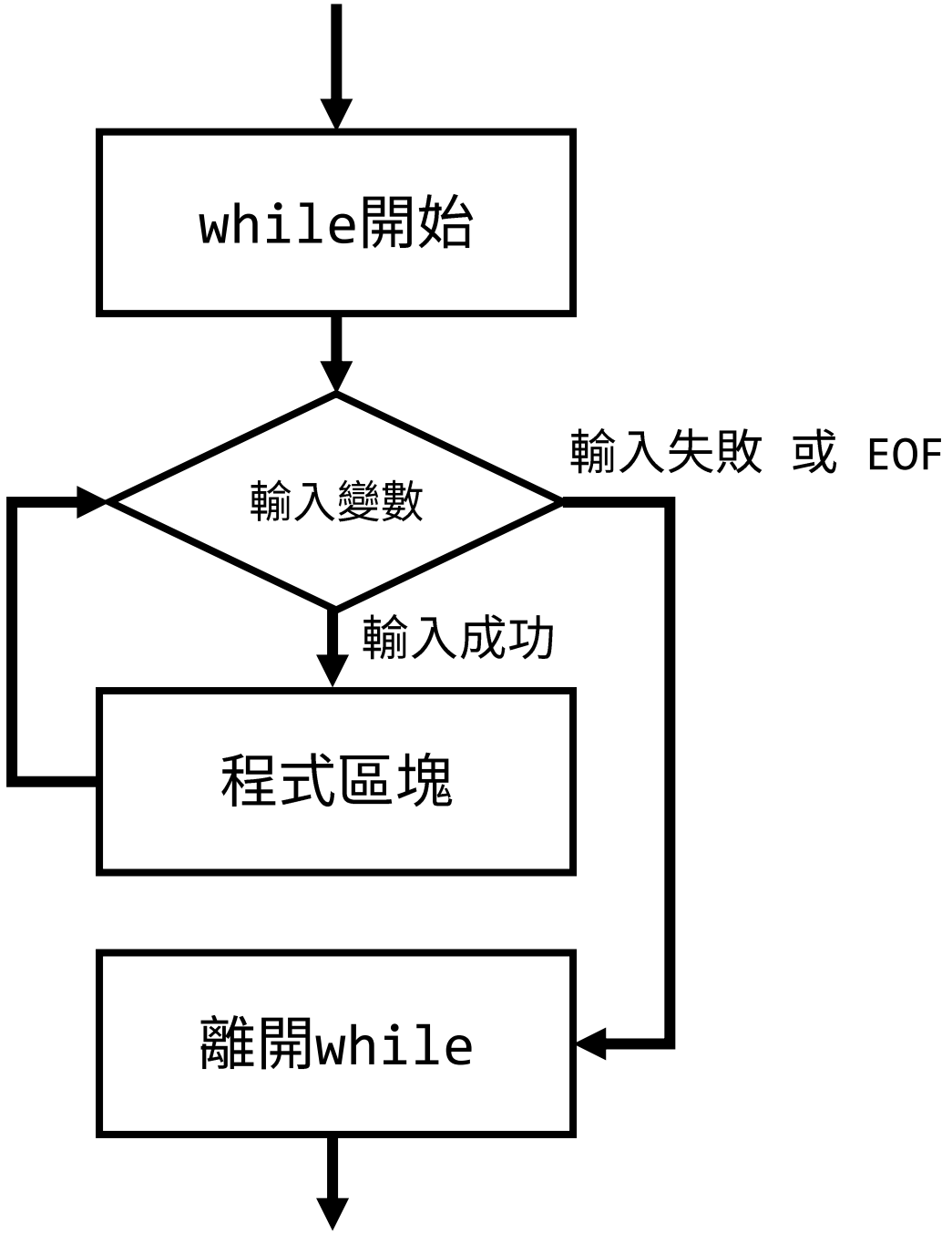
while 重複輸入

```
while(cin >> 變數){
    程式區塊;
}
```

EOF (End of File) 檔案結尾

顧名思義，就是檔案的結尾

在 Windows 上可以利用 Ctrl-Z 輸入
在 Linux 上可以利用 Ctrl-D 輸入



do-while 迴圈

與while迴圈類似，
不過會先執行程式
區塊再檢查條件式

```
do{
    程式區塊;
} while(條件式(布林值));
```

do-while 迴圈

與while迴圈類似，
不過會先執行程式
區塊再檢查條件式

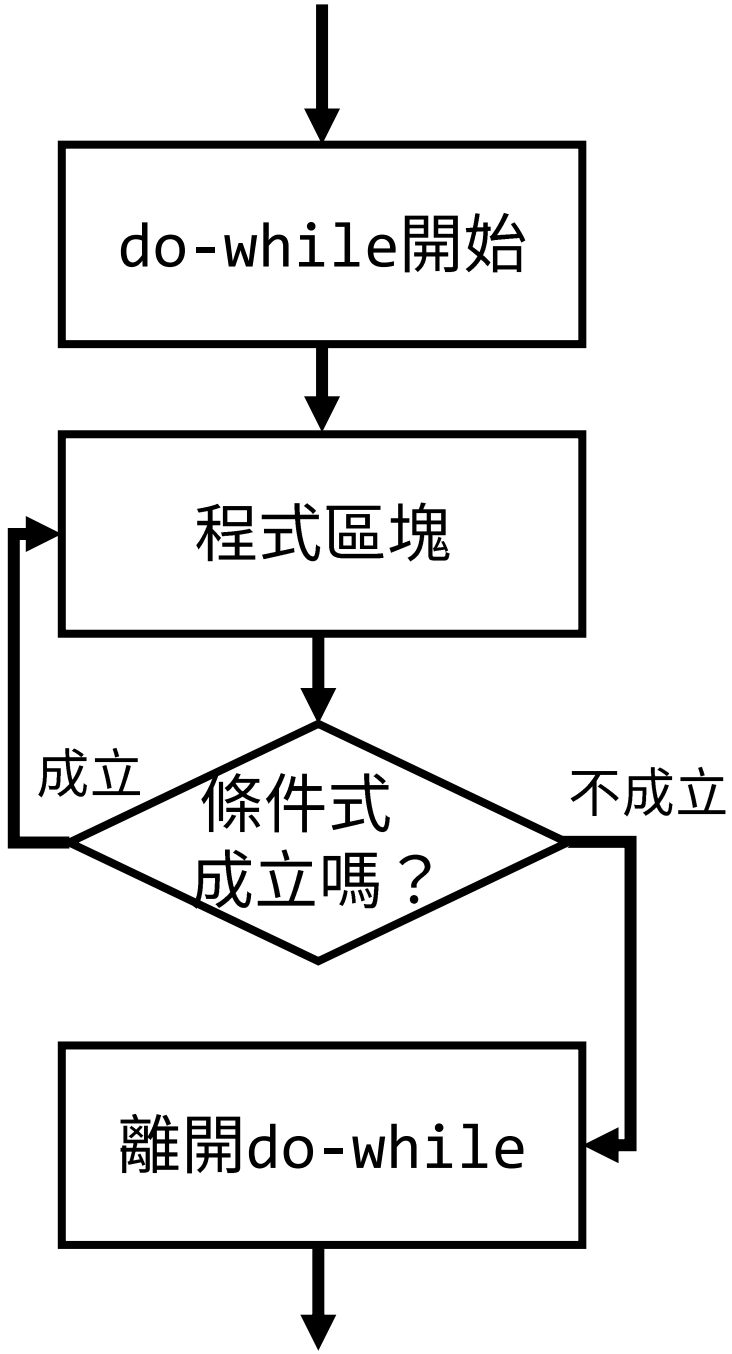
```
do{
    程式區塊;
} while(條件式(布林值));
```

要加;

do-while 迴圈

```
do{
    程式區塊;
} while(條件式(布林值));
```

流程圖：

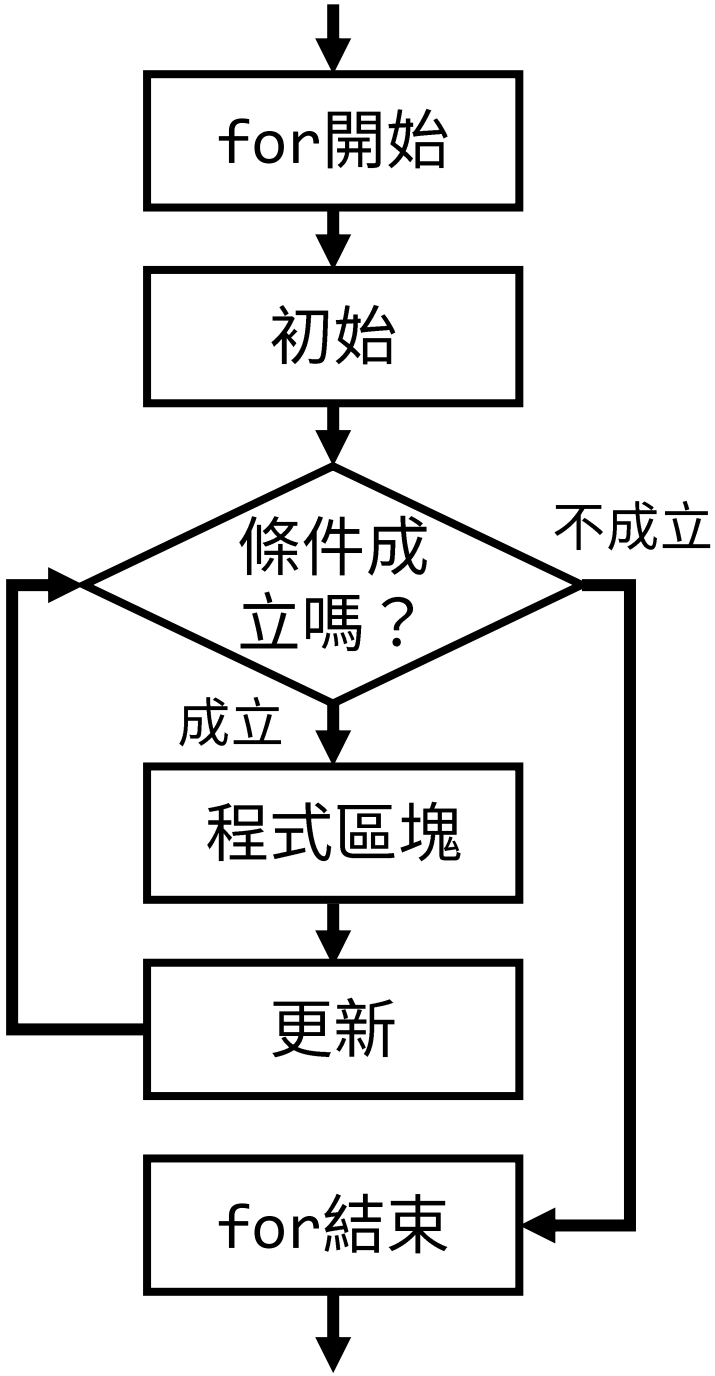


for 迴圈

```
for (初始; 條件; 更新){
    程式區塊;
}
```

初始：只會執行一次的程式
條件：繼續執行迴圈的判斷依據
更新：執行完程式區塊後執行的程式

流程圖：



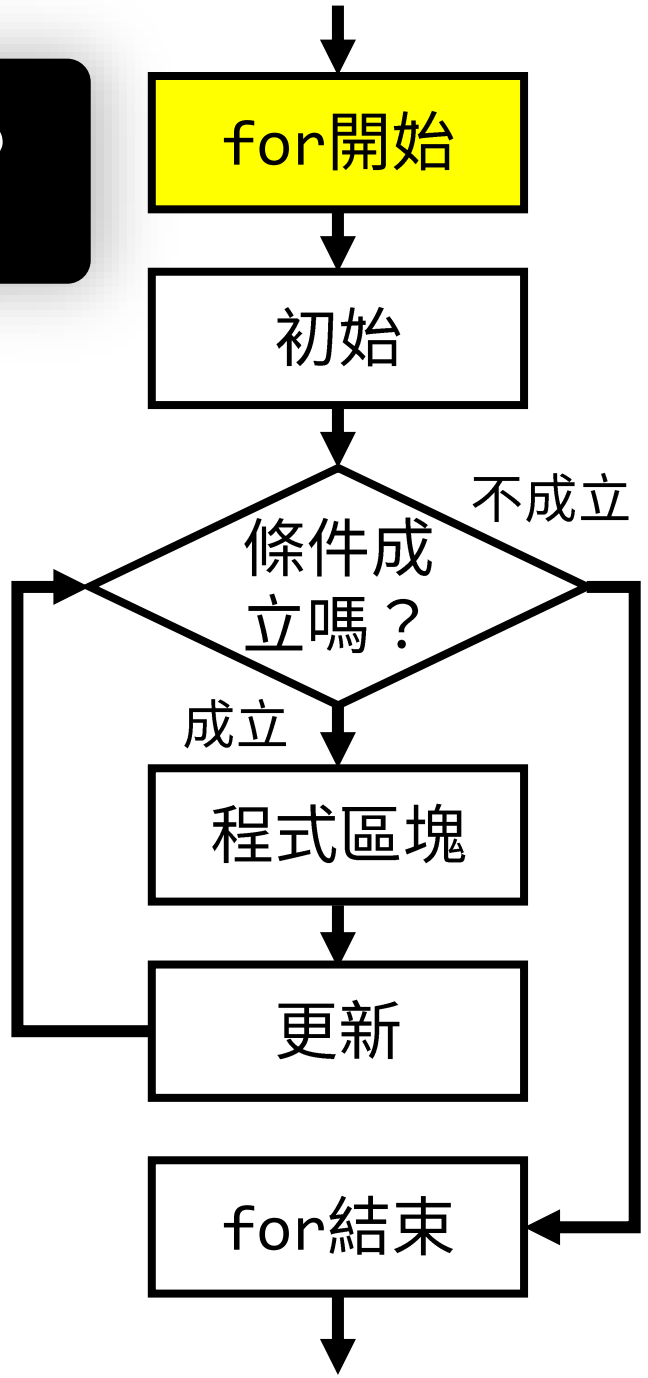
如果今天想要列出1~5間的整數？

```
 1  #include <iostream>
 2  using namespace std;
 3
 4  int main()
 5  {
 6      for (int i = 1; i <= 5; i++){
 7          cout << i << endl;
 8      }
 9      return 0;
10  }
```

如果今天想要列出1~5間的整數？

```
1   #include <iostream>
2   using namespace std;
3
4   int main()
5   {
6       for (int i = 1; i <= 5; i++){
7           cout << i << endl;
8       }
9       return 0;
10  }
```

輸出：

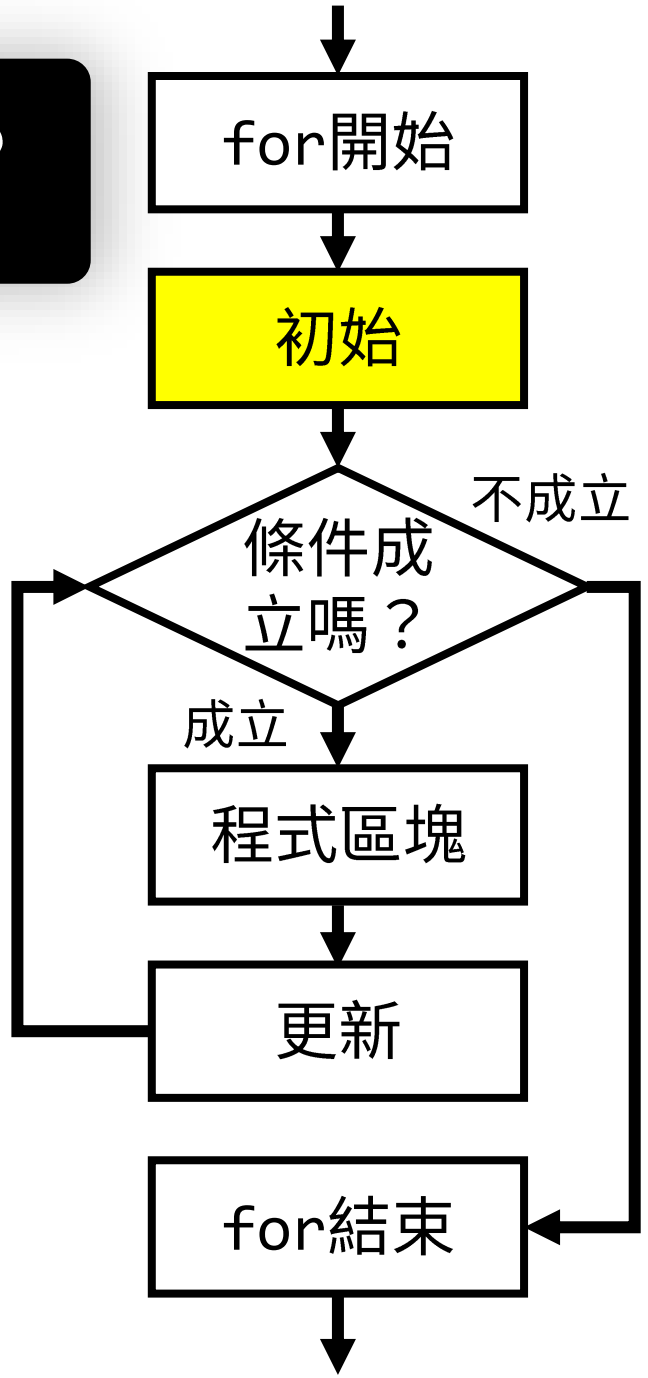


如果今天想要列出1~5間的整數？

```
1   #include <iostream>
2   using namespace std;
3
4   int main()
5   {
6       for (int i = 1; i <= 5; i++){
7           cout << i << endl;
8       }
9       return 0;
10  }
```

宣告整數 i = 1

輸出：



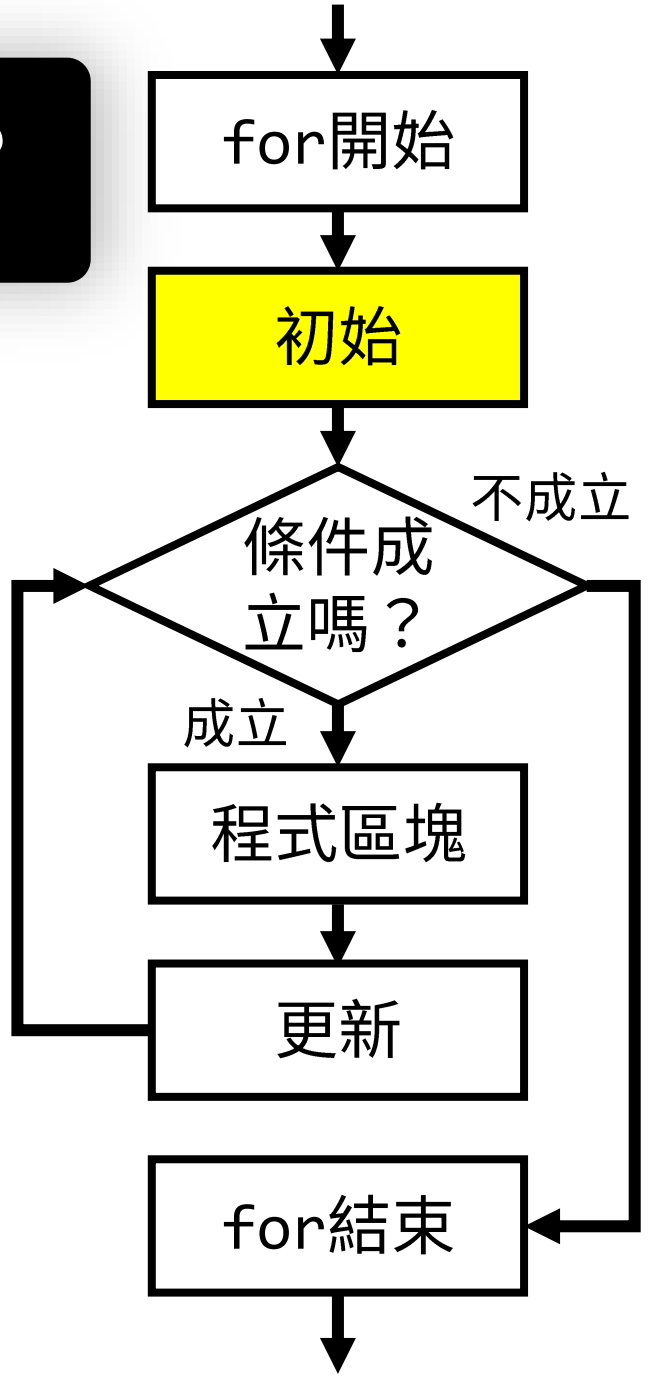
如果今天想要列出1~5間的整數？

```
1   #include <iostream>
2   using namespace std;
3
4   int main()
5   {
6       for (int i = 1; i <= 5; i++){
7           cout << i << endl;
8       }
9       return 0;
10  }
```

i = 1

宣告整數 i = 1

輸出：

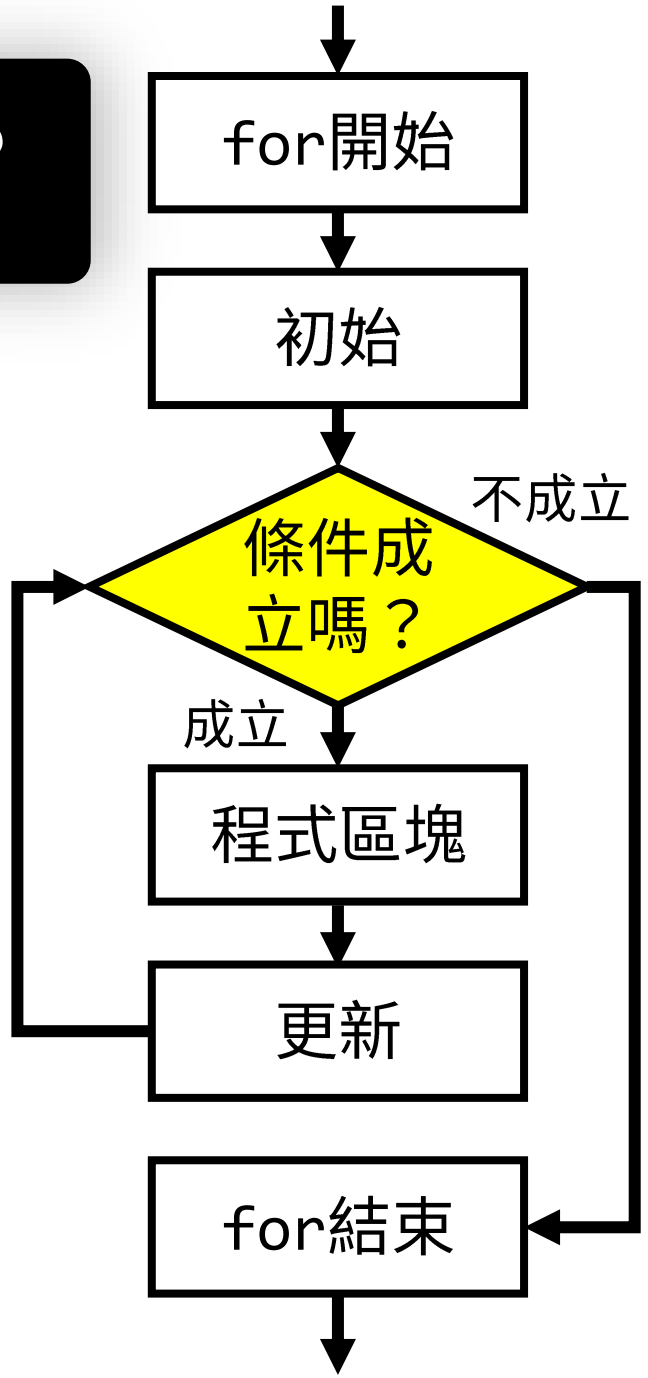


如果今天想要列出1~5間的整數？

```
1   #include <iostream>
2   using namespace std;
3
4   int main()
5   {
6       for (int i = 1; i <= 5; i++){
7           cout << i << endl;
8       }
9       return 0;
10  }
```

i = 1

輸出：



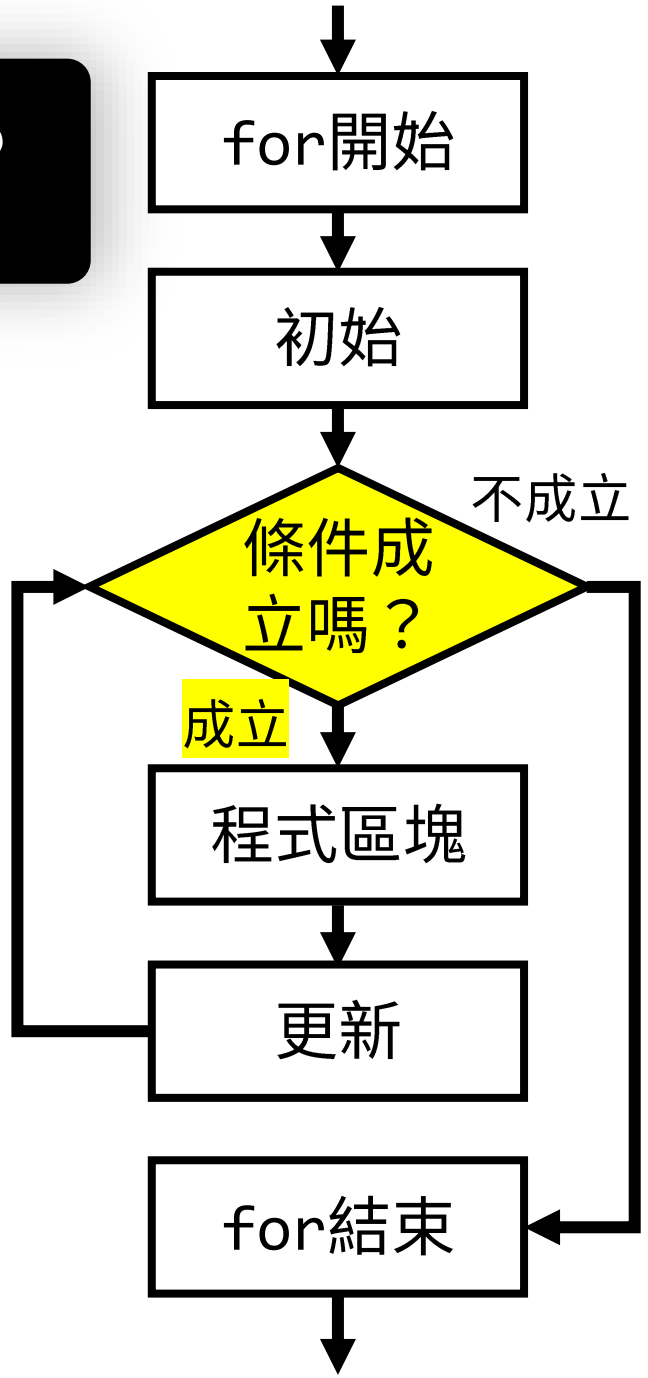
如果今天想要列出1~5間的整數？

```
1   #include <iostream>
2   using namespace std;
3
4   int main()
5   {
6       for (int i = 1; i <= 5; i++){
7           cout << i << endl;
8       }
9       return 0;
10  }
```

i = 1

i <= 5為真

輸出：



如果今天想要列出1~5間的整數？

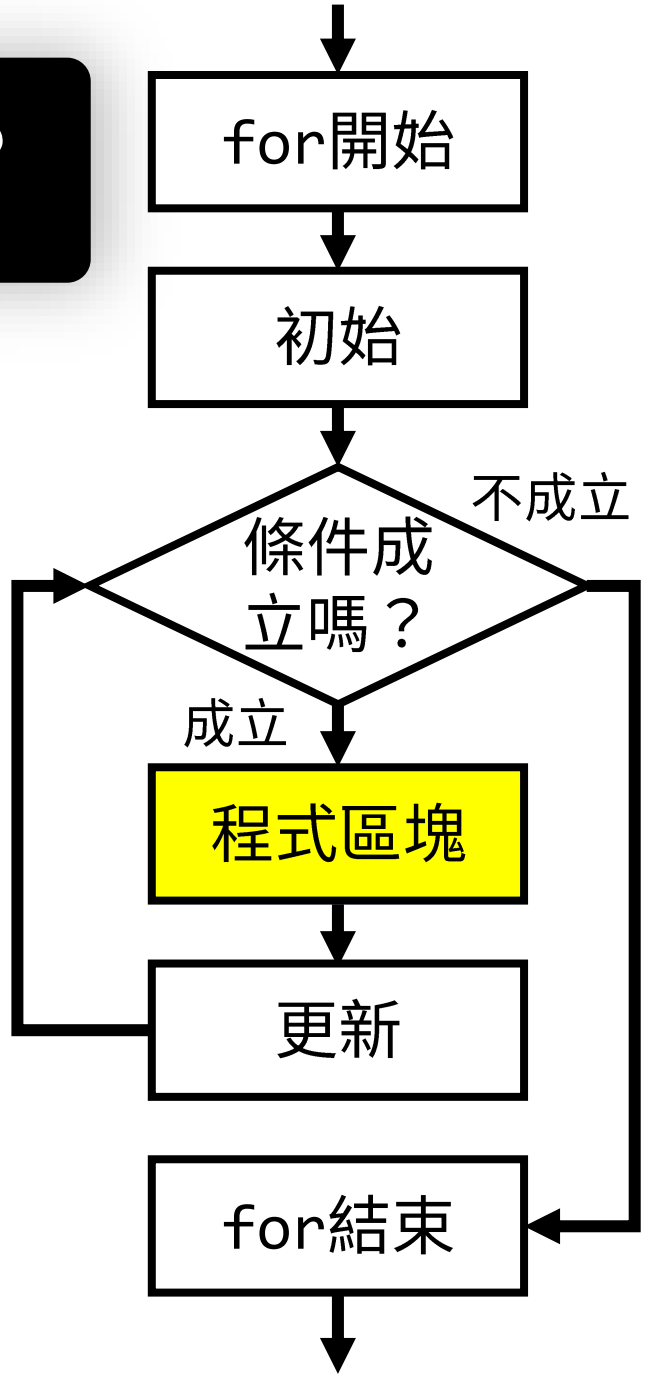
```
1   #include <iostream>
2   using namespace std;
3
4   int main()
5   {
6       for (int i = 1; i <= 5; i++){
7           cout << i << endl;
8       }
9       return 0;
10  }
```

i = 1

i <= 5為真

輸出i

輸出：
1



如果今天想要列出1~5間的整數？

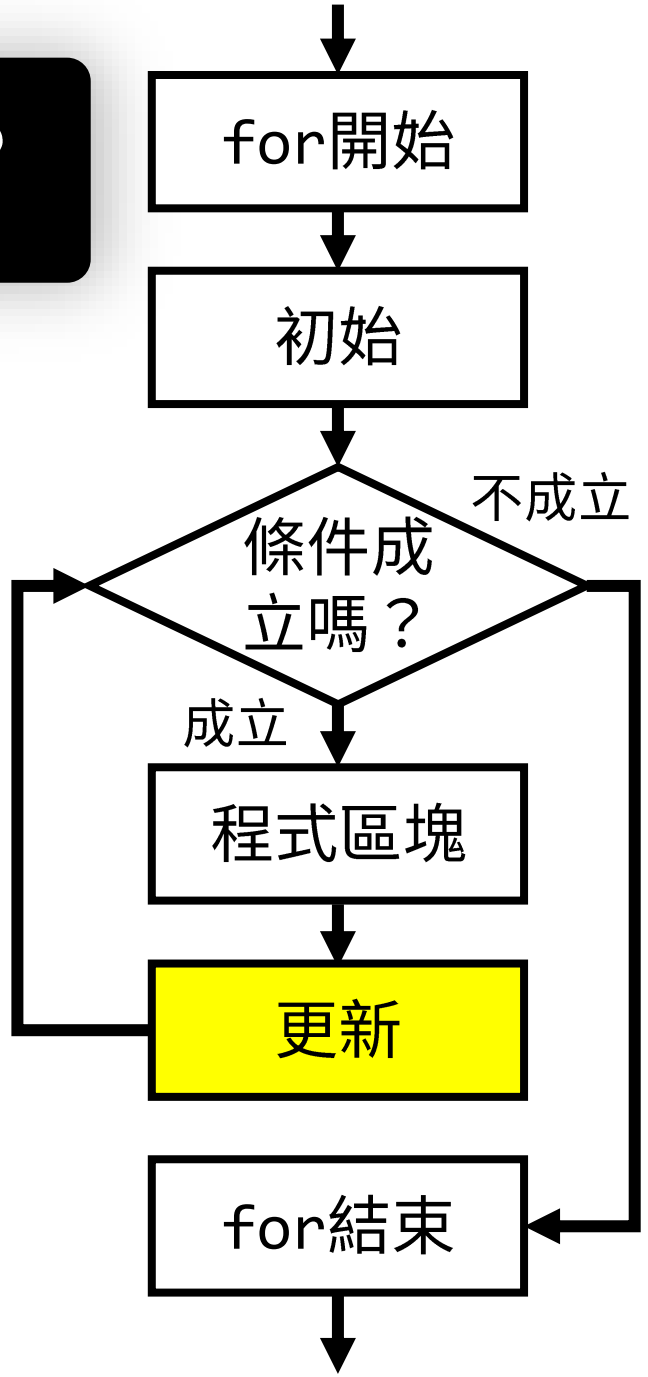
```
1   #include <iostream>
2   using namespace std;
3
4   int main()
5   {
6       for (int i = 1; i <= 5; i++){
7           cout << i << endl;
8       }
9       return 0;
10  }
```

i = 2

i <= 5為真

i = i + 1

輸出：
1

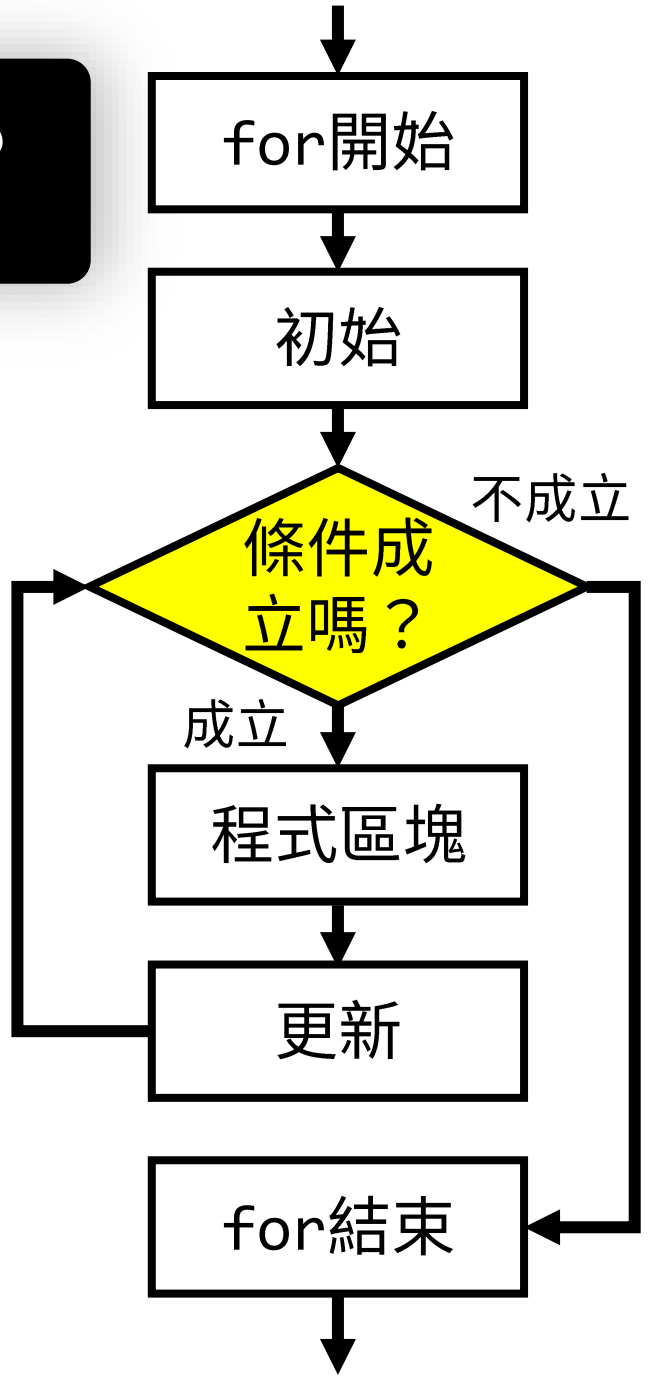


如果今天想要列出1~5間的整數？

```
1   #include <iostream>
2   using namespace std;
3
4   int main()
5   {
6       for (int i = 1; i <= 5; i++){
7           cout << i << endl;
8       }
9       return 0;
10  }
```

i = 2

輸出：
1



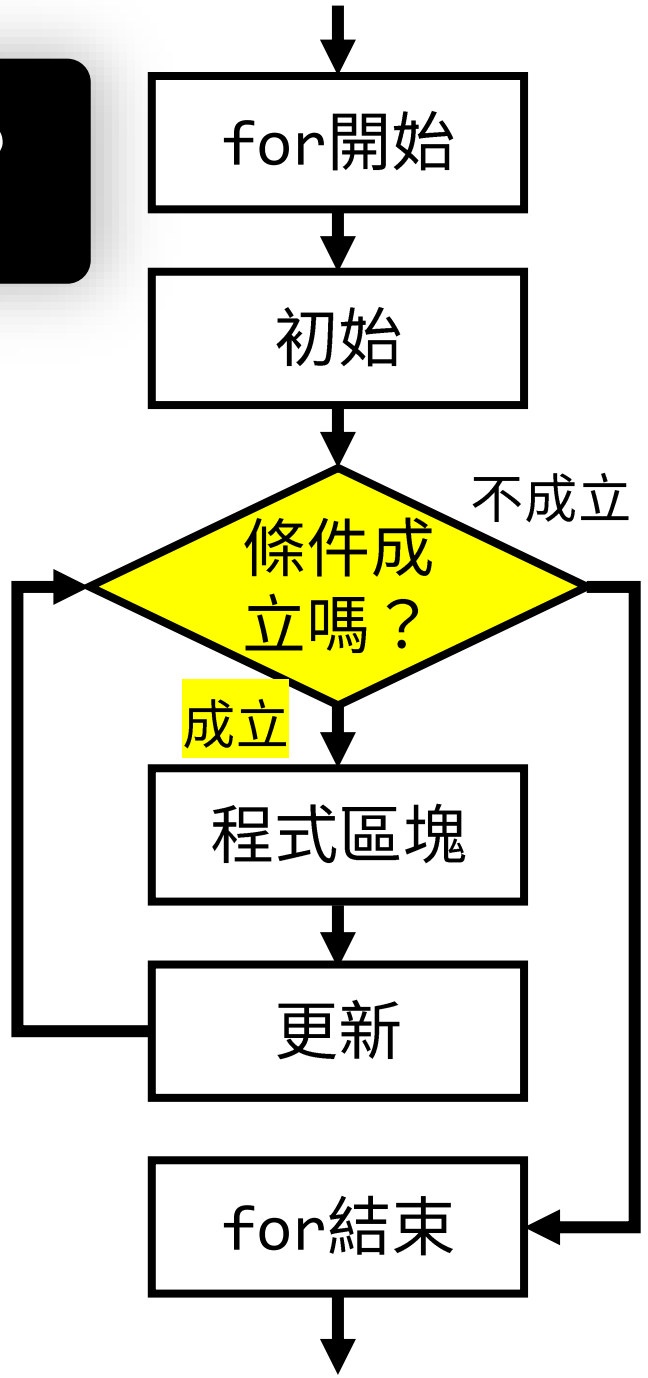
如果今天想要列出1~5間的整數？

```
1   #include <iostream>
2   using namespace std;
3
4   int main()
5   {
6       for (int i = 1; i <= 5; i++){
7           cout << i << endl;
8       }
9       return 0;
10  }
```

i = 2

i <= 5為真

輸出：
1



如果今天想要列出1~5間的整數？

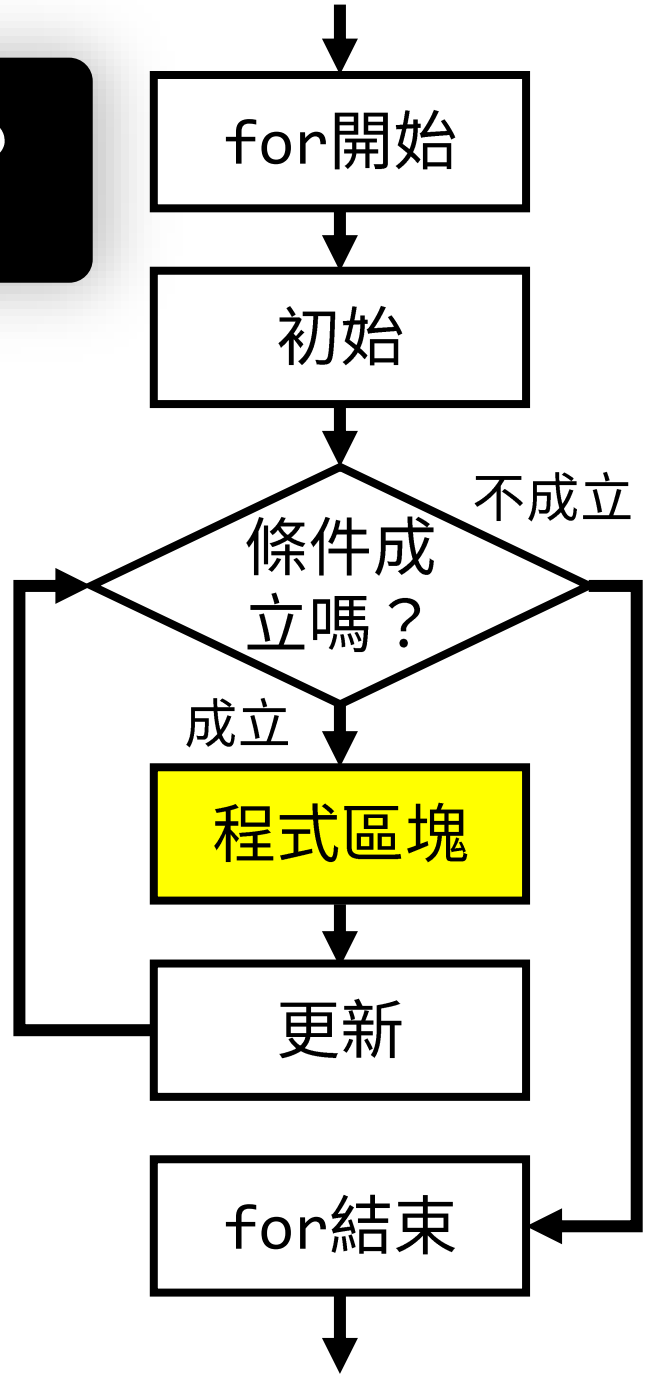
```
1   #include <iostream>
2   using namespace std;
3
4   int main()
5   {
6       for (int i = 1; i <= 5; i++){
7           cout << i << endl;
8       }
9       return 0;
10  }
```

i = 2

i <= 5為真

輸出i

```
輸出：
1
2
```



如果今天想要列出1~5間的整數？

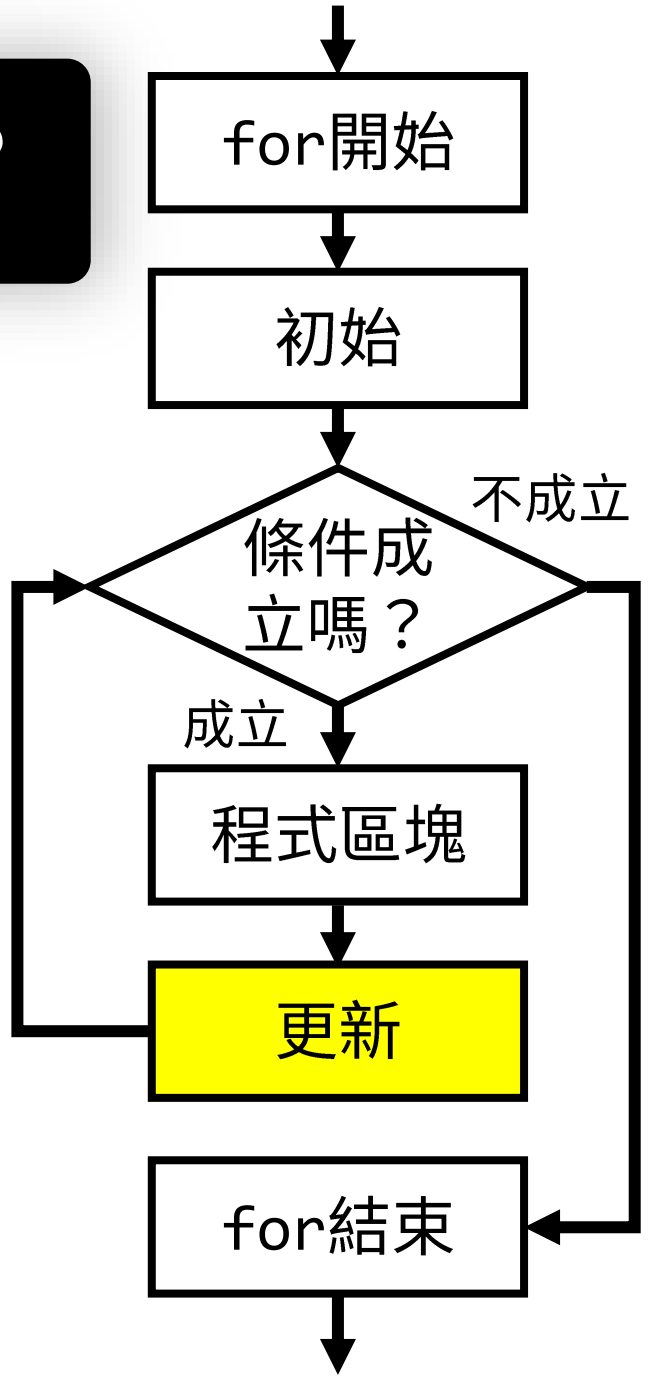
```
1   #include <iostream>
2   using namespace std;
3
4   int main()
5   {
6       for (int i = 1; i <= 5; i++){
7           cout << i << endl;
8       }
9       return 0;
10  }
```

i = 3

i <= 5為真

i = i + 1

輸出：
1
2

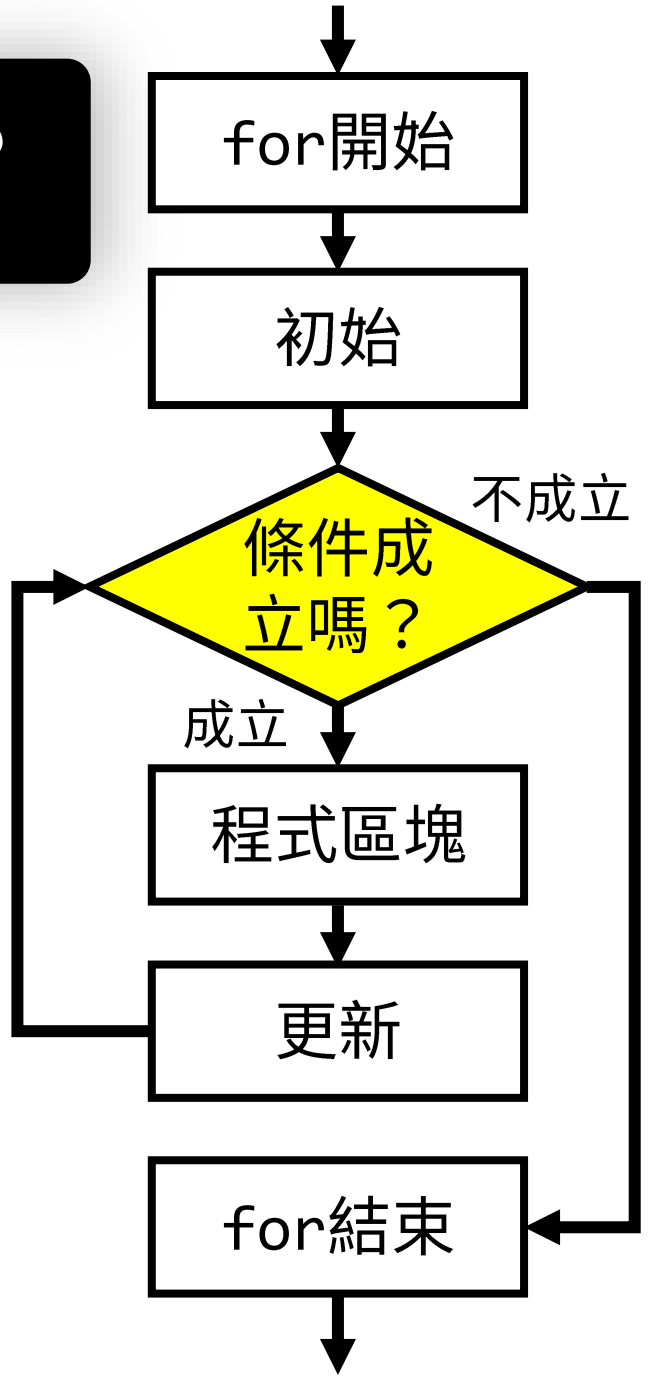


如果今天想要列出1~5間的整數？

```
1   #include <iostream>
2   using namespace std;
3
4   int main()
5   {
6       for (int i = 1; i <= 5; i++){
7           cout << i << endl;
8       }
9       return 0;
10  }
```

i = 3

輸出：
1
2



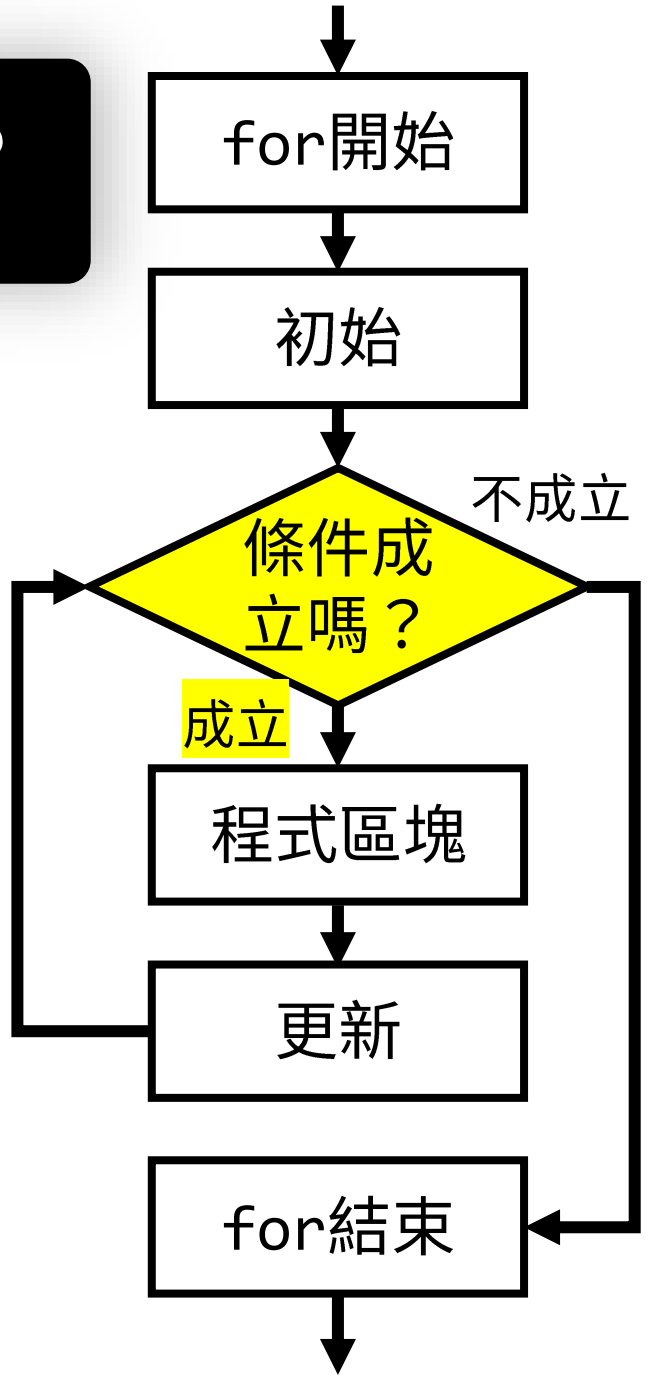
如果今天想要列出1~5間的整數？

```
1   #include <iostream>
2   using namespace std;
3
4   int main()
5   {
6       for (int i = 1; i <= 5; i++){
7           cout << i << endl;
8       }
9       return 0;
10  }
```

i = 3

i <= 5為真

輸出：
1
2



如果今天想要列出1~5間的整數？

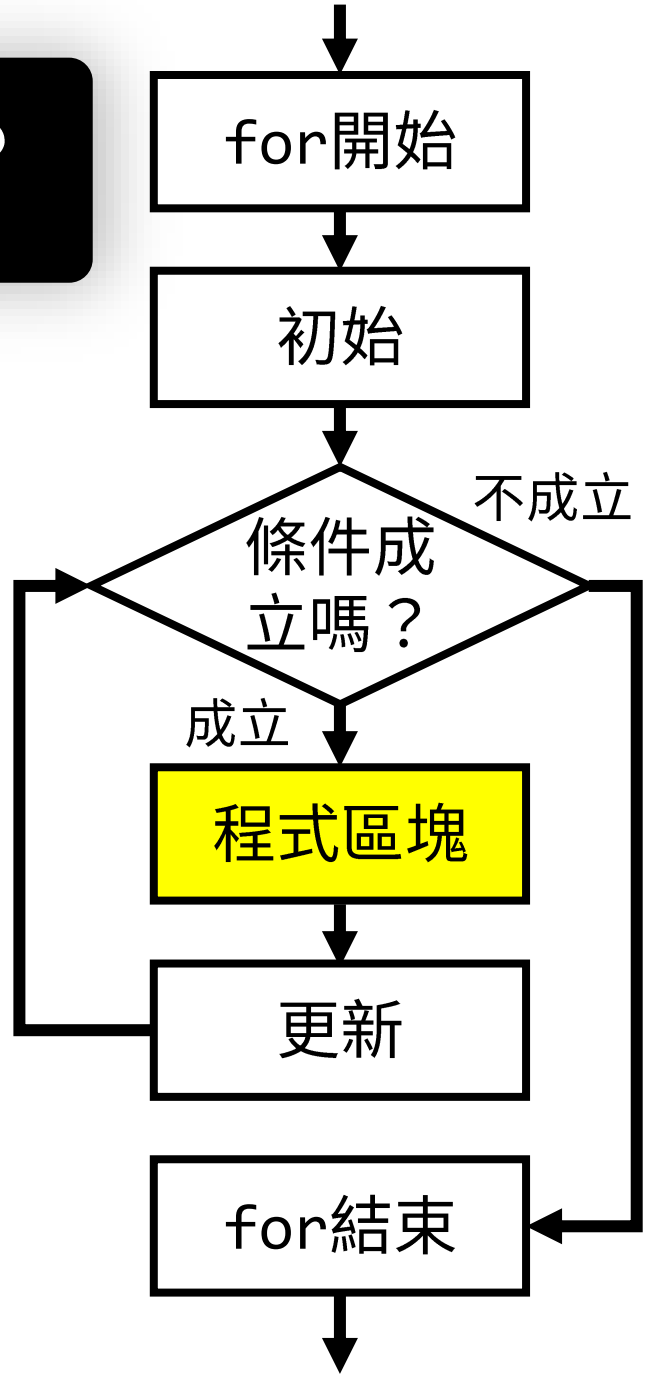
```
1   #include <iostream>
2   using namespace std;
3
4   int main()
5   {
6       for (int i = 1; i <= 5; i++){
7           cout << i << endl;
8       }
9       return 0;
10  }
```

i = 3

i <= 5為真

輸出i

```
輸出：
1
2
3
```



如果今天想要列出1~5間的整數？

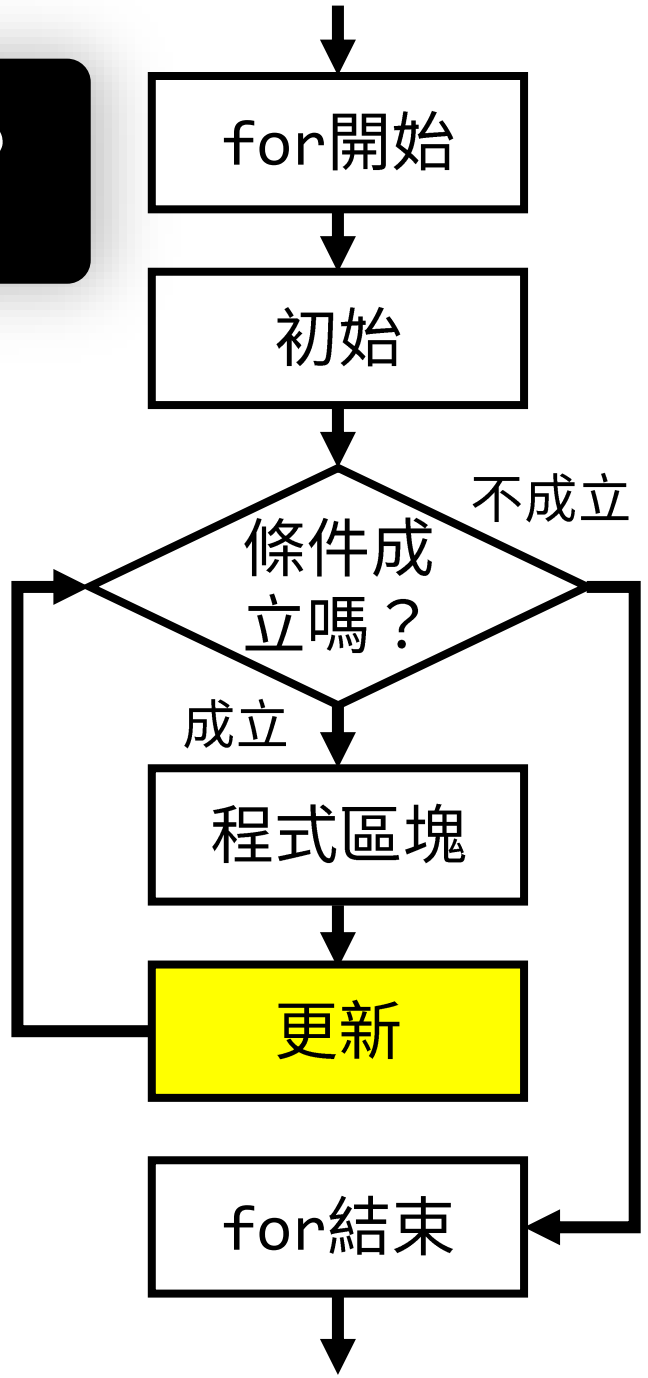
```
1   #include <iostream>
2   using namespace std;
3
4   int main()
5   {
6       for (int i = 1; i <= 5; i++){
7           cout << i << endl;
8       }
9       return 0;
10  }
```

i = 4

i <= 5為真

i = i + 1

輸出：
1
2
3

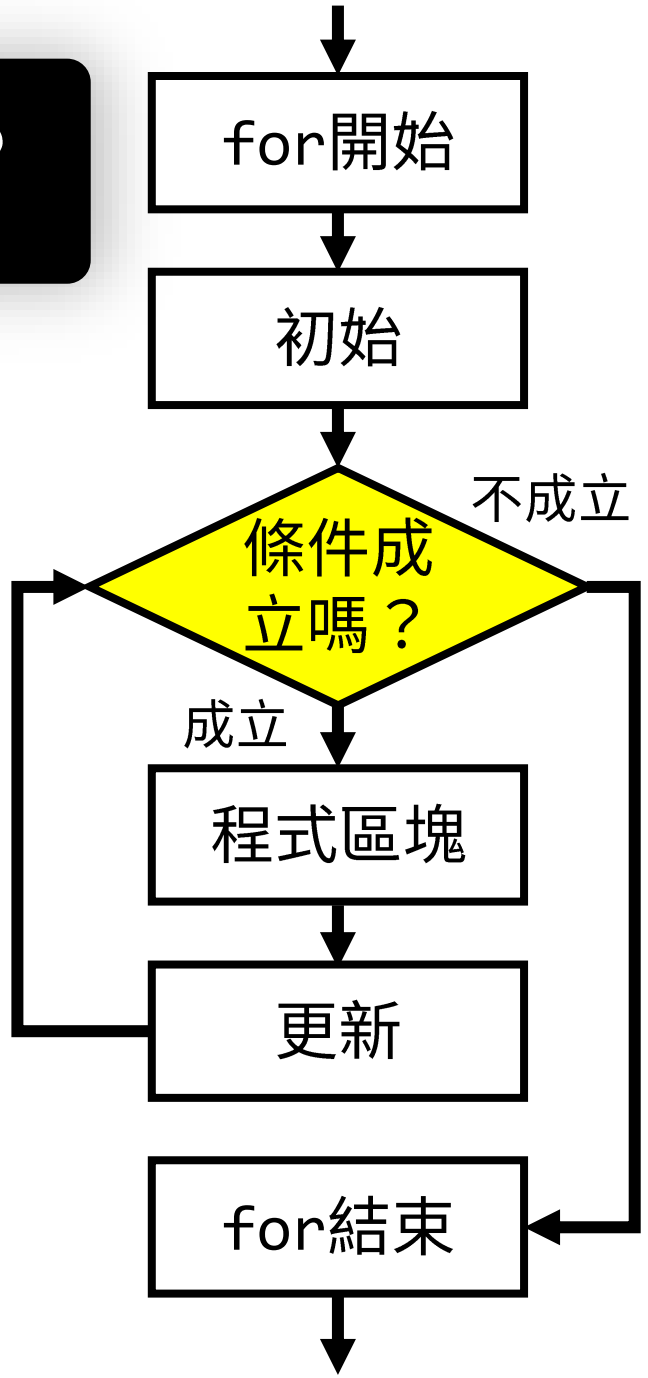


如果今天想要列出1~5間的整數？

```
1   #include <iostream>
2   using namespace std;
3
4   int main()
5   {
6       for (int i = 1; i <= 5; i++){
7           cout << i << endl;
8       }
9       return 0;
10  }
```

i = 4

輸出：
1
2
3



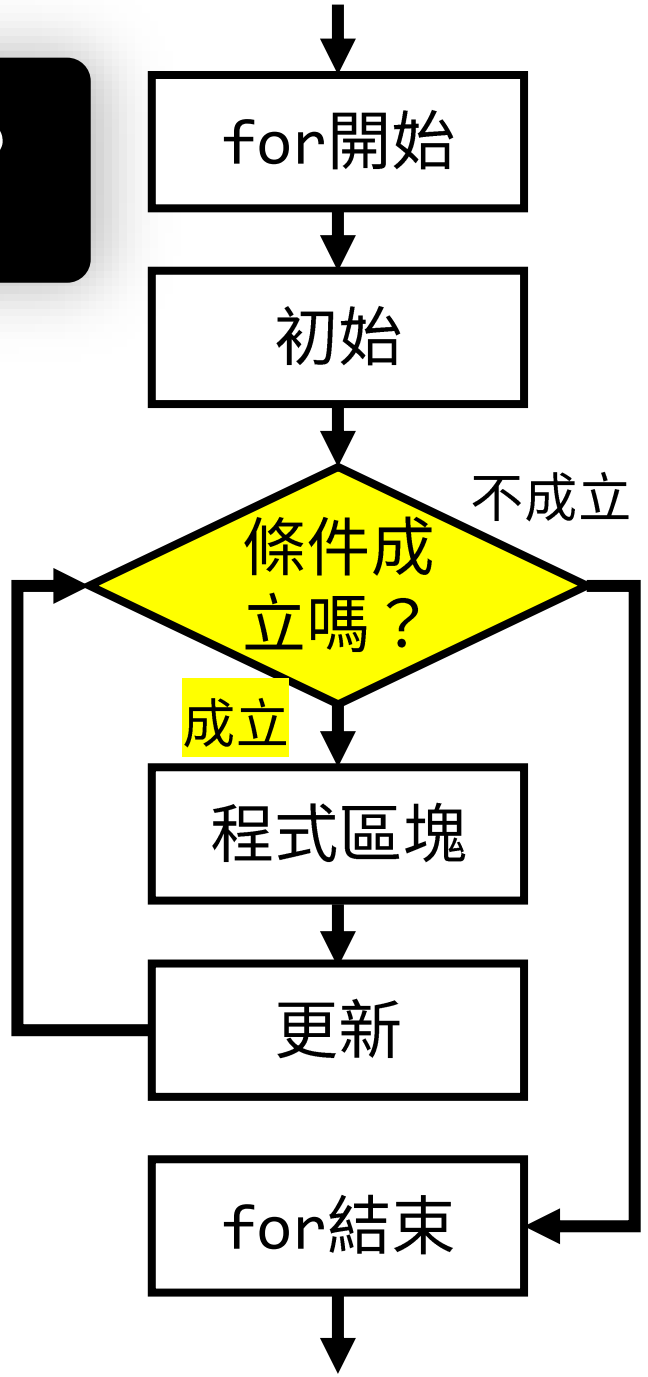
如果今天想要列出1~5間的整數？

```
1   #include <iostream>
2   using namespace std;
3
4   int main()
5   {
6       for (int i = 1; i <= 5; i++){
7           cout << i << endl;
8       }
9       return 0;
10  }
```

i = 4

i <= 5為真

輸出：
1
2
3



如果今天想要列出1~5間的整數？

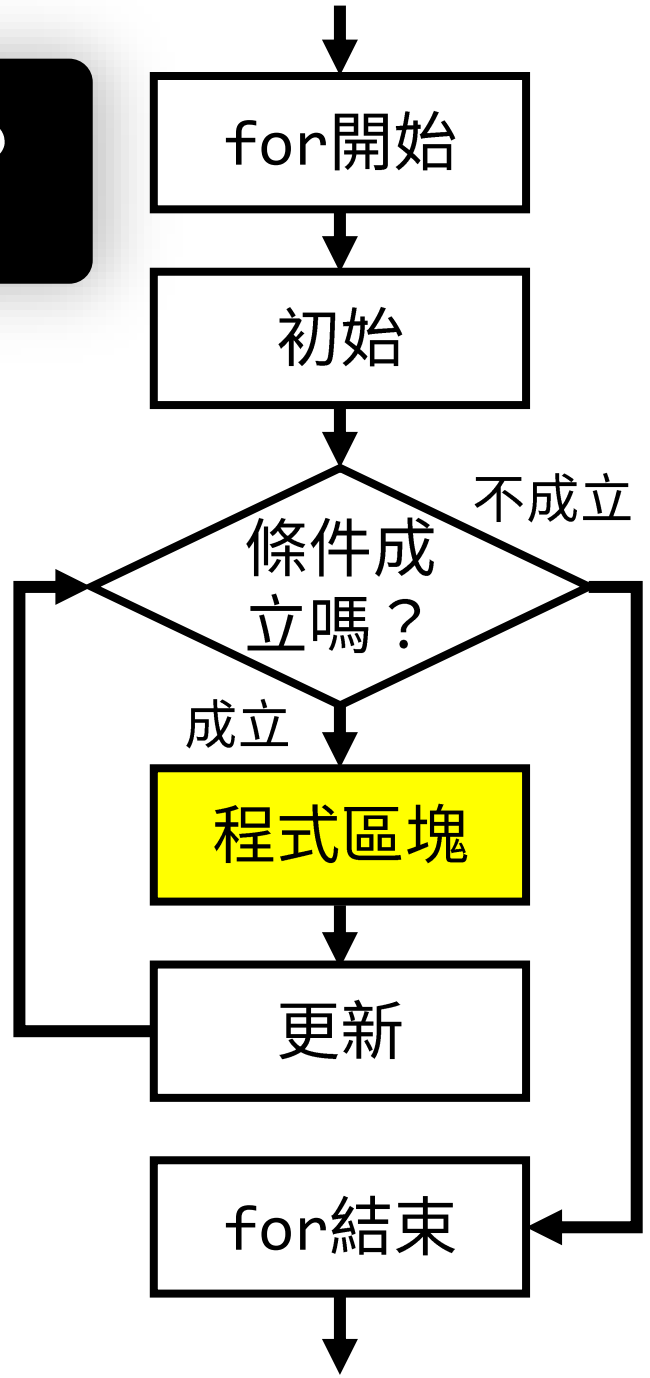
```
1   #include <iostream>
2   using namespace std;
3
4   int main()
5   {
6       for (int i = 1; i <= 5; i++){
7           cout << i << endl;
8       }
9       return 0;
10  }
```

i = 4

i <= 5為真

輸出i

```
輸出：
1
2
3
4
```



如果今天想要列出1~5間的整數？

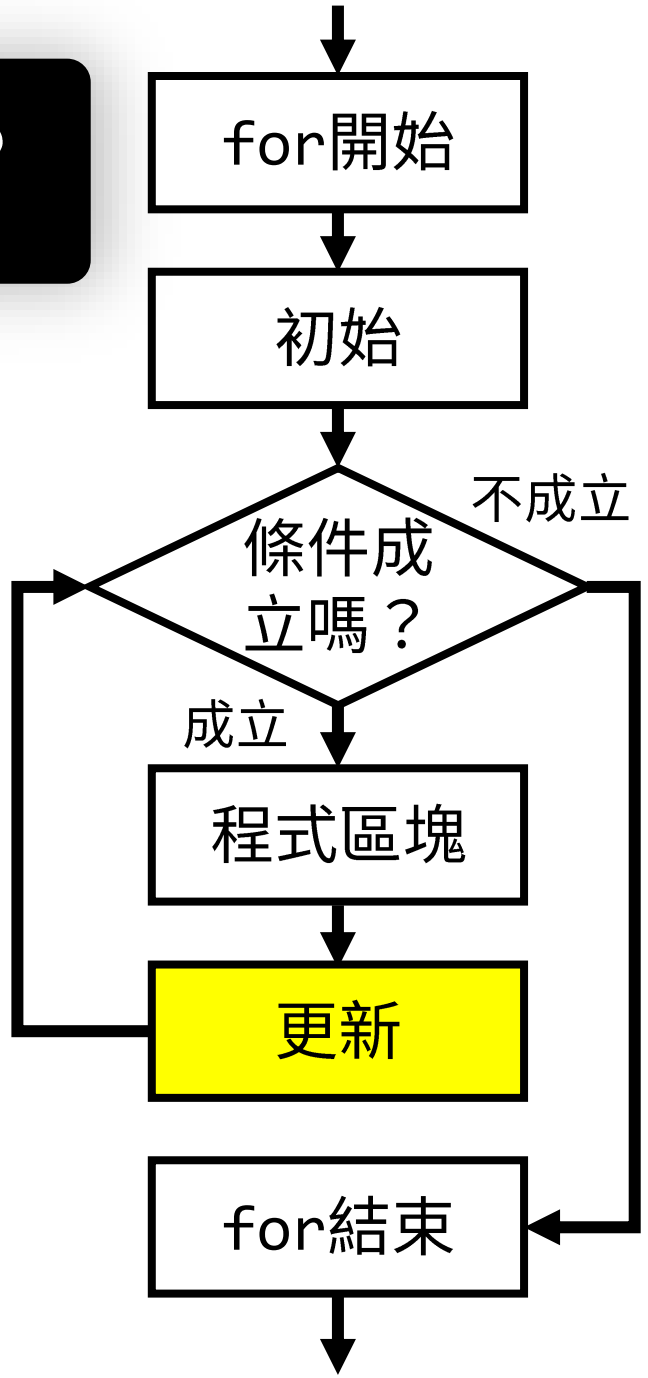
```
1   #include <iostream>
2   using namespace std;
3
4   int main()
5   {
6       for (int i = 1; i <= 5; i++){
7           cout << i << endl;
8       }
9       return 0;
10  }
```

i = 5

i <= 5為真

i = i + 1

輸出：
1
2
3
4

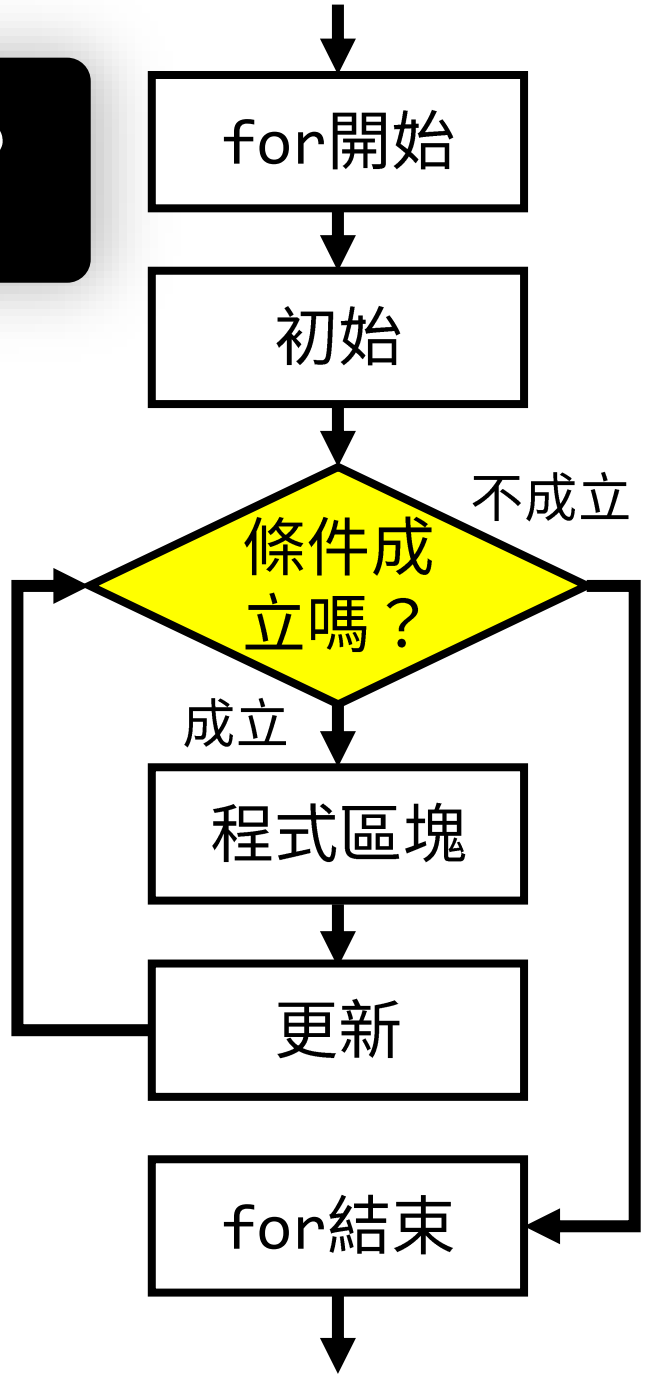


如果今天想要列出1~5間的整數？

```
1   #include <iostream>
2   using namespace std;
3
4   int main()
5   {
6       for (int i = 1; i <= 5; i++){
7           cout << i << endl;
8       }
9       return 0;
10  }
```

i = 5

```
輸出：
1
2
3
4
```



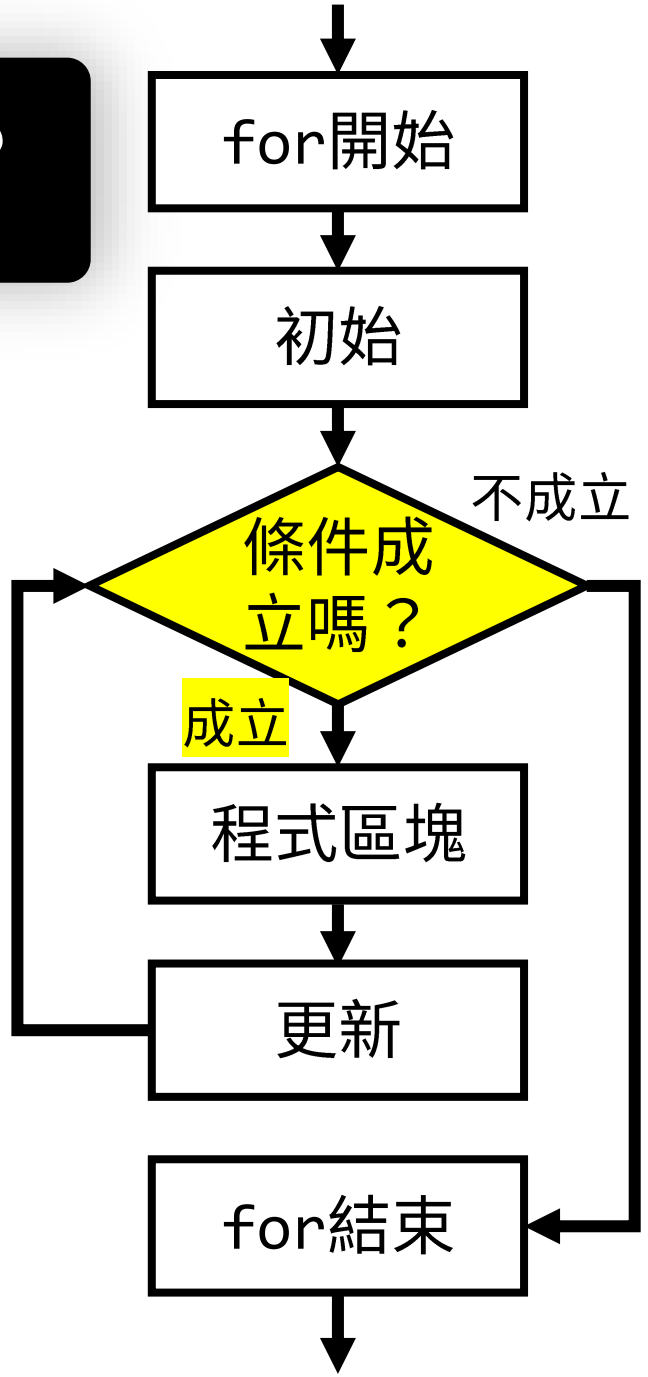
如果今天想要列出1~5間的整數？

```
1   #include <iostream>
2   using namespace std;
3
4   int main()
5   {
6       for (int i = 1; i <= 5; i++){
7           cout << i << endl;
8       }
9       return 0;
10  }
```

i = 5

i <= 5為真

輸出：
1
2
3
4



如果今天想要列出1~5間的整數？

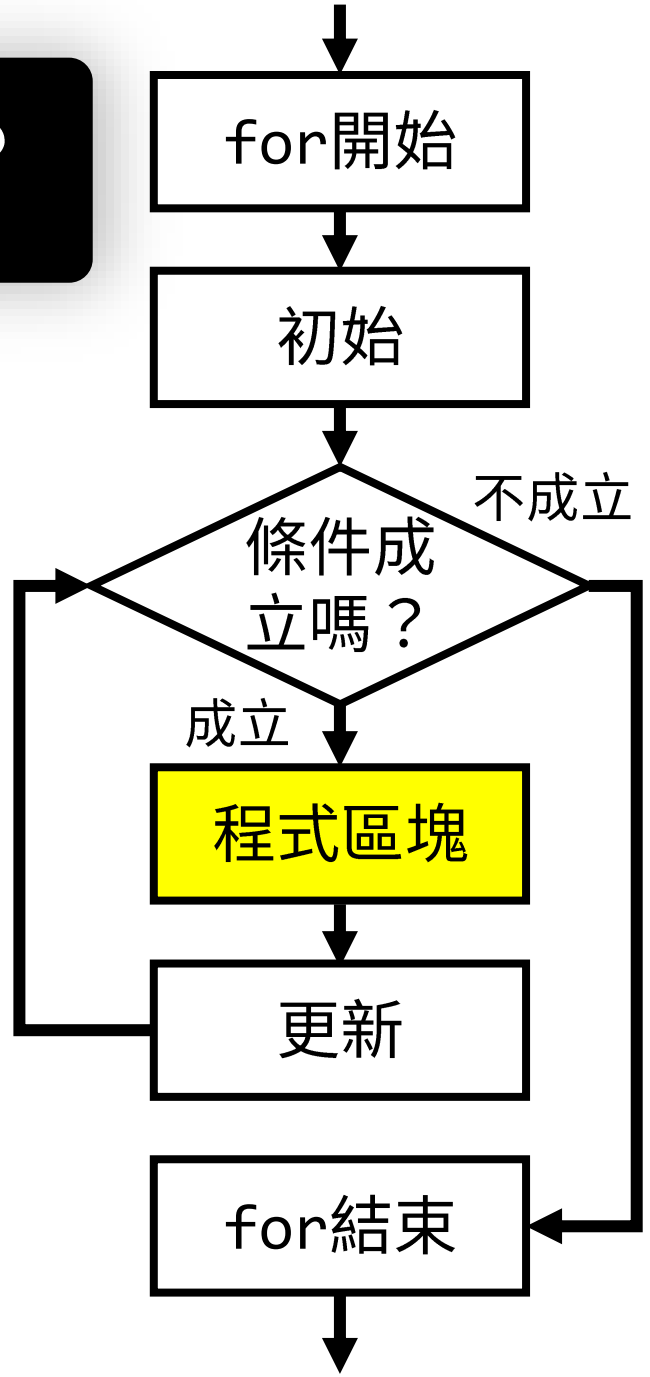
```
1   #include <iostream>
2   using namespace std;
3
4   int main()
5   {
6       for (int i = 1; i <= 5; i++){
7           cout << i << endl;
8       }
9       return 0;
10  }
```

i = 5

i <= 5為真

輸出i

```
輸出：
1
2
3
4
5
```



如果今天想要列出1~5間的整數？

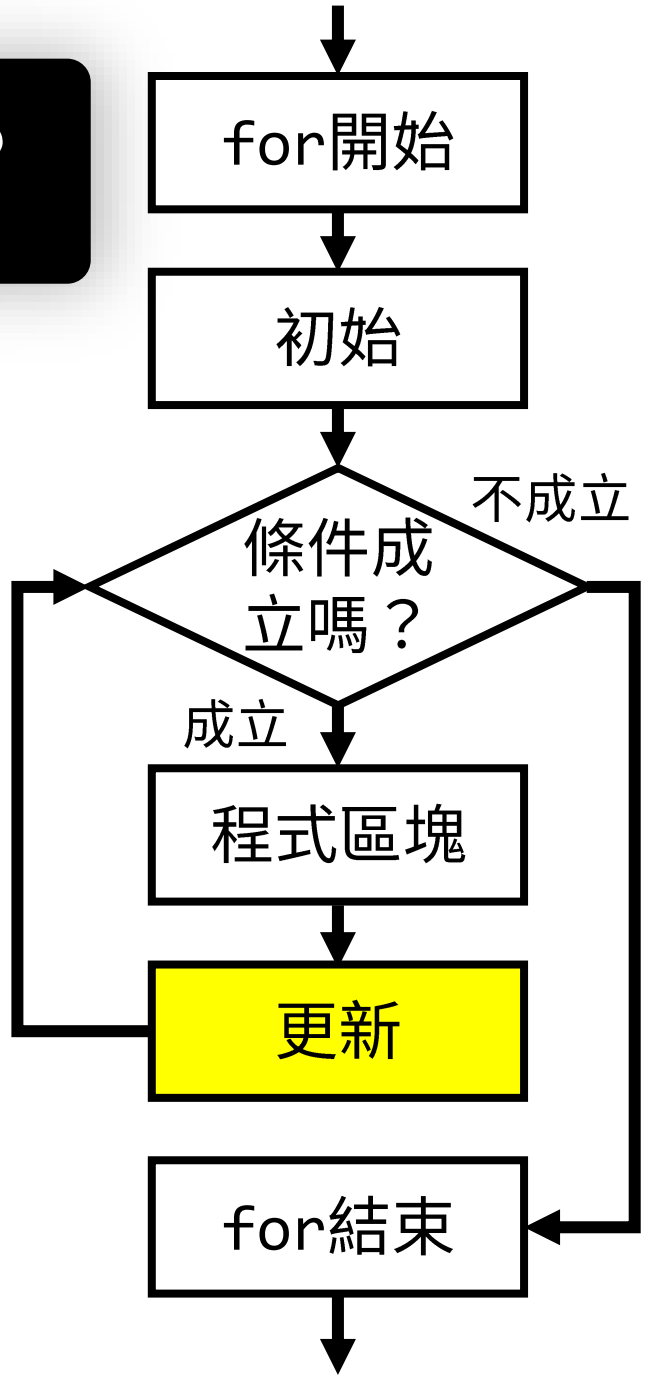
```
1   #include <iostream>
2   using namespace std;
3
4   int main()
5   {
6       for (int i = 1; i <= 5; i++){
7           cout << i << endl;
8       }
9       return 0;
10  }
```

i = 6

i <= 5為真

i = i + 1

輸出：
1
2
3
4
5

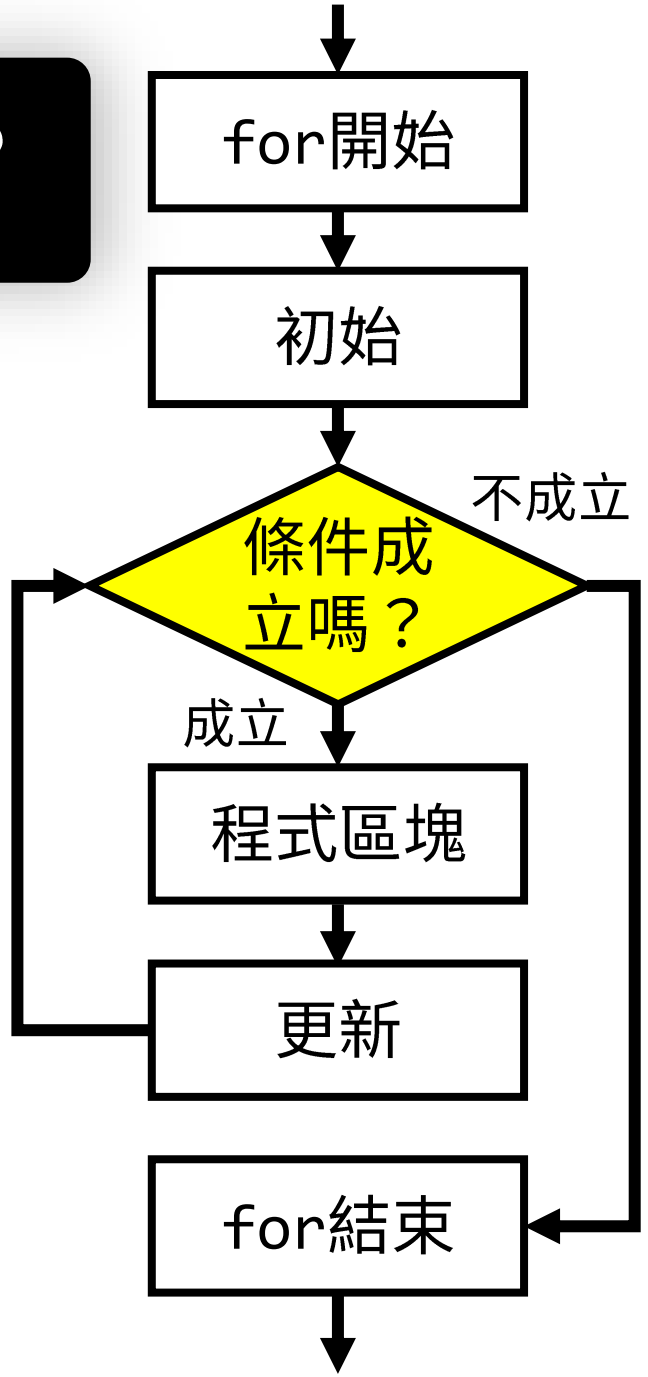


如果今天想要列出1~5間的整數？

```
1   #include <iostream>
2   using namespace std;
3
4   int main()
5   {
6       for (int i = 1; i <= 5; i++){
7           cout << i << endl;
8       }
9       return 0;
10  }
```

i = 6

輸出：
1
2
3
4
5



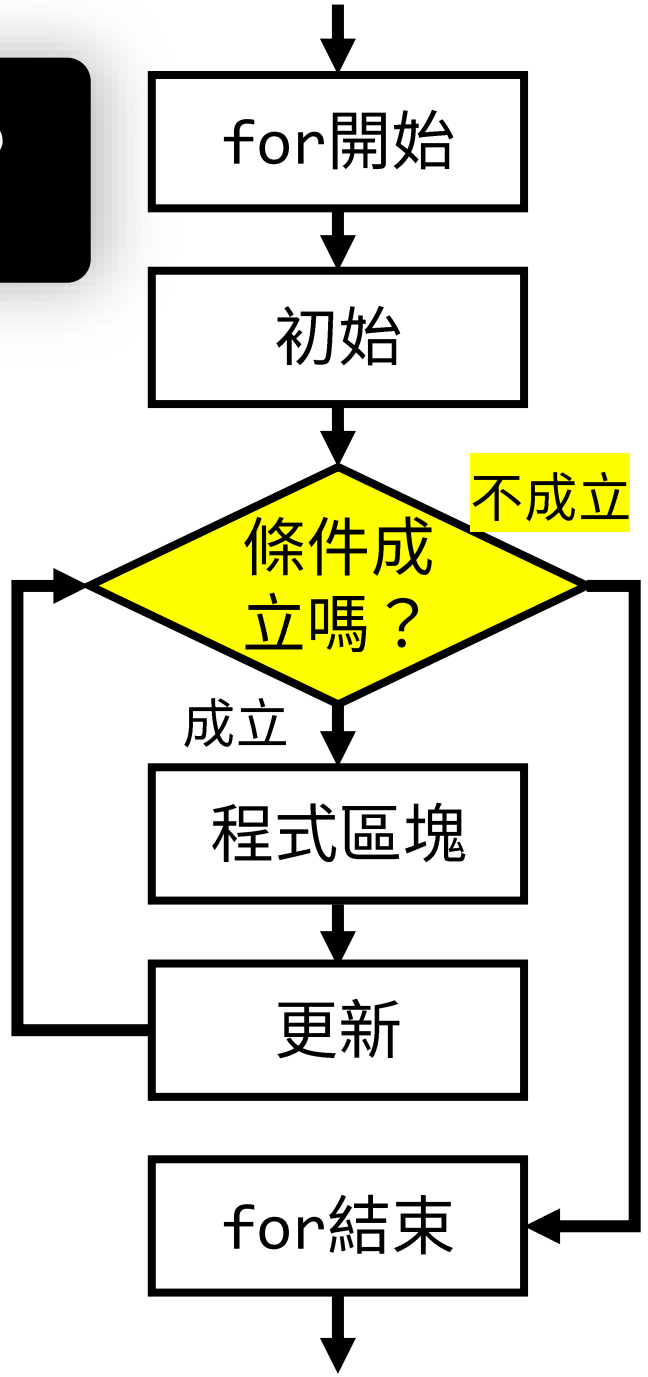
如果今天想要列出1~5間的整數？

```
1   #include <iostream>
2   using namespace std;
3
4   int main()
5   {
6       for (int i = 1; i <= 5; i++){
7           cout << i << endl;
8       }
9       return 0;
10  }
```

i = 6

i <= 5為非

輸出：
1
2
3
4
5

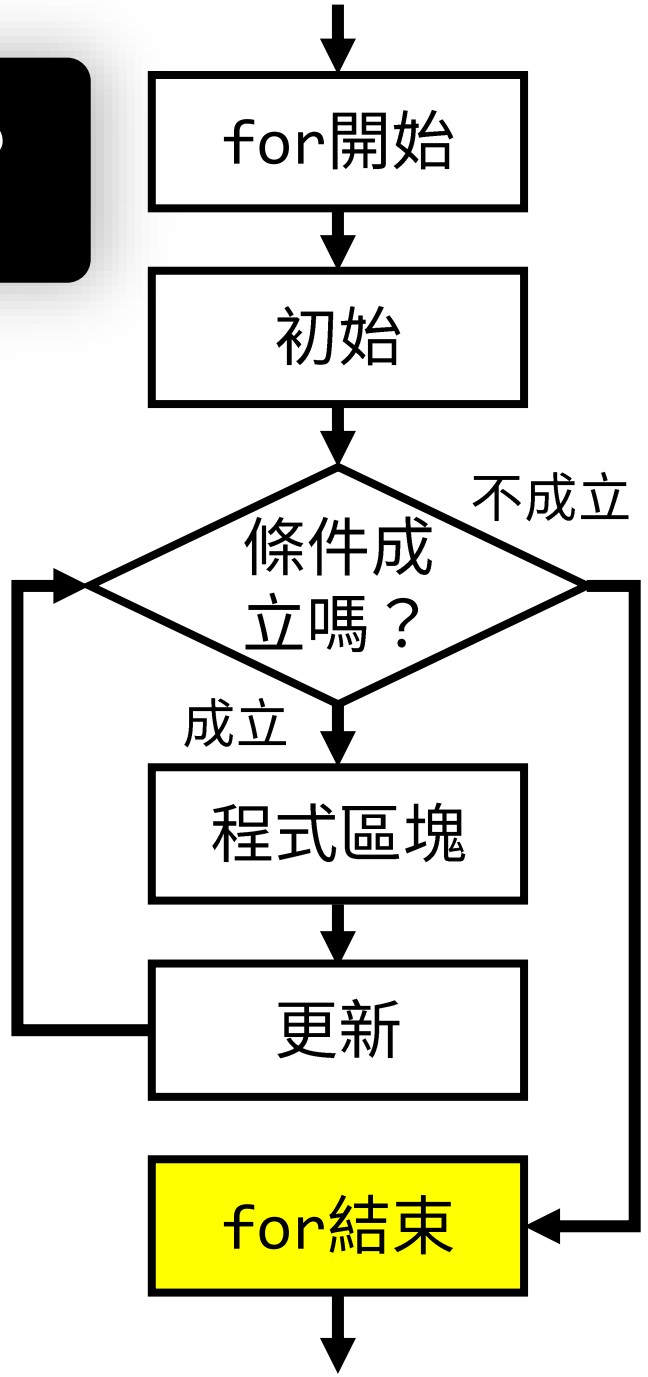


如果今天想要列出1~5間的整數？

```
1   #include <iostream>
2   using namespace std;
3
4   int main()
5   {
6       for (int i = 1; i <= 5; i++){
7           cout << i << endl;
8       }
9       return 0;
10  }
```

i = 6

輸出：
1
2
3
4
5



b970: 我不說髒話（繼）

內容

文文上次說髒話被老師罰在黑板上寫 50 遍「I don't say swear words!」，結果他只寫了 45 遍就跑出去玩了，以為老師不會發現。這次老師要求他罰寫的每一遍都要加上編號。

輸入說明

輸入只有一行，其中含有一個正整數 n，代表文文被罰寫的次數。

輸出說明

輸出 n 行「I don't say swear words!」，每行前面要有流水編號。請參考範例輸出。

範例輸入 #1

```
10
```

範例輸出 #1

```
1. I don't say swear words!
2. I don't say swear words!
3. I don't say swear words!
4. I don't say swear words!
5. I don't say swear words!
6. I don't say swear words!
7. I don't say swear words!
8. I don't say swear words!
9. I don't say swear words!
10. I don't say swear words!
```

b970: 我不說髒話（繼）

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

int main()
{
    int n;
    cin >> n;
    for (int i = 1; i <= n; i++){
        cout << i << ". I don't say swear words!\n";
    }

    return 0;
}
```

宣告整數n，紀錄被罰寫次數

讀入文文的被罰寫次數

利用for迴圈，讓整數i從1數到n

宣告整數i = 1

當 i <= n時結束迴圈

迴圈內程式執行完後i++

輸出 流水編碼 和 罰寫

a148: You Cannot Pass?!

內容

你考了 n 科筆試題目，每科的滿分都是 100 分。老師說，如果平均大於 59 你就過關了。

輸入說明

輸入第一行為一個數字 n，接著有 n 個正整數。

輸出說明

若你被當了，請輸出「yes」，否則輸出「no」。

範例輸入 #1

```
1 60
3 0 80 75
5 61 61 61 61 55
```

範例輸出 #1

```
no
yes
no
```

a148: You Cannot Pass?!

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

int main()
{
    int n, score, total;
    float avg;

    while (cin >> n){
        total = 0;
        for (int i = 0; i < n; i++){
            cin >> score;
            total += score;
        }

        avg = total * 1.0 / n;
        if (avg > 59){
            cout << "no\n";
        }
        else{
            cout << "yes\n";
        }
    }

    return 0;
}
```

宣告整數n、score、total，分別用來記錄共有多少個正整數、輸入的分數、總分

宣告浮點數avg，用於記錄平均分數

a148: You Cannot Pass?!

total -> 整數
total * 1.0 -> 浮點數
total * 1.0 / n -> 浮點數

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

int main()
{
    int n, score, total;
    float avg;

    while (cin >> n){
        total = 0;
        for (int i = 0; i < n; i++){
            cin >> score;
            total += score;
        }

        avg = total * 1.0 / n;
        if (avg > 59){
            cout << "no\n";
        }
        else{
            cout << "yes\n";
        }
    }

    return 0;
}
```

當有輸入一數字n時

總分歸零

利用for迴圈從0讀到n-1，共n次

讀入一分數

分數加入總分

判斷是否被當

算平均分數

b294: 經濟大恐荒

內容

西元 2505 年 1 月 1 日，發生了世界經濟大恐荒。從那天起，物價飛漲。第一天一個饅頭只要一元，第二天就要二元，第三天要賣三元，以此類推。

給你從第一天起文文每天所買的饅頭數，請問他總共花了多少錢？

輸入說明

輸入第一行有一個整數 n，代表文文從第一天起，連續買了 n 天的饅頭。

第二行會有 n 個整數，依序為第一天到第 n 天文文所買的饅頭數量。

輸出說明

輸出文文買饅頭所花的金額。

範例輸入 #1

```
5
1 2 3 4 5
```

範例輸出 #1

```
55
```

b294: 經濟大恐荒

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

int main()
{
    int n, cnt, total = 0;
    cin >> n;

    for (int i = 1; i <= n; i++){
        cin >> cnt;
        total += cnt * i;
    }

    cout << total;

    return 0;
}
```

宣告整數n, cnt, total，分別用於記錄買了幾天饅頭、第幾天買的饅頭數、所花的金額

讀入整數n

利用for迴圈，讓 i 從 1 數到 n，代表第i天的饅頭價格

宣告整數 i = 1

當 I <= n，結束迴圈

迴圈內程式執行完後i++

讀入第i天買的饅頭數

加總

計算第i天花了多少錢

輸出所花的金額

a024: 最大公因數(GCD)

內容

給定兩個整數，請求出它們的最大公因數

輸入說明

輸入包含兩個整數，以空白鍵隔開，兩個整數均 大於 0，小於 2^{31}

輸出說明

輸出兩個整數的最大公因數

範例輸入 #1

```
12 15
```

範例輸出 #1

```
3
```

範例輸入 #2

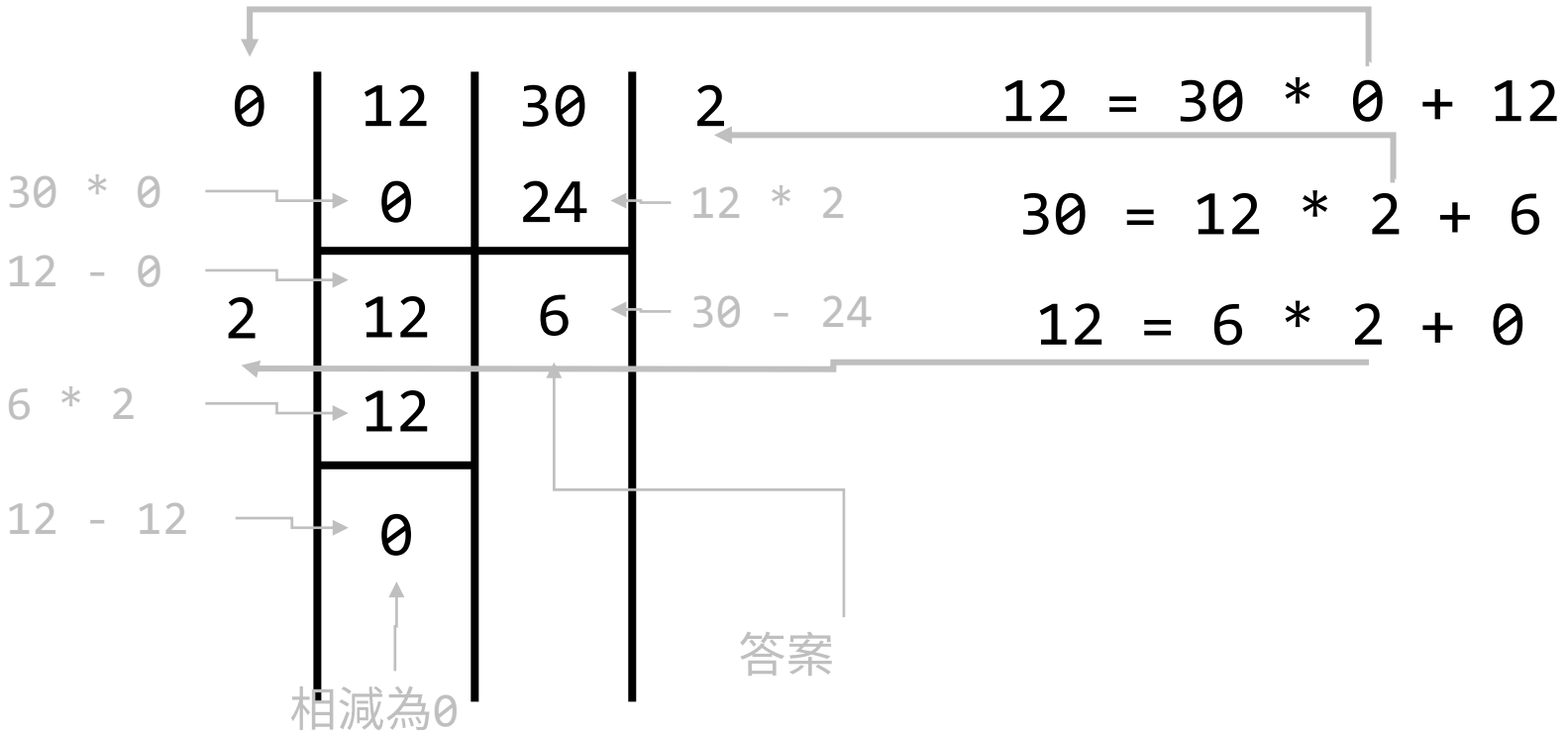
```
1 100
```

範例輸出 #2

```
1
```

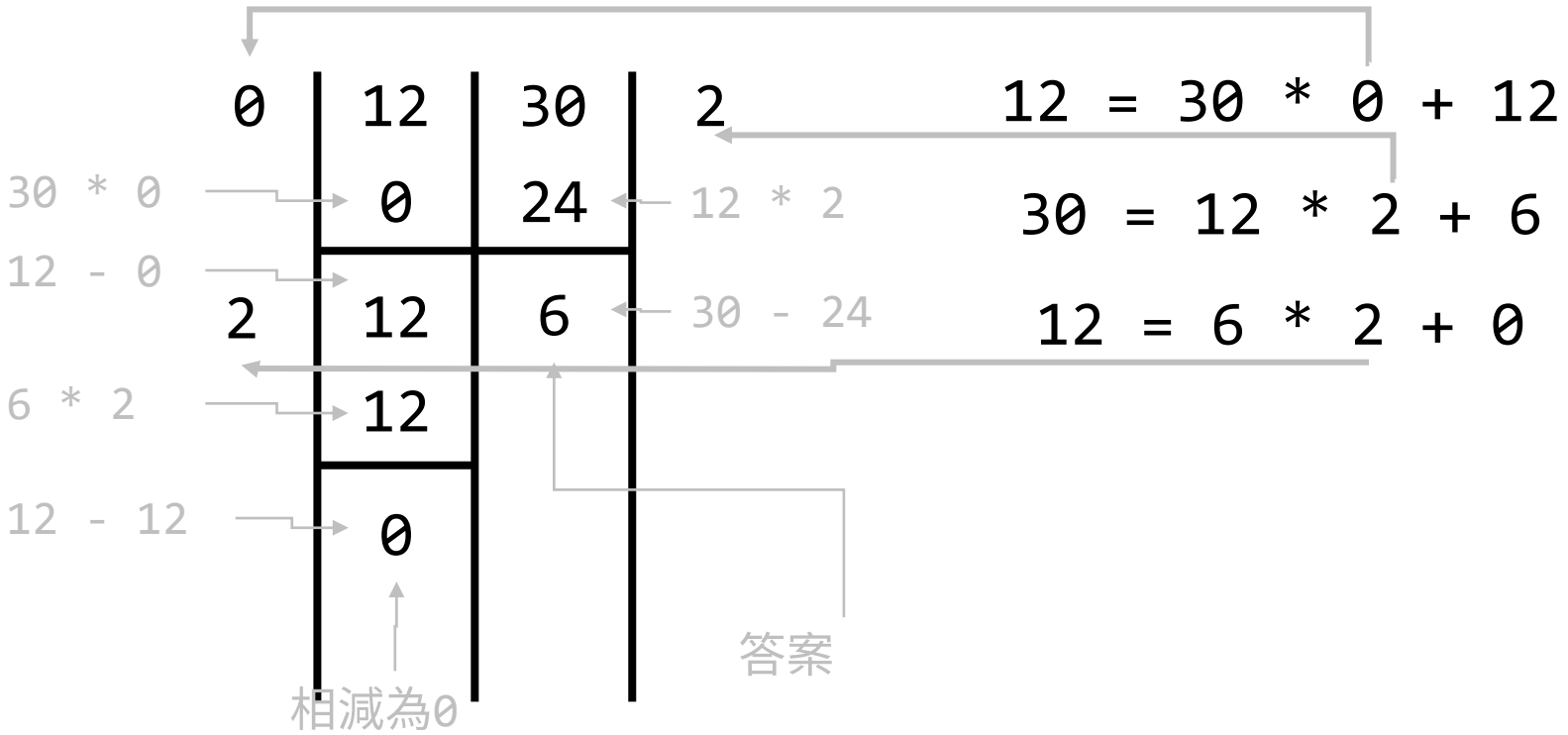
輾轉相除法

- gcd(12, 30) = ?



輾轉相除法

- gcd(12, 30) = 30



輾轉相除法

- 轉化成C++，可以這樣寫：

```
int a = 12, b = 30, c = 1;

while (c != 0){
    c = a % b;
    a = b;
    b = c;
}

cout << a;
```

輾轉相除法

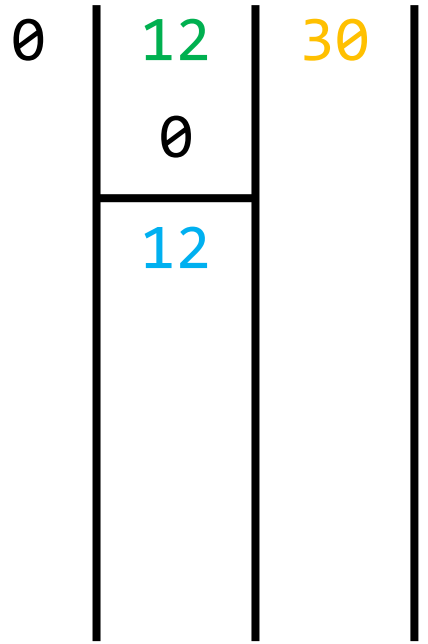
- 轉化成C++，可以這樣寫：

```
int a = 12, b = 30, c = 1;

while (c != 0){
    c = a % b;
    a = b;
    b = c;
}

cout << a;
```

a: 12

b: 30

c: 12



輾轉相除法

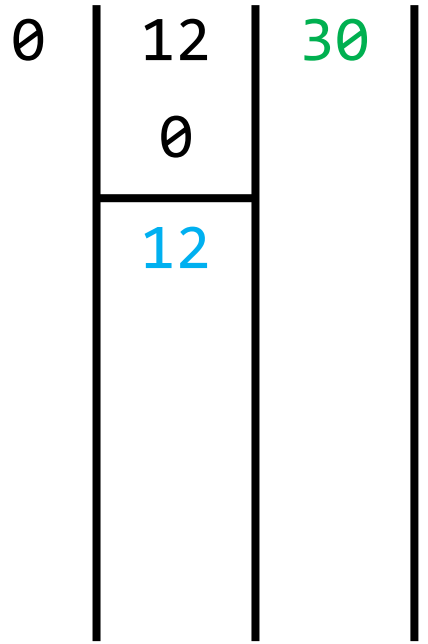
- 轉化成C++，可以這樣寫：

```
int a = 12, b = 30, c = 1;

while (c != 0){
    c = a % b;
    a = b;
    b = c;
}

cout << a;
```

a: 30

b: 30

c: 12



輾轉相除法

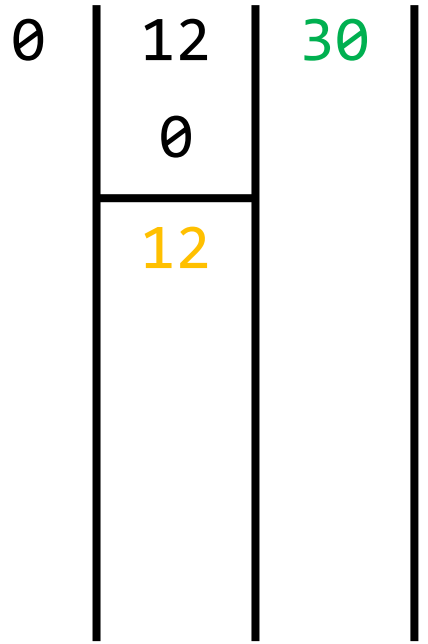
- 轉化成C++，可以這樣寫：

```
int a = 12, b = 30, c = 1;

while (c != 0){
    c = a % b;
    a = b;
    b = c;
}

cout << a;
```

a: 30

b: 12

c: 12



輾轉相除法

- 轉化成C++，可以這樣寫：

```
int a = 12, b = 30, c = 1;

while (c != 0){
    c = a % b;
    a = b;
    b = c;
}

cout << a;
```

a: 30

b: 12

c: 6

0	12	30	2
	0	24	
	12	6	

輾轉相除法

- 轉化成C++，可以這樣寫：

```
int a = 12, b = 30, c = 1;

while (c != 0){
    c = a % b;
    a = b;
    b = c;
}

cout << a;
```

a: 12

b: 12

c: 6

0	12	30	2
	0	24	
	12	6	

輾轉相除法

- 轉化成C++，可以這樣寫：

```
int a = 12, b = 30, c = 1;

while (c != 0){
    c = a % b;
    a = b;
    b = c;
}

cout << a;
```

a: 12

b: 6

c: 6

0	12	30	2
	0	24	
	12	6	

輾轉相除法

- 轉化成C++，可以這樣寫：

```
int a = 12, b = 30, c = 1;

while (c != 0){
    c = a % b;
    a = b;
    b = c;
}

cout << a;
```

a: 12

b: 6

c: 0

0	12	30	2
	0	24	
2	12	6	
	12		
	0		

輾轉相除法

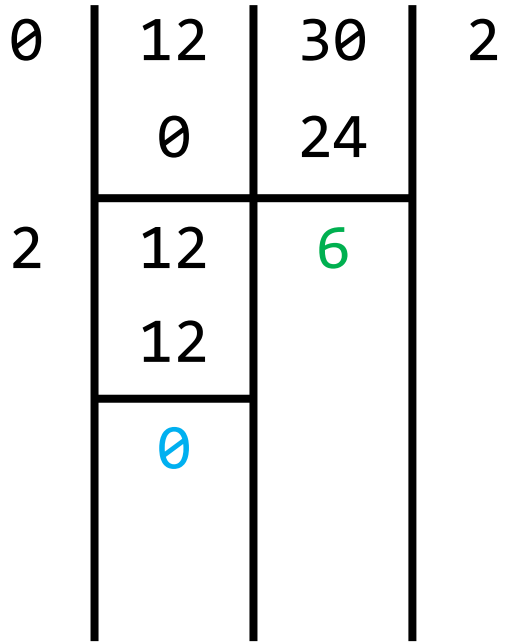
- 轉化成C++，可以這樣寫：

```
int a = 12, b = 30, c = 1;

while (c != 0){
    c = a % b;
    a = b;
    b = c;
}

cout << a;
```

a: 6

b: 6

c: 0



輾轉相除法

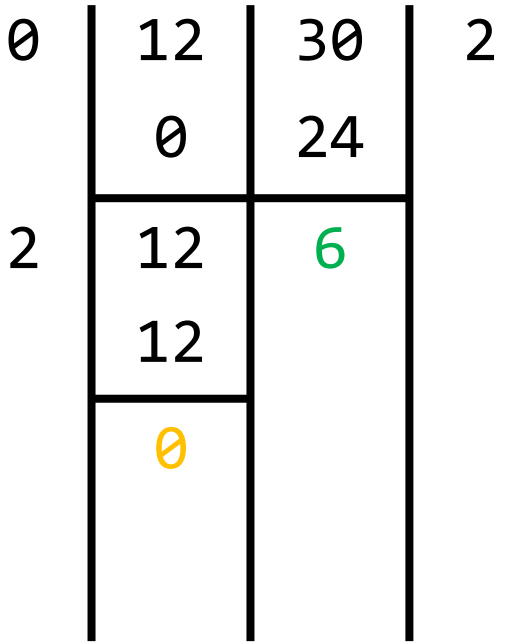
- 轉化成C++，可以這樣寫：

```
int a = 12, b = 30, c = 1;

while (c != 0){
    c = a % b;
    a = b;
    b = c;
}

cout << a;
```

a: 6

b: 0

c: 0



輾轉相除法

- 轉化成C++，可以這樣寫：

```
int a = 12, b = 30, c = 1;

while (c != 0){
    c = a % b;
    a = b;
    b = c;
}

cout << a;
```

a: 6

b: 0

c: 0

ANS: 6

0	12	30	2
	0	24	
2	12	6	
	12		
	0		

a024: 最大公因數(GCD)

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

int main()
{
    int a, b, c = 1;
    cin >> a >> b;

    while (c != 0){
        c = a % b;
        a = b;
        b = c;
    }

    cout << a;

    return 0;
}
```

a024: 最大公因數(GCD)

- 在C++14以下的版本，可使用：

__gcd(a, b) 來找 a 和 b 的最大公因數

*需要加入標頭檔 <algorithm>

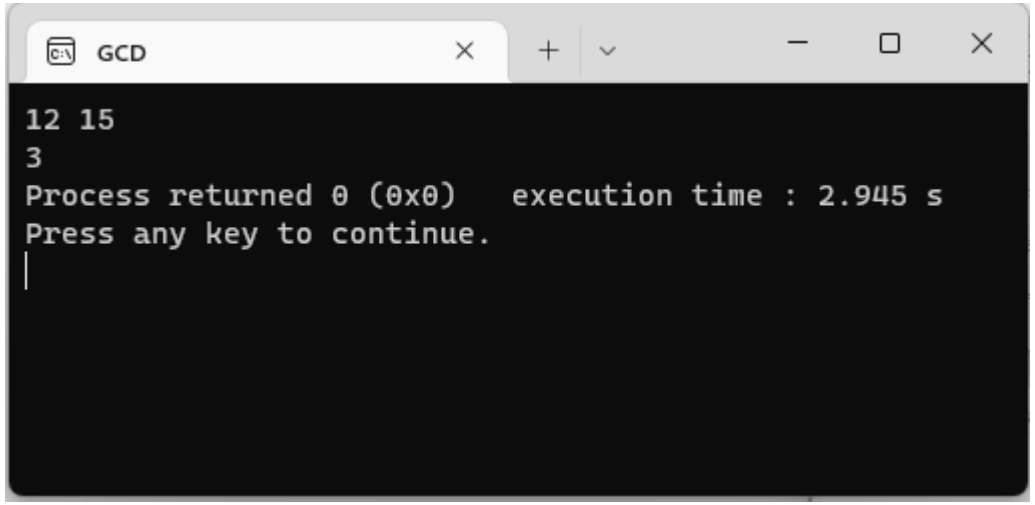
a024: 最大公因數(GCD)

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

int main()
{
    int a, b;
    cin >> a >> b;

    cout << __gcd(a, b);

    return 0;
}
```

使用前需
#include <algorithm>

algorithm N. [C] 演算法

a set of mathematical instructions or rules that, especially if given to a computer, will help to calculate an answer to a problem



a024: 最大公因數(GCD)

- 在C++17以上的版本，可使用：

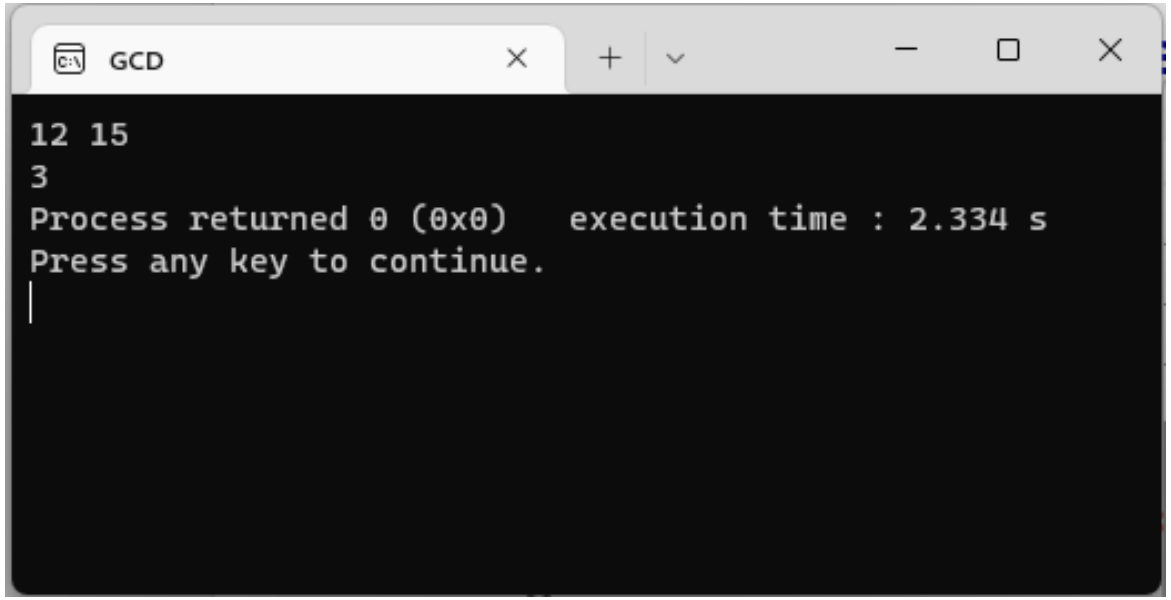
gcd(a, b) 來找 a 和 b 的最大公因數

*需要加入標頭檔 <numeric>

a024: 最大公因數(GCD)

```
1  #include <iostream>
2  #include <numeric>
3  using namespace std;
4
5  int main()
6  {
7      int a, b;
8      cin >> a >> b;
9
10     cout << gcd(a, b);
11
12     return 0;
13 }
```

使用前需
#include <numeric>



b538: 分數運算-2

內容

上次老師的教甄題後，就想說出個分數的加、減、乘、除，也許有人出過類似題，但還是想出這題為下一題準備

輸入說明

每組測資有多列以EOF結束，每列四個整數 -9999 <=a,b,c,d <=9999 一個字元{ + - * / }以空白隔開
代表兩個分數 a/b @ c/d ，其中@為加減乘除之一。{b,d不為0，若為除法運算則c亦不為0}

輸出說明

對輸入的每一列，輸出1個 分數的運算結果且為最簡分數 p/q ，若 p被q整除，則只顯示一個整除後的整數。

範例輸入 #1

```
-1 2 4 -3 +
1 1 1 1 -
1 1 1 2 +
2 3 1 2 *
2 3 2 3 /
```

範例輸出 #1

```
-11/6
0
3/2
1/3
1
```

b538: 分數運算-2

```cpp
#include <iostream>
#include <algorithm>
#include <cmath>
using namespace std;

int main()
{
    int a, b, c, d, p, q, GCD, LCM;
    char calc;

    while (cin >> a >> b >> c >> d >> calc){
        if (calc == '/'){
            swap(c, d);
            calc = '*';
        }
        if (calc == '*'){
            p = a * c;
            q = b * d;
        }
        else{
            if (b < 0){
                b *= -1;
                a *= -1;
            }
            if (d < 0){
                d *= -1;
                c *= -1;
            }

            if (calc == '-'){
                c *= -1;
                calc = '+';
            }
            if (calc == '+'){
                LCM = b * d / __gcd(b, d);
                a *= LCM / b;
                c *= LCM / d;
                p = a + c;
                q = LCM;
            }
        }

        GCD = __gcd(abs(p), abs(q));

        p /= GCD;
        q /= GCD;

        if (q < 0){
            q *= -1;
            p *= -1;
        }

        if (p % q == 0){
            cout << p / q << endl;
        }
        else{
            cout << p << '/' << q << endl;
        }
    }

    return 0;
}
```

b538: 分數運算-2

```cpp
#include <iostream>
#include <algorithm>
#include <cmath>
using namespace std;

int main()
{
    int a, b, c, d, p, q, GCD, LCM;
    char calc;

    while (cin >> a >> b >> c >> d >> calc){
        if (calc == '/'){
            swap(c, d);
            calc = '*';
        }
        if (calc == '*'){
            p = a * c;
            q = b * d;
        }
        else{
            if (b < 0){
                b *= -1;
                a *= -1;
            }
            if (d < 0){
                d *= -1;
                c *= -1;
            }
            if (calc == '-'){
                c *= -1;
                calc = '+';
            }
            if (calc == '+'){
                LCM = b * d / __gcd(b, d);
                a *= LCM / b;
                c *= LCM / d;
                p = a + c;
                q = LCM;
            }
        }
        GCD = __gcd(abs(p), abs(q));
        p /= GCD;
        q /= GCD;
        if (q < 0){
            q *= -1;
            p *= -1;
        }
        if (p % q == 0){
            cout << p / q << endl;
        }
        else{
            cout << p << '/' << q << endl;
        }
    }

    return 0;
}
```

變數宣告

輸入

把除法當乘法做

乘法計算

使分子為負，而非分母為負

把減法當加法做

加法計算

加法計算

化簡

使分母恆正

輸出

輸出