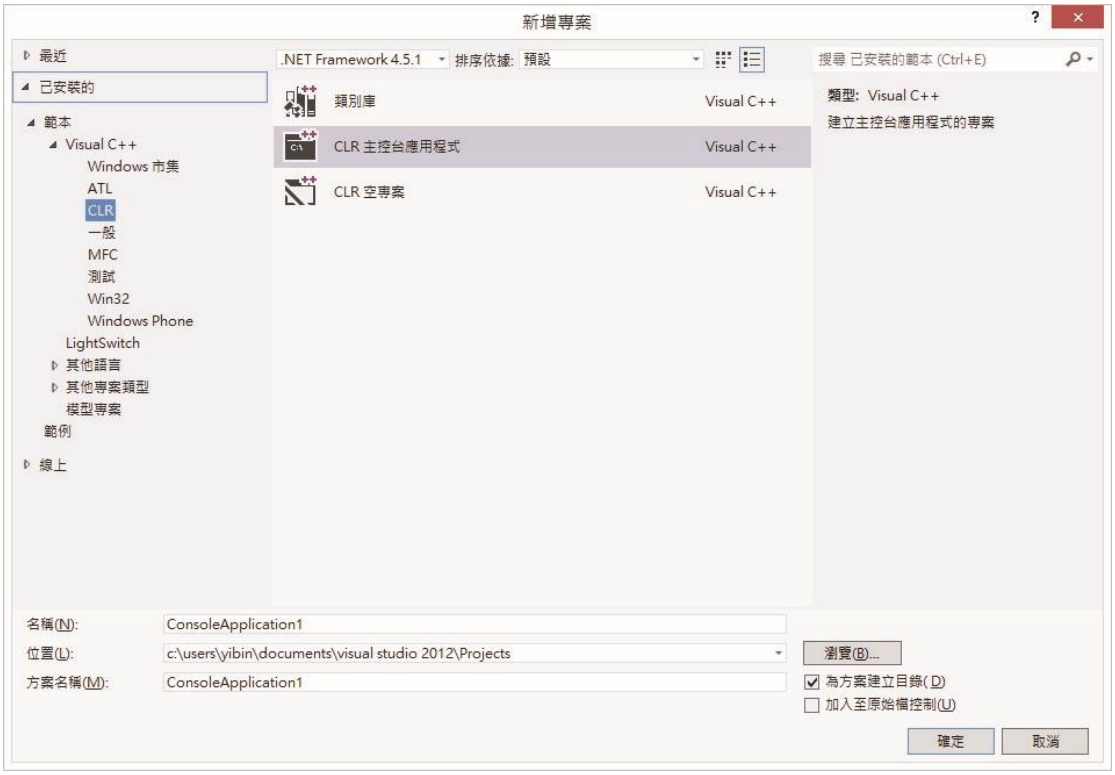
VS DEBUG 教學

1. 設中斷點(breakpoint)

- 逐步執行(F11)
- 跳過函式(F10)
- 繼續執行(F5)
- 執行到游標處(ctrl + F10)

範例 1: 計算 BMI 新增一個新專案

選擇 CLR 主控台應用程式

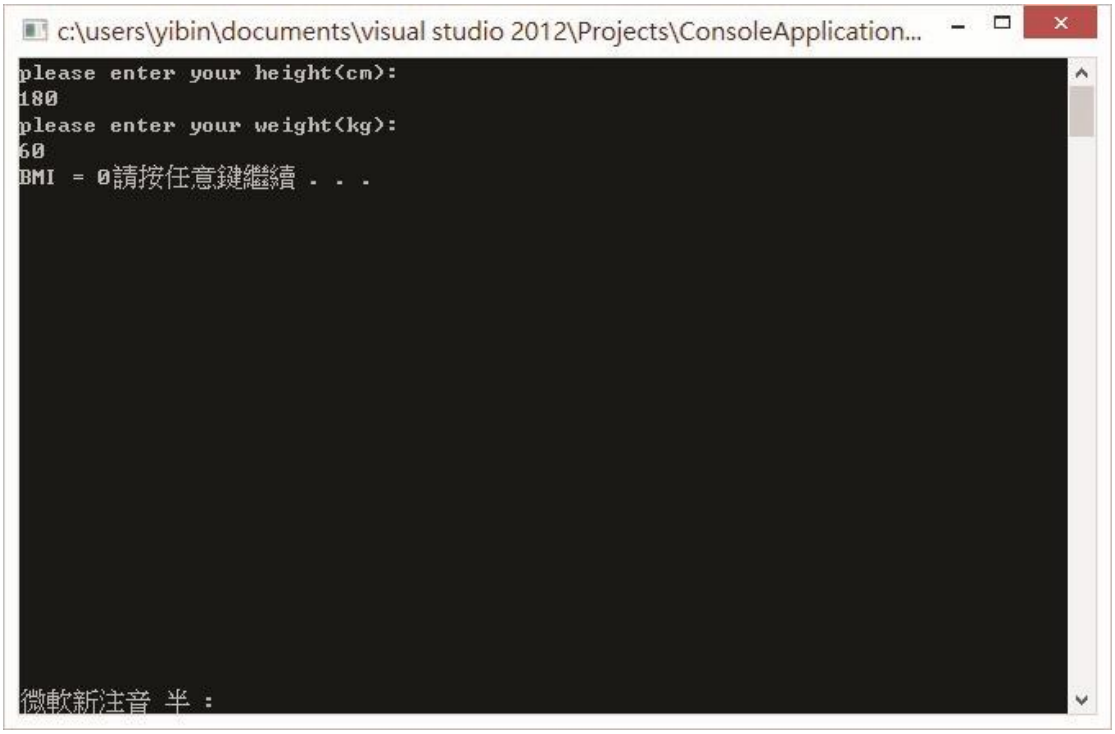


貼上 Sample code 並執行

```
#include "stdafx.h"
#include "stdafx.h"
#include <iostream>

using namespace std;
void InputWeight(double weight)
{
	cout << "please enter your weight(kg): " << endl;
	cin >> weight;
}
double GetBMI(double height, double weight)
{
	double bmi;
	bmi = weight / pow(height/100, 2);
	return bmi;
}
int main(array<System::String ^> ^args)
{
```

```
    double height = 0;
     double weight = 0;
     cout << "please enter your height(cm): " << endl;
     cin >> height;       InputWeight(weight);
     cout << "BMI = ";
     cout << GetBMI(height, weight);
     system("pause");     return 0;
}
```

執行結果發現 BMI 為 0



開始 DEBUG

- 插入中斷點在要中斷的的位置連點兩下

ConsoleApplication1.cpp

(全域範圍)

```
#include "stdafx.h"
#include "stdafx.h"
#include <iostream>

using namespace std;
void InputWeight(double weight)
{
    cout << "please enter your weight(kg): " << endl;
    cin >> weight;
}

double GetBMI(double height, double weight)
{
    double bmi;
    bmi = weight / pow(height/100, 2);
    return bmi;
}

int main(array<System::String ^> ^args)
{
    double height = 0;
    double weight = 0;
    cout << "please enter your height(cm): " << endl;
    cin >> height;
    InputWeight(weight);
    cout << "BMI = ";
    cout << GetBMI(height, weight);
    system("pause");
    return 0;
}
```

- 逐步執行(F11)

按下 F11 黃色箭頭為目前執行到的位置

ConsoleApplication1.cpp

(全域範圍)

```
#include "stdafx.h"
#include "stdafx.h"
#include <iostream>

using namespace std;
void InputWeight(double weight)
{
    cout << "please enter your weight(kg): " << endl;
    cin >> weight;
}

double GetBMI(double height, double weight)
{
    double bmi;
    bmi = weight / pow(height/100, 2);
    return bmi;
}

int main(array<System::String ^> ^args)
{
    double height = 0;
    double weight = 0;
    cout << "please enter your height(cm): " << endl;
    cin >> height;
    InputWeight(weight);
    cout << "BMI = ";
    cout << GetBMI(height, weight);
    system("pause");
    return 0;
}
```

- 跳過函式(F10)

按下 F10 可直接跳過函式, 並可在下方的區域變數視窗查看目前變數數值

ConsoleApplication1.cpp

(全域範圍)

```cpp
#include "stdafx.h"
#include "stdafx.h"
#include <iostream>

using namespace std;
void InputWeight(double weight)
{
    cout << "please enter your weight(kg): " << endl;
    cin >> weight;
}

double GetBMI(double height, double weight)
{
    double bmi;
    bmi = weight / pow(height/100, 2);
    return bmi;
}

int main(array<System::String ^> ^args)
{
    double height = 0;
    double weight = 0;
    cout << "please enter your height(cm): " << endl;
    cin >> height;
    InputWeight(weight);
    cout << "BMI = ";
    cout << GetBMI(height, weight);
    system("pause");
    return 0;
}
```

區域變數

名稱	值	型別
args	{Length=0}	array<Sy
height	180.00000000000000	double
weight	0.00000000000000000	double

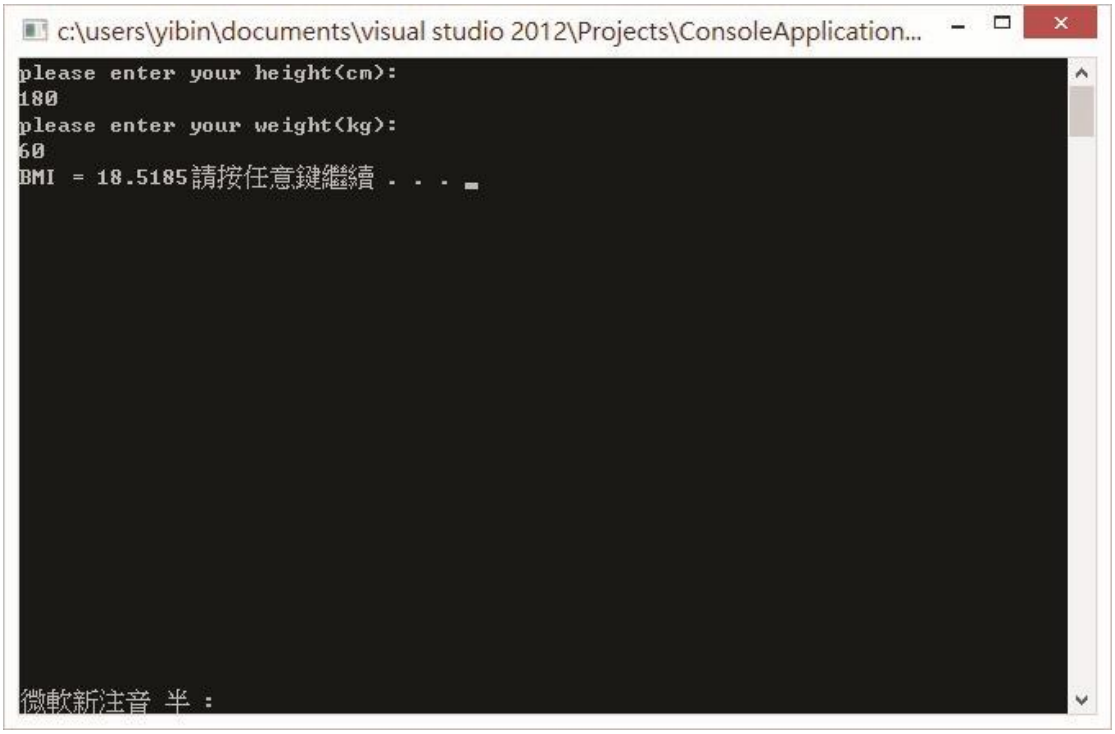
區域變數 監看式 1

就緒

發現 InputWeight 函式執行後，weight 沒有改變，函式應該傳址。

```
void InputWeight(double* weight)
{
     cout << "please enter your weight(kg): " << endl;
     cin >> *weight;
}
double GetBMI(double height, double weight)
{
     double bmi;
     bmi = weight / pow(height/100, 2);
     return bmi;
}
int main(array<System::String ^> ^args)
{
    double height = 0;
     double weight = 0;
     cout << "please enter your height(cm): " << endl;
     cin >> height;       InputWeight(&weight);
     cout << "BMI = ";
     cout << GetBMI(height, weight);
     system("pause");     return 0;
}
```

重新執行後結果正確



2. 設中斷點停止條件

- 條件成立(Is true) ex: x==100
- 變數改變(Has changed)

範例 2: 泡沫排序法

和範例一相同先建立一個 CLR 主控台應用程式

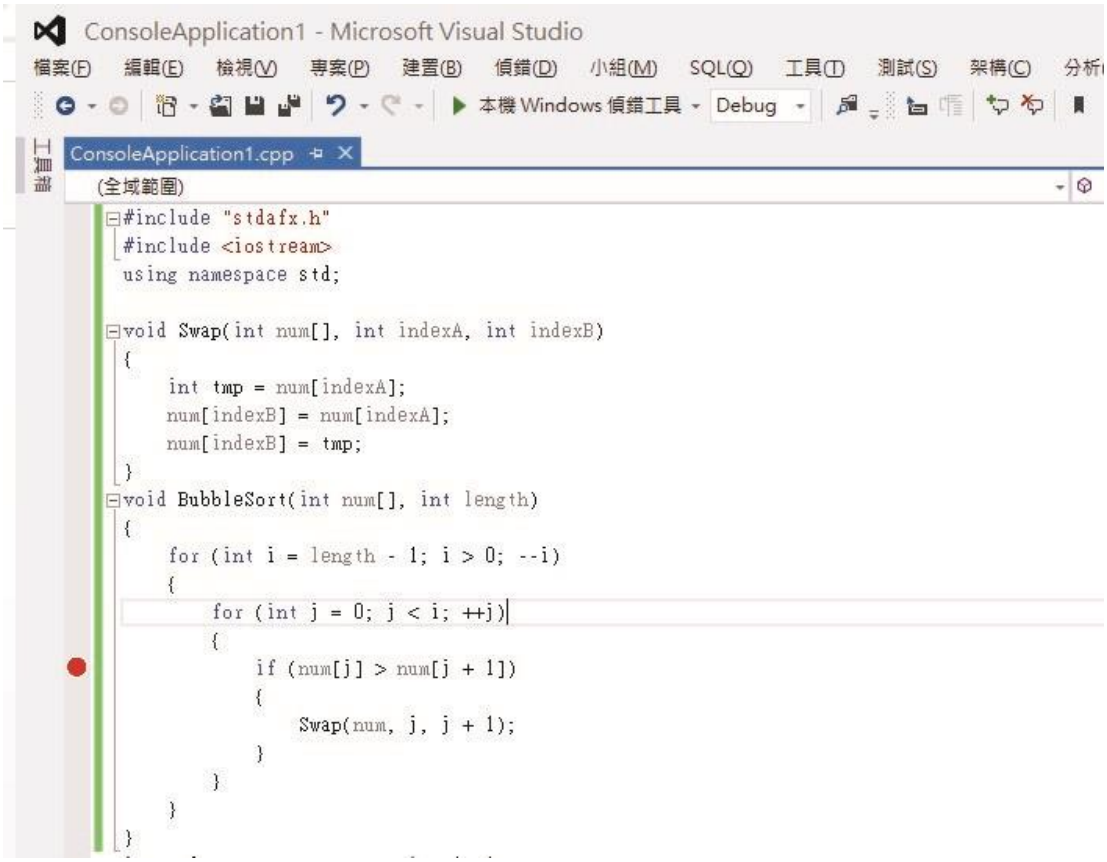
貼上 Sample code

```
#include "stdafx.h"
#include <iostream> using
namespace std;

void Swap(int num[], int indexA, int indexB)
{
    int tmp = num[indexA];
num[indexB] = num[indexA];
num[indexB] = tmp;
}
void BubbleSort(int num[], int length)
{
    for (int i = length - 1; i > 0; --i)
```

```
    {
        for (int j = 0; j < i; ++j)
          {
            if (num[j] > num[j + 1])
{
                Swap(num, j, j + 1);
            }
          }
    } }
int main(array<System::String ^> ^args)
{
   int num[5];       int length =
5;   num[0] = 12;    num[1] = 42;
     num[2] = 23;    num[3] = 51;
     num[4] = 8;     cout <<
"Array: ";      for(int i =0;
i<length; i++)
     {
          cout << num[i] << " ";
     }
     cout << endl;
     BubbleSort(num, length);
     cout << "Bubble Sort: ";
     for(int i =0; i<length; i++)
     {
          cout << num[i] << " ";
     }
     system("pause");
     return 0;
}
```

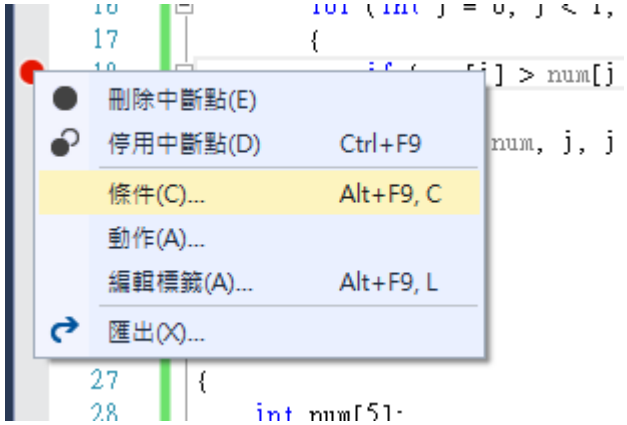
執行結果

發現結果不如預期, 發生數字重複問題設中斷點在
BubbleSort Function 內



但是若將中斷點設在多重迴圈內會發生多次中斷的問題, 需要更有效率的中斷條件

- 中斷點停止條件在中斷點上點選右鍵設定中斷條件

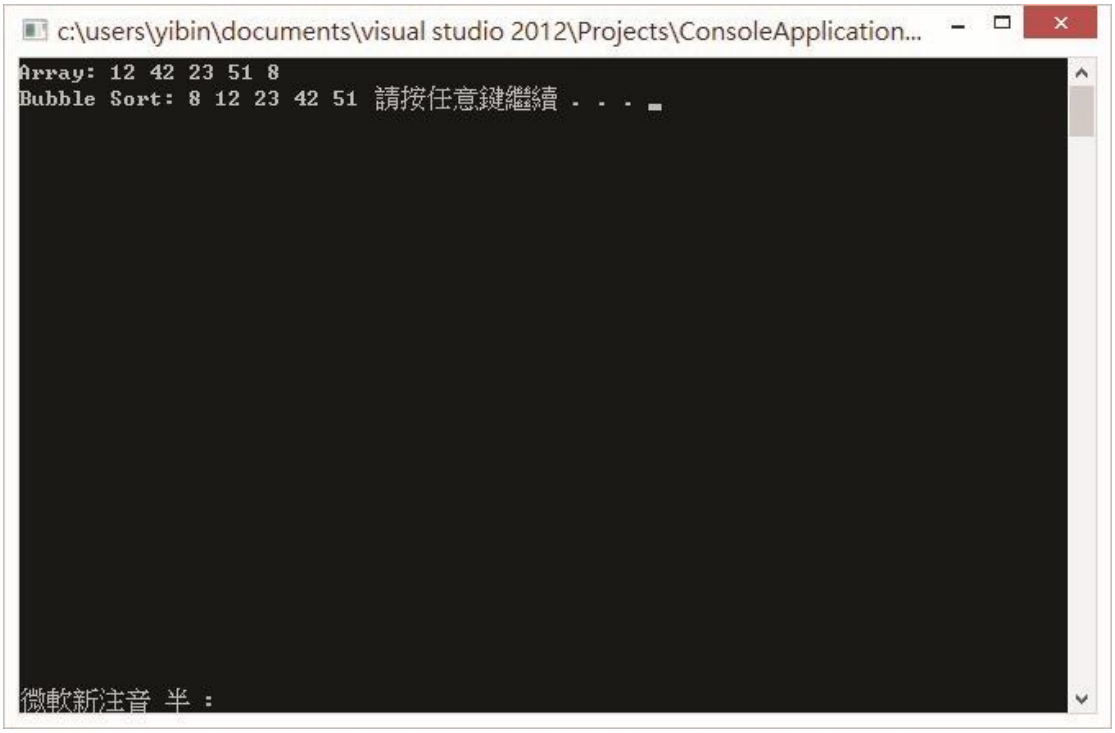


將條件設為 num[j]為 42 時才停止

當執行到 Swap 函式時發現 Swap 函式有錯誤應修改為

```
void Swap(int num[], int indexA, int indexB)
{
    int tmp = num[indexA];
num[indexA] = num[indexB];
num[indexB] = tmp;
}
```

重新執行結果正確



3. 設中斷點執行次數停止

- Hit Count ex: loop 第 10 次範例

3: 費伯納西遞迴式

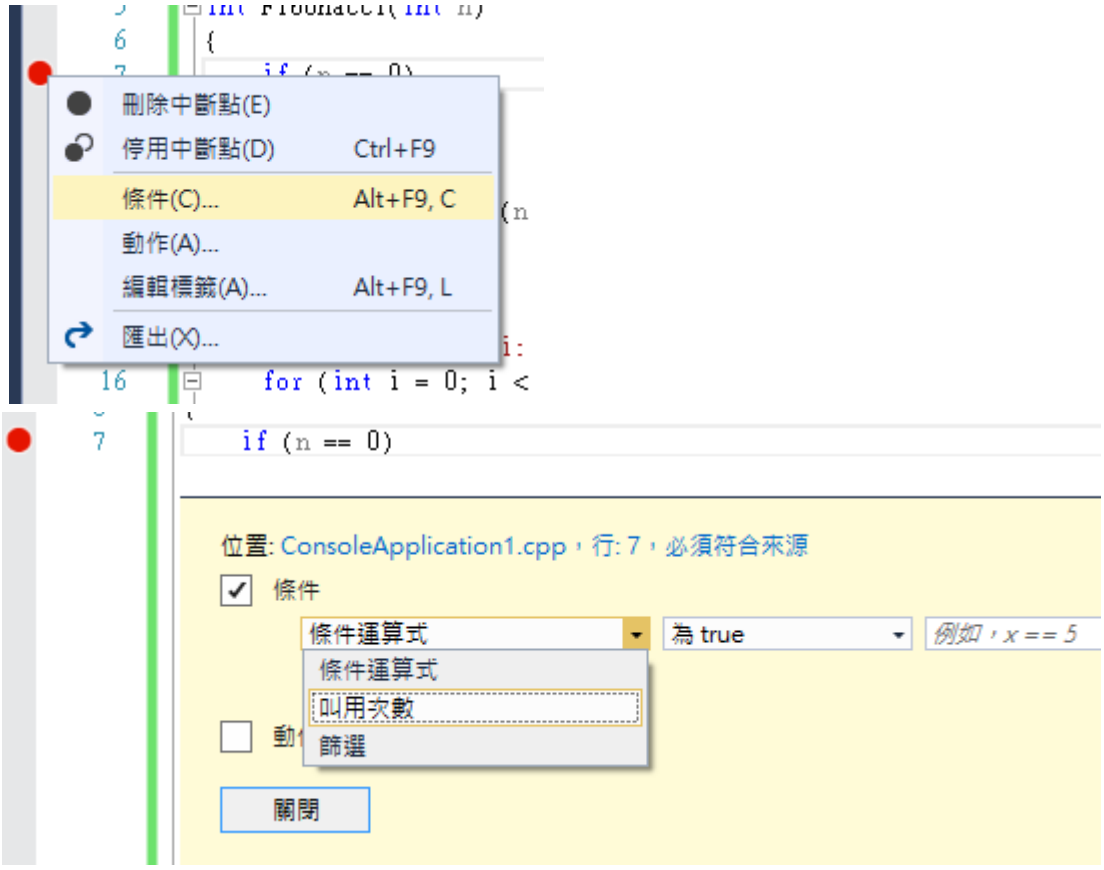
建立新專案, 貼上 Sample code

```
#include "stdafx.h"
#include <iostream> using
namespace std;


int Fibonacci(int n)
{
    if(n==0)
return 0;
if(n==1)
return 1;
    return (Fibonacci(n-1)+Fibonacci(n-2));


}
int main(array<System::String ^> ^args)
```

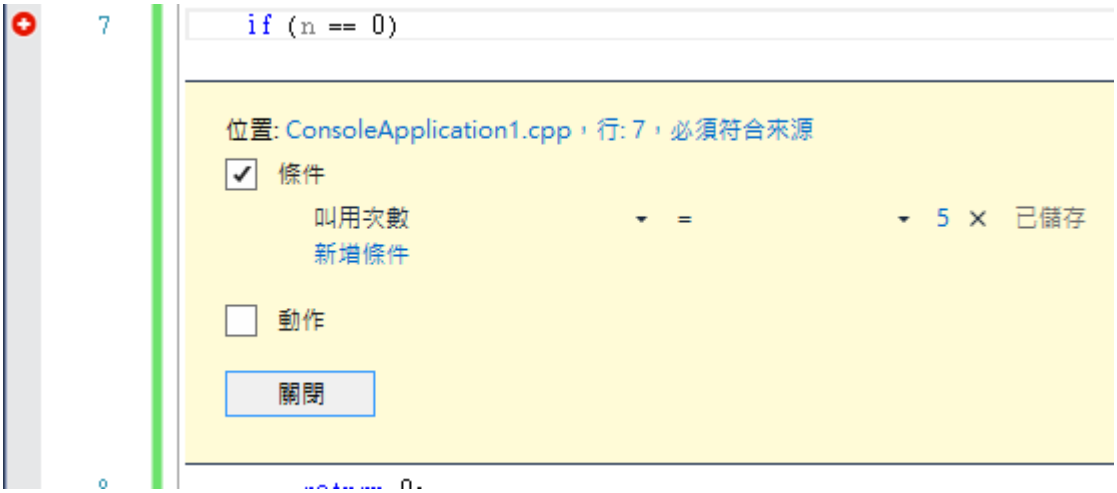
```
{
	cout << "Fibonacci: ";
	for(int i = 0; i<10; i++)
	{
		cout << Fibonacci(i) << " ";
	}
	system("pause");
	return 0;
}
```

● 叫用次數當遇到像遞迴式或是多重迴圈時, 除了範例 2 中的中斷點停止條件也可以設定叫用次數。

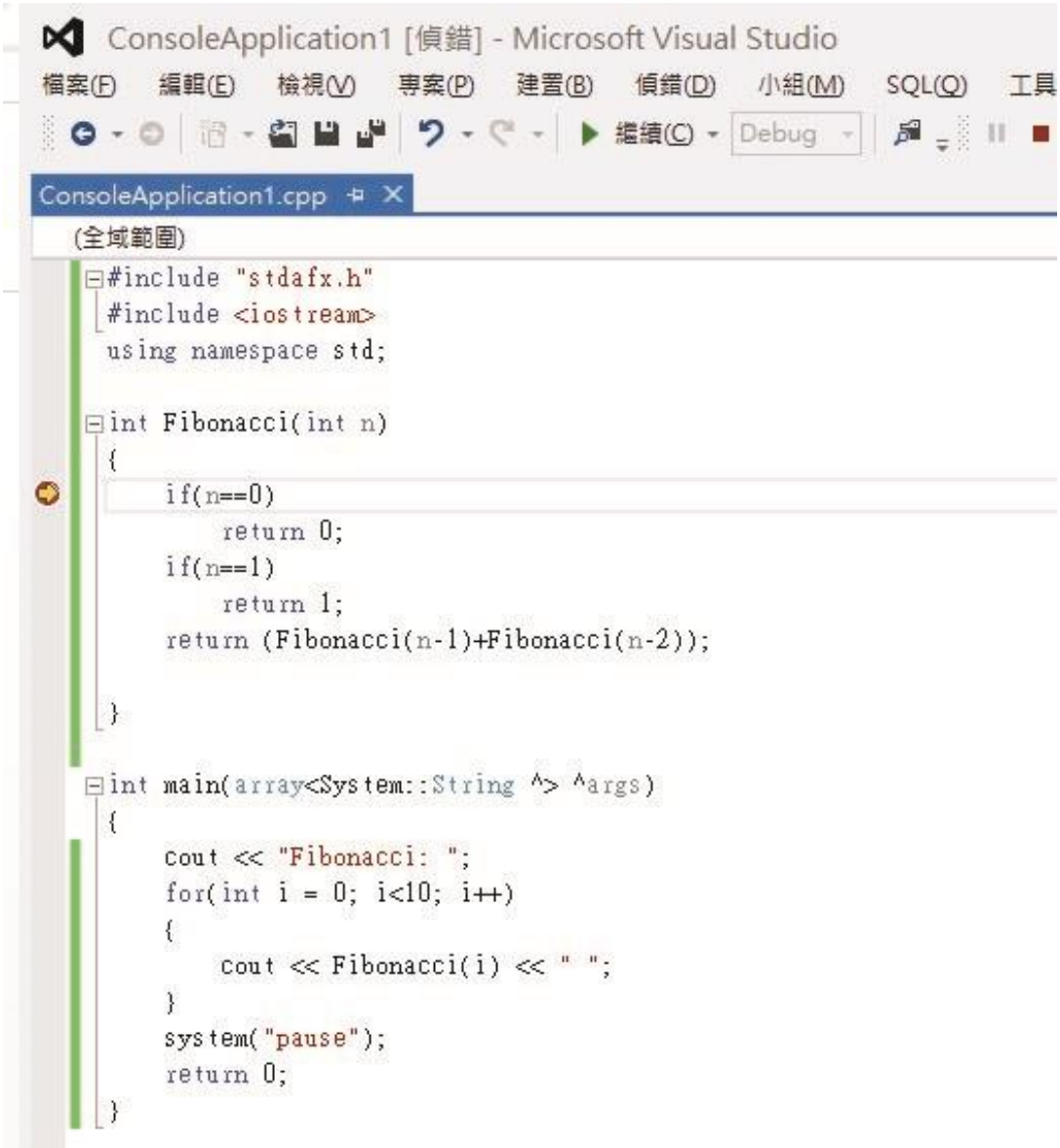
中斷點右鍵, 選擇條件，並在下拉式選單選擇叫用次數。



可以設呼叫次數等於或大於某數後才中斷



當 Fibonacci 呼叫到第 5 次才會觸發中斷



4. 設追蹤點(Trace Point)

- Output 視窗顯示

- 顯示 function 名稱($FUNCTION)
- 顯示呼叫的 function 名稱($CALLER)
- 顯示變數值({variable})

範例 4:泡沫排序法&選擇排序法

建立新專案 貼上 Sample Code

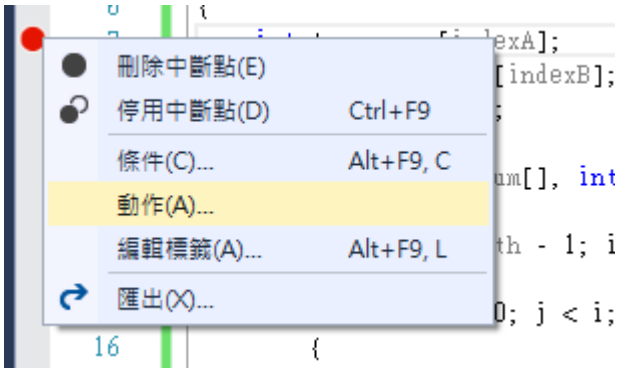
```
#include "stdafx.h"
#include <iostream> using
namespace std;

void Swap(int num[], int indexA, int indexB)
{
    int tmp = num[indexA];
num[indexA] = num[indexB];
num[indexB] = tmp;
}
void BubbleSort(int num[], int length)
{
    for (int i = length - 1; i > 0; --i)
    {
        for (int j = 0; j < i; ++j)
          {
            if (num[j] > num[j + 1])
{
                Swap(num, j, j + 1);
            }
          }
    }
}
void SelectionSort(int num[], int length)
{
    int i, j, max;
```

```
    for(i = 0; i<length; i++)
      {
        max = i;
        for(j = i + 1; j<length; j++)
           {
            if(num[j] > num[max])
max = j;
           }
           Swap(num, i, max);
      }
}

int main(array<System::String ^> ^args)
{
     int num[10] = {12, 42, 23, 51, 8, 31, 24, 57, 78, 33};
     int length = 10;     cout << "Array: ";  for(int i =0;
i<length; i++)
      {
           cout << num[i] << " ";
      }
     cout << endl;  //Bubble
Sort
     BubbleSort(num, length);
     cout << "Bubble Sort: ";
     for(int i =0; i<length; i++)
      {
           cout << num[i] << " ";
      }
      cout << endl;
      //Selection Sort
     SelectionSort(num, length);
     cout << "Selection Sort: ";
     for(int i =0; i<length; i++)
      {
           cout << num[i] << " ";
      }
```

```
    cout << endl;
    system("pause"); return
    0;


}
```

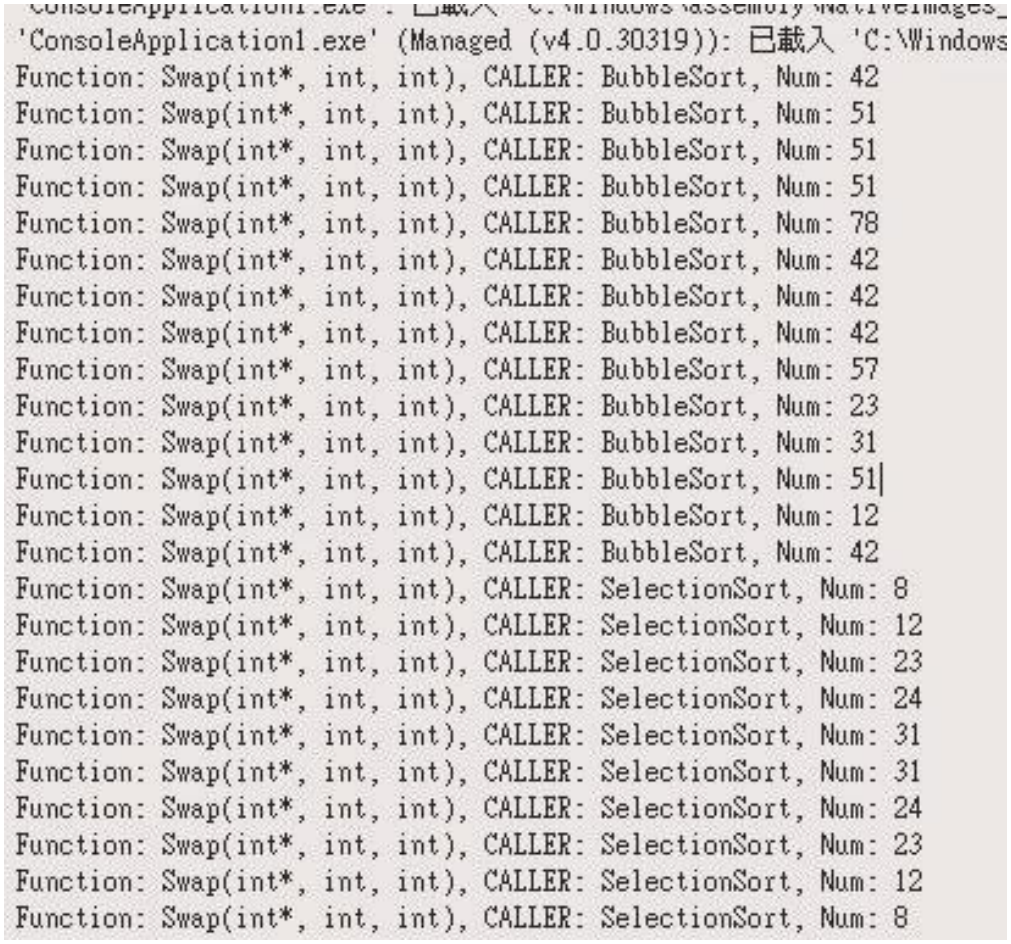
當想中斷的函式可能同時被多種函式呼叫, 如範例中的 Swap 函式在 BubbleSort 和 SelectionSort 都有被呼叫到

在 Swap 函式中設中斷點, 並對中斷點右鍵選擇動作



在列印訊息中可顯示出目前中斷的 Function 名稱, 和呼叫該 Function 的函式名稱
並可設定列印訊息和是否繼續執行

執行結果



可在下方輸出視窗中顯示目前中斷點的 Function 名稱, 呼叫的 Function 名稱和變數數值。