



Chapter 3

Flow Control & Exception



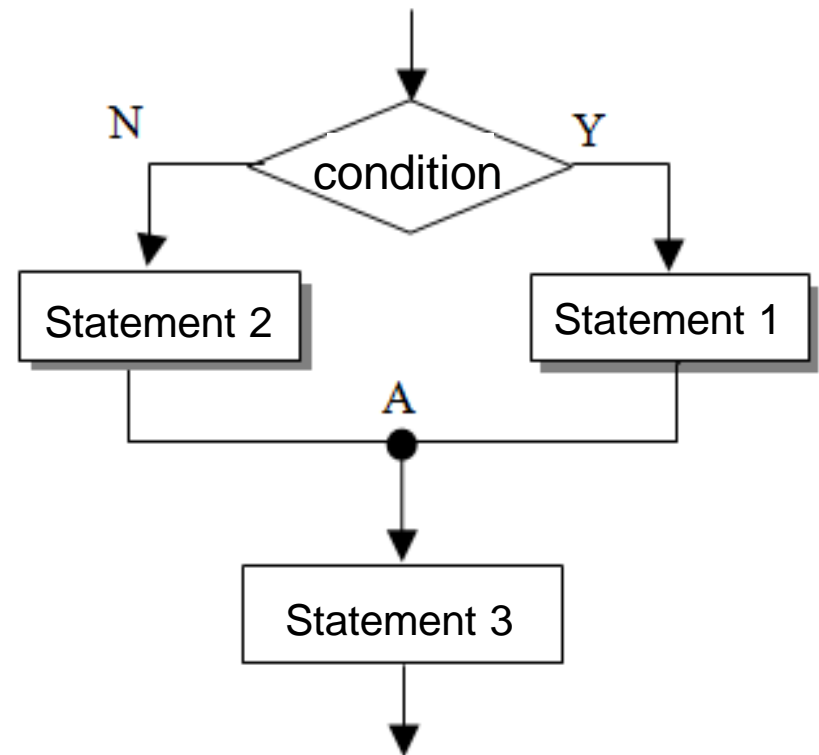
3.1 Selection Statements

3 selection statements in C#

1. if... else
2. if ... else if ... else
3. switch

3.1.1 if...else

```
if ( [condition] )  
{  
    [statement 1]  
}  
else  
{  
    [statement 2]  
}  
[statement 3]
```



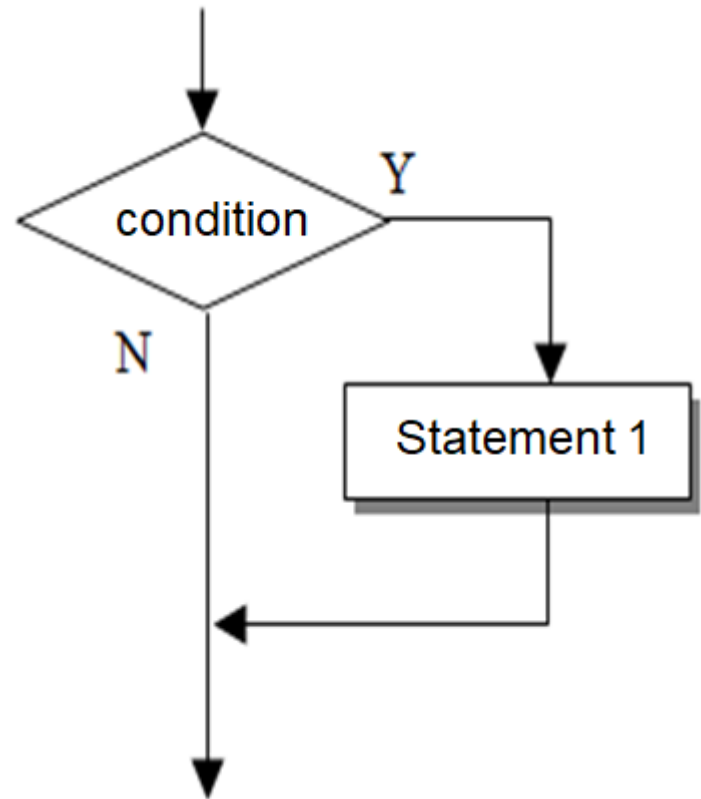
```
if ([condition])
```

```
{
```

```
.....
```

```
    [statement1]
```

```
}
```



{ } can be leave out if the statement is only in 1 line, usage:

```
if ( [condition] )  
    [statement]
```

Ex1: get the absolute value of the number “num”: usage:

```
if ( num < 0 )  
    num = -num;
```

Ex2: if “num” is a multiple of 3, show the quotient of “num” which is divided by 3.

Usage:

```
if ( num % 3 == 0)
{
    quotient = num / 3;
    Console.WriteLine("{0}被3整除的商為{1}", num, quotient);
}
```

Ex3: the price is 100 dollars if the age is ≤ 10 or > 60 , otherwise, the price is 200 dollars. Usage:

```
if ((age <= 10) || (age > 60))
{
    price = 100;
}
else
{
    price = 200;
}
```



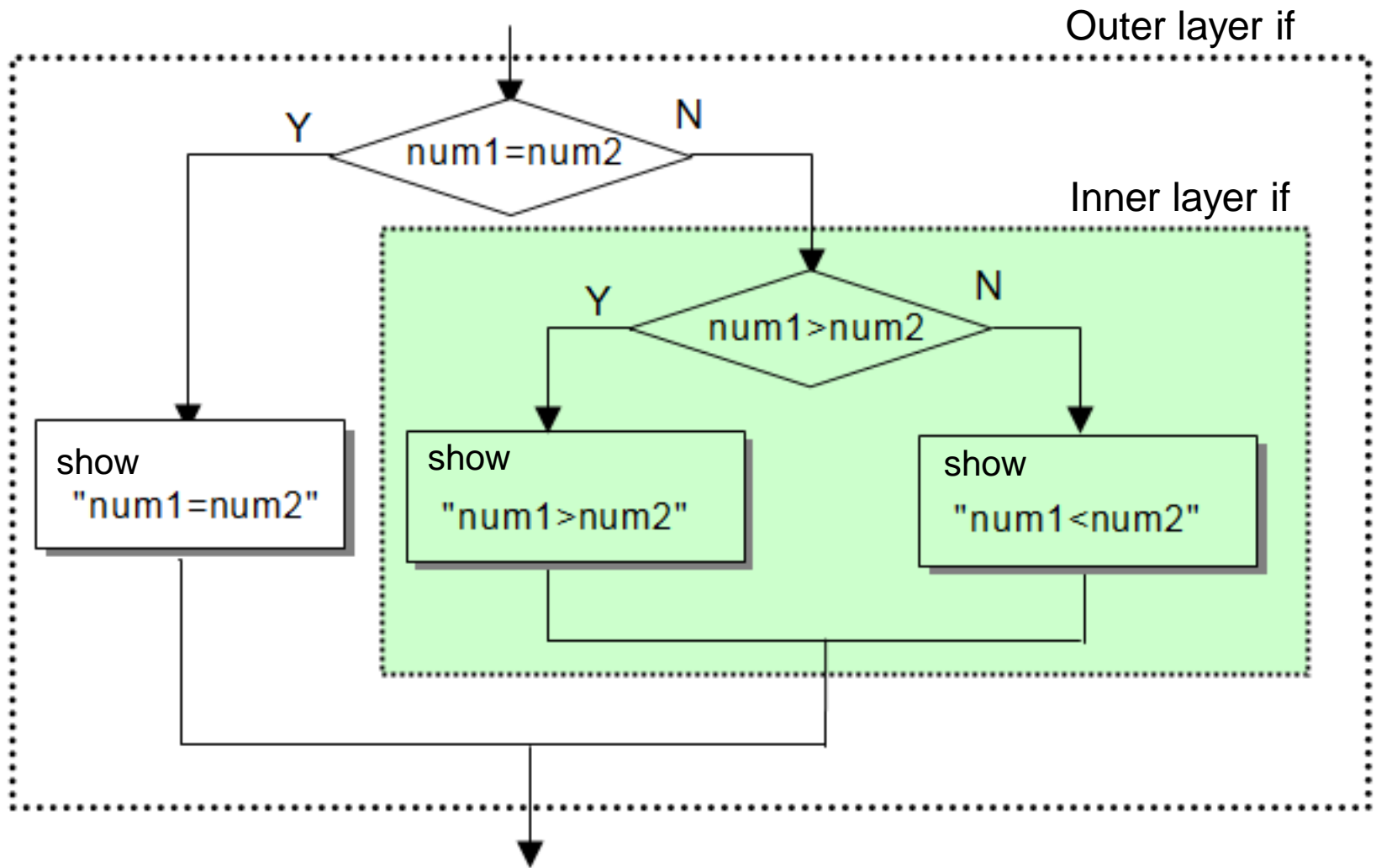
Nested if

- **Nested if is formed by the if-else section which has another if-else section inside**
- **3 conditions to determine complete 2 if-else statements to form nested if**

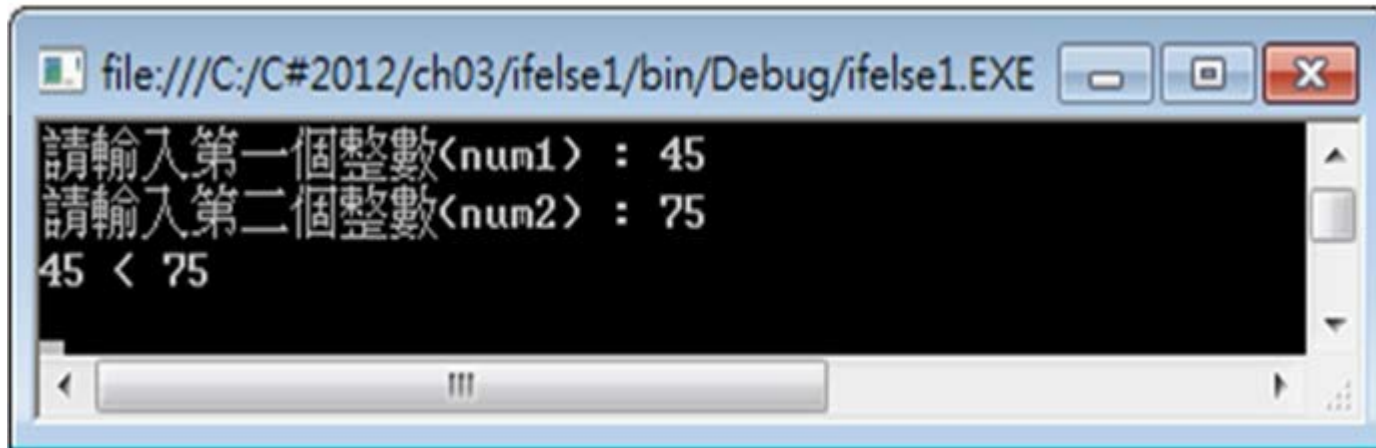
Example(ifelse1):

Try to write a program to get integer input from keyboard, then:

1. If $\text{num1} = \text{num2}$, show “ $\text{num1} = \text{num2}$ ”
2. If $\text{num1} > \text{num2}$, show “ $\text{num1} > \text{num2}$ ”
3. If $\text{num1} < \text{num2}$, show “ $\text{num1} < \text{num2}$ ”



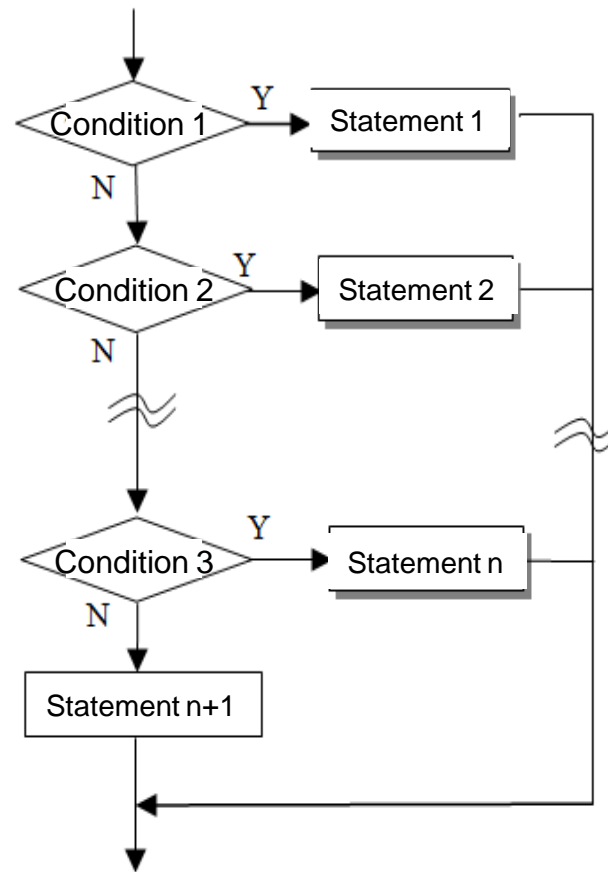
Example(ifelse1):



```
file:///C:/C#2012/ch03/ifelse1/bin/Debug/ifelse1.EXE
請輸入第一個整數<num1> : 45
請輸入第二個整數<num2> : 75
45 < 75
```

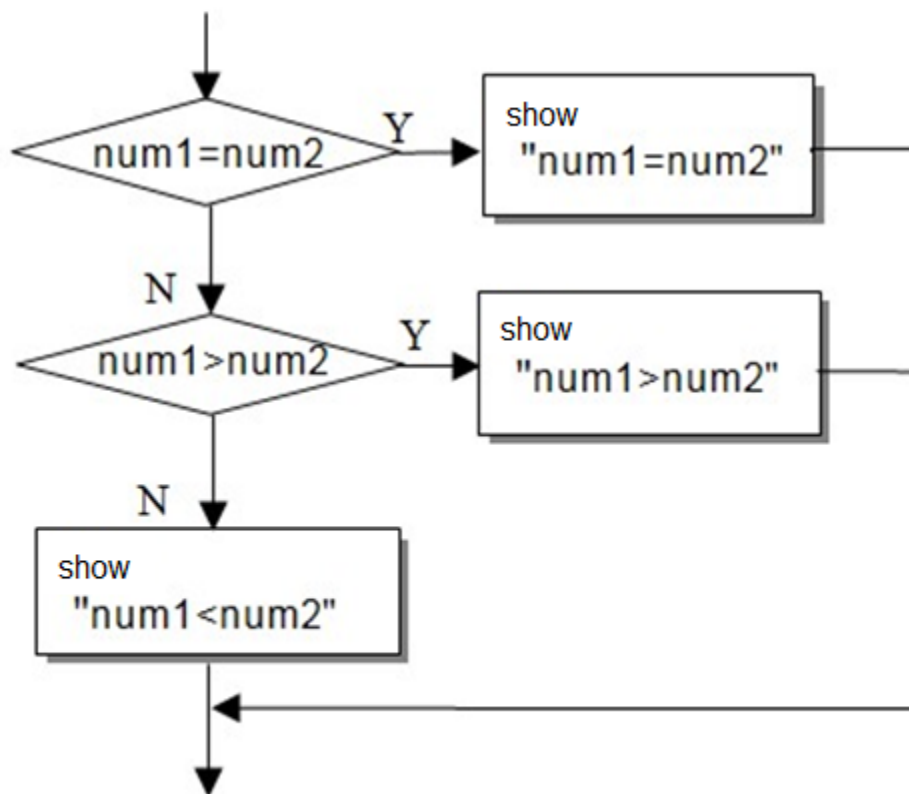
3.1.2 if else if else Multiple Selection

```
if ( [condition 1] )
{
    [statement 1]
}
else if ( [condition 2] )
{
    [statement 2]
}
...
else if ( [condition n] )
{
    [statement n]
}
else
{
    [statement n+1]
}
```



Example(ifelseif1):

From the former practice, use if...else if...else multiple selection statements to rewrite the program



3.1.3 switch Multiple Selection

- **Difference**

- ① **if ... else if ... else can use many different conditions**

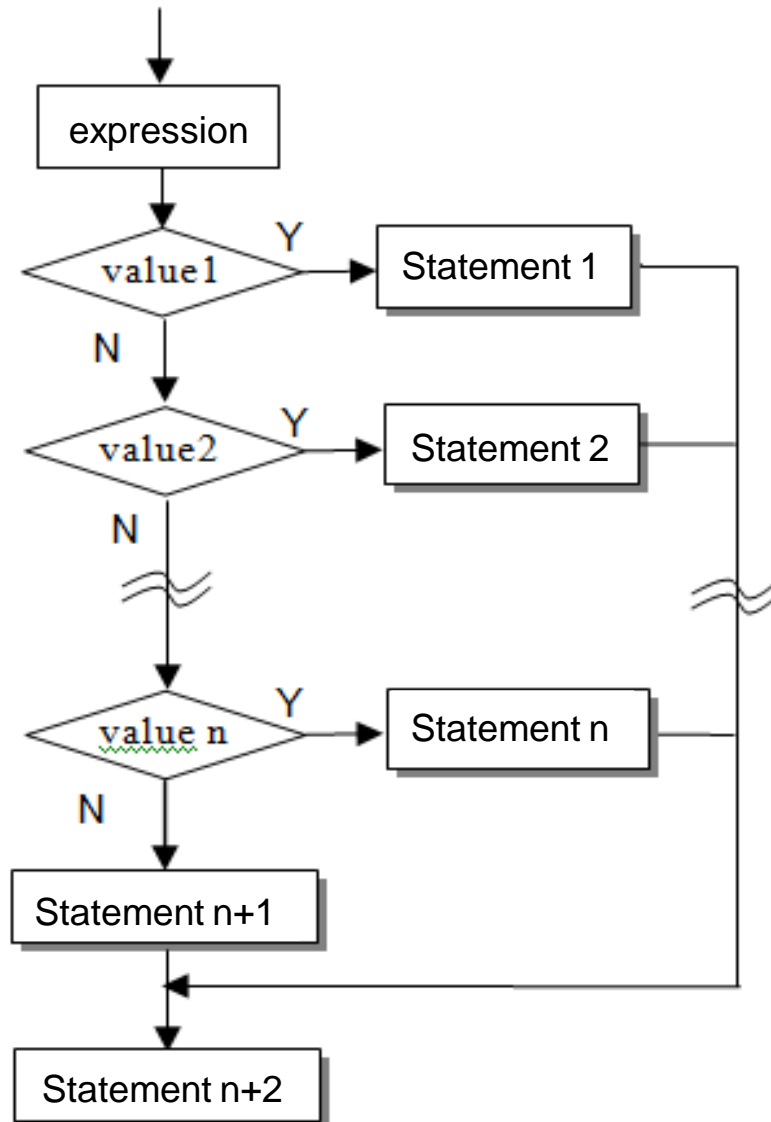
- ② **switch allows only 1 statement**

- **Too many “if” statements cause complexity and low maintainability, but “switch” statement does not**

```

switch ( [expression] )
{
    case [value 1]:
        [statement 1]
        break;
    case [value 2]:
        [statement 2]
        break;
    ...
    case [value n]:
        [statement n]
        break;
    default:
        [statement n+1];
}
[statement n+2];

```



● Case in different style

① If condition 1, 2, 4 is true:

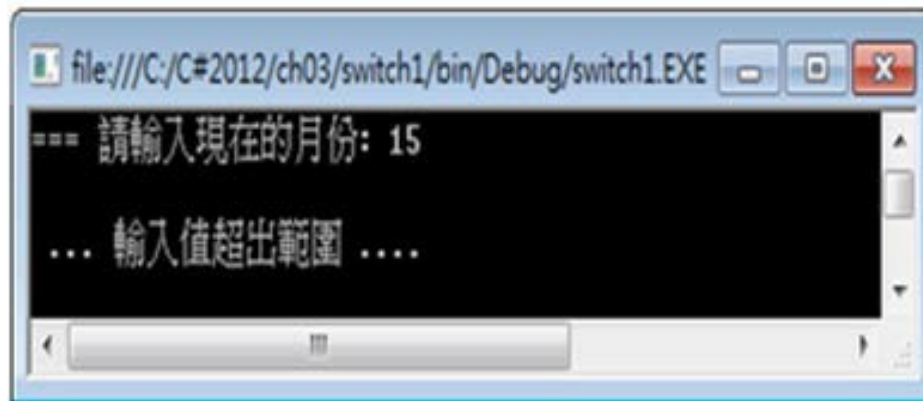
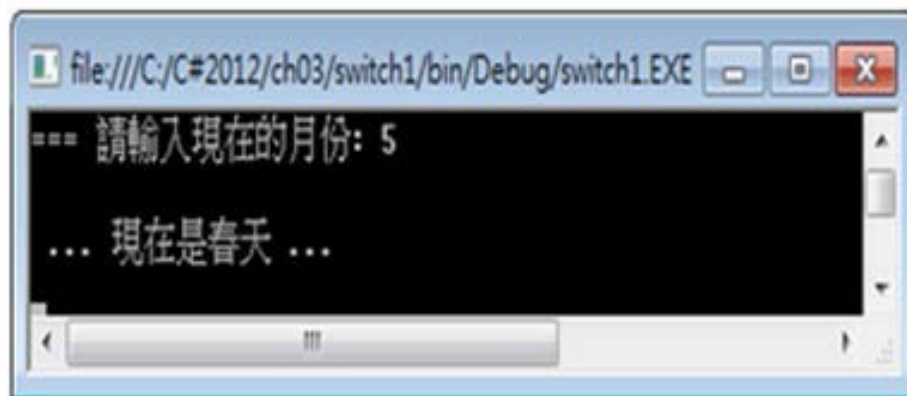
```
case 1 :  
case 2:  
case 4:  
    :  
    : Statement;  
    :  
break;
```

② If result of condition is "Y" or "y" or true:

```
case "y" :  
case "Y" :  
    :  
    : Statement;  
    :  
break;
```

Example(switch1):

Try to use switch statement to get month input from keyboard and show the season of the month. If the input is not 1~12, show message.

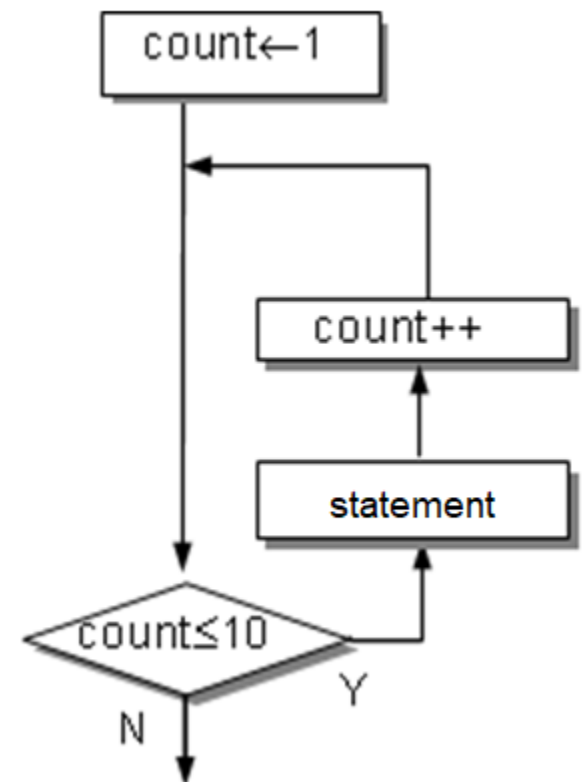
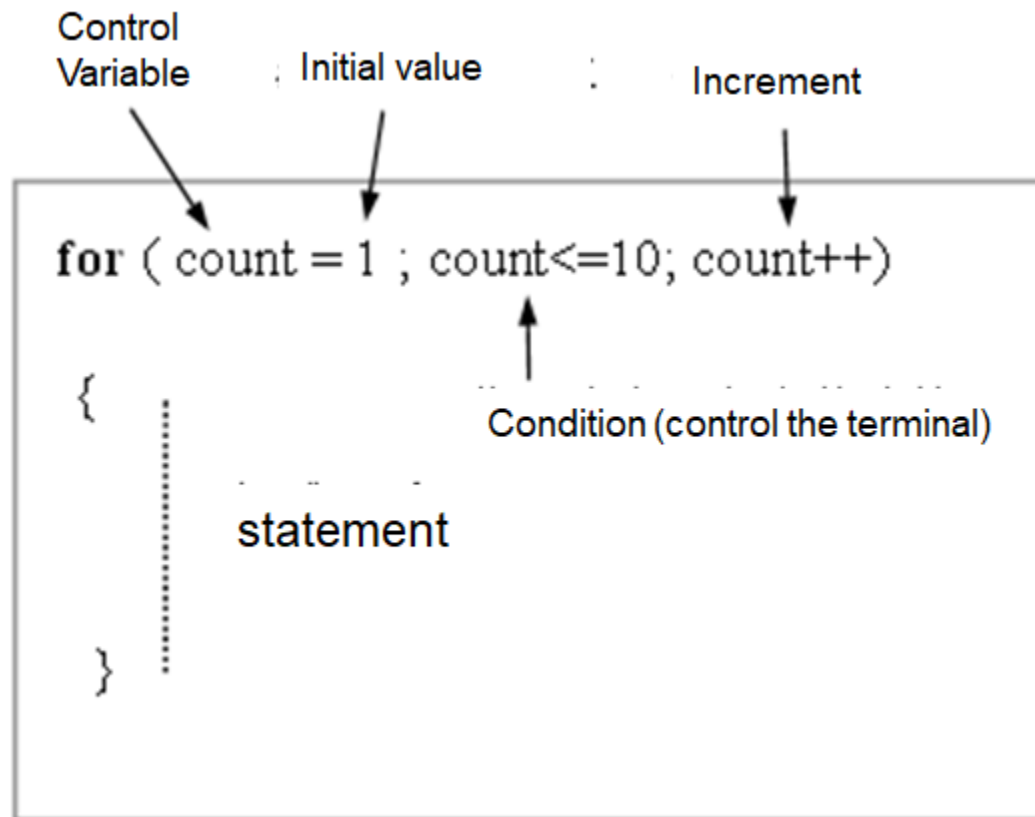


3.2 Iteration Statements

- Also called **loop**
- Some sections in the program have to be repeated in several times or until the condition is not fulfilled
- “for” statement: assign the number of times, called counter controlled loop
- “while” statement: run according to the condition, called condition controlled loop

3.2.1 for

- Counter controlled loop begins from left bracket of for loop and ends at right bracket



- Use “break” statement to leave from halfway for loop
- Use “continue” statement to jump to the beginning of “for” loop immediately and carry on execution

Ordinary for loop usage:

① **for (k=1 ; k<= 5 ; k++)**

k = 1, 2, 3, 4, 5. The loop executes 5 times

② **for (k=1 ; k<=5 ; k+=2)**

k = 1, 3, 5. The loop executes 3 times

③ Initial value and iterator can be a decimal

for ($k = -0.5$; $k \leq 1.5$; $k += 0.5$)

$k = -0.5, 0, 0.5, 1.0, 1.5$. The loop executes 5 times

④ Iterator is decrement

for ($k = 6$; $k \geq 1$; $k -= 2$)

$k = 6, 4, 2$. The loop executes 3 times

⑤ 2 or more initial values, separate them by comma(,):

for (x=1, y=5 ; x<3 && y>2 ;x++ , y--)

x=1 & y=5; x=2 & y=4; the loop executes 2 times

⑥ Initial values and condition can have expressions

for (k=x ; k<=y+9 ; k+=2)

if x=1, y=-2, then k = 1,3,5,7. The loop executes 4 times

⑦ Infinite loop

for (; ;)

Example(series):

Get the sum of the following series

$$\sum_{x=1}^5 (3x+2) = \frac{5}{x=1} + \frac{8}{x=2} + \frac{11}{x=3} + \frac{14}{x=4} + \frac{17}{x=5} = ?$$



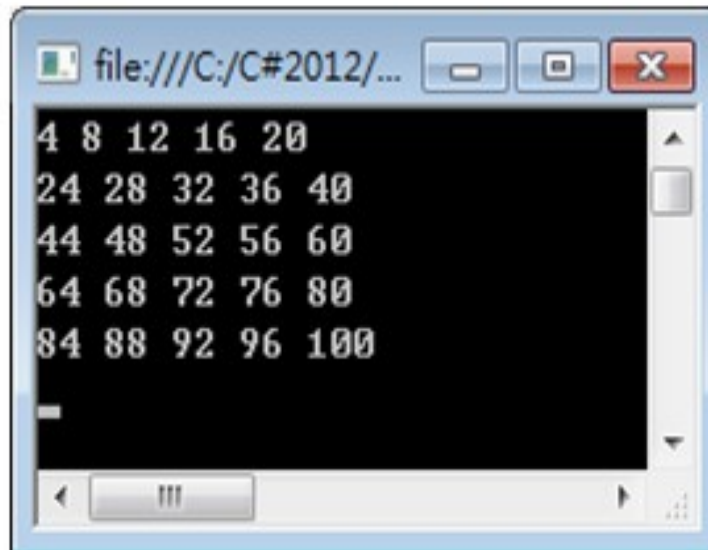
```
file:///C:/C#2012/ch03/series/bin/Debug/series.EXE

=== 求級數的總和 ===

  x      3x+2
=====
  1         5
  2         8
  3        11
  4        14
  5        17
-----
此級數總和為 : 55
```

Example(for1):

Try to write a program to show the multiples of 4 between 1 and 100.
Print 5 numbers in 1 line



```
file:///C:/C#2012/...
4 8 12 16 20
24 28 32 36 40
44 48 52 56 60
64 68 72 76 80
84 88 92 96 100
```

3.2.2 Nested Loop

- A loop which has loops inside forms a nested loop usually used in 2-d array
- Use nested loop to show numbers like a ladder
1st stair shows 1,
2nd stair shows 1 2,
3rd stair shows 1 2 3 ...
a space between numbers, show 6 stairs

Example(forsample):

Assume i is the number of lines, k stands for the number to show

```
1
1 2
1 2 3
1 2 3 4
1 2 3 4 5
1 2 3 4 5 6
```

When i = 1, show k = 1~1

When i = 2, show k = 1~2

...

When i = 6, show k = 1~6

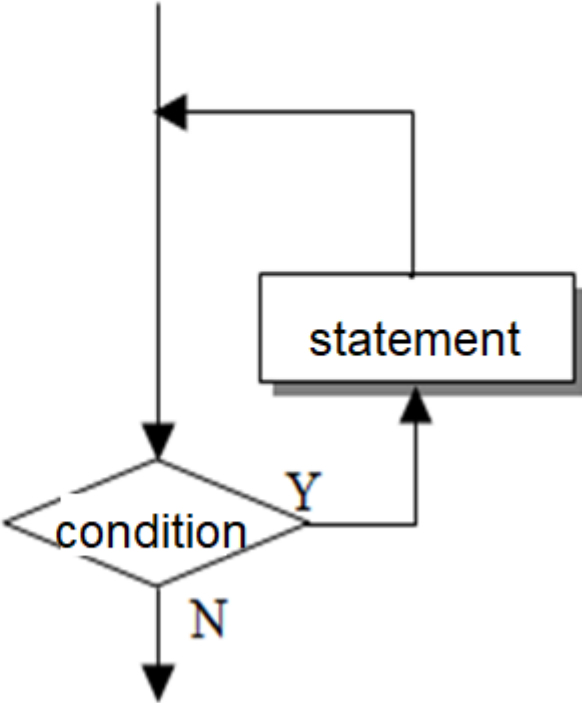
Source code:

```
for (int i = 1; i <= 6; i++)
    for (int k = 1; k <= i; k++)
        Console.Write("{0} ", k);
    Console.WriteLine();
```

3.2.3 Pre-test Loop

- The condition statement is at the beginning of the loop
- Decide to enter the loop or not by the result of condition
 - ① fulfilled, execute the statements in the loop once and back to the beginning of the loop
 - ② not fulfilled, exit the loop
- First time enter the loop, and the condition is false, exit the loop immediately

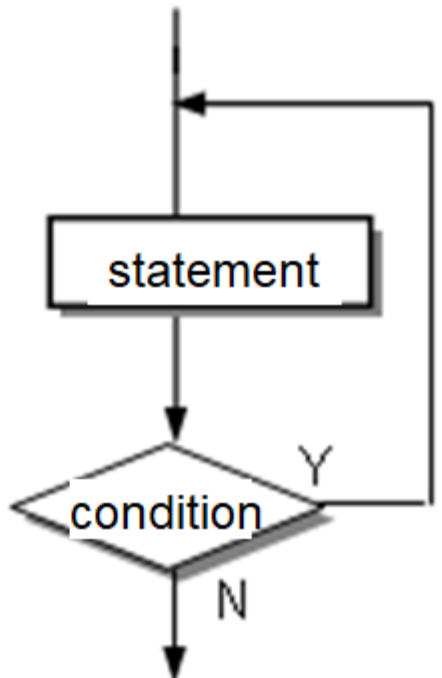
3.2.3 Pre-test Loop

Grammar	Flow chart
<pre>while ([condition]) { [statement] }</pre>	 <pre>graph TD; Entry(()) --> Decision{condition}; Decision -- Y --> Statement(statement); Statement --> Decision; Decision -- N --> Exit(());</pre> <p>The flowchart illustrates the execution of a pre-test loop. It begins with an entry point (represented by a downward arrow) leading to a decision diamond labeled 'condition'. If the condition is true (labeled 'Y'), the flow proceeds to a rectangular process box labeled 'statement'. After the statement is executed, the flow loops back to the entry point just before the decision diamond. If the condition is false (labeled 'N'), the flow proceeds to an exit point (represented by a downward arrow).</p>

3.2.4 Post-test Loop

- The condition statement is put at the end of the loop
- First run does not examine the condition, then the condition is checked at the end of loop
 - ① the statements run one more time if the condition is fulfilled, then the condition at the end of loop is examined again
 - ② exit the loop until the condition is not fulfilled
- The statement in the loop runs at least one time

3.2.4 Post-test Loop

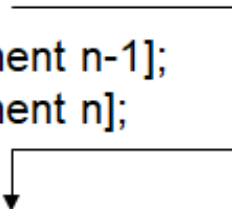
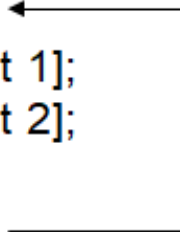
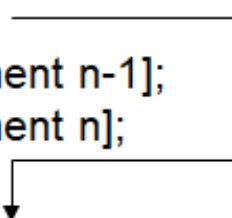
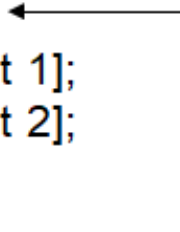
Grammar	Flow chart
<pre>do { [statement] } while ([condition]);</pre>	 <pre>graph TD; Entry(()) --> Statement[statement]; Statement --> Condition{condition}; Condition -- Y --> Entry; Condition -- N --> Exit(());</pre> <p>The flowchart illustrates the execution of a post-test loop. It begins with an entry point (represented by a vertical line with an arrow) leading into a rectangular process box labeled 'statement'. An arrow from the bottom of the 'statement' box points down to a diamond-shaped decision box labeled 'condition'. From the right side of the 'condition' diamond, an arrow labeled 'Y' loops back up to the entry point above the 'statement' box, indicating that if the condition is true, the loop repeats. From the bottom of the 'condition' diamond, an arrow labeled 'N' points down, representing the exit from the loop.</p>

Example(factorial):

Try to write a program which uses pre-test loop to calculate factorial. First the user inputs an integer, then the factorial of the number is computed.

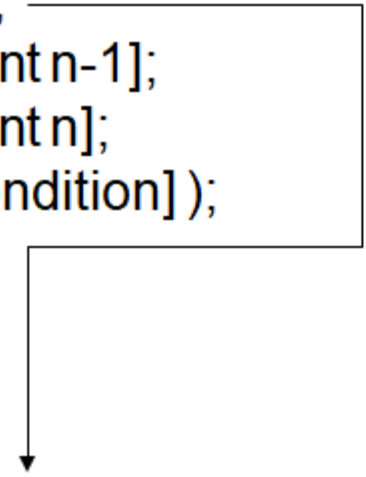


3.3 break and continue

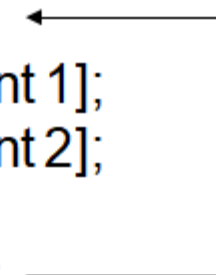
break	continue
<pre>for (...) { [statement 1]; [statement 2]; ... break; [statement n-1]; [statement n]; }</pre> 	<pre>for (...) { [statement 1]; [statement 2]; ... continue; [statement n-1]; [statement n]; }</pre> 
<pre>while ([condition]) { [statement 1]; [statement 2]; ... break; [statement n-1]; [statement n]; }</pre> 	<pre>while (...) { [statement 1]; [statement 2]; ... continue; [statement n-1]; [statement n]; }</pre> 

3.3 break and continue

```
do
{
    [statement 1];
    [statement 2];
    ... break;
    [statement n-1];
    [statement n];
} while ( [condition] );
```

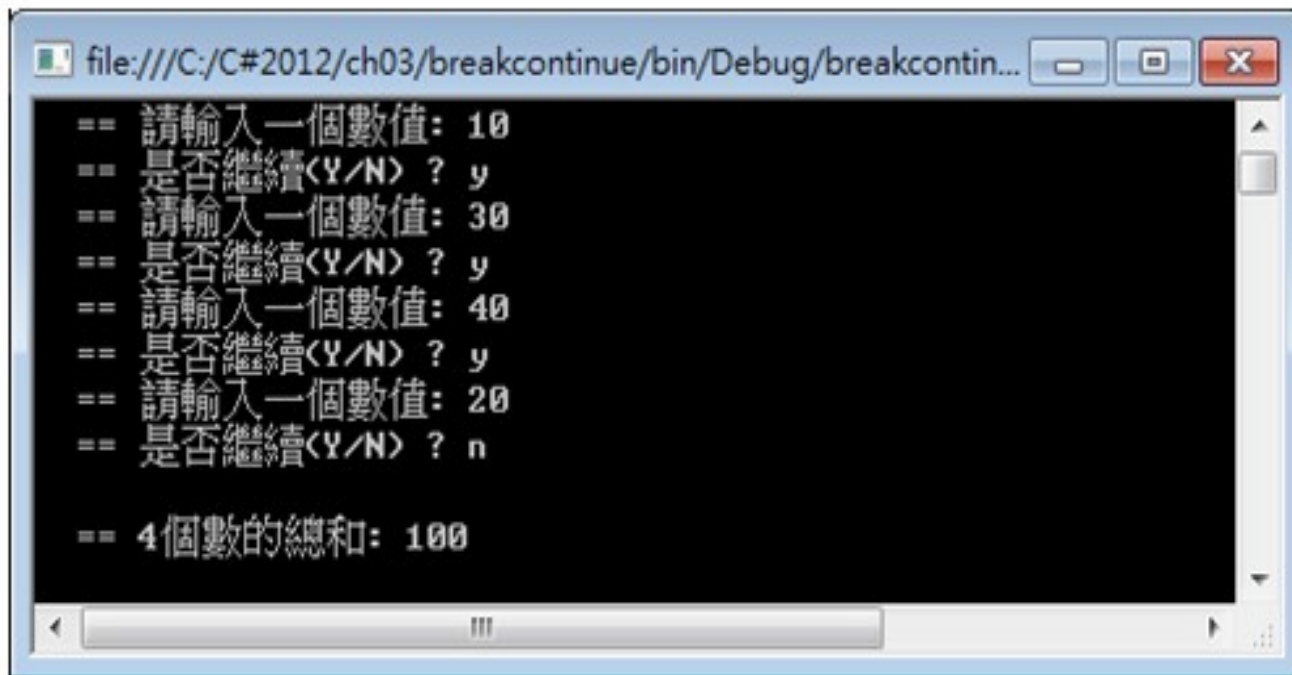


```
do
{
    [statement 1];
    [statement 2];
    ...
    continue;
    [statement n-1];
    [statement n];
} while ( [condition] );
```



Example(breakcontinue):

Try to write a program which accumulate numbers, use break and continue to decide whether continue accumulating or not in do...while loop.



```
file:///C:/C#2012/ch03/breakcontinue/bin/Debug/breakcontin...  
== 請輸入一個數值: 10  
== 是否繼續(Y/N) ? y  
== 請輸入一個數值: 30  
== 是否繼續(Y/N) ? y  
== 請輸入一個數值: 40  
== 是否繼續(Y/N) ? y  
== 請輸入一個數值: 20  
== 是否繼續(Y/N) ? n  
  
== 4個數的總和: 100
```

3.4 Program Debug

- The unexpected result may cause
 - ① compilation error
 - ② runtime error
- Syntax error
the error occurs during compilation. The identifier is marked by **wavy blue** underline and unrecognizable.

3.4 Program Debug

Continue

- **Logic Error**

no error occurs after compilation completed, but the expected result does not happen when the program is running

- **Logic error is not grammatical error**

- **program flow**
- **statement**
- **wrong variable application**



3.5 Exception

- **Error occurs when the program is running**
- **C# provides a structured and easy-to-control solution to handle the unexpected condition**

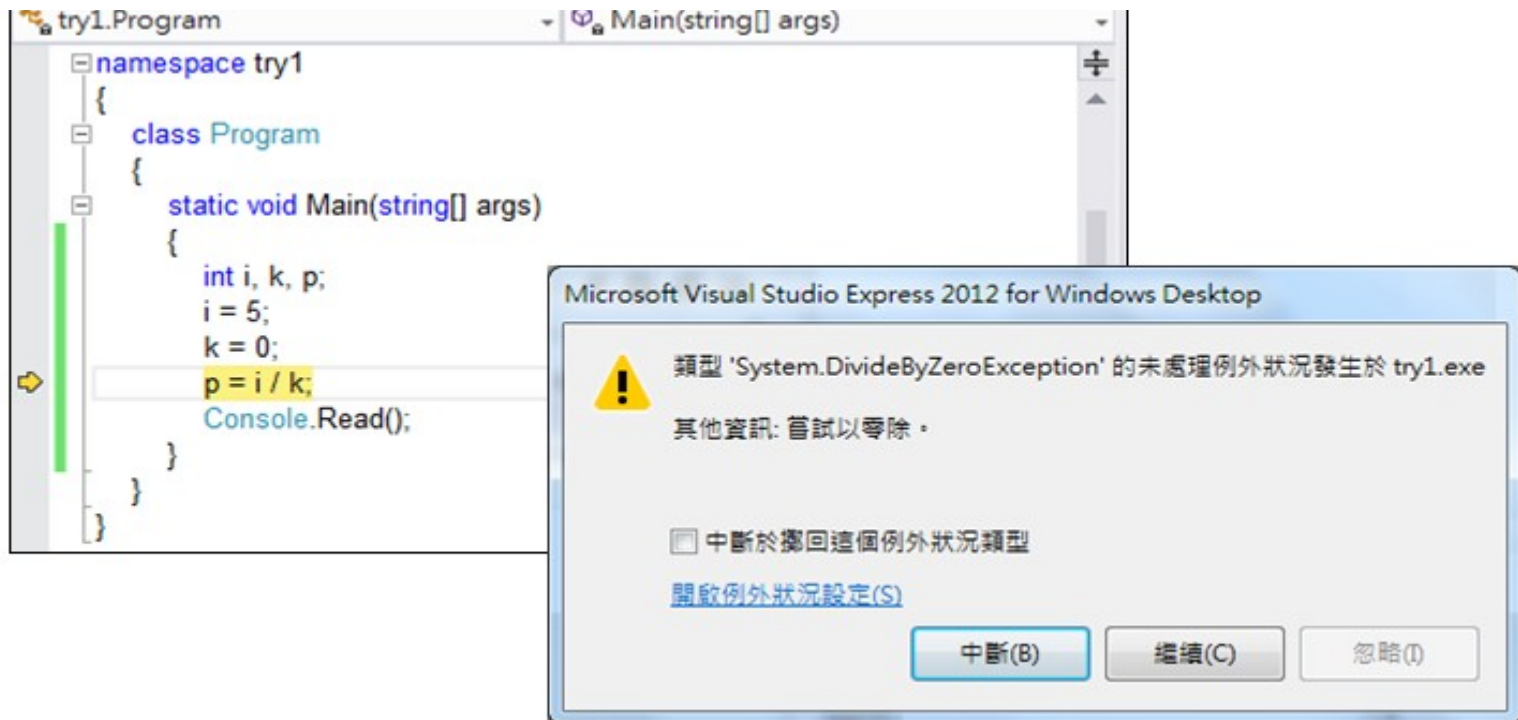
```
try
{
    [try statement]
}
catch (exception1 ex)
{
    [catch statement]
}
catch (exception2 ex)
{
    [catch statement]
}
...
finally
{
    [finally statement]
}
```

3.5 Exception

Exception classes	Error reason
ArgumentOutOfRangeException	Argument's data type is out of the range defined by the function parameter
DivideByZeroException	Divisor is zero
IndexOutOfRangeException	Array index is out of the maximum size
InvalidCastException	Data type conversion error
OverflowException	Data over flow
Exception	Runtime error

Example(try1):

Try to write a program which can cause DivideByZeroException. First i, k, p are declared. The initial value of i is 5, k is 0. The program is terminated when i/k causes DivideByZeroException.



Example(try2):

From the former practice, try to insert try...catch to handle the exception.

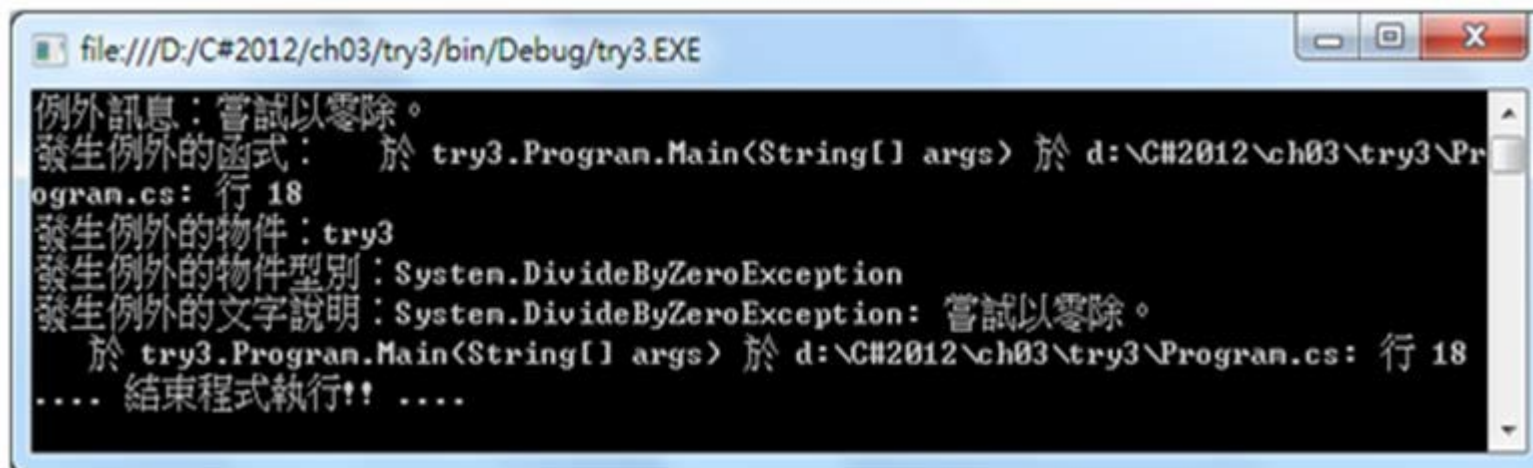
FileName : try2.sln

```
01 static void Main(string[] args)
02 {
03     int i, k, p;
04     i = 5;
05     k = 0;
06     try
07     {
08         p = i / k;          // 將可能發生例外的程式碼置於 try 區塊
09     }
10     catch (Exception ex) // 當發生的例外符合 Exception 時會執行此處
11     {
12         Console.WriteLine("發生例外");
13     }
14     finally                // 無論是否發生例外皆會執行 finally 區塊
15     {
16         Console.WriteLine(".... 結束程式執行!! ....");
17     }
18     Console.Read();
19 }
```



● Attributes and methods in common use

Members of exception	Description
GetType	Get data type of exception object
ToString	Get text description of exception object
Message	Get exception message
Source	Get application or object which cause exception
StackTrace	Get methods or functions which cause exception

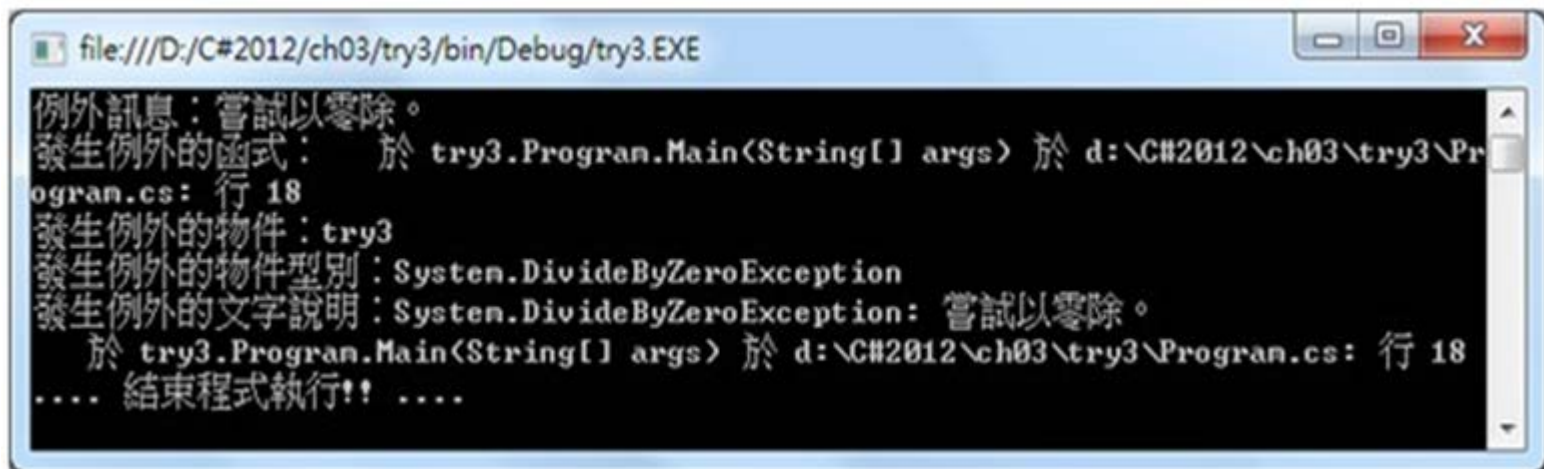


file:///D:/C#2012/ch03/try3/bin/Debug/try3.EXE

```
例外訊息：嘗試以零除。  
發生例外的函式： 於 try3.Program.Main(String[] args) 於 d:\C#2012\ch03\try3\Pr  
ogram.cs: 行 18  
發生例外的物件：try3  
發生例外的物件型別：System.DivideByZeroException  
發生例外的文字說明：System.DivideByZeroException: 嘗試以零除。  
    於 try3.Program.Main(String[] args) 於 d:\C#2012\ch03\try3\Program.cs: 行 18  
.... 結束程式執行!! ....
```

Example(try3):

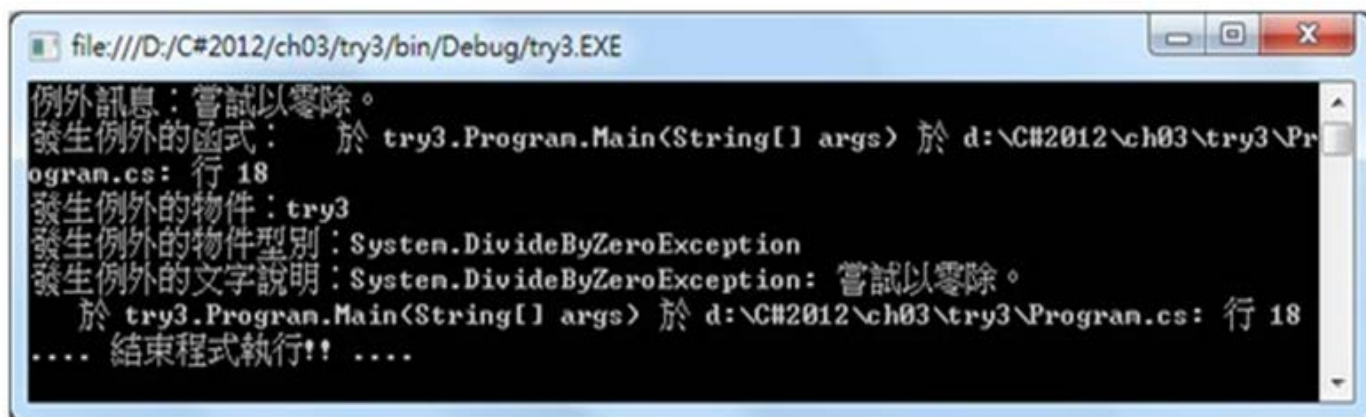
From the former practice, please use GetType, ToString, Message, Source, StackTrace members of exception object to show the information of exception



```
file:///D:/C#2012/ch03/try3/bin/Debug/try3.EXE
例外訊息：嘗試以零除。
發生例外的函式： 於 try3.Program.Main(String[] args) 於 d:\C#2012\ch03\try3\Program.cs: 行 18
發生例外的物件：try3
發生例外的物件型別：System.DivideByZeroException
發生例外的文字說明：System.DivideByZeroException: 嘗試以零除。
    於 try3.Program.Main(String[] args) 於 d:\C#2012\ch03\try3\Program.cs: 行 18
.... 結束程式執行!! ....
```

FileName : try3.sln

```
01  static void Main(string[] args)
02  {
03      int i, k, p;
04      i = 5;
05      k = 0;
06      try
07      {
08          p = i / k; // 將可能發生例外的程式碼置於 try 區塊
09      }
10      catch (DivideByZeroException ex)
11      {
12          Console.WriteLine("例外訊息：{0}", ex.Message);
13          Console.WriteLine("發生例外的函式：{0}", ex.StackTrace);
14          Console.WriteLine("發生例外的物件：{0}", ex.Source);
15          Console.WriteLine("發生例外的物件型別：{0}", ex.GetType());
16          Console.WriteLine("發生例外的文字說明：{0}", ex.ToString());
17      }
18      finally          // 無論是否發生例外，皆會執行 finally 區塊中的程式碼
19      {
20          Console.WriteLine(".... 結束程式執行!! ....");
21      }
22      Console.Read();
23  }
```



```
file:///D:/C#2012/ch03/try3/bin/Debug/try3.EXE
例外訊息：嘗試以零除。
發生例外的函式： 於 try3.Program.Main(String[] args) 於 d:\C#2012\ch03\try3\Program.cs: 行 18
發生例外的物件：try3
發生例外的物件型別：System.DivideByZeroException
發生例外的文字說明：System.DivideByZeroException: 嘗試以零除。
    於 try3.Program.Main(String[] args) 於 d:\C#2012\ch03\try3\Program.cs: 行 18
.... 結束程式執行!! ....
```



The End

Take a Break.....