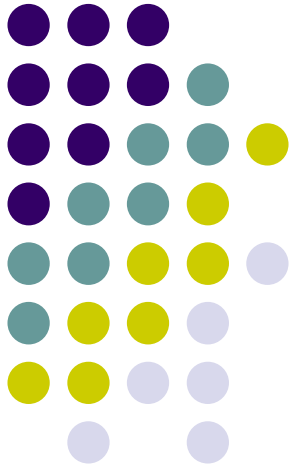
第五章
選擇性敘述與迴圈



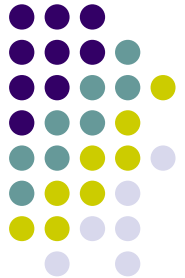
認識程式的結構設計

學習選擇性敘述與各種迴圈的用法

學習多重選擇敘述的用法

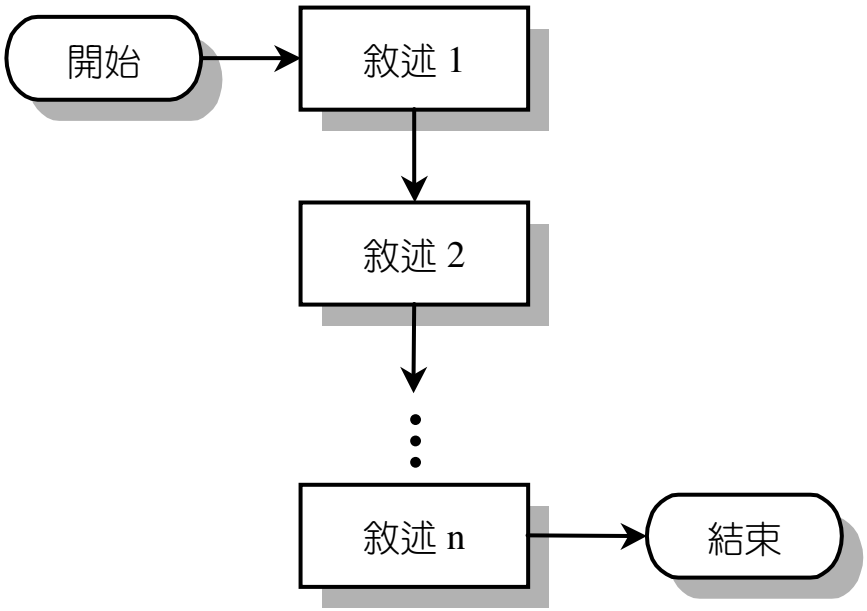
運算式、運算元與運算子

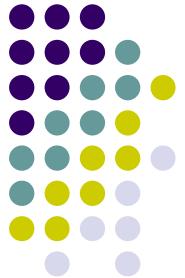
- 程式的結構包含有下面三種：
  - 循序性結構（sequence structure）
  - 選擇性結構（selection structure）
  - 重複性結構（iteration structure）



循序性結構

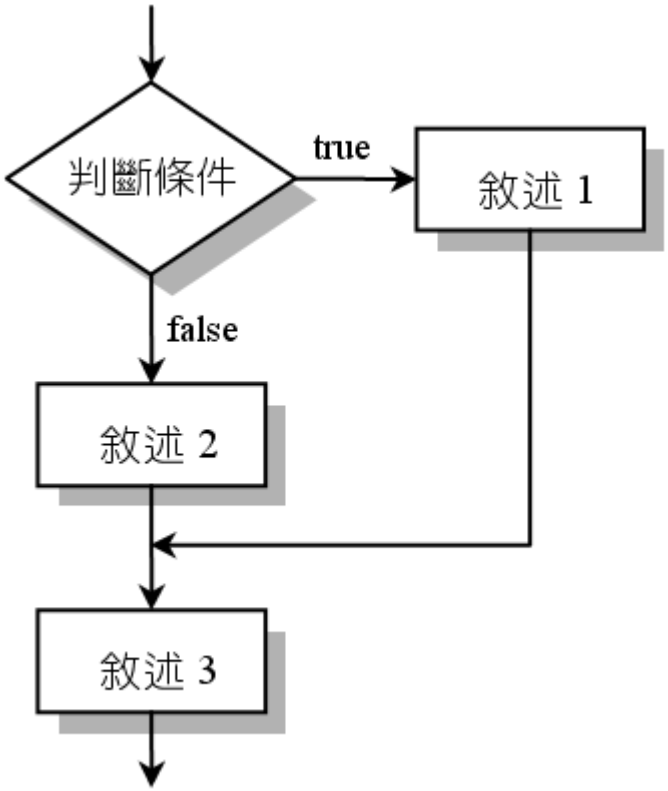
- 採上至下（top to down）的敘述執行





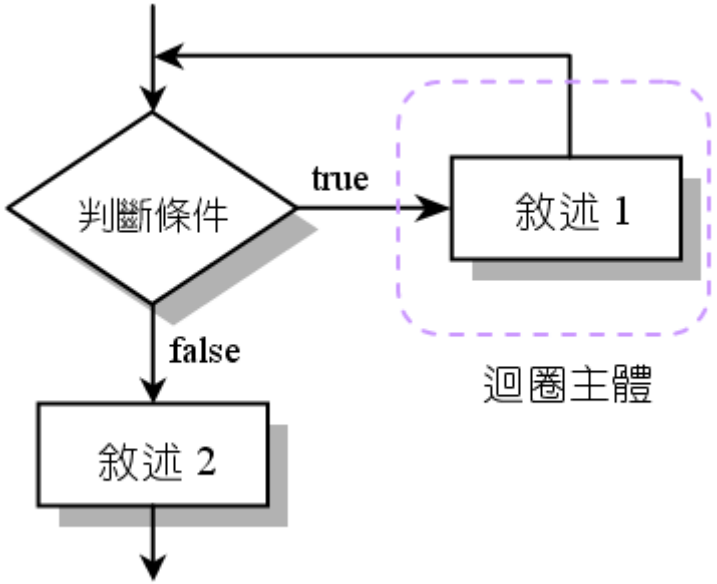
選擇性結構

- 根據條件的成立與否，再決定要執行哪些敘述



重複性結構

- 根據判斷條件的成立與否，決定程式段落的執行次數
- 重複性結構有for、while及do while三種迴圈

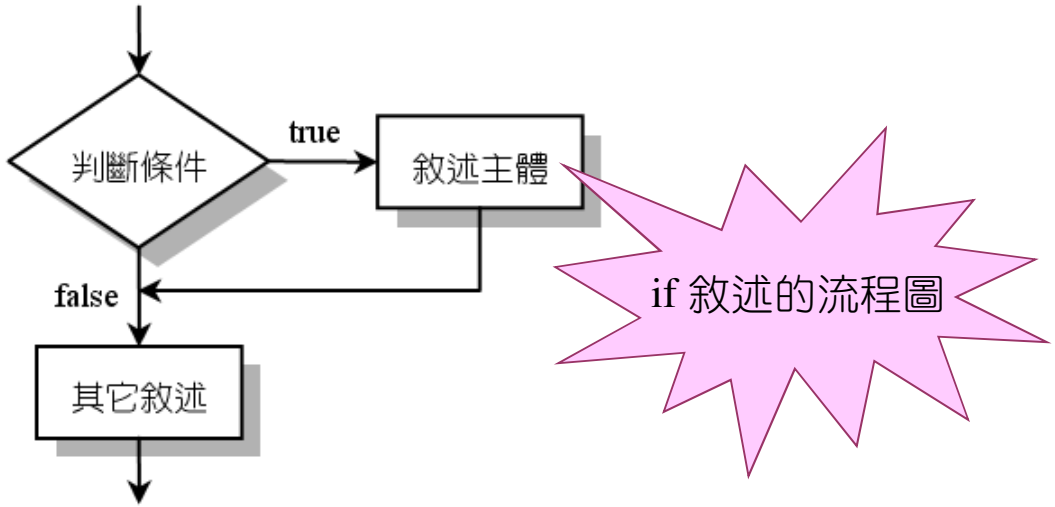


if 敘述

- 根據判斷的結果來執行不同的敘述

if 敘述的格式

```
if(判斷條件)
{
    敘述主體;
}
```

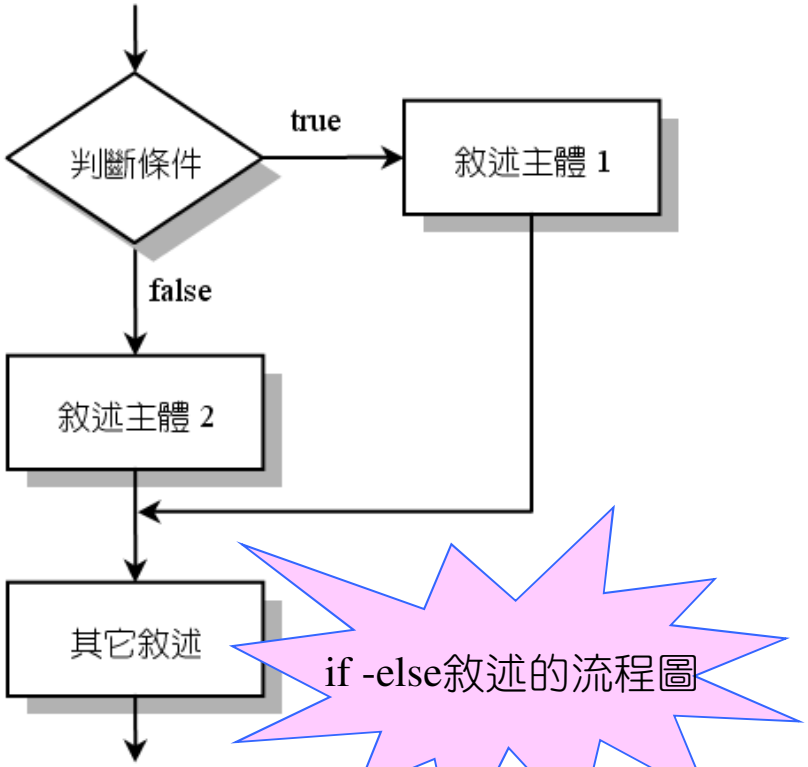


if-else 敘述 (1/2)

- if-else敘述的格式與流程圖如下：

if-else 敘述的格式

```
if(判斷條件)
{
    敘述主體1;
}
else
{
    敘述主體2;
}
```



if -else敘述的流程圖

if-else 敘述 (2/2)

- 下面的範例可用來判斷變數a是奇數或是偶數

```
01  // app5_1, if-else 敘述
02  public class app5_1
03  {
04     public static void main(String args[])
05     {
06        int a=15;
07
08        if (a%2==0)           // 如果可被 2 整除
09           System.out.println(a+" is an even number"); // 印出 a 為偶數
10        else
11           System.out.println(a+" is an odd number"); // 印出 a 為奇數
12     }
13  }
```

```
/* app5_1 OUTPUT-----

15 is an odd number
--------------------*/
```

巢狀 if 敘述

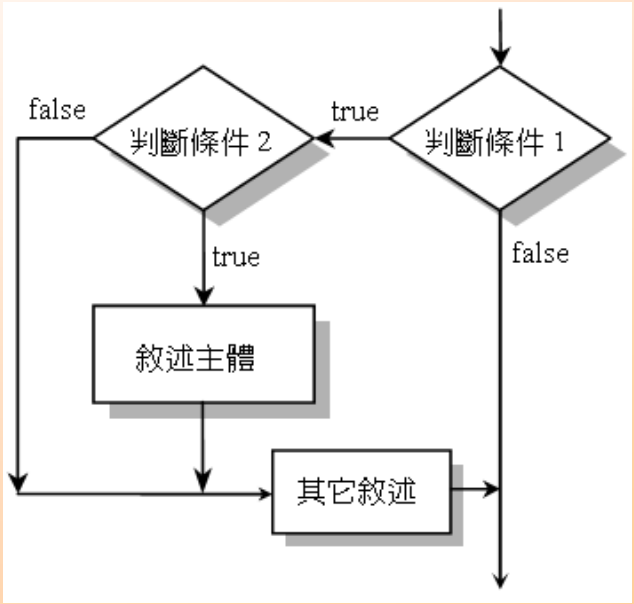
- if 敘述中又包含其它 if 敘述時，稱為巢狀 if 敘述（nested if）

若判斷條件1成立，則執行這個部份

巢狀if敘述的格式

```
if(判斷條件1)
{
    if(判斷條件2)
    {
        敘述主體;
    }
    ...
    其它敘述;
}
```

若判斷條件2成立，則執行這個部份



條件運算子 (1/2)

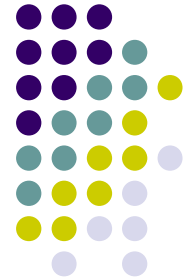
- 條件運算子的說明：

條件運算子	意義
?:	根據條件的成立與否，來決定結果為?或:後的運算式

- ?: 的格式：

?:的敘述格式

```
傳回值 = 判斷條件 ? 運算式1 : 運算式2;
```



條件運算子 (1/2)

- 下面的程式可找出二數之間較大的數：

```
01  // app5_2, 條件運算子?:的使用
02  public class app5_2
03  {
04     public static void main(String args[])
05     {
06        int a=8,b=3,max;
07
08        max=(a>b)?a:b;                    // a>b 時,max=a,否則 max=b
09
10        System.out.println("a="+a+", b="+b);
11        System.out.println(max+"是較大的數");
12     }
13  }
```

```
/* app5_2 OUTPUT----
a=8, b=3
8 是較大的數
-------------------*/
```

for迴圈 (1/2)

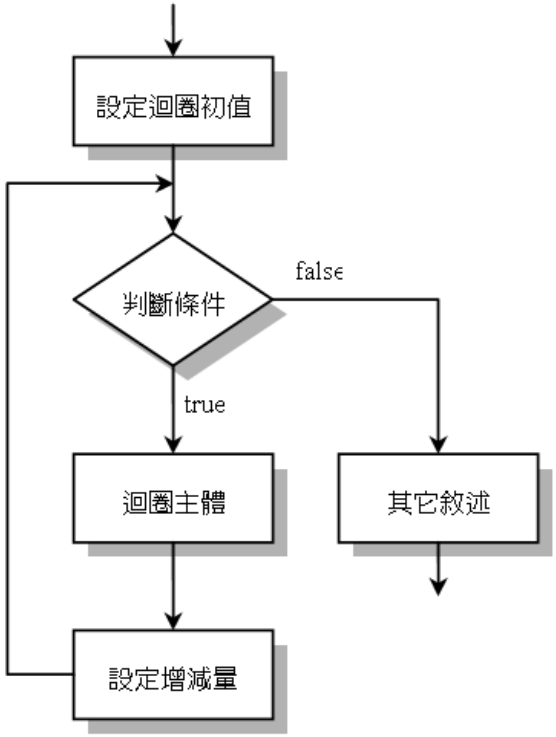
- for迴圈的格式及執行流程：

for迴圈敘述格式

```
for(設定迴圈初值; 判斷條件; 設定增減量)
{
    迴圈主體;
}
```

這兒不可以加分號

這兒不可以加分號



1. 第一次進入 for 迴圈時，設定迴圈控制變數的起始值。
2. 根據判斷條件的內容，檢查是否要繼續執行迴圈，當條件判斷值為真（true），繼續執行迴圈主體；條件判斷值為假（false），則跳出迴圈執行其它敘述。
3. 執行完迴圈主體內的敘述後，迴圈控制變數會根據增減量的設定，更改迴圈控制變數的值，再回到步驟 2 重新判斷是否繼續執行迴圈。

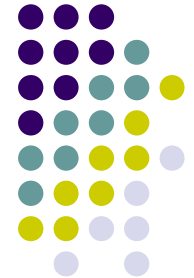
for迴圈 (2/2)

- 下面的程式利用for迴圈計算1+2+...+10：

```
01  // app5_3,for 迴圈
02  public class app5_3
03  {
04     public static void main(String args[])
05     {
06        int i,sum=0;
07
08        for(i=1;i<=10;i++)
09           sum+=i;    // 計算 sum=sum+i
10        System.out.println("1+2+...+10="+sum);  // 印出結果
11     }
12  }
```

```
/* app5_3 OUTPUT---

1+2+...+10=55
-----------------*/
```



for迴圈裡的區域變數

- 迴圈裡宣告的變數是區域變數（local variable），跳出迴圈，這個變數便不能再使用
- for迴圈裡的區域變數使用範例：

```
01  // app5_4, 區域變數
02  public class app5_4
03  {
04     public static void main(String args[])
05     {
06        int sum=0;
07
08        for(int i=1;i<=5;i++)     // 在迴圈內宣告變數 i
09        {
10           sum=sum+i;
11           System.out.println("i="+i+", sum="+sum);
12        }
13     }
14  }
```

```
/* app5_4 OUTPUT---
i=1, sum=1
i=2, sum=3
i=3, sum=6
i=4, sum=10
i=5, sum=15
-----------------*/
```

變數 i 的有效範圍

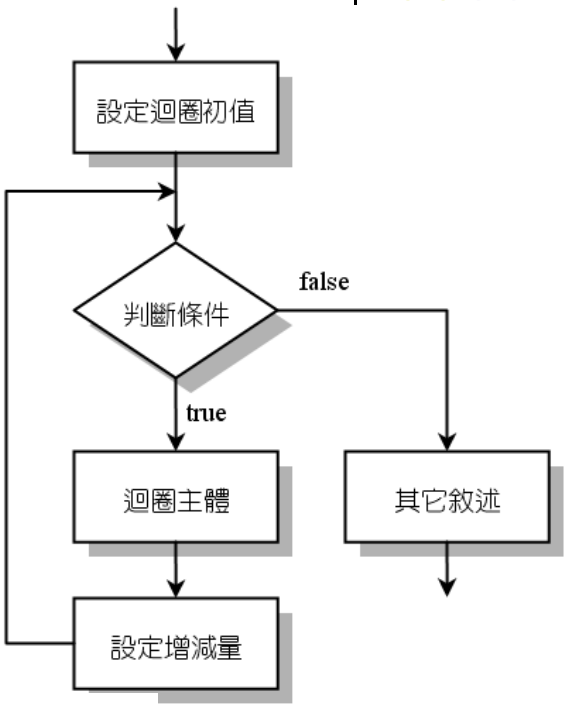
while 迴圈 (1/2)

while迴圈敘述格式

```
設定迴圈初值;
while(判斷條件)
{
    迴圈主體;
    設定增減量;
}
```

這兒不可以加分號

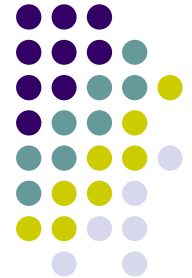
這兒不可以加分號



1. 第一次進入 **while** 迴圈前，就必須先設定迴圈控制變數的起始值。

2. 根據判斷條件的內容，檢查是否要繼續執行迴圈，如果條件判斷值為 **true**，則繼續執行迴圈主體；如果條件判斷值為 **false**，則跳出迴圈執行後續的敘述。

3. 執行完迴圈主體內的敘述後，重新設定（增加或減少）迴圈控制變數的值，由於 **while** 迴圈不會主動更改迴圈控制變數的內容，所以在 **while** 迴圈中，設定迴圈控制變數的工作要由我們自己來做，再回到步驟 2 重新判斷是否繼續執行迴圈。



while 迴圈 (2/2)

- 利用while迴圈計算1+2+...+10：

```
01  // app5_5, while 迴圈
02  public class app5_5
03  {
04     public static void main(String args[])
05     {
06        int i=1,sum=0;
07
08        while(i<=10)
09        {
10           sum+=i;                          // 累加計算
11           i++;
12        }
13        System.out.println("1+2+...+10="+sum);     // 印出結果
14     }
15  }
```

```
/* app5_5 OUTPUT---

1+2+...+10=55
-----------------*/
```

在程式設計的慣例上，會在確定迴圈次數時選擇for迴圈，而在不確定迴圈次數時選擇while迴圈，這樣的做法能讓語意更清楚的表達

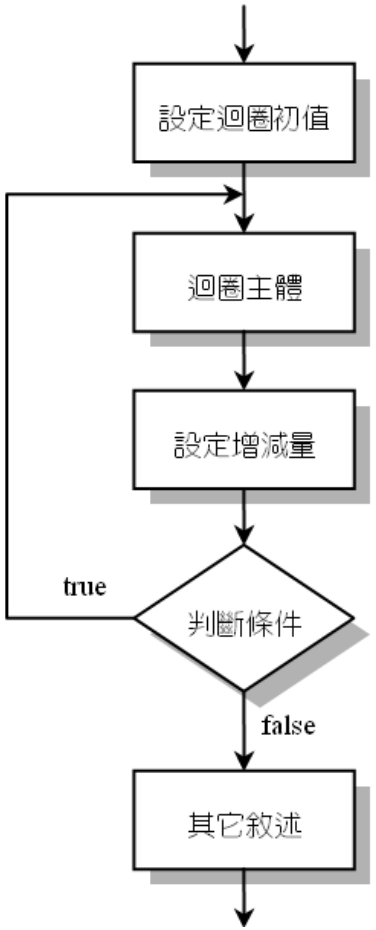
do while 迴圈 (1/2)

- do while是用於迴圈執行的次數未知時
- do while至少會執行1次迴圈主體

do while迴圈敘述格式

```
設定迴圈初值;
do
{
    迴圈主體;
    設定增減量;
} while(判斷條件);
```

要加分號



do while 迴圈 (2/2)

```
01  // app5_6, do while 迴圈
02  import java.io.*;
03  public class app5_6
04  {
05     public static void main(String args[]) throws IOException
06     {
07        int n,i=1,sum=0;
08        String str;
09        BufferedReader buf;
10
11        buf=new BufferedReader(new InputStreamReader(System.in));
12
13        do{
14           System.out.print("請輸入累加的最大值: ");
15           str=buf.readLine();
16           n=Integer.parseInt(str);
17        }while(n<1);          // 輸入 n,n 要大於等於 1,否則會一直重複輸入
18
19        do
20           sum+=i++;          // 計算 sum=sum+i,然後 i 值再加 1
21        while(i<=n);
22
23        System.out.println("1+2+...+"+n+"="+sum);       // 印出結果
24     }
25  }
```

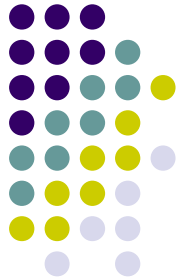
```
/* app5_6 OUTPUT-------
請輸入累加的最大值: -8
請輸入累加的最大值: 10
1+2+...+10=55
----------------------*/
```

巢狀迴圈 (nested loops)

- 迴圈敘述中又有其它迴圈敘述時，稱為巢狀迴圈
- 以列印部份的九九乘法表為例，練習巢狀迴圈：

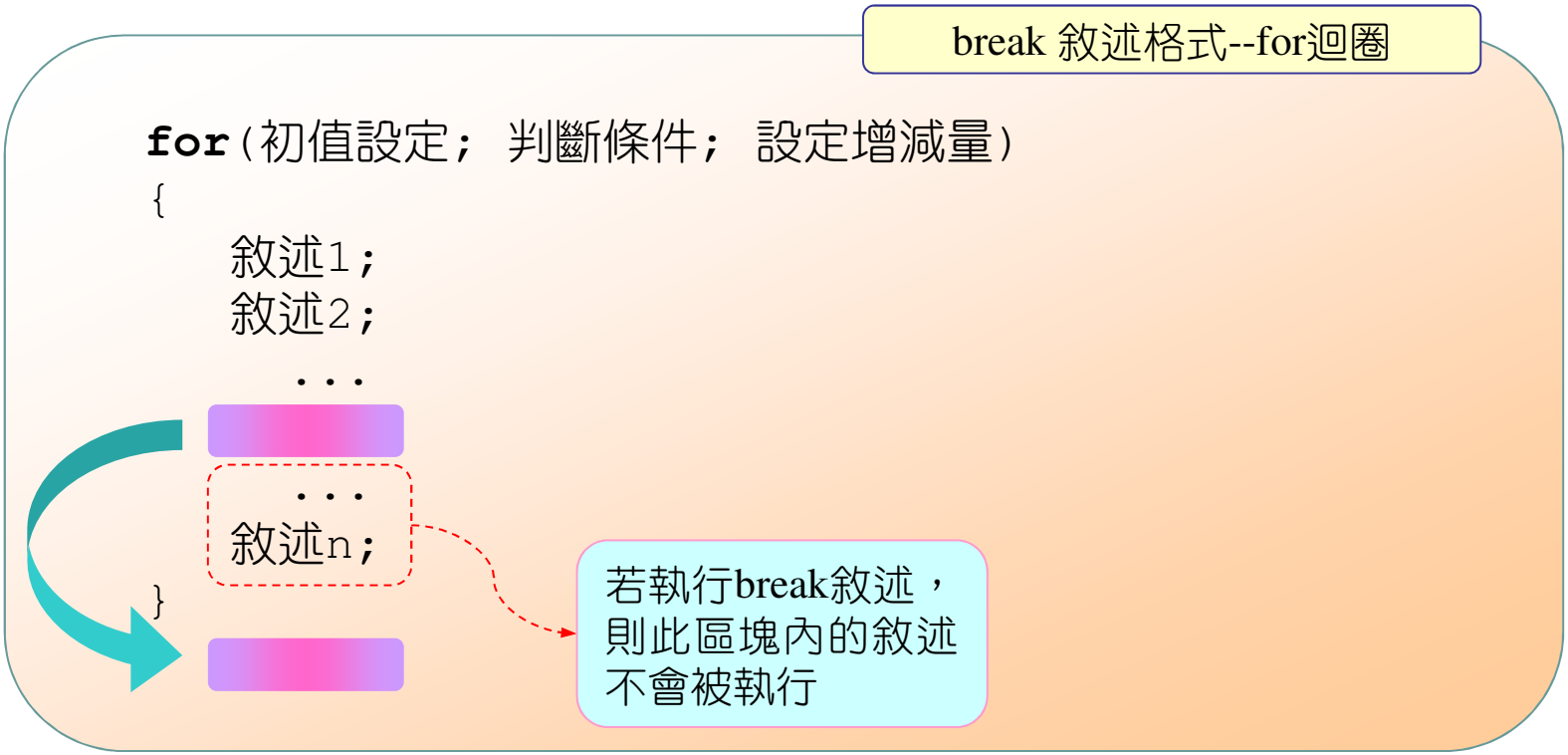
```
01  // app5_7, 巢狀 for 迴圈求 9*9 乘法表
02  public class app5_7
03  {
04     public static void main(String args[])
05     {
06        int i,j;
07
08        for (i=1;i<=3;i++)              // 外層迴圈
09        {
10           for (j=1;j<=3;j++)           // 內層迴圈
11              System.out.print(i+"*"+j+"="+(i*j)+"\t");
12           System.out.println();
13        }
14     }
15  }
```

```
/* app5_7 OUTPUT------
1*1=1   1*2=2   1*3=3
2*1=2   2*2=4   2*3=6
3*1=3   3*2=6   3*3=9
--------------------*/
```



break敘述 (1/2)

- break敘述格式：

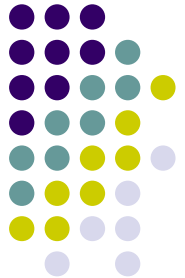


break敘述 (2/2)

- 在for迴圈中使用break敘述的範例：

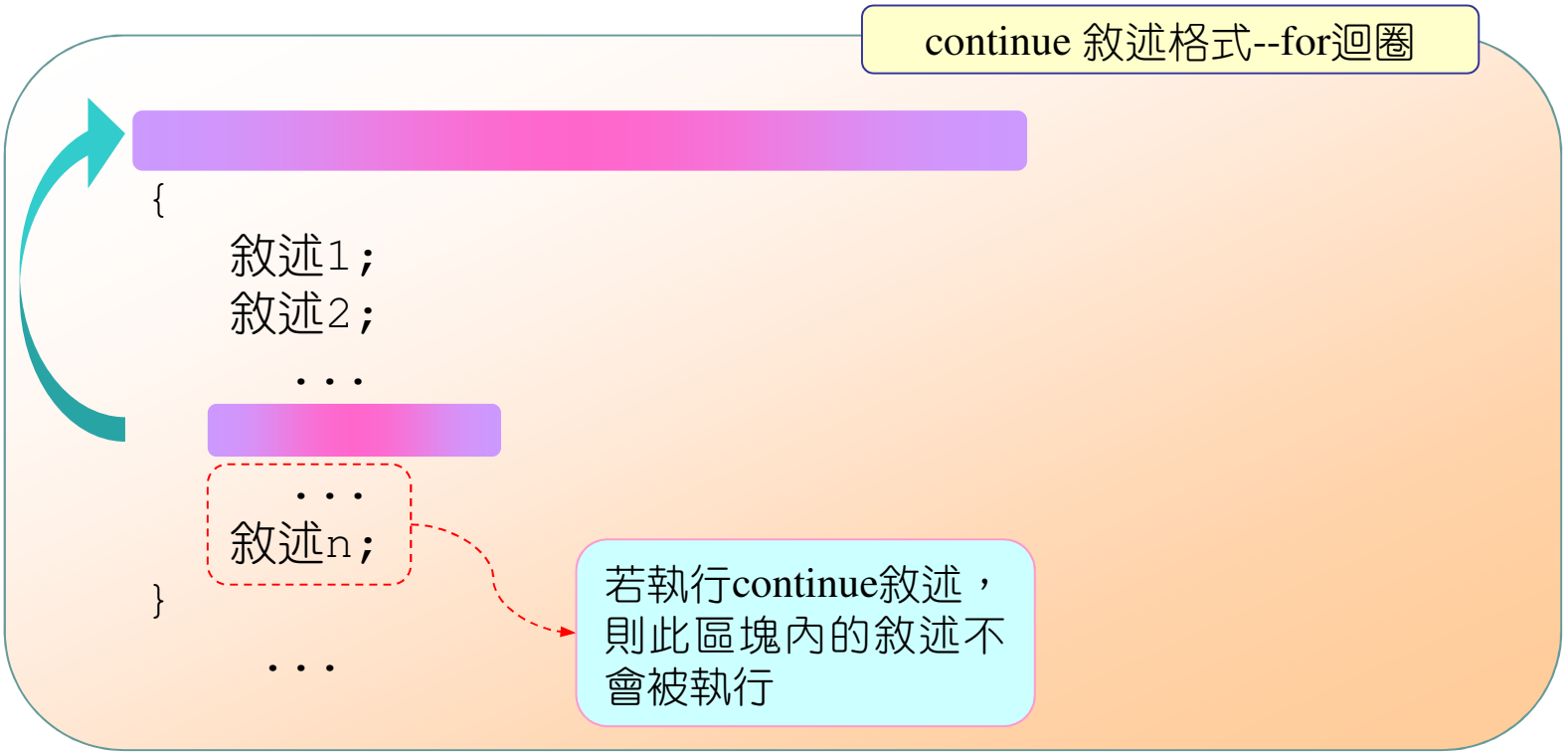
```
01  // app5_8, break 的使用
02  public class app5_8
03  {
04     public static void main(String args[])
05     {
06        int i;
07
08        for (i=1;i<=10;i++)
09        {
10           if(i%3==0)                          // 判斷 i%3 是否為 0
11              break;
12           System.out.println("i="+i);          // 印出 i 的值
13        }
14        System.out.println("when loop interruped,i="+i);
15     }
16  }
```

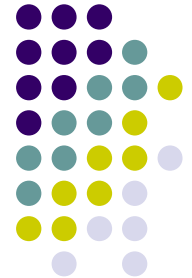
```
/* app5_8 OUTPUT----------
i=1
i=2
when loop interruped,i=3
-------------------------*/
```



continue敘述 (1/2)

- continue敘述會強迫程式跳到迴圈的起頭
- continue敘述的格式：





continue敘述 (2/2)

- 使用continue敘述的範例：

```
01  // app5_9, continue 的使用
02  public class app5_9
03  {
04     public static void main(String args[])
05     {
06        int i;
07
08        for (i=1;i<=10;i++)
09        {
10           if(i%3==0)                    // 判斷 i%3 是否為 0
11              continue;
12           System.out.println("i="+i);    // 印出 i 的值
13        }
14        System.out.println("when loop interruped,i="+i);
15     }
16  }
```

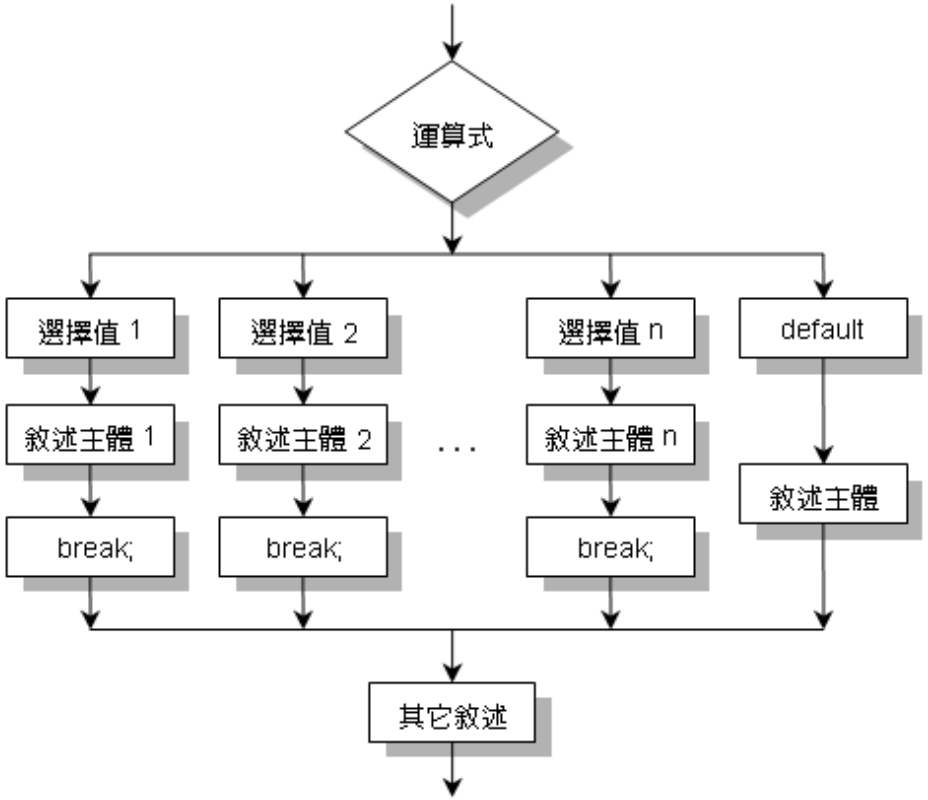
```
/* app5_9 OUTPUT----------
i=1
i=2
i=4
i=5
i=7
i=8
i=10
when loop interruped,i=11
-------------------------*/
```

switch敘述 (1/2)

- switch敘述可將多選一的情況簡化，格式如下：

switch 敘述格式

```
switch(運算式)
{
      case 選擇值1:
            敘述主體1;
            break;
      case 選擇值2:
            敘述主體2;
            break;
             ...
      case 選擇值n:
            敘述主體n;
            break;
      default:
             敘述主體;
   }
```



switch敘述 (2/2)

```
01  // app5_10, switch 敘述
02  public class app5_10
03  {
04     public static void main(String args[])
05     {
06        int a=50,b=20;
07        char oper='*';
08
09        switch (oper)
10        {
11           case '+':          // 印出 a+b
12              System.out.println(a+"+"+b+"="+(a+b));
13              break;
14           case '-':          // 印出 a-b
15              System.out.println(a+"-"+b+"="+(a-b));
16              break;
17           case '*':          // 印出 a*b
18              System.out.println(a+"*"+b+"="+(a*b));
19              break;
20           case '/':          // 印出 a/b
21              System.out.println(a+"/"+b+"="+((float)a/b));
22              break;
23           default:           // 印出字串
24              System.out.println("Unknown expression!!");
25        }
26     }
27  }
```

```
/* app5_10 OUTPUT---
50*20=1000
--------------------*/
```

如果沒有在case敘述結尾處加上break，則會一直執行到switch敘述的尾端，才會離開switch敘述，如此將造成執行結果的錯誤