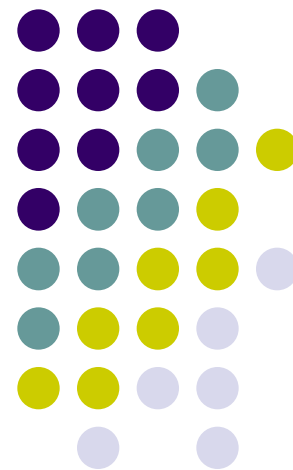


# 第五章

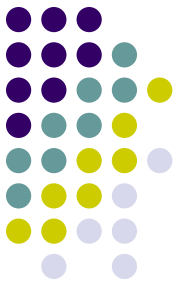
## 程式流程控制

認識程式的結構設計  
學習選擇性敘述的用法  
學習各種迴圈的用法  
學習迴圈的跳離

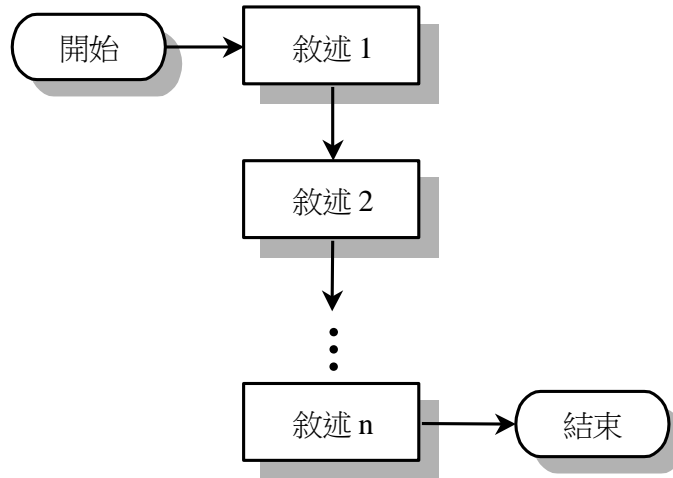


# 程式的結構

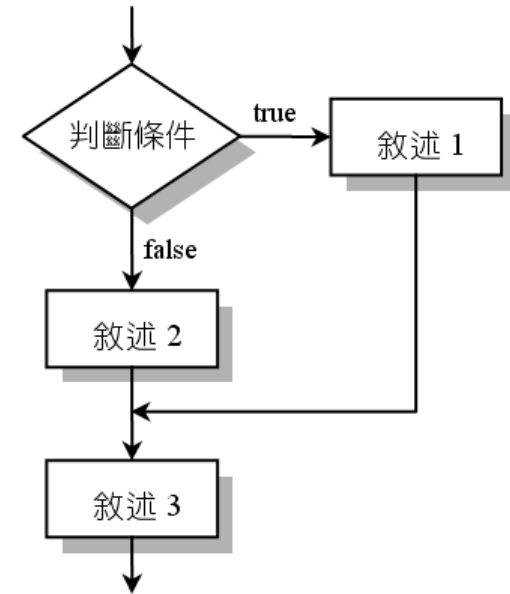
## 5.1 程式的結構設計



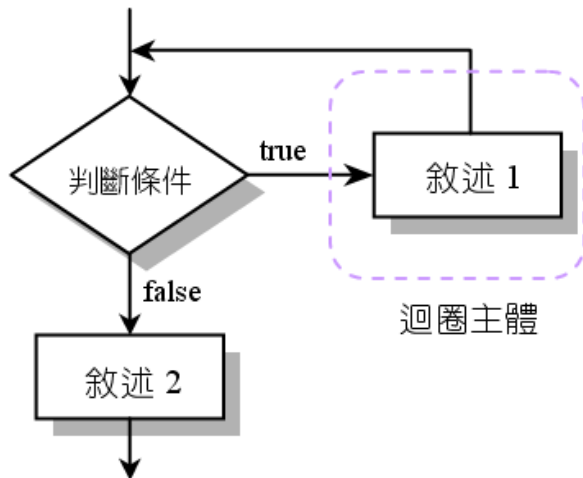
- 循序性結構 (sequence structure)



- 選擇性結構 (selection structure)



- 重複性結構 (iteration structure)



重複性結構有for、while及do while三種迴圈

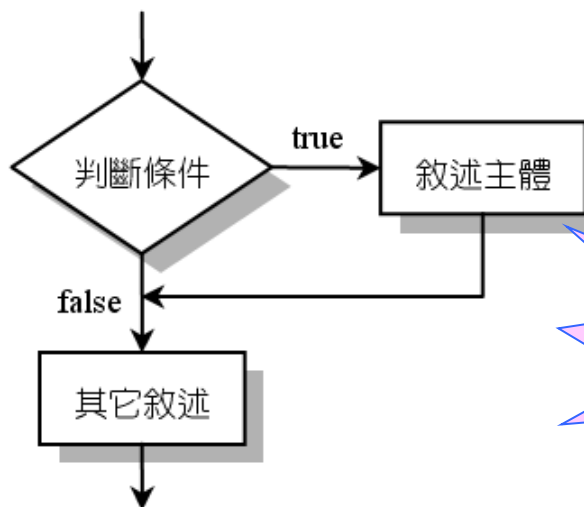


# if 敘述(1/2)

- 根據判斷的結果來執行不同的敘述

- if 敘述的語法

語法	說明
<pre>if(判斷條件){     敘述主體; }</pre>	如果判斷條件成立 (true)，則執行敘述主體。



if 敘述的流程圖



# if 敘述(2/2)

- if 敘述的練習

```
01 // Ch5_1, if 敘述的練習-判別是否為偶數
02 public class Ch5_1{
03     public static void main(String[] args){
04         int a=4;
05         if(a%2==0){        // 判別 a 除以 2 的餘數是否為 0
06             System.out.printf("%d is an even number",a); //若成立則執行這行
07         }
08     }
09 }
```

- 執行結果：

```
4 is an even number
```

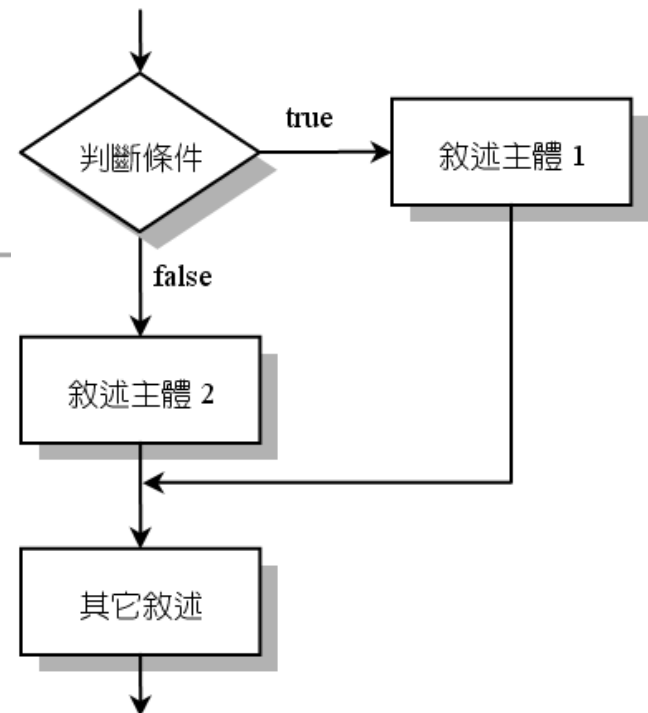


# if-else 敘述 (1/2)

- if-else敘述的格式與流程圖如下：

- if-else 敘述的語法

語法	說明
<pre>if(判斷條件){     敘述主體 1; } else{     敘述主體 2; }</pre>	如果判斷條件成立（結果為 true），則執行敘述主體 1，否則執行敘述主體 2。





## 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 // Ch5_2, if-else 敘述的練習-判別奇偶數
02 public class Ch5_2{
03     public static void main(String[] args){
04         int a=15;
05         if (a%2==0)        // 如果可被 2 整除
06             System.out.printf("%d is an even number",a); // 印出 a 為偶數
07         else
08             System.out.printf("%d is an odd number",a);  // 印出 a 為奇數
09     }
10 }
```

- 執行結果：

```
15 is an odd number
```



# 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

- ?: 的語法格式：

- 條件運算子的語法

語法	說明
變數 = 判斷條件 ? 運算式 1 : 運算式 2;	若判斷條件成立，則傳回運算式 1 的結果，否則傳回運算式 2 的結果



## 條件運算子 (2/2)

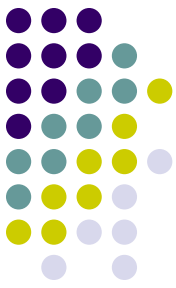
- 下面的程式可找出二數之間較大的數：

```
01 // Ch5_3, 條件運算子?:的使用-找出較大的數
02 public class Ch5_3{
03     public static void main(String[] args){
04         int a=8,b=3,max;
05
06         max=(a>b)?a:b;           // a>b 時,max=a,否則 max=b
07         System.out.printf("a=%d, b=%d, %d 是較大的數\n",a,b,max);
08     }
09 }
```

- 執行結果：

a=8, b=3, 8 是較大的數





# 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)

- 巢狀 if 敘述的語法

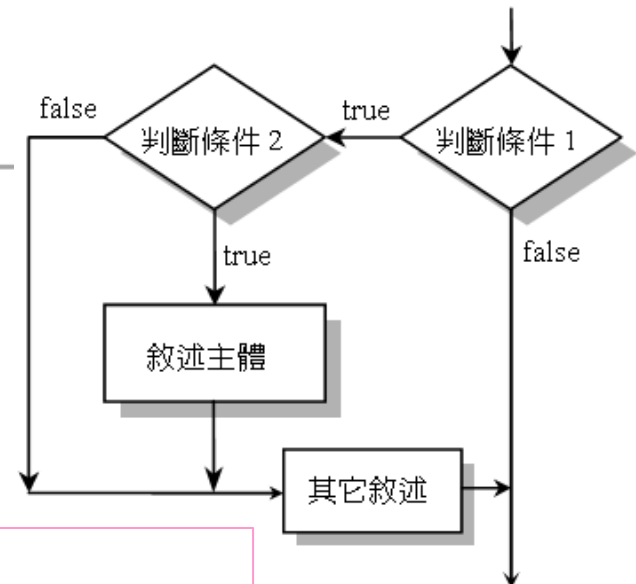
語法	說明
<pre> if(判斷條件1){     if(判斷條件2){         敘述主體;     }     其它敘述; } </pre>	<p>如果外層 if 敘述裡的判斷條件 1 成立，則進到內層的 if 敘述，若判斷條件 2 也成立，則執行敘述主體。無論判斷條件 2 是否成立，只要判斷條件 1 成立，其它敘述就會被執行。</p>

- 巢狀 if 敘述的使用範例：

```

01  if(num>0)
02      if(num%2==0)
03          System.out.printf("%d 是大於 0 的偶數",num);

```



沒有else的情況下，  
可用邏輯運算子改寫  
成單一if 敘述

```

01  if (num > 0 && num % 2 == 0){
02      System.out.printf("%d 是大於 0 的偶數", num);
03  }

```

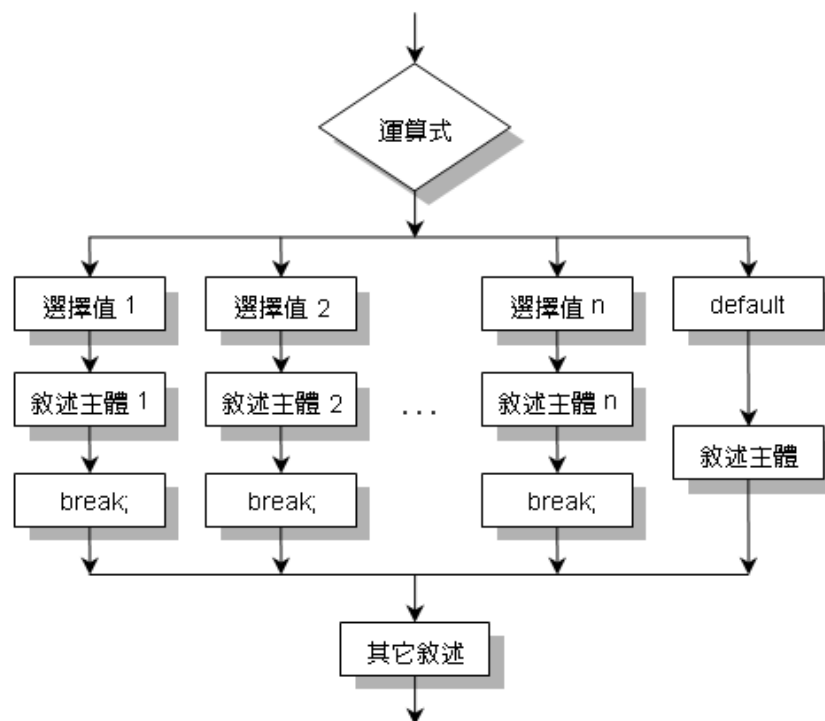
# switch敘述 (1/2)

- switch敘述可將多選一的情況簡化，格式如下：

- switch 敘述的語法

語法	說明
<pre>switch(運算式){   case 選擇值 1:     敘述主體 1;     break;     ...   case 選擇值 n:     敘述主體 n;     break;   default:     預設的敘述主體; }</pre>	由運算式算出一個選擇值，然後根據算出的選擇值執行相對應的敘述主體。如果算出的選擇值和列出的選擇值都不符合，則執行預設的敘述主體。

可以是字元、  
字串或是整數





# switch敘述 (2/2)

```
01 // Ch5_4, switch 敘述-根據選擇值來進行加法或減法計算
02 public class Ch5_4{
03     public static void main(String[] args){
04         int a=50,b=20;
05         char oper='+';
06
07         switch(oper){
08             case '+':          // 選擇值為 '+'
09                 System.out.println(a+"+"+b+"="+ (a+b)); // 印出 a+b
10                 break;
11             case '-':          // 選擇值為 '-'
12                 System.out.println(a+"-"+b+"="+ (a-b)); // 印出 a-b
13                 break;
14             default:           // 沒有相對應的選擇值
15                 System.out.println("Unknown expression!!"); // 印出字串
16         }
17     }
18 }
```

• 執行結果：

50+20=70


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



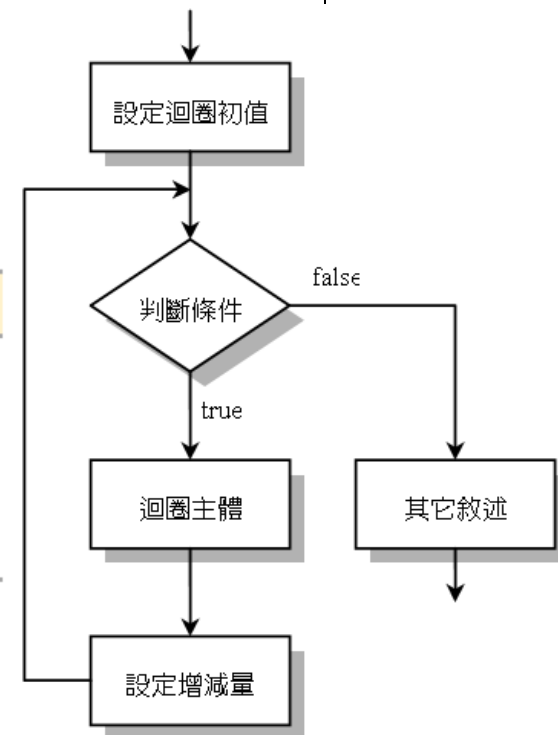
# for迴圈 (1/2)

- for迴圈的格式及執行流程：

- for 迴圈的語法

語法	說明
<pre>for(迴圈初值; 判斷條件; 設定增減量){     迴圈主體; }</pre>	根據迴圈初值、判斷條件與增減量的設定來決定迴圈主體的執行方式
	

- 第一次進入 for 迴圈時，設定迴圈控制變數的起始值。
- 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。
- 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 $1+2+\dots+10$ ：

```
01 // Ch5_5, 利用 for 迴圈計算 1 加到 10 的總和
02 public class Ch5_5{
03     public static void main(String[] args){
04         int i,sum=0;
05
06         for(i=1;i<=10;i++)
07             sum+=i;    // 計算 sum=sum+i
08         System.out.printf("1+2+...+10=%d",sum); // 印出結果
09     }
10 }
```

- 執行結果：

1+2+...+10=55



# 在VSCode裡偵錯迴圈 (1/4)

- 啟動偵錯模組：

The screenshot shows the VS Code interface with a Java file 'Ch5\_5.java' open. A breakpoint is set at line 4. The 'Run' menu is open, showing options like '啟動偵錯' (Start Debugging) and 'F5'. A red dashed box highlights '啟動偵錯'. A red arrow points from 'F5' in the menu to a callout box that says '也可以直接按下F5' (You can also press F5 directly). Another red arrow points from a hand icon to the breakpoint on line 4, with a callout box that says '1. 點選這裡設置中斷點' (1. Click here to set a breakpoint). A second callout box says '2. 選擇「執行」功能表 - 「啟動偵錯」' (2. Select the 'Run' menu - 'Start Debugging').

1. 點選這裡設置中斷點







2. 選擇「執行」功能表 - 「啟動偵錯」

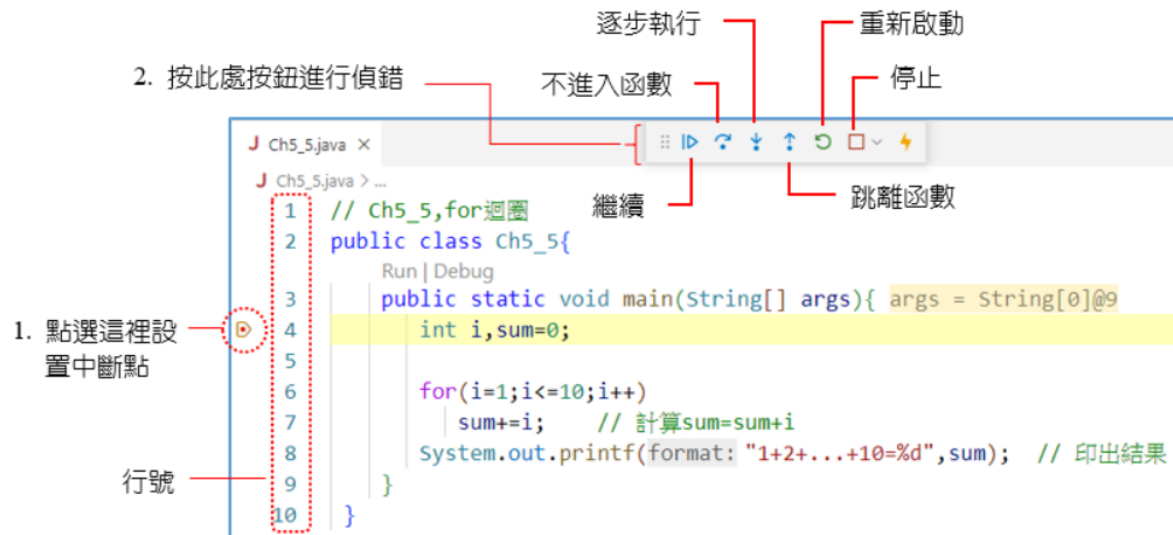
也可以直接按下F5



# 在VSCode裡偵錯迴圈 (2/4)


## ● 偵錯模組的功能說明：


1.  繼續：繼續執行到下一個中斷點(在行號左側空白區點選滑鼠即可增加中斷點)。
2.  不進入函式：一行一行執行，若該行有一個函數，會直接執行完該函數，不進入函數進行逐步偵錯。
3.  逐步執行：一行一行執行，若該行有一個函數，會跳到該函數裡執行。
4.  跳離函式：若現在正在一個函數裡執行，則立即執行完該函數然後跳離。
5.  重新啟動：重新啟動偵錯模組。
6.  停止：退出偵錯模組。





# 在VSCode裡偵錯迴圈 (3/4)

- Ch5\_5按了二下  的畫面：

目前正要執行這行 — 

```
1 // Ch5_5,for迴圈
2 public class Ch5_5{
3     Run | Debug
4     public static void main(String[] args){ args = String[0]@9
5         int i,sum=0; i = 1, sum = 0
6         for(i=1;i<=10;i++) i = 1
7         sum+=i; // 計算sum=sum+i sum = 0, i = 1
8         System.out.printf(format: "1+2+...+10=%d",sum); // 印出結果
9     }
10 }
```

目前變數的值

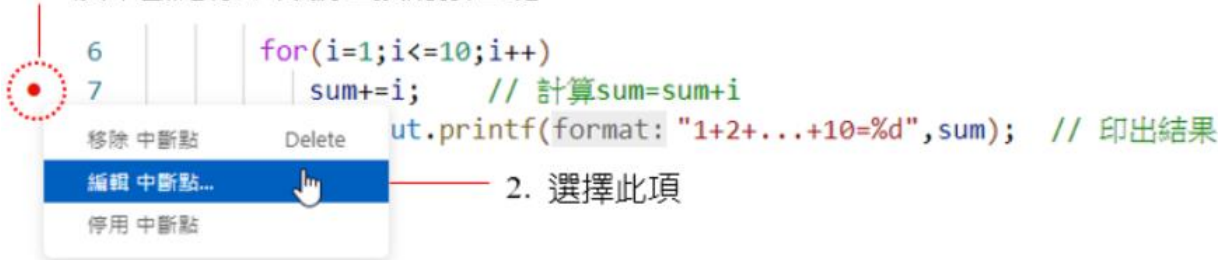




# 在VSCode裡偵錯迴圈 (4/4)

## ● 偵錯過程：

1. 於中斷點標示符號上按滑鼠右鍵



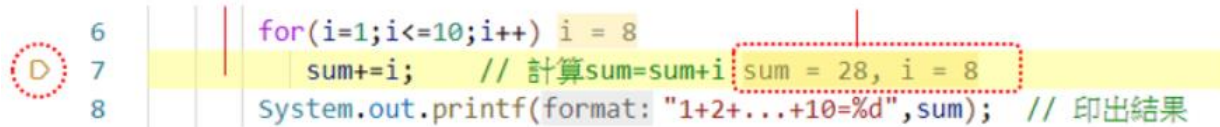
2. 選擇此項



3. 鍵入 `i==8`，然後按下 Enter 鍵

正要執行第 7 行

目前 sum 的值為 28，i 的值為 8





# for迴圈裡的區域變數(1/2)

- 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 // Ch5_6, for 迴圈裡的區域變數
02 public class Ch5_6{
03     public static void main(String[] args){
04         int sum=0;
05
06         for(int i=1;i<=4;i++){    // 在迴圈內宣告變數 i
07             sum=sum+i;
08             System.out.printf("i=%2d, sum=%2d\n",i,sum);
09         }
10     }
11 }
```

變數 i 的有效範圍

• 執行結果：

```
i= 1, sum= 1
i= 2, sum= 3
i= 3, sum= 6
i= 4, sum=10
```



## for迴圈裡的區域變數(2/2)

- 在for迴圈外試著列印i值：

```
6   for(int i=1;i<=4;i++){
7       sum=sum+i;
8       System.out.printf(format: "i=%2d, sum=%2d\n",i,sum);
9   }
10  System.out.printf(format: "i=%2d\n",i);
11  }
12 }
```

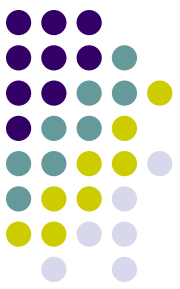
在 for 迴圈外面列印 i 的值

問題 1 輸出 偵錯主控台 終端機

Ch5\_6.java 1

⊗ i cannot be resolved to a variable Java(33554515) [第 10 行, 第 37 欄]

無法找到 i



# for迴圈裡的迴圈初值之設定

- 正確的「設定迴圈初值」方式

```
for(int i=0,j=0; i+j<10; i++,j+=2){           // 正確的迴圈初值設定方式
    迴圈主體
}
```

- 錯誤的「設定迴圈初值」方式

```
for(int i=0,int j=0; i+j<10; i++,j+=2){      // 錯誤，關鍵字 int 只能出現一次
```

```
for(i=0,int j=0; i+j<10; i++,j+=2){         // 錯誤，int 要寫在第 1 個宣告變數之前
```

```
for(int i=0,short j=0; i+j<10; i++,j+=2){   // 錯誤，i 和 j 的型別必須相同
```



# 無窮迴圈

- 當迴圈控制變數使用不當，就有可能導致無窮迴圈
- 省略判斷條件、不設定增減量、或是判斷條件永不成立的情況都將造成無窮迴圈

```
for(int i=1;     ; i++){  
    // 迴圈主體;  
}
```

└─ 省略判斷條件（或是填上 true）

按下Ctrl+C強  
制中斷程式



# while 迴圈 (1/2)

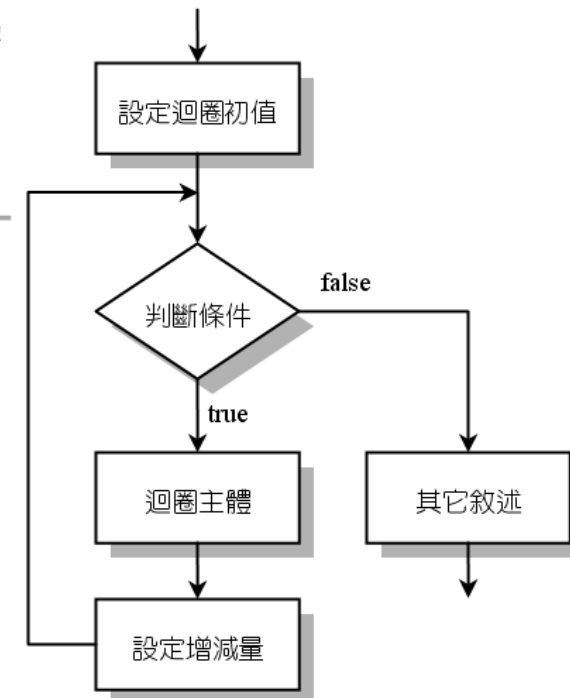
## • while 迴圈的語法

語法	說明
設定迴圈初值; <code>while(判斷條件){</code> 迴圈主體; <code>}</code>	當判斷條件成立時，就執行迴圈的主體，直到判斷條件不成立為止。

這兒不可以加分號

使用 while 迴圈時應注意的事項：

1. 第一次進入 while 迴圈前，就必須先設定迴圈控制變數的初值。
2. 在迴圈主體內應包含有一些敘述，在某些情況下會使得判斷條件不成立，如此才能跳離迴圈，否則將造成無窮迴圈。





# while 迴圈 (2/2)

- 當sum>20即跳離迴圈：

```
01 // Ch5_7, while 迴圈-當 sum>20 時就跳離迴圈
02 public class Ch5_7{
03     public static void main(String[] args){
04         int n=1,sum=0;
05         while(sum<=20){
06             sum+=n;    // 累加計算
07             System.out.printf("n=%d, sum=%2d\n",n, sum);
08             n++;       // 將 n 值加 1
09         }
10     }
11 }
```

- 執行結果：

```
n=1, sum= 1
n=2, sum= 3
n=3, sum= 6
n=4, sum=10
n=5, sum=15
n=6, sum=21
```

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，這樣的作法能讓語意更清楚的表達

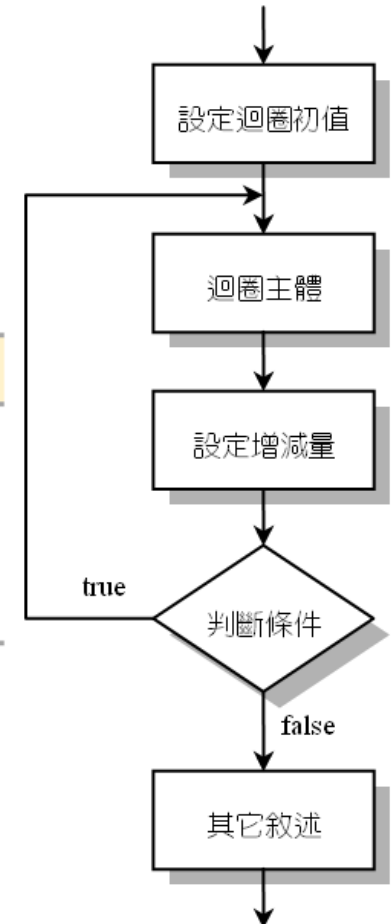


# do while 迴圈 (1/2)

- 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 迴圈的語法

語法	說明
設定迴圈初值; do{ 迴圈主體; }while(判斷條件) ; — 這裡要加分號	在設定迴圈初值後，先執行迴圈的主體，然後根據 while 裡的判斷條件決定迴圈是否繼續執行。







## do while 迴圈 (2/2)

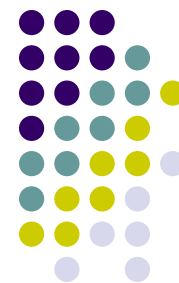
```
01 // Ch5_8, do-while 迴圈-累加 1 至 n 的程式
02 import java.util.Scanner;
03 public class Ch5_8{
04     public static void main(String[] args){
05         Scanner scn=new Scanner(System.in);
06         int n,sum=0;
07
08         do{
09             System.out.print("請輸入累加的最大值: ");
10             n=scn.nextInt();    // 輸入一個整數
11         }while(n<1);           // n 要大於等於 1,否則會要求重複輸入
12
13         for(int i=1;i<=n;i++)
14             sum+=i;           // 計算 sum=sum+i
15         System.out.printf("1+2+...+%d=%d\n",n,sum);    // 印出結果
16         scn.close();
17     }
18 }
```

• 執行結果：

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 // Ch5_9, 巢狀 for 迴圈求 9*9 乘法表
02 public class Ch5_9{
03     public static void main(String[] args){
04         for (int i=1;i<=3;i++){           // 外層迴圈
05             for (int j=1;j<=4;j++)        // 內層迴圈
06                 System.out.printf("%d*%d=%2d  ",i,j,i*j);
07             System.out.println();         // 換行
08         }
09     }
10 }
```

• 執行結果：

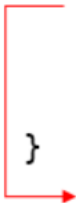
```
1*1= 1   1*2= 2   1*3= 3   1*4= 4
2*1= 2   2*2= 4   2*3= 6   2*4= 8
3*1= 3   3*2= 6   3*3= 9   3*4=12
```



# break敘述 (1/2)

- break敘述格式：

- break 敘述的語法

語法	說明
<pre>for(初值設定; 判斷條件; 設定增減量){     敘述 1;     ...     break;     ...     敘述 n; }</pre> 	<p>在迴圈內執行到 <b>break</b> 敘述時，程式會跳離當層迴圈的主體，繼續執行當層迴圈外面的敘述。</p> <p>} 此區塊內的敘述 不會被執行</p>



## break敘述 (2/2)

- 在for迴圈中使用break敘述的範例：

```
01 // Ch5_10, break 的使用-當 i%3==0 時跳離迴圈
02 public class Ch5_10{
03     public static void main(String[] args){
04         int i;
05         for (i=1;i<=10;i++){
06             if(i%3==0)                // 判斷 i%3 是否為 0
07                 break;
08             System.out.println("i="+i);    // 印出 i 的值
09         }
10         System.out.println("when loop interrupted, i="+i);
11     }
12 }
```

- 執行結果：

i=1

i=2


when loop interrupted, 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 敘述的語法

語法	說明
<pre>for(初值設定; 判斷條件; 設定增減量){     敘述 1;     ...     continue;     ...     敘述 n; }</pre> 	<p>當程式執行到迴圈主體中的 continue 敘述時，會跳過 continue 敘述後面未執行的部分，然後回到迴圈的起點繼續執行剩餘的迴圈。</p> <p>此區塊內的敘述 不會被執行</p>



## continue敘述 (2/2)

- 使用continue敘述的範例：

```
01 // Ch5_11, continue 的使用-回到迴圈起點，繼續執行剩餘的部分
02 public class Ch5_11{
03     public static void main(String[] args){
04         for (int i=1;i<=10;i++){
05             if(i%3==0)                // 判斷 i%3 是否為 0
06                 continue;
07             System.out.printf("%3d",i);    // 印出 i 的值
08         }
09     }
10 }
```

- 執行結果：

1 2 4 5 7 8 10



-The End-