J01-09

使用 Loop Constructs (重複結構)

曾瑞君 (Jim_Tzeng)

學習目標

- 使用 while 迴圈
- 使用 for 迴圈
- 使用 nested loop (巢狀迴圈)
- 使用 for 迴圈存取陣列
- 使用 do/while 迴圈
- 比較迴圈結構

1/6

使用 While 迴圈

Loops (迴圈)

- 程式碼中使用特定條件 (expression)，滿足時即可重複某些行為 (code block)。
- 有 3 種主要型態：

1) while loop

   滿足 expression = true 將持續進行

2) do/while loop

   執行一次後，滿足 expression = true 將持續進行

3) for loop

   重複特定次數

Repeating Behavior

```
while (!doesTheRainStop) {
        walk back and forth;
        ask, " does the rain stop?";
}
Ya!;
Get out of door;
```

Creating while Loops

- 語法：

```
while (boolean_expression) {
    code_block;
} // 迴圈結束
// 程式結束迴圈後，繼續其他
```

while Loop in Elevator

- 指定目標電梯樓層後，反覆 up()、down() 以抵達目標樓層。

```
public void toFloor(int targetFloor) {
    while ( currentFloor != targetFloor ){
        if (currentFloor < targetFloor) {
            up();
        } else {
            down();
        }
    }
}
```

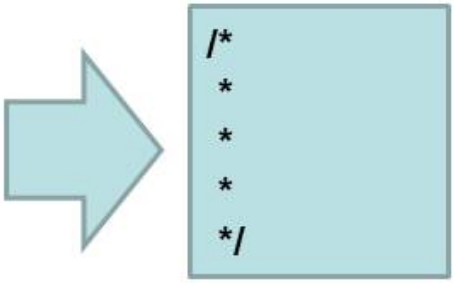
複利年息計算 (何時倍增?)

```
public static void main(String[] args) {
    double money = 1000;
    double interest = 0.18;                    // 年 18%
    int years = 0;
    while (money <= 2000) {                    // 本金倍增後停止
        money += money * interest;             // 複利計息
        years++;
        System.out.println("Year " + years + ": " + money);
    }
}
```

```
Year 1: 1180.0
Year 2: 1392.4
Year 3: 1643.0320000000002
Year 4: 1938.7777600000002
Year 5: 2287.7577568
```

輸出 comment /**/ 區塊

```
public static void main(String[] args) {
    System.out.println("/*");
    int i = 0;
    while (i < 3) {
        System.out.println(" *");
        i++;
    }
    System.out.println(" */");
}
```



2/6

使用 For 迴圈

Developing a for Loop

- 語法：

```
for (initialize[,initialize]; boolean_expression; update[,update]) {
    code_block;
}
```

```
public static void main(String[] args) {
    for (        String i = "$", t = "~";
                 i.length() < 5;
                 i += "$", t += "~") {
        System.out.println(i + t);
    }
}
```

```
$~
$$~~
$$$~~~
$$$$~~~~
```

for Loop

迴圈組成要素：

1) 初始條件

2) 變動條件

3) 滿足條件

```
int i = 0;
while ( i < 7 ) {
        System.out.println("$");
        i ++;
}
```

```
for ( int i = 0 ;  i < 7 ; i ++ ) {
        System.out.println("$");
}
```

3/6

使用 Nested Loop (巢狀迴圈)

Nested for Loop

- For loop 裡還有 for loop：

```
public static void main(String[] args) {
    int num = 5;
    for (int i = 0; i < num; i++) {
        for (int j = 0; j <= i; j++) {
            System.out.print('*');
        }
        System.out.println();
    }
}
```

```
*
**
***
****
*****
```

由隨機浮點數轉換為隨機大寫字母

擴充目標	擴充後的不等式				
原始隨機函數	0	<=	Math.random()	<	1
全部乘上26	(0*26)	<=	Math.random()*26	<	(1*26)
全部加上65	0+65	<=	Math.random()*26+65	<	26+65
轉換為隨機字元	A	<=	(char)(Math.random()*26+65)	<=	Z

Nested while Loop (猜字串)

```
public static void main(String[] args) {
    String yuName = "jim";
    String guessYuName = "";
    int tryCounts = 0;
    while (!guessYuName.equals(yuName.toUpperCase())) {   // 字串不同時繼續
        guessYuName = "";
        while (guessYuName.length() < yuName.length()) {    // 隨機組出長度相同字串
            char c = (char) (Math.random() * 26 + 65);
            guessYuName = guessYuName + c;
        }
        //System.out.println(guessYuName);            // 印出隨機組成的字串
        tryCounts++;
    }
    System.out.println(yuName + " was found!!");
    System.out.println("After " + tryCounts + " tries!!");
}
```

4/6

使用 for 迴圈存取陣列

Setting/getting values in an Array

```
public static void main(String[] args) {
    long[] longArray = new long[9];

    // 設定陣列內容
    for (int i = 0; i < longArray.length; i++) {
        longArray[i] = Math.round(Math.random() * 100);
    }
    // 讀取陣列內容
    for (int i = 0; i < longArray.length; i++) {
        System.out.println(i + ": " + longArray[i]);
    }
}
```

```
0: 50
1: 26
2: 98
3: 15
4: 46
5: 56
6: 25
7: 13
8: 76
```

Enhanced for Loop with Arrays

- Enhanced for Loop (forEach) 除 Array 外，也可用於 ArrayList

```
public static void main(String[] args) {

    int[] intArray = { 12, 23, 45, 3, 67, 34, 87, 96, 89 };
    for (int element : intArray) {
        System.out.println(element);
    }

    String[] names = {"jim", "bill", "albert", "sue", "mary", "elsa"};
    for (String name : names) {
        System.out.println(name);
    }
}
```

```
public static void enhancedLoopArrayList() {
	ArrayList names = new ArrayList();
	names.add("jim");
	names.add("bill");
	names.add("albert");
	names.add("sue");
	names.add("mary");
	names.add("elsa");

	for (Object name : names) {
		System.out.println(name);
	}
}
```

Using break with Loops

- 使用 break 敘述結束 loop：

```
public static void main(String[] args) {
    int passScore = 60;
    int[] scores = { 40, 36, 52, 58, 65, 34, 93 };
    int passAt = 0;
    for (int s : scores) {
        passAt ++;
        if (s > passScore) {
            break;
        }
    }
    System.out.println("Finally pass at: " + passAt);
}
```

```
Finally pass at: 5
```

Using continue with Loops

- 使用 continue 敘述回到 loop 內的起始點

```
public static void main(String[] args) {
    int passScore = 60;
    int[] scores = { 40, 36, 52, 58, 65, 34, 93 };
    for (int s : scores) {
        if (s > passScore)
            continue;
        System.out.println("the score: " + s + " is failed to pass.");
    }
}
```

```
the score: 40 is failed to pass.
the score: 36 is failed to pass.
the score: 52 is failed to pass.
the score: 58 is failed to pass.
the score: 34 is failed to pass.
```

5/6

使用 do/while 迴圈

使用 do/while 迴圈

- 語法：

```
do {
    code_block;
} while (boolean_expression) ; // 注意結尾加上；
```

- 特色：至少執行一次。又稱“後測式迴圈”

```
public static void main(String[] args) {
    int count = 0;
    do {
        System.out.println("DoWhile Count is: " + count);
    } while (count < 0);
}
```

6/6

比較迴圈結構

迴圈比較

迴圈種類	執行次數
while	執行 0 到 多次
do/while	執行 1 到 多次
for	執行事先定義的次數

END ~~

Thank you!!