Java 程式設計 作業與範例總整理

共 42 支程式，用 Notepad++ 撰寫，在命令提示字元（CMD）編譯執行，更新日期 2026-11-07

第0章 環境建置：Notepad++ 撰寫、CMD 執行

課程規定用 Notepad++ 撰寫 Java 程式，再用命令提示字元（CMD）編譯並執行。第一次使用前，照下面的步驟建置環境；之後每次寫程式只要做步驟 4～6。

1 下載並安裝 JDK

JDK（Java Development Kit）提供編譯用的 `javac` 和執行用的 `java`。本專案使用 Eclipse Adoptium 的 Temurin JDK 25（LTS 長期支援版）。

- 到 https://adoptium.net/ 下載 Windows x64 的 `.msi` 安裝檔。
- 執行安裝檔，在「Custom Setup」畫面確認「Add to PATH」與「Set JAVA_HOME variable」都是「Will be installed on local hard drive」。
- 其餘照預設，按 Next 完成安裝。預設安裝位置是 `C:\Program Files\Eclipse Adoptium\jdk-25...\bin`。

---

2 確認安裝與環境變數 Path

安裝完後要重新開啟 CMD（已經開著的視窗讀不到新的 Path）。按 `Win + R`，輸入 `cmd` 按 Enter，然後輸入：

```
java -version
javac -version
where java
```

- 看到 `openjdk version "25..."` 和 `javac 25...` 就代表安裝成功。
- `where java` 會列出找到的 java 位置；有好幾個 JDK 時，**排在第一個的會被使用**。

> 如果出現「'javac' 不是內部或外部命令」，代表 Path 沒設定好：開啟「設定 → 系統 → 系統資訊 → 進階系統設定 → 環境變數」，在「系統變數」選 Path → 編輯 → 新增 JDK 的 bin 資料夾（例如 `C:\Program Files\Eclipse Adoptium\jdk-25.0.4.101-hotspot\bin`），按確定後重新開啟 CMD。

3 安裝 Notepad++，建立程式資料夾

- 到 https://notepad-plus-plus.org/downloads/ 下載安裝 Notepad++。
- 建立放程式的資料夾 `C:\java`（本課程的程式都放在這裡，CMD 路徑比較好打）。

```
mkdir C:\java
```

4 用 Notepad++ 撰寫程式

- 「檔案 → 開新檔案」。
- 「語言 → J → Java」：程式碼會有顏色，比較好檢查。
- 「編碼 → UTF-8」：下方狀態列要顯示 `UTF-8`，不要選 `UTF-8-BOM`。
- 輸入程式，例如下面的 Hello.java。
- 「檔案 → 另存新檔」：存到 `C:\java`，**檔名必須和 `public class` 後面的名稱完全相同（大小寫也要一樣）**，副檔名是 `.java`，例如 `Hello.java`。
- 之後修改程式，按 `Ctrl + S` 存檔即可。

```
public class Hello {
    public static void main(String[] args) {
        System.out.println("Hello, Java!");
        System.out.println("我的第一支 Java 程式");
    }
}
```

5 開啟 CMD，切換到程式資料夾

- 按 `Win + R`，輸入 `cmd`，按 Enter。
- 用 `cd /d` 切換到 `C:\java`（`/d` 可以連磁碟機一起切換），提示文字會變成 `C:\java>`。
- 用 `dir` 確認程式檔有在資料夾裡。

```
cd /d C:\java
dir
```

6 編譯與執行（沒有 package 的程式）

`javac` 編譯，成功時**不會有任何訊息**，並產生 `Hello.class`；`java` 執行，後面接類別名稱，**不加 .java 也不加 .class**。

```
javac -encoding UTF-8 Hello.java
java Hello
```

- 執行結果會直接顯示在 CMD：`Hello, Java!`
- `-encoding UTF-8` 讓 javac 正確讀取程式裡的中文。
- 每次修改程式都要重做：存檔 → `javac` → `java`。在 CMD 按 `↑` 方向鍵可以叫出上一個指令。

7 編譯與執行（有 package 的程式）

課本範例和第 3～6 章作業的程式第一行有 `package myJava.ch04;` 之類的宣告，編譯要加 `-d .`，執行要打完整名稱（package 名稱 + 類別名稱）。

```
javac -encoding UTF-8 -d . Season.java
java myJava.ch04.Season
```

- `-d .` 會依照 package 名稱自動建立資料夾，`.class` 檔會放在 `C:\java\myJava\ch04\` 裡。
- 只打 `java Season` 會出現「找不到或無法載入主要類別」。

8 鍵盤輸入、命令列引數與中文顯示

- 用 `Console` 讀鍵盤輸入的程式（例如 Triangle、PhoneBook）執行後會停下來等你打字，每輸入一項按 Enter。必須在**真正的 CMD 視窗**執行，`System.console()` 才不會是 null。
- 需要命令列引數的程式，引數接在類別名稱後面、用空白隔開，例如 `java ex6_03 46 10`，程式裡用 `args[0]`、`args[1]` 讀取。
- 中文變成亂碼時，先執行 `chcp 65001`（把 CMD 切換成 UTF-8），再執行程式。

```
chcp 65001
java ex6_03 46 10
```

9 常見錯誤

錯誤訊息	原因與解法
`'javac' 不是內部或外部命令`	Path 沒有 JDK 的 bin 資料夾，見步驟 2；設定後要重開 CMD
`class X is public, should be declared in a file named X.java`	檔名和 public class 名稱不一樣（含大小寫）
`file not found: X.java`	沒有先 cd /d C:\java，或檔案沒存在那裡
`找不到或無法載入主要類別`	執行時多打了 .java / .class，或有 package 卻沒打完整名稱
`';' expected`	某一行結尾少了分號
`possible lossy conversion from int to short`	大型態放進小型態，要改型態或強制轉型
`NullPointerException（System.console()）`	不是在真正的 CMD 視窗執行鍵盤輸入的程式
`NumberFormatException`	輸入的不是數字
`ArrayIndexOutOfBoundsException`	陣列索引超出範圍，或執行時忘了加命令列引數

期中考範圍對照

switch
已練習 3 支程式

for
已練習 16 支程式

overloading
已練習 5 支程式

recursive
已練習 4 支程式

（標示「尚未練習」的主題，目前的作業裡還沒有對應的程式。）

環境測試

01 第一支 Java 程式 `Hello.java`

輸出

題目：確認 JDK 安裝與 Path 設定正確：用 Notepad++ 撰寫程式，在 CMD 用 javac 編譯、java 執行。

程式碼

```
public class Hello {
    public static void main(String[] args) {
        System.out.println("Hello, Java!");
        System.out.println("我的第一支 Java 程式");
    }
}
```

CMD 指令

```
cd /d C:\java
javac -encoding UTF-8 Hello.java
java Hello
```

執行結果

```
Hello, Java!
我的第一支 Java 程式
```

重點

- 檔名必須和 public class 的名稱完全相同（大小寫也要一樣）。
- javac 檔名.java 會產生 .class 檔；java 類別名稱 執行時不加副檔名。

第3章 作業（運算子）

02 作業1：強制型別轉換 Ex1.java

型別轉換

題目：撰寫程式，求出下列 result 的值：int x=15,y=12; float result; result = x/(float)y;

程式碼

```
public class Ex1 {
    public static void main(String[] args) {
        int   x = 15, y = 12;
        float result;

        result = x / (float) y;

        System.out.println("x = " + x + ", y = " + y);
        System.out.println("result = x / (float) y = " + result);
    }
}
```

CMD 指令

```
cd /d C:\java
javac -encoding UTF-8 Ex1.java
java Ex1
```

執行結果

```
x = 15, y = 12
result = x / (float) y = 1.25
```

重點

- (float)y 先把 12 轉成 12.0，只要有一邊是小數，就做小數除法，15/12.0 = 1.25。
- 如果寫成 x/y，就是整數除法，小數會被丟掉，結果是 1.0。

03 作業2：前置與後置遞增 ex3_05.java

遞增運算子

題目：修正下列程式（改變其中一個運算子），使其符合執行結果 a=23, b=4, c=6。原程式第 9 行為 a+=b*++c;

程式碼

```
import java.lang.*;

public class ex3_05
{
    public static void main(String args[])
    {
        int a=3,b=4,c=5;
        a+=b*c++;

        System.out.println("a=" + a);
        System.out.println("b=" + b);
        System.out.println("c=" + c);
    }
}
```

CMD 指令

```
cd /d C:\java
javac -encoding UTF-8 ex3_05.java
java ex3_05
```

執行結果

```
a=23
b=4
c=6
```

重點

- 把 ++c 改成 c++。
- ++c（前置）：先加 1 再使用，a = 3 + 4×6 = 27。
- c++（後置）：先使用再加 1，a = 3 + 4×5 = 23，最後 c 一樣變成 6。

04 作業3：字串串接與運算順序 ex3_07.java

遞增運算子 字串串接

題目：修正程式第 10 行（仍需使用 x, y 變數），使其輸出「台北最高樓為101大樓」。原程式為 "台北最高樓為" + x + y + "大樓"。

程式碼

```
import java.lang.*;

public class ex3_07
{
    public static void main(String args[])
    {
        int x=50,y=x++;

        System.out.println("台北最高樓為" + (x + y) + "大樓");
    }
}
```

CMD 指令

```
cd /d C:\java
javac -encoding UTF-8 ex3_07.java
java ex3_07
```

執行結果

```
台北最高樓為101大樓
```

重點

- int x=50, y=x++; 執行後 y=50、x=51。
- + 由左往右算，左邊已是字串時，後面的 + 都變成「接文字」，所以沒括號會印出 5150。
- 把 x + y 加上括號，先算 51+50=101 再接字串。

05 作業4：overflow（溢位） HW4.java

資料型態 overflow

題目：修改資料型態的宣告，使 short a=30000, b=30000; c=a+b; 能輸出 60000。（檔名指定用 HW4）

程式碼

```
import java.lang.*;

public class HW4
{
    public static void main(String args[])
    {
        short a=30000,b=30000;
        int c;
        c=a+b;
        System.out.println(c);
    }
}
```

CMD 指令

```
cd /d C:\java
javac -encoding UTF-8 HW4.java
java HW4
```

執行結果

```
60000
```

重點

- short 範圍只有 -32768 ~ 32767，存不下 60000。
- 把 c 改成 int 即可。a+b 運算時會自動變成 int，原本寫 short c 連編譯都過不了。
- 若硬寫 c=(short)(a+b); 會真的溢位，印出 -5536。

第3章 課本範例

06 範例3-1：鍵盤輸入 ch3_01.java

鍵盤輸入 package

題目：使用 Console 從鍵盤讀取兩個字串並印出。

程式碼

```
/*  檔名:ch3_01.java    功能:鍵盤輸入範例    */

package myJava.ch03;
import java.lang.*;
import java.io.Console;

public class ch3_01          //主類別
{
    public static void main(String args[])
    {
        Console console=System.console();
        String str1,str2;

        System.out.print("請輸入第一個字串:");
        str1 = console.readLine();
        System.out.print("請輸入第二個字串:");
        str2 = console.readLine();
        System.out.println("您所輸入的字串如下:");
        System.out.println(str1);
        System.out.println(str2);
    }
}
```

CMD 指令

```
cd /d C:\java
javac -encoding UTF-8 -d . ch3_01.java
java myJava.ch03.ch3_01
```

範例輸入（每行輸入後按 Enter）

```
Hello
Java
```

執行結果

```
請輸入第一個字串:Hello
請輸入第二個字串:Java
您所輸入的字串如下:
Hello
Java
```

重點

- System.console() 只有在真正的 CMD 視窗才有值，用檔案導入輸入會得到 null。
- 有 package 的程式要用 javac -d . 編譯，執行時要打完整名稱 java myJava.ch03.ch3_01。

07 範例3-3：算術運算子 ch3_03.java

算術運算子

題目：示範 + - * / % 以及 float 運算 b²-4ac。

程式碼

```
/*  檔名:ch3_03.java     功能:算術運算子範例    */

package myJava.ch03;
import java.lang.*;

public class ch3_03          //主類別
{
  public static void main(String args[])
  {
    float answer;
    float a=2.1f,b=3.5f,c=4.0f;
    int x,y;
    x = 20;
    y = 7;
    System.out.print("當x=" + x + "     ");
    System.out.println("y=" + y + "時");
    System.out.println("x + y = " + (x+y));
    System.out.println("x - y = " + (x-y));
    System.out.println("x * y = " + (x*y));
    System.out.println("x / y = " + (x/y));
    System.out.println("x % y = " + (x%y));
    System.out.println("----------------------");
    System.out.print("當a=" + a + "    ");
    System.out.print("b=" + b + "    ");
    System.out.println("c" + c + "時");
    answer = b*b-4*a*c;
    System.out.println("b^2-4ac=" + answer);
  }
}
```

CMD 指令

```
cd /d C:\java
javac -encoding UTF-8 -d . ch3_03.java
java myJava.ch03.ch3_03
```

執行結果

```
當x=20     y=7時
x + y = 27
```

```
x - y = 13
x * y = 140
x / y = 2
x % y = 6
----------------------
當a=2.1    b=3.5    c4.0時
b^2-4ac=-21.349998
```

重點

- int / int 是整數除法：20/7 = 2；% 取餘數：20%7 = 6；y 不可為 0。
- float 存小數不完全精準，所以 -21.35 會變成 -21.349998。
- 課本第 25 行 "c" 少了等號，所以印出 c4.0，照抄未修改。

08 範例3-4：比較與邏輯運算子 ch3_04.java

比較運算子 邏輯運算子

題目：示範 == != > < >= <= 以及 ! && || 與 nand、nor。

程式碼

```
/*   檔名:ch3_04.java   功能:比較運算子與邏輯運算子範例   */

package myJava.ch03;
import java.lang.*;

public class ch3_04          //主類別
{
  public static void main(String args[])
  {
       int x=10,y=20;
       boolean a=true,b=false;

       System.out.print(" x=" + x);
       System.out.println(" y=" + y);
       System.out.println("---------------");
       System.out.println("x==y ==> " + (x==y));
       System.out.println("x!=y ==> " + (x!=y));
       System.out.println("x>y  ==> " + (x>y));
       System.out.println("x<y  ==> " + (x<y));
       System.out.println("x>=y ==> " + (x>=y));
       System.out.println("x<=y ==> " + (x<=y));
       System.out.println("====================");
       System.out.print(" a=" + a);
       System.out.println(" b=" + b);
       System.out.println("---------------");
       System.out.println("not a    ==> " + !a);
       System.out.println("a and b  ==> " + (a && b));
       System.out.println("a or b   ==> " + (a || b));
       System.out.println("a nand b ==> " + !(a && b));
       System.out.println("a nor b  ==> " + !(a || b));
       System.out.println("a and (x<y) ==> " + (a && (x<y)));
  }
}
```

CMD 指令

```
cd /d C:\java
javac -encoding UTF-8 -d . ch3_04.java
java myJava.ch03.ch3_04
```

執行結果

```
 x=10 y=20
---------------
x==y ==> false
x!=y ==> true
x>y  ==> false
x<y  ==> true
x>=y ==> false
x<=y ==> true
====================
 a=true b=false
---------------
not a    ==> false
a and b  ==> false
a or b   ==> true
a nand b ==> true
a nor b  ==> false
a and (x<y) ==> true
```

重點

- 等於要寫兩個等號 ==。
- &&：兩個都 true 才是 true；||：有一個 true 就是 true；! 反過來。
- nand = !(a && b)，nor = !(a || b)。

09 範例3-5：位元邏輯與移位運算子 ch3_05.java

位元運算子

題目：x=100(01100100)、y=50(00110010)、p=3，示範 ~ & | ^ >>> <<。

程式碼

```
/*    檔名:ch3_05.java     功能:位元邏輯及移位運算子範例     */

package myJava.ch03;
import java.lang.*;

public class ch3_05          //主類別
{
  public static void main(String args[])
  {
        short x=100,y=50,p=3;
        int xx=0;

        System.out.println("p=3");
        System.out.println("x=01100100");
        System.out.println("y=00110010");
        System.out.println("--------------------");
        System.out.println("not xx   ==> " + ~xx);
        System.out.println("x and y  ==> " + (x & y));
        System.out.println("x or y   ==> " + (x | y));
        System.out.println("x xor y  ==> " + (x ^ y));
        System.out.println("x >>> p  ==> " + (x >>> p));
        System.out.println("x << p   ==> " + (x << p));
  }
}
```

CMD 指令

```
cd /d C:\java
javac -encoding UTF-8 -d . ch3_05.java
java myJava.ch03.ch3_05
```

執行結果

```
p=3
x=01100100
y=00110010
--------------------
not xx   ==> -1
x and y  ==> 32
x or y   ==> 118
```

```
x xor y  ==> 86
x >>> p  ==> 12
x << p   ==> 800
```

重點

- &：兩位都是 1 才是 1；|：有一個 1 就是 1；^：兩位不同才是 1。
- x >>> 3 等於 100 ÷ 8 取整數 = 12；x << 3 等於 100 × 8 = 800。
- ~0 把 32 個 0 全部反轉成 1，用二補數表示就是 -1。

10 範例3-7：運算式的自動轉型 ch3_07.java

自動轉型

題目：short、int、char、float、double 混合運算：(s+a-b+c)+(c*d)。

程式碼

```
/*  檔名:ch3_07.java       功能:運算式的自動轉型範例    */

package myJava.ch03;
import java.lang.*;

public class ch3_07       //主類別
{
    public static void main(String args[])
    {
       short s=23;
       int a=100;
       char b='a';
       float c=3.5f;
       double d=4.8d;  //相當於 double d=4.8;

       System.out.print("result = ");
       System.out.println((s+a-b+c)+(c*d));
    }
}
```

CMD 指令

```
cd /d C:\java
javac -encoding UTF-8 -d . ch3_07.java
java myJava.ch03.ch3_07
```

執行結果

```
result = 46.3
```

重點

- 混合運算會自動轉成範圍較大的型態：short → int → long → float → double。
- char 'a' 參與計算時用 ASCII 碼 97：23+100-97+3.5 = 29.5。
- 3.5×4.8 = 16.8，29.5+16.8 = 46.3（double）。

11 範例3-8：字串轉數值 ch3_08.java

鍵盤輸入 字串轉數字

題目：從鍵盤輸入 x，用 Double.parseDouble 轉成數字後求 x 的 3 次方。

程式碼

```
/*  檔名:ch3_08.java      功能:字串轉原始資料型態範例   */

package myJava.ch03;
import java.lang.*;
import java.io.Console;

public class ch3_08          //主類別
{
    public static void main(String args[])
    {
        Console console=System.console();
        String str1;
        double x;

        System.out.print("欲求x的3次方,請輸入x:");
        str1 = console.readLine();
        x=Double.parseDouble(str1);
        System.out.println("x的3次方為:" + (x*x*x));
    }
}
```

CMD 指令

```
cd /d C:\java
javac -encoding UTF-8 -d . ch3_08.java
java myJava.ch03.ch3_08
```

範例輸入（每行輸入後按 Enter）

```
6.5
```

執行結果

```
欲求x的3次方,請輸入x:6.5
x的3次方為:274.625
```

重點

- 鍵盤讀進來的一定是字串，要先轉成數字才能計算。

- Double.parseDouble、Integer.parseInt、Float.parseFloat。
- 輸入不是數字會出現 NumberFormatException。

12 範例3-9：數值轉字串 ch3_09.java

數字轉字串

題目：用 String.valueOf、Float.toString、Double.toString 把數值轉成字串後串接。

程式碼

```
/*  檔名:ch3_09.java       功能:數值轉字串範例   */

package myJava.ch03;
import java.lang.*;

public class ch3_09          //主類別
{
    public static void main(String args[])
    {
       short a=100;
       int b=200;
       float d=3.5f;
       double e=4.8d;  //相當於 double d=4.8;
       String s1,s2;

       s1=String.valueOf(a+b)+String.valueOf(a)+String.valueOf(b);
       s2=Float.toString(d)+Double.toString(e);
       System.out.println("s1 = " + s1);
       System.out.println("s2 = " + s2);
    }
}
```

CMD 指令

```
cd /d C:\java
javac -encoding UTF-8 -d . ch3_09.java
java myJava.ch03.ch3_09
```

執行結果

```
s1 = 300100200
s2 = 3.54.8
```

重點

- String.valueOf(a+b) 括號裡先相加得 300，再轉成 "300"。
- 字串的 + 是接起來，"3.5" + "4.8" = "3.54.8"。
- valueOf 是多載（overloading）方法，任何型態都能放；toString 要用對應的類別。

第4章 作業（if-else、for）

13 判斷月份的季節 Season.java

if-else

題目：使用 IF ELSE 設計判斷結構，由月份判斷季節，月份由程式中直接定義。12,1,2 冬天；3,4,5 春天；6,7,8 夏天；9,10,11 秋天。經過判斷後印出「5月是春天」。

程式碼

```
/*  檔名:Season.java     功能:if-else判斷月份的季節    */

package myJava.ch04;
import java.lang.*;

public class Season          //主類別
{
    public static void main(String args[])
    {
      int month=5;           //月份直接在程式中定義

      if(month==12 || month==1 || month==2)
          System.out.println(month + "月是冬天");
      else if(month>=3 && month<=5)
          System.out.println(month + "月是春天");
      else if(month>=6 && month<=8)
          System.out.println(month + "月是夏天");
      else if(month>=9 && month<=11)
          System.out.println(month + "月是秋天");
      else
          System.out.println("月份錯誤");
    }
}
```

CMD 指令

```
cd /d C:\java
javac -encoding UTF-8 -d . Season.java
java myJava.ch04.Season
```

執行結果

```
5月是春天
```

重點

- 12、1、2 不連續，用 || 一個一個列；其他季節是連續範圍，用 && 表示。
- 最後的 else 處理不合法的月份。
- 此題也可以改用 switch 寫（case 12: case 1: case 2: ...），是期中考範圍。

14 判斷直角三角形 `Triangle.java`

if-else　鍵盤輸入

題目：由使用者輸入三角形的三邊 x, y, z（假設只會輸入正整數），判別此三角形是否為直角三角形。

程式碼

```
/*  檔名:Triangle.java     功能:判斷是否為直角三角形    */

package myJava.ch04;
import java.lang.*;
import java.io.Console;

public class Triangle          //主類別
{
    public static void main(String args[])
    {
      Console console=System.console();
      int x,y,z;

      System.out.print("請輸入第一邊x:");
      x=Integer.parseInt(console.readLine());
      System.out.print("請輸入第二邊y:");
      y=Integer.parseInt(console.readLine());
      System.out.print("請輸入第三邊z:");
      z=Integer.parseInt(console.readLine());

      if((x*x + y*y == z*z) || (x*x + z*z == y*y) || (y*y + z*z == x*x))
          System.out.println("此三角形是直角三角形");
      else
          System.out.println("此三角形不是直角三角形");
    }
}
```

CMD 指令

```
cd /d C:\java
javac -encoding UTF-8 -d . Triangle.java
java myJava.ch04.Triangle
```

範例輸入（每行輸入後按 Enter）

```
3
4
5
```

執行結果

```
請輸入第一邊x:3
請輸入第二邊y:4
請輸入第三邊z:5
此三角形是直角三角形
```

重點

- 畢氏定理：兩股平方和 = 斜邊平方。
- 使用者不一定把最長邊放最後，所以三種情況都要檢查，用 || 連接。

15 for 迴圈 9×9 乘法表 `Multiply.java`

for

題目：使用 for 迴圈印出 9×9 乘法表，每一列的第二個數字相同（1*1=1 2*1=2 … 9*1=9）。

程式碼

```
/*  檔名:Multiply.java    功能:for迴圈印出9X9乘法表   */

package myJava.ch04;
import java.lang.*;

public class Multiply          //主類別
{
  public static void main(String args[])
  {
     for(int i=1;i<=9;i++)
     {
       for(int j=1;j<=9;j++)
           System.out.print(j + "*" + i + "=" + j*i + "\t");
       System.out.println();
     }
  }
}
```

CMD 指令

```
cd /d C:\java
javac -encoding UTF-8 -d . Multiply.java
java myJava.ch04.Multiply
```

執行結果

```
1*1=1   2*1=2   3*1=3   4*1=4   5*1=5   6*1=6   7*1=7   8*1=8   9*1=9
1*2=2   2*2=4   3*2=6   4*2=8   5*2=10  6*2=12  7*2=14  8*2=16  9*2=18
1*3=3   2*3=6   3*3=9   4*3=12  5*3=15  6*3=18  7*3=21  8*3=24  9*3=27
1*4=4   2*4=8   3*4=12  4*4=16  5*4=20  6*4=24  7*4=28  8*4=32  9*4=36
1*5=5   2*5=10  3*5=15  4*5=20  5*5=25  6*5=30  7*5=35  8*5=40  9*5=45
1*6=6   2*6=12  3*6=18  4*6=24  5*6=30  6*6=36  7*6=42  8*6=48  9*6=54
1*7=7   2*7=14  3*7=21  4*7=28  5*7=35  6*7=42  7*7=49  8*7=56  9*7=63
1*8=8   2*8=16  3*8=24  4*8=32  5*8=40  6*8=48  7*8=56  8*8=64  9*8=72
1*9=9   2*9=18  3*9=27  4*9=36  5*9=45  6*9=54  7*9=63  8*9=72  9*9=81
```

重點

- 外層 i 控制第幾列，內層 j 控制第幾欄，共 81 個算式。
- 和課本 ch4_17 的差別只是印出時 i、j 對調（j + "*" + i）。

- print 不換行、println() 換行、\t 對齊。

16 英文字母大小寫轉換 `CharCase.java`

if-else　字元　鍵盤輸入

題目：輸入一個字元，小寫轉大寫、大寫轉小寫後輸出；不是英文字母則輸出「您所輸入的並不是英文字母」。

程式碼

```
/*  檔名:CharCase.java     功能:英文字母大小寫轉換    */

package myJava.ch04;
import java.lang.*;
import java.io.Console;

public class CharCase          //主類別
{
    public static void main(String args[])
    {
      Console console=System.console();
      char ch;

      System.out.print("請輸入一個字元:");
      ch=console.readLine().charAt(0);       //讀取一行字串,取出第一個字元

      if(ch>='a' && ch<='z')
          System.out.println("轉換為大寫:" + (char)(ch-32));
      else if(ch>='A' && ch<='Z')
          System.out.println("轉換為小寫:" + (char)(ch+32));
      else
          System.out.println("您所輸入的並不是英文字母");
    }
}
```

CMD 指令

```
cd /d C:\java
javac -encoding UTF-8 -d . CharCase.java
java myJava.ch04.CharCase
```

範例輸入（每行輸入後按 Enter）

```
b
```

執行結果

```
請輸入一個字元:b
轉換為大寫:B
```

重點

- console.readLine().charAt(0) 取出輸入的第一個字元。
- 字元可以直接比大小：ch>='a' && ch<='z'。
- 大小寫的 ASCII 碼差 32（'A'=65、'a'=97），計算後要 (char) 轉回字元。

第5章 作業（陣列）

17 陣列查詢電話號碼 PhoneBook.java

陣列 for 字串比對

題目：使用陣列記錄 3 個人的姓名與電話（Tom 032125678、Bob 0226713456、Jack 075534321），由使用者輸入姓名，輸出對應的電話；找不到則輸出「查無此人」。

程式碼

```
/*  檔名:PhoneBook.java     功能:用陣列查詢電話號碼    */

package myJava.ch05;
import java.lang.*;
import java.io.Console;

public class PhoneBook          //主類別
{
    public static void main(String args[])
    {
      Console console=System.console();
      String name[]={"Tom","Bob","Jack"};
      String phone[]={"032125678","0226713456","075534321"};
      String str1;
      boolean found=false;

      System.out.print("請輸入要查詢的對象:");
      str1=console.readLine();

      for(int i=0;i<name.length;i++)
      {
         if(str1.equals(name[i]))
         {
            System.out.println("電話是:" + phone[i]);
            found=true;
         }
      }
      if(found==false)
         System.out.println("查無此人");
    }
}
```

CMD 指令

```
cd /d C:\java
javac -encoding UTF-8 -d . PhoneBook.java
java myJava.ch05.PhoneBook
```

範例輸入（每行輸入後按 Enter）

```
Tom
```

執行結果

```
請輸入要查詢的對象:Tom
電話是:032125678

（輸入 Mary 時）
請輸入要查詢的對象:Mary
查無此人
```

重點

- 兩個陣列用同一個索引對應：name[i] 的電話在 phone[i]。
- 電話用 String 存，用 int 開頭的 0 會不見。
- 字串比對要用 str1.equals(str2)，不要用 ==。
- 用 boolean found 記錄有沒有找到，迴圈跑完還是 false 才印查無此人。

18 成績分布橫條圖 ScoreChart.java

陣列 for

題目：使用陣列存放學生的成績（最多 50 人），並畫出學生分數分布圖（100、90~99 … 0~9，每人一個 = ）。

程式碼

```
/*  檔名:ScoreChart.java     功能:用陣列存放成績並畫出分數分布圖    */

package myJava.ch05;
import java.lang.*;
import java.io.Console;

public class ScoreChart          //主類別
{
    public static void main(String args[])
    {
      Console console=System.console();
      int n;
      int score[]=new int[50];      //最多50人
      int count[]=new int[11];      //count[0]:0~9分 ... count[9]:90~99分, count[10]:100分

      System.out.print("請輸入學生人數(最多50人):");
      n=Integer.parseInt(console.readLine());
      if(n<1 || n>50)
      {
         System.out.println("人數必須在1~50之間");
         return;
      }

      System.out.println("請輸入學生分數");
      for(int i=0;i<n;i++)
      {
         System.out.print("第 " + (i+1) + " 位 : ");
         score[i]=Integer.parseInt(console.readLine());
         count[score[i]/10]++;      //依照分數的十位數分組計數
      }

      System.out.println("=== 成績分布橫條圖===");
      for(int k=10;k>=0;k--)
      {
         if(k==10)
            System.out.print(" 100  :");
         else
         {
            int lo=k*10, hi=k*10+9;
            System.out.print((lo<10 ? " " : "") + lo + "~" + (hi<10 ? " " : "") + hi + "
:");
```

```
            }
            for(int j=0;j<count[k];j++)
               System.out.print("=");
            System.out.println();
         }
      }
}
```

CMD 指令

```
cd /d C:\java
javac -encoding UTF-8 -d . ScoreChart.java
java myJava.ch05.ScoreChart
```

範例輸入（每行輸入後按 Enter）

```
10
93
100
84
72
46
48
77
4
75
65
```

執行結果

```
=== 成績分布橫條圖===
 100  :=
90~99 :=
80~89 :=
70~79 :===
60~69 :=
50~59 :
40~49 :==
30~39 :
20~29 :
10~19 :
 0~ 9 :=
```

重點

- count[score[i]/10]++：整數除法取十位數當作分組索引，100 分會放在 count[10]。
- 外層 k 從 10 倒數到 0，內層依 count[k] 印出幾個 =。

- (lo<10 ? " " : "") 是條件運算子，用來補空格對齊。

19 3 階行列式 Determinant.java

二維陣列 for

題目：3 階行列式 = aez + bfx + cdy - xec - yfa - zdb。求行列式 {1 2 3; 5 7 9; 10 5 8} 的值。

程式碼

```
/*  檔名:Determinant.java     功能:用二維陣列計算3階行列式    */

package myJava.ch05;
import java.lang.*;

public class Determinant          //主類別
{
    public static void main(String args[])
    {
      int m[][]={{ 1, 2, 3},
                 { 5, 7, 9},
                 {10, 5, 8}};
      int det;

      System.out.println("行列式為:");
      for(int i=0;i<3;i++)
      {
         System.out.print("| ");
         for(int j=0;j<3;j++)
            System.out.print(m[i][j] + "\t");
         System.out.println("|");
      }

      // aez + bfx + cdy - xec - yfa - zdb
      det = m[0][0]*m[1][1]*m[2][2]
          + m[0][1]*m[1][2]*m[2][0]
          + m[0][2]*m[1][0]*m[2][1]
          - m[2][0]*m[1][1]*m[0][2]
          - m[2][1]*m[1][2]*m[0][0]
          - m[2][2]*m[1][0]*m[0][1];

      System.out.println("其值為:" + det);
    }
}
```

CMD 指令

```
cd /d C:\java
javac -encoding UTF-8 -d . Determinant.java
java myJava.ch05.Determinant
```

執行結果

```
行列式為:
| 1     2       3       |
| 5     7       9       |
| 10    5       8       |
其值為:-24
```

重點

- 手算：56 + 180 + 75 - 210 - 45 - 80 = -24。
- m[列][行]，索引從 0 開始：a=m[0][0]、e=m[1][1]、z=m[2][2]、x=m[2][0]。
- 把陣列改成課本例子 {-2 8 7; 3 0 5; 4 1 6} 可驗算得到 47。

20 矩陣轉置 Transpose.java

二維陣列 for 鍵盤輸入

題目：輸入一個矩陣 A（最大 10×10），輸出 A 的轉置矩陣。

程式碼

```
/*  檔名:Transpose.java    功能:矩陣轉置   */

package myJava.ch05;
import java.lang.*;
import java.io.Console;

public class Transpose          //主類別
{
    public static void main(String args[])
    {
      Console console=System.console();
      int A[][]=new int[10][10];    //原始矩陣,最大10*10
      int B[][]=new int[10][10];    //轉置矩陣
      int row,col;

      System.out.print("請輸入原始矩陣的列數=>");
      row=Integer.parseInt(console.readLine());
      System.out.print("請輸入原始矩陣的行數=>");
      col=Integer.parseInt(console.readLine());
      if(row<1 || row>10 || col<1 || col>10)
      {
         System.out.println("列數與行數必須在1~10之間");
         return;
      }

      for(int i=0;i<row;i++)
         for(int j=0;j<col;j++)
         {
            System.out.print("A[" + i + "][" + j + "]元素值=>");
            A[i][j]=Integer.parseInt(console.readLine());
         }

      for(int i=0;i<row;i++)        //轉置:B[j][i]=A[i][j]
         for(int j=0;j<col;j++)
            B[j][i]=A[i][j];

      System.out.println("原始矩陣");
      for(int i=0;i<row;i++)
      {
         for(int j=0;j<col;j++)
            System.out.print(A[i][j] + "\t");
         System.out.println();
```

```
      }

      System.out.println("轉置矩陣");
      for(int i=0;i<col;i++)
      {
         for(int j=0;j<row;j++)
            System.out.print(B[i][j] + "\t");
         System.out.println();
      }
   }
}
```

CMD 指令

```
cd /d C:\java
javac -encoding UTF-8 -d . Transpose.java
java myJava.ch05.Transpose
```

範例輸入（每行輸入後按 Enter）

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

執行結果

```
請輸入原始矩陣的列數=>3
請輸入原始矩陣的行數=>2
A[0][0]元素值=>1
A[0][1]元素值=>2
A[1][0]元素值=>3
A[1][1]元素值=>4
A[2][0]元素值=>5
A[2][1]元素值=>6
原始矩陣
1       2
3       4
5       6
轉置矩陣
1       3       5
2       4       6
```

重點

- 轉置的關鍵：B[j][i] = A[i][j]，列和行對調。
- 陣列先開最大 10×10，只使用前 row 列、col 行。
- 印轉置矩陣時迴圈範圍也要對調：i<col、j<row。

第6章 作業（函式）

21 作業1：遞迴函式 myPower MyPower.java

recursive 函式

題目：利用遞迴的方式設計一個函式 double myPower(double x,int y)，計算出 x 的 y 次方。（若 y<0，則回傳 0）

程式碼

```
/*  檔名:MyPower.java     功能:遞迴函式計算x的y次方    */

package myJava.ch06;
import java.lang.*;

public class MyPower          //主類別
{
    public static void main(String args[])
    {
        System.out.println("2的10次方=" + myPower(2,10));
        System.out.println("1.5的3次方=" + myPower(1.5,3));
        System.out.println("3.5的0次方=" + myPower(3.5,0));
        System.out.println("2的-1次方=" + myPower(2,-1) + "  (y<0回傳0)");
    }

    public static double myPower(double x,int y)
    {
        if(y<0)                       //y<0,依題意回傳0
            return 0;
        if(y==0)                      //邊界條件:x的0次方為1
            return 1;
        return x*myPower(x,y-1);      //遞迴呼叫:x的y次方 = x * x的(y-1)次方
    }
}
```

CMD 指令

```
cd /d C:\java
javac -encoding UTF-8 -d . MyPower.java
java myJava.ch06.MyPower
```

執行結果

```
2的10次方=1024.0
1.5的3次方=3.375
```

```
3.5的0次方=1.0
2的-1次方=0.0  (y<0回傳0)
```

重點

- 遞迴定義：x 的 y 次方 = x × (x 的 y-1 次方)，所以寫成 return x*myPower(x,y-1);
- 邊界條件：y==0 時回傳 1，遞迴才會停止（講義 6.9：遞迴函式必須有邊界條件）。
- 每呼叫一次 y 就減 1，逐漸往邊界條件移動；y<0 依題意直接回傳 0。
- 函式要宣告成 static，main 才能直接呼叫（講義 6.2）。

22 作業2：函式 print_total_mul 列印乘法表 `TotalMul.java`

函式 for

題目：設計一個函式 void print_total_mul(int a,int b)，列印 ab 乘法表。例如 a=5、b=8 時印出 5*5 到 8*8；a=8、b=5 時輸出相同；a=1、b=9 時恰為九九乘法表。

程式碼

```
/*  檔名:TotalMul.java     功能:函式列印a~b的乘法表    */

package myJava.ch06;
import java.lang.*;

public class TotalMul          //主類別
{
    public static void main(String args[])
    {
        System.out.println("a=5,b=8時:");
        print_total_mul(5,8);
        System.out.println();
        System.out.println("a=8,b=5時:");
        print_total_mul(8,5);
        System.out.println();
        System.out.println("a=1,b=9時(九九乘法表):");
        print_total_mul(1,9);
    }

    public static void print_total_mul(int a,int b)
    {
        int start,end;
        if(a<b)                        //不論a,b誰大,都由小印到大
        {  start=a;  end=b;  }
        else
        {  start=b;  end=a;  }

        for(int i=start;i<=end;i++)
        {
            for(int j=start;j<=end;j++)
                System.out.print(i + "*" + j + "=" + i*j + "\t");
            System.out.println();
        }
    }
}
```

CMD 指令

```
cd /d C:\java
javac -encoding UTF-8 -d . TotalMul.java
```

```
java myJava.ch06.TotalMul
```

執行結果

```
a=5,b=8時:
5*5=25  5*6=30  5*7=35  5*8=40
6*5=30  6*6=36  6*7=42  6*8=48
7*5=35  7*6=42  7*7=49  7*8=56
8*5=40  8*6=48  8*7=56  8*8=64

a=8,b=5時:
(與上面相同)

a=1,b=9時(九九乘法表):
1*1=1   1*2=2   1*3=3   ...     1*9=9
...
9*1=9   9*2=18  9*3=27  ...     9*9=81
```

重點

- 先判斷 a、b 誰小，存成 start、end，所以 (5,8) 和 (8,5) 印出來一樣。
- 兩層 for：外層 i 是列、內層 j 是欄，都從 start 跑到 end，印 i*j。
- void 表示函式沒有回傳值，只負責印出結果。

23 作業3：多載函式 maxEle `MaxEle.java`

overloading　函式　二維陣列　for

題目：使用多載技術，設計兩個函式 double maxEle(int x[][]) 與 double maxEle(double x[][])，可找出二維陣列之最大值並回傳。

程式碼

```
/*  檔名:MaxEle.java     功能:多載函式找出二維陣列的最大值    */

package myJava.ch06;
import java.lang.*;

public class MaxEle          //主類別
{
    public static void main(String args[])
    {
        int    a[][]={{12,45,7},
                      {88,23},
                      {5,61,30,9}};
        double b[][]={{3.5,9.25},
                      {12.75,0.5,6.0},
                      {7.125}};

        System.out.println("int陣列的最大值為:" + maxEle(a));     //呼叫 maxEle(int x[][])
        System.out.println("double陣列的最大值為:" + maxEle(b));  //呼叫 maxEle(double x[]
[])
    }

    public static double maxEle(int x[][])
    {
        int max=x[0][0];
        for(int i=0;i<x.length;i++)
            for(int j=0;j<x[i].length;j++)
                if(x[i][j]>max)
                    max=x[i][j];
        return max;
    }

    public static double maxEle(double x[][])
    {
        double max=x[0][0];
        for(int i=0;i<x.length;i++)
            for(int j=0;j<x[i].length;j++)
                if(x[i][j]>max)
                    max=x[i][j];
        return max;
```

```
    }
}
```

CMD 指令

```
cd /d C:\java
javac -encoding UTF-8 -d . MaxEle.java
java myJava.ch06.MaxEle
```

執行結果

```
int陣列的最大值為:88.0
double陣列的最大值為:12.75
```

重點

- 多載（overloading）：函式名稱相同，但參數的型態或個數不同；Java 會依照傳進去的引數型態自動選擇要呼叫哪一個。
- maxEle(a) 傳 int 陣列 → 呼叫 maxEle(int x[][])；maxEle(b) 傳 double 陣列 → 呼叫 maxEle(double x[][])。
- 只有回傳型態不同不算多載，參數一定要不同。
- x.length 是列數、x[i].length 是第 i 列的元素個數，每列長度不同也能正確找。
- 先假設 max = x[0][0]，再逐一比較；int 版本回傳時會自動轉成 double，所以印出 88.0。

24 作業4：命令列引數與函式 mod `ex6_03.java`

函式 命令列引數

題目：撰寫函式 mod，傳入 x,y 給 mod，x,y 使用命令列引數輸入，計算 x/y 的商數及餘數並回傳。（x,y 皆為 int，宣告時設定初值，y 不為 0）執行：java ex6_03 46 10

程式碼

```
/*  檔名:ex6_03.java     功能:命令列引數輸入x,y,用函式mod回傳商數及餘數    */

import java.lang.*;

public class ex6_03          //主類別
{
    public static void main(String args[])
    {
        int x=Integer.parseInt(args[0]),y=Integer.parseInt(args[1]);    //宣告時設定初值(命令列引數)
        int ans[];

        ans=mod(x,y);
        System.out.println("被除數為" + x + "\t" + "除數為" + y);          //印出第1行結果
        System.out.println("商數為" + ans[0] + " 餘數為" + ans[1]);        //印出第2行結果
    }

    public static int[] mod(int x,int y)
    {
        int result[]=new int[2];
        result[0]=x/y;          //商數
        result[1]=x%y;          //餘數
        return result;          //一次回傳兩個值,所以用陣列
    }
}
```

CMD 指令

```
cd /d C:\java
javac -encoding UTF-8 ex6_03.java
java ex6_03 46 10
```

執行結果

```
C:\java>java ex6_03 46 10
被除數為46        除數為10
商數為4 餘數為6
```

重點

- 命令列引數：java ex6_03 46 10 後面的 46、10 會放進 main 的 args 陣列，args[0]="46"、args[1]="10"（字串），要用 Integer.parseInt 轉成 int。
- int x=Integer.parseInt(args[0]),y=Integer.parseInt(args[1]); 就是「宣告時設定初值」。
- Java 的函式只能 return 一個值，要回傳商數和餘數兩個值，所以讓 mod 回傳 int 陣列：result[0] 放商數 x/y、result[1] 放餘數 x%y。
- 題目的 mod(…); 修正為 ans=mod(x,y);，用陣列接住回傳值，再印出第 2 行。
- 此題沒有 package，才能照題目用 java ex6_03 46 10 執行；執行時忘了加引數會出現 ArrayIndexOutOfBoundsException。

期中考補充：switch 練習

25 switch：數字 1~7 對應星期幾 `Weekday.java`

switch　鍵盤輸入

題目：寫一個 Java 程式，接收使用者輸入的整數（1 到 7），利用 switch 判斷並印出對應的星期幾（1 為星期一 … 7 為星期日）；輸入其他數字時印出錯誤訊息。

程式碼

```
/*  檔名:Weekday.java     功能:switch判斷數字1~7對應星期幾    */

package myJava.ch04;
import java.lang.*;
import java.io.Console;

public class Weekday          //主類別
{
    public static void main(String args[])
    {
      Console console=System.console();
      int day;

      System.out.print("請輸入數字(1~7):");
      day=Integer.parseInt(console.readLine());

      switch(day)
      {
         case 1:
            System.out.println("星期一");
            break;
         case 2:
            System.out.println("星期二");
            break;
         case 3:
            System.out.println("星期三");
            break;
         case 4:
            System.out.println("星期四");
            break;
         case 5:
            System.out.println("星期五");
            break;
         case 6:
            System.out.println("星期六");
            break;
         case 7:
            System.out.println("星期日");
```

```
            break;
         default:                          //不是1~7的數字
            System.out.println("輸入錯誤,請輸入1~7");
      }
   }
}
```

CMD 指令

```
cd /d C:\java
javac -encoding UTF-8 -d . Weekday.java
java myJava.ch04.Weekday
```

範例輸入（每行輸入後按 Enter）

```
5
```

執行結果

```
請輸入數字(1~7):5
星期五

（輸入 8 時）
請輸入數字(1~7):8
輸入錯誤,請輸入1~7
```

重點

- switch(day) 會跳到值相同的 case 開始執行。
- 每個 case 最後都要加 break，否則會「貫穿」（fall-through）繼續執行下一個 case。
- default 處理所有 case 都不符合的情況，相當於 if-else 最後的 else。
- 最後一個區塊（default）後面的 break 可省略。

26 switch：分數判斷等級 A~E `ScoreGrade.java`

switch　鍵盤輸入

題目：輸入 0~100 的分數，用 switch(分數/10) 判斷等級：90~100 得到A、80~89 得到B、70~79 得到C、60~69 得到D、0~59 得到E；超出範圍則印出錯誤訊息。（參考 iT 邦幫忙〈[iT鐵人賽 Day16]JAVA-switch範例〉https://ithelp.ithome.com.tw/articles/10264840 ）

程式碼

```
/*  檔名:ScoreGrade.java     功能:switch依分數判斷等級A~E    */

package myJava.ch04;
import java.lang.*;
import java.io.Console;

public class ScoreGrade          //主類別
{
    public static void main(String args[])
    {
      Console console=System.console();
      int score;

      System.out.print("請輸入分數(0~100):");
      score=Integer.parseInt(console.readLine());

      if(score<0 || score>100)          //先確保分數在0~100之內
      {
         System.out.println("輸入錯誤,請輸入0~100");
         return;
      }

      switch(score/10)                  //整數除法取十位數
      {
         case 10:                       //100分
         case 9:                        //90~99分
            System.out.println("得到A");
            break;
         case 8:
            System.out.println("得到B");
            break;
         case 7:
            System.out.println("得到C");
            break;
         case 6:
            System.out.println("得到D");
            break;
         default:                       //0~59分
            System.out.println("得到E");
      }
```

```
    }
}
```

CMD 指令

```
cd /d C:\java
javac -encoding UTF-8 -d . ScoreGrade.java
java myJava.ch04.ScoreGrade
```

範例輸入（每行輸入後按 Enter）

```
83
```

執行結果

```
請輸入分數(0~100):83
得到B

（輸入 100）得到A
（輸入 59）得到E
（輸入 105）輸入錯誤,請輸入0~100
```

重點

- score/10 是整數除法，取出十位數：83/10 = 8、100/10 = 10。
- case 10: case 9: 合併寫法：兩個 case 之間沒有 break，所以 100 分和 90~99 分都會執行「得到 A」。
- 先用 if 檢查 0~100 的範圍再進 switch（講義第 4 章 switch-case 範例的做法）。如果沒有檢查，101~109 分的 score/10 也是 10，會被誤判成 A。
- 範圍檢查過後，0~59 分都交給 default 處理，不用寫 case 5: case 4: … case 0:。

27 switch：12 位數長整數中 0~9 各出現幾次 DigitCount.java

switch while 資料型態

題目：試設定一個剛好 12 個位數的長整數變數，並利用 switch 計算其 0 到 9 這十個數字中每一個數字出現的次數。

程式碼

```
/*  檔名:DigitCount.java     功能:switch計算12位數長整數中0~9各出現幾次    */

package myJava.ch04;
import java.lang.*;

public class DigitCount          //主類別
{
    public static void main(String args[])
    {
      long num=314159265358L;           //剛好12位數的長整數,Long常數要加L
      long n=num;
      int c0=0,c1=0,c2=0,c3=0,c4=0,c5=0,c6=0,c7=0,c8=0,c9=0;

      while(n>0)
      {
         switch((int)(n%10))            //取出個位數;switch不能用long,所以轉成int
         {
            case 0: c0++; break;
            case 1: c1++; break;
            case 2: c2++; break;
            case 3: c3++; break;
            case 4: c4++; break;
            case 5: c5++; break;
            case 6: c6++; break;
            case 7: c7++; break;
            case 8: c8++; break;
            case 9: c9++; break;
         }
         n=n/10;                        //去掉個位數
      }

      System.out.println("長整數:" + num);
      System.out.println("數字0出現" + c0 + "次");
      System.out.println("數字1出現" + c1 + "次");
      System.out.println("數字2出現" + c2 + "次");
      System.out.println("數字3出現" + c3 + "次");
      System.out.println("數字4出現" + c4 + "次");
      System.out.println("數字5出現" + c5 + "次");
      System.out.println("數字6出現" + c6 + "次");
      System.out.println("數字7出現" + c7 + "次");
      System.out.println("數字8出現" + c8 + "次");
```

```
      System.out.println("數字9出現" + c9 + "次");
    }
}
```

CMD 指令

```
cd /d C:\java
javac -encoding UTF-8 -d . DigitCount.java
java myJava.ch04.DigitCount
```

執行結果

```
長整數:314159265358
數字0出現0次
數字1出現2次
數字2出現1次
數字3出現2次
數字4出現1次
數字5出現3次
數字6出現1次
數字7出現0次
數字8出現1次
數字9出現1次
```

重點

- 12 位數超過 int 的範圍（約 21 億），要用 long，常數後面要加 L：314159265358L。
- n%10 取出個位數，n=n/10 去掉個位數，用 while 迴圈一位一位取，直到 n 變成 0。
- switch 不能用 long，所以要先轉型：switch((int)(n%10))。switch 可以用 int、char、short、byte、String。
- 每個 case 讓對應的計數器加 1（c0++ … c9++）；所有次數加起來剛好是 12。

Java 練習題（HackMD）

28 練習1：兔子問題（費氏數列） Rabbit.java

recursive 函式 for 鍵盤輸入

題目：古典問題：有一對兔子，從出生後第 3 個月起每個月都生一對兔子，小兔子長到第三個月後每個月又生一對兔子，假如兔子都不死，問每個月的兔子對數為多少？（題目來源：HackMD〈Java練習題〉https://hackmd.io/@Boom17o/ryjHxEkqY ）

程式碼

```
/*  執行方式:
    在 PowerShell 輸入 cmd 按 Enter,提示字元會從 PS C:\Users\USER> 變成 C:\Users\USER>,
    之後照下面的指令打:

    cmd
    cd /d C:\Users\USER\Java
    javac -encoding UTF-8 -d . Rabbit.java
    java myJava.practice.Rabbit

    也可以按 Win + R,輸入 cmd 按 Enter,直接開 CMD 視窗(就不用打第一行的 cmd)。
*/

/*  檔名:Rabbit.java     功能:遞迴計算每個月的兔子對數(費氏數列)    */

package myJava.practice;
import java.lang.*;
import java.io.Console;

public class Rabbit          //主類別
{
    public static void main(String args[])
    {
      Console console=System.console();
      int month;

      System.out.print("請輸入要計算到第幾個月:");
      month=Integer.parseInt(console.readLine());

      for(int i=1;i<=month;i++)
         System.out.println("第" + i + "個月的兔子對數:" + rabbit(i));
    }

    public static int rabbit(int n)
    {
        if(n==1 || n==2)                  //邊界條件:第1、2個月都只有1對
            return 1;
        return rabbit(n-1)+rabbit(n-2);   //遞迴呼叫:前兩個月的對數相加
```

```
    }
}
```

CMD 指令

```
cd /d C:\java
javac -encoding UTF-8 -d . Rabbit.java
java myJava.practice.Rabbit
```

範例輸入（每行輸入後按 Enter）

```
10
```

執行結果

```
請輸入要計算到第幾個月:10
第1個月的兔子對數:1
第2個月的兔子對數:1
第3個月的兔子對數:2
第4個月的兔子對數:3
第5個月的兔子對數:5
第6個月的兔子對數:8
第7個月的兔子對數:13
第8個月的兔子對數:21
第9個月的兔子對數:34
第10個月的兔子對數:55
```

重點

- 列出每個月的對數：1, 1, 2, 3, 5, 8, 13 ...，規律是「這個月 = 上個月 + 上上個月」，就是費氏數列（講義 ch6_19）。
- 遞迴寫法：return rabbit(n-1)+rabbit(n-2);
- 邊界條件：第 1、2 個月都是 1 對（n==1 || n==2 時 return 1），遞迴才會停止。
- 這種遞迴每次會呼叫自己兩次，月份很大時（例如 40 以上）會算很久。

29 練習2：101~200 之間的質數 PrimeNumber.java

for 函式

題目：判斷 101~200 之間有多少個質數，並輸出所有質數。

程式碼

```
/*  執行方式:
    在 PowerShell 輸入 cmd 按 Enter,提示字元會從 PS C:\Users\USER> 變成 C:\Users\USER>,
    之後照下面的指令打:

    cmd
    cd /d C:\Users\USER\Java
    javac -encoding UTF-8 -d . PrimeNumber.java
    java myJava.practice.PrimeNumber

    也可以按 Win + R,輸入 cmd 按 Enter,直接開 CMD 視窗(就不用打第一行的 cmd)。
*/

/*  檔名:PrimeNumber.java     功能:找出101~200之間的質數並計算個數    */

package myJava.practice;
import java.lang.*;

public class PrimeNumber          //主類別
{
    public static void main(String args[])
    {
      int count=0;                        //質數個數

      System.out.println("101~200之間的質數:");
      for(int i=101;i<=200;i++)
      {
         if(isPrime(i))
         {
            count++;
            System.out.print(i + "\t");
            if(count%10==0)               //每印10個換一行
               System.out.println();
         }
      }
      System.out.println();
      System.out.println("共有" + count + "個質數");
    }

    public static boolean isPrime(int n)
    {
        for(int j=2;j*j<=n;j++)           //只要檢查到根號n
        {
```

```
            if(n%j==0)                      //能被整除,不是質數
                return false;
        }
        return true;                        //都不能整除,是質數
    }
}
```

CMD 指令

```
cd /d C:\java
javac -encoding UTF-8 -d . PrimeNumber.java
java myJava.practice.PrimeNumber
```

執行結果

```
101~200之間的質數:
101     103     107     109     113     127     131     137     139     149
151     157     163     167     173     179     181     191     193     197
199
共有21個質數
```

重點

- 質數：只能被 1 和自己整除。用 n%j==0 檢查能不能被 j 整除。
- 把判斷寫成函式 boolean isPrime(int n)，找到因數就馬上 return false；迴圈跑完都沒有才 return true。
- j 只要檢查到根號 n（j*j<=n）就夠了：如果 n 有大於根號 n 的因數，一定也有對應的小於根號 n 的因數。
- count%10==0 時換行，每列印 10 個。

30 練習3：水仙花數 Narcissistic.java

for

題目：印出所有的「水仙花數」，所謂「水仙花數」是指一個三位數，其各位數字立方和等於該數本身。例如：153 是一個水仙花數，因為 153 = 1^3 + 5^3 + 3^3。

程式碼

```
/*  執行方式:
    在 PowerShell 輸入 cmd 按 Enter,提示字元會從 PS C:\Users\USER> 變成 C:\Users\USER>,
    之後照下面的指令打:

    cmd
    cd /d C:\Users\USER\Java
    javac -encoding UTF-8 -d . Narcissistic.java
    java myJava.practice.Narcissistic

    也可以按 Win + R,輸入 cmd 按 Enter,直接開 CMD 視窗(就不用打第一行的 cmd)。
*/

/*  檔名:Narcissistic.java     功能:印出所有三位數的水仙花數    */

package myJava.practice;
import java.lang.*;

public class Narcissistic          //主類別
{
    public static void main(String args[])
    {
      int hun,ten,one;                  //百位、十位、個位

      System.out.println("所有的水仙花數:");
      for(int i=100;i<=999;i++)
      {
         hun=i/100;                     //百位數
         ten=i/10%10;                   //十位數
         one=i%10;                      //個位數
         if(hun*hun*hun + ten*ten*ten + one*one*one == i)
            System.out.println(i + "=" + hun + "^3+" + ten + "^3+" + one + "^3");
      }
    }
}
```

CMD 指令

```
cd /d C:\java
javac -encoding UTF-8 -d . Narcissistic.java
java myJava.practice.Narcissistic
```

執行結果

```
所有的水仙花數:
153=1^3+5^3+3^3
370=3^3+7^3+0^3
371=3^3+7^3+1^3
407=4^3+0^3+7^3
```

重點

- 三位數是 100~999，for 從 100 跑到 999。
- 拆位數：百位 i/100、十位 i/10%10、個位 i%10（整數除法和取餘數）。
- 立方直接寫 hun*hun*hun；如果用 Math.pow 會得到 double，要再 (int) 轉型。

31 練習4：分解質因數 PrimeFactor.java

for while 鍵盤輸入

題目：將一個正整數分解質因數。例如：輸入 90，印出 90=2*3*3*5。

程式碼

```
/*  執行方式:
    在 PowerShell 輸入 cmd 按 Enter,提示字元會從 PS C:\Users\USER> 變成 C:\Users\USER>,
    之後照下面的指令打:

    cmd
    cd /d C:\Users\USER\Java
    javac -encoding UTF-8 -d . PrimeFactor.java
    java myJava.practice.PrimeFactor

    也可以按 Win + R,輸入 cmd 按 Enter,直接開 CMD 視窗(就不用打第一行的 cmd)。
*/

/*  檔名:PrimeFactor.java     功能:將正整數分解質因數    */

package myJava.practice;
import java.lang.*;
import java.io.Console;

public class PrimeFactor          //主類別
{
    public static void main(String args[])
    {
      Console console=System.console();
      int num,n;

      System.out.print("請輸入一個大於1的正整數:");
      num=Integer.parseInt(console.readLine());
      if(num<=1)
      {
         System.out.println("輸入錯誤,請輸入大於1的正整數");
         return;
      }

      n=num;
      System.out.print(num + "=");
      for(int i=2;i<=n;i++)
      {
         while(n%i==0)                    //i是因數就一直除,直到不能整除
         {
            n=n/i;
            System.out.print(i);
            if(n!=1)                      //還沒除完才印*
```

```
                System.out.print("*");
            }
        }
        System.out.println();
    }
}
```

CMD 指令

```
cd /d C:\java
javac -encoding UTF-8 -d . PrimeFactor.java
java myJava.practice.PrimeFactor
```

範例輸入（每行輸入後按 Enter）

```
90
```

執行結果

```
請輸入一個大於1的正整數:90
90=2*3*3*5

（輸入 97 時）
請輸入一個大於1的正整數:97
97=97
```

重點

- 從 i=2 開始試除；能整除就一直除（while），把同一個質因數全部除完，例如 90 → 45 時 2 除完，再換 3。
- 因為小的因數先被除光，所以能整除 n 的 i 一定是質數，不必另外判斷。
- n 除到剩 1 就是分解完了；n!=1 時才印 *，最後一個因數後面就不會多一個 *。
- 迴圈條件 i<=n 會隨著 n 變小而提早結束。

32 練習5：巢狀條件運算子判斷成績 CondGrade.java

條件運算子 鍵盤輸入

題目：利用條件運算子的巢狀來完成此題：學習成績 >=90 分的同學用 A 表示，60~89 分之間的用 B 表示，60 分以下的用 C 表示。

程式碼

```
/*  執行方式:
    在 PowerShell 輸入 cmd 按 Enter,提示字元會從 PS C:\Users\USER> 變成 C:\Users\USER>,
    之後照下面的指令打:

    cmd
    cd /d C:\Users\USER\Java
    javac -encoding UTF-8 -d . CondGrade.java
    java myJava.practice.CondGrade

    也可以按 Win + R,輸入 cmd 按 Enter,直接開 CMD 視窗(就不用打第一行的 cmd)。
*/

/*  檔名:CondGrade.java     功能:用巢狀條件運算子判斷成績等級A、B、C    */

package myJava.practice;
import java.lang.*;
import java.io.Console;

public class CondGrade          //主類別
{
    public static void main(String args[])
    {
      Console console=System.console();
      int score;
      char grade;

      System.out.print("請輸入成績:");
      score=Integer.parseInt(console.readLine());

      grade=(score>=90) ? 'A' : ((score>=60) ? 'B' : 'C');   //巢狀條件運算子
      System.out.println(score + "分的等級為" + grade);
    }
}
```

CMD 指令

```
cd /d C:\java
javac -encoding UTF-8 -d . CondGrade.java
java myJava.practice.CondGrade
```

範例輸入（每行輸入後按 Enter）

```
75
```

執行結果

```
請輸入成績:75
75分的等級為B

（輸入 95）95分的等級為A
（輸入 45）45分的等級為C
```

重點

- 條件運算子：條件？成立時的值 : 不成立時的值，例如 (5>3) ? "YES" : "NO" 得到 YES。
- 巢狀：把「不成立時的值」再放一個條件運算子：(score>=90) ? 'A' : ((score>=60) ? 'B' : 'C')。
- 效果和 if … else if … else 相同；和 ScoreGrade.java 的 switch 寫法比較看看。

33 練習6：最大公因數與最小公倍數 GcdLcm.java

while 鍵盤輸入

題目：輸入兩個正整數 m 和 n，求其最大公因數和最小公倍數。

程式碼

```
/*  執行方式:
    在 PowerShell 輸入 cmd 按 Enter,提示字元會從 PS C:\Users\USER> 變成 C:\Users\USER>,
    之後照下面的指令打:

    cmd
    cd /d C:\Users\USER\Java
    javac -encoding UTF-8 -d . GcdLcm.java
    java myJava.practice.GcdLcm

    也可以按 Win + R,輸入 cmd 按 Enter,直接開 CMD 視窗(就不用打第一行的 cmd)。
*/

/*  檔名:GcdLcm.java     功能:輾轉相除法求最大公因數與最小公倍數    */

package myJava.practice;
import java.lang.*;
import java.io.Console;

public class GcdLcm          //主類別
{
    public static void main(String args[])
    {
      Console console=System.console();
      int m,n,a,b,r,gcd,lcm;

      System.out.print("請輸入正整數m:");
      m=Integer.parseInt(console.readLine());
      System.out.print("請輸入正整數n:");
      n=Integer.parseInt(console.readLine());

      a=m;
      b=n;
      while(b!=0)                        //輾轉相除法:餘數為0時,a就是最大公因數
      {
         r=a%b;
         a=b;
         b=r;
      }
      gcd=a;
      lcm=m/gcd*n;                       //兩數乘積 = 最大公因數 × 最小公倍數

      System.out.println("最大公因數為:" + gcd);
```

```
        System.out.println("最小公倍數為:" + lcm);
    }
}
```

CMD 指令

```
cd /d C:\java
javac -encoding UTF-8 -d . GcdLcm.java
java myJava.practice.GcdLcm
```

範例輸入（每行輸入後按 Enter）

```
12
18
```

執行結果

```
請輸入正整數m:12
請輸入正整數n:18
最大公因數為:6
最小公倍數為:36
```

重點

- 輾轉相除法：r=a%b，再 a=b、b=r，一直做到 b 變成 0，此時的 a 就是最大公因數。
- 12、18：12%18=12 → (18,12)；18%12=6 → (12,6)；12%6=0 → (6,0)，最大公因數為 6。
- 不用先比大小：a<b 時第一次 a%b 等於 a，兩數會自動對調。
- 兩數乘積 = 最大公因數 × 最小公倍數，所以 lcm = m/gcd*n（先除再乘比較不會溢位）。

34 練習7：統計英文字母、空格、數字和其他字元 CharCount.java

for 字元 鍵盤輸入

題目：輸入一行字元，分別統計出其英文字母、空格、數字和其它字元的個數。

程式碼

```
/*  執行方式:
    在 PowerShell 輸入 cmd 按 Enter,提示字元會從 PS C:\Users\USER> 變成 C:\Users\USER>,
    之後照下面的指令打:

    cmd
    cd /d C:\Users\USER\Java
    javac -encoding UTF-8 -d . CharCount.java
    java myJava.practice.CharCount

    也可以按 Win + R,輸入 cmd 按 Enter,直接開 CMD 視窗(就不用打第一行的 cmd)。
*/

/*  檔名:CharCount.java     功能:統計一行字元中英文字母、空格、數字和其他字元的個數    */

package myJava.practice;
import java.lang.*;
import java.io.Console;

public class CharCount          //主類別
{
    public static void main(String args[])
    {
      Console console=System.console();
      String str;
      char ch;
      int letter=0,space=0,digit=0,other=0;

      System.out.print("請輸入一行字元:");
      str=console.readLine();

      for(int i=0;i<str.length();i++)
      {
         ch=str.charAt(i);              //取出第i個字元
         if((ch>='a' && ch<='z') || (ch>='A' && ch<='Z'))
            letter++;
         else if(ch==' ')
            space++;
         else if(ch>='0' && ch<='9')
            digit++;
         else
            other++;
      }
```

```
      System.out.println("英文字母:" + letter + "個");
      System.out.println("空格:" + space + "個");
      System.out.println("數字:" + digit + "個");
      System.out.println("其他字元:" + other + "個");
    }
}
```

CMD 指令

```
cd /d C:\java
javac -encoding UTF-8 -d . CharCount.java
java myJava.practice.CharCount
```

範例輸入（每行輸入後按 Enter）

```
Hello World 123!
```

執行結果

```
請輸入一行字元:Hello World 123!
英文字母:10個
空格:2個
數字:3個
其他字元:1個
```

重點

- readLine() 讀整行（包含空格），str.length() 是字元個數，str.charAt(i) 取出第 i 個字元。
- 字元可以直接比大小：ch>='a' && ch<='z'、ch>='0' && ch<='9'，不用背 ASCII 碼（'A'=65、'a'=97、'0'=48、空格=32）。
- 用 if … else if … else 分類，最後的 else 就是「其他字元」。
- 中文字也會算進其他字元。

35 練習8：求 s=a+aa+aaa+... SumAAA.java

for 鍵盤輸入

題目：求 s=a+aa+aaa+aaaa+aa...a 的值，其中 a 是一個數字。假設 a=2，項數為 3，則輸出結果的形式如：2+22+222=246。

程式碼

```
/*  執行方式:
    在 PowerShell 輸入 cmd 按 Enter,提示字元會從 PS C:\Users\USER> 變成 C:\Users\USER>,
    之後照下面的指令打:

    cmd
    cd /d C:\Users\USER\Java
    javac -encoding UTF-8 -d . SumAAA.java
    java myJava.practice.SumAAA

    也可以按 Win + R,輸入 cmd 按 Enter,直接開 CMD 視窗(就不用打第一行的 cmd)。
*/

/*  檔名:SumAAA.java     功能:求s=a+aa+aaa+...的值    */

package myJava.practice;
import java.lang.*;
import java.io.Console;

public class SumAAA          //主類別
{
    public static void main(String args[])
    {
      Console console=System.console();
      int a,n,term=0,s=0;

      System.out.print("請輸入數字a(1~9):");
      a=Integer.parseInt(console.readLine());
      System.out.print("請輸入項數n:");
      n=Integer.parseInt(console.readLine());

      for(int i=1;i<=n;i++)
      {
         term=term*10+a;                  //2 → 22 → 222 ...
         s=s+term;
         System.out.print(term);
         if(i<n)                          //最後一項後面不印+
            System.out.print("+");
      }
      System.out.println("=" + s);
```

```
    }
}
```

CMD 指令

```
cd /d C:\java
javac -encoding UTF-8 -d . SumAAA.java
java myJava.practice.SumAAA
```

範例輸入（每行輸入後按 Enter）

```
2
3
```

執行結果

```
請輸入數字a(1~9):2
請輸入項數n:3
2+22+222=246

（輸入 a=3、n=5）
3+33+333+3333+33333=37035
```

重點

- 下一項 = 上一項 × 10 + a：2 → 22 → 222，用 term=term*10+a 產生。
- s=s+term 把每一項加起來。
- i<n 時才印 +，最後一項後面接 =。
- int 最多約 21 億，項數超過 9 項就會溢位，要改用 long。

36 練習9：1000 以內的完數 `PerfectNumber.java`

for

題目：一個數如果恰好等於它的因子之和，這個數就稱為「完數」。例如 6=1+2+3。程式設計找出 1000 以內的所有完數。

程式碼

```
/*  執行方式:
    在 PowerShell 輸入 cmd 按 Enter,提示字元會從 PS C:\Users\USER> 變成 C:\Users\USER>,
    之後照下面的指令打:

    cmd
    cd /d C:\Users\USER\Java
    javac -encoding UTF-8 -d . PerfectNumber.java
    java myJava.practice.PerfectNumber

    也可以按 Win + R,輸入 cmd 按 Enter,直接開 CMD 視窗(就不用打第一行的 cmd)。
*/

/*  檔名:PerfectNumber.java     功能:找出1000以內的所有完數    */

package myJava.practice;
import java.lang.*;

public class PerfectNumber          //主類別
{
    public static void main(String args[])
    {
      int sum;

      System.out.println("1000以內的完數:");
      for(int i=2;i<=1000;i++)
      {
         sum=0;                           //每個數都要重新計算
         for(int j=1;j<i;j++)
         {
            if(i%j==0)                    //j是i的因數
               sum=sum+j;
         }
         if(sum==i)
         {
            System.out.print(i + "=1");  //1一定是因數
            for(int j=2;j<i;j++)
               if(i%j==0)
                  System.out.print("+" + j);
            System.out.println();
         }
      }
```

```
    }
}
```

CMD 指令

```
cd /d C:\java
javac -encoding UTF-8 -d . PerfectNumber.java
java myJava.practice.PerfectNumber
```

執行結果

```
1000以內的完數:
6=1+2+3
28=1+2+4+7+14
496=1+2+4+8+16+31+62+124+248
```

重點

- 兩層 for：外層 i 是要檢查的數，內層 j 從 1 跑到 i-1 找因數（i%j==0），加進 sum。
- sum=0 要放在外層迴圈裡面，每個數都重新計算。
- 因數不包含自己，所以內層條件是 j<i。
- sum==i 時，再跑一次迴圈把因數印出來。

補充練習：費氏數列兩種解法

37 費氏數列 解法一：迴圈（Iterative） FibLoop.java

for 函式

題目：費氏數列 1, 1, 2, 3, 5, 8 ...（第 1、2 項是 1，之後每一項 = 前兩項相加）。輸入 n，用迴圈印出前 n 項與第 n 項的值。解法一使用迴圈：利用變數迭代，時間複雜度 O(n)、空間複雜度 O(1)，不會發生 Stack Overflow（堆疊溢位），效能最佳、最推薦。

程式碼

```
/*  執行方式:
    在 PowerShell 輸入 cmd 按 Enter,提示字元會從 PS C:\Users\USER> 變成 C:\Users\USER>,
    之後照下面的指令打:

    cmd
    cd /d C:\Users\USER\Java
    javac -encoding UTF-8 -d . FibLoop.java
    java myJava.practice.FibLoop

    也可以按 Win + R,輸入 cmd 按 Enter,直接開 CMD 視窗(就不用打第一行的 cmd)。
*/

/*  檔名:FibLoop.java     功能:費氏數列 解法一 用迴圈計算(效能最佳)    */

package myJava.practice;
import java.lang.*;
import java.io.Console;

public class FibLoop          //主類別
{
    public static void main(String args[])
    {
      Console console=System.console();
      int n;

      System.out.print("請輸入n(1~92):");
      n=Integer.parseInt(console.readLine());
      if(n<1 || n>92)                    //第93項會超過Long的範圍
      {
         System.out.println("輸入錯誤,請輸入1~92");
         return;
      }

      System.out.print("前" + n + "項:");
      for(int i=1;i<=n;i++)
         System.out.print(fib(i) + " ");
      System.out.println();
```

```
      System.out.println("第" + n + "項為:" + fib(n));
    }

    public static long fib(int n)
    {
        long a=1,b=1,c;                    //a是前兩項,b是前一項
        if(n<=2)                           //第1、2項都是1
            return 1;
        for(int i=3;i<=n;i++)
        {
            c=a+b;                         //這一項 = 前兩項相加
            a=b;                           //兩個變數往後移一格
            b=c;
        }
        return b;
    }
}
```

CMD 指令

```
cd /d C:\java
javac -encoding UTF-8 -d . FibLoop.java
java myJava.practice.FibLoop
```

範例輸入（每行輸入後按 Enter）

```
10
```

執行結果

```
請輸入n(1~92):10
前10項:1 1 2 3 5 8 13 21 34 55
第10項為:55

（輸入 93 時）
請輸入n(1~92):93
輸入錯誤,請輸入1~92
```

重點

- 只用兩個變數 a（前兩項）、b（前一項）：c=a+b 算出這一項，再 a=b、b=c 往後移一格。
- 時間複雜度 O(n)：第 n 項只要做 n-2 次加法；空間複雜度 O(1)：不管 n 多大都只用 a、b、c 三個變數。
- 沒有函式呼叫自己，所以不會因為呼叫太多層而 Stack Overflow（堆疊溢位）。
- 數字長得很快，要用 long；第 93 項會超過 long 的範圍（約 9.2×10^{18}），所以限制 n 在 1~92。

38 費氏數列 解法二：遞迴（Recursive）與記憶化 `FibRecursive.java`

recursive　函式　陣列

題目：用遞迴計算費氏數列第 n 項，並分析優缺點：為什麼一般遞迴寫法效能不好？如何用記憶化（Memoization）優化？程式同時算出兩種遞迴的結果，並比較函式被呼叫的次數。

程式碼

```
/*  執行方式:
    在 PowerShell 輸入 cmd 按 Enter,提示字元會從 PS C:\Users\USER> 變成 C:\Users\USER>,
    之後照下面的指令打:

    cmd
    cd /d C:\Users\USER\Java
    javac -encoding UTF-8 -d . FibRecursive.java
    java myJava.practice.FibRecursive

    也可以按 Win + R,輸入 cmd 按 Enter,直接開 CMD 視窗(就不用打第一行的 cmd)。
*/

/*  檔名:FibRecursive.java     功能:費氏數列 解法二 遞迴,並比較加上記憶化(Memoization)的差別
*/

package myJava.practice;
import java.lang.*;
import java.io.Console;

public class FibRecursive          //主類別
{
    static long count=0;               //記錄函式被呼叫幾次
    static long memo[]=new long[93];   //記憶化:存算過的答案,0代表還沒算過

    public static void main(String args[])
    {
      Console console=System.console();
      int n;

      System.out.print("請輸入n(1~40):");
      n=Integer.parseInt(console.readLine());
      if(n<1 || n>40)                    //一般遞迴n太大會算很久
      {
         System.out.println("輸入錯誤,請輸入1~40");
         return;
      }

      count=0;
      System.out.println("一般遞迴:  第" + n + "項為" + fib(n) + ",函式呼叫" + count +
"次");
```

```
        count=0;
        System.out.println("記憶化遞迴:第" + n + "項為" + fibMemo(n) + ",函式呼叫" + count +
"次");
    }

    public static long fib(int n)        //一般遞迴
    {
        count++;
        if(n<=2)                         //邊界條件:第1、2項都是1
            return 1;
        return fib(n-1)+fib(n-2);        //同一項會被重複計算很多次
    }

    public static long fibMemo(int n)    //記憶化遞迴
    {
        count++;
        if(n<=2)
            return 1;
        if(memo[n]!=0)                   //算過了,直接拿出來用
            return memo[n];
        memo[n]=fibMemo(n-1)+fibMemo(n-2);   //第一次算,存起來
        return memo[n];
    }
}
```

CMD 指令

```
cd /d C:\java
javac -encoding UTF-8 -d . FibRecursive.java
java myJava.practice.FibRecursive
```

範例輸入（每行輸入後按 Enter）

```
30
```

執行結果

```
請輸入n(1~40):30
一般遞迴:  第30項為832040,函式呼叫1664079次
記憶化遞迴:第30項為832040,函式呼叫57次

（輸入 10 時）
一般遞迴:  第10項為55,函式呼叫109次
記憶化遞迴:第10項為55,函式呼叫17次
```

重點

- 一般遞迴：return fib(n-1)+fib(n-2); 邊界條件 n<=2 時回傳 1。寫法最直覺，和數學定義一模一樣（優點）。
- 缺點一：重複計算。fib(5) 會呼叫 fib(4) 和 fib(3)，fib(4) 又再算一次 fib(3)……呼叫次數以指數成長，時間複雜度約 $O(2^n)$；n=30 就呼叫了 166 萬次。
- 缺點二：每呼叫一層都要佔用堆疊（stack）空間，深度是 n，空間複雜度 O(n)；n 非常大時會 Stack Overflow。
- 優化：記憶化（Memoization）用陣列 memo[] 存算過的答案，算之前先檢查 memo[n]!=0 就直接回傳，每一項只算一次，時間複雜度降到 O(n)，n=30 只呼叫 57 次。
- static 變數 count 放在函式外面，兩個函式都能使用，用來數呼叫次數；比較前要先歸零。
- 考試常問：「為什麼這個寫法不好？」答重複計算、指數時間；「如何優化？」答記憶化或改用迴圈（解法一）。

補充練習：overloading（多載）

39 多載1：參數個數不同與型態不同的 add、max `AddMax.java`

overloading　函式

題目：使用多載設計 int add(int a,int b)、int add(int a,int b,int c)、double add(double a,double b)，以及 max(int,int)、max(int,int,int)、max(double,double)，並分別呼叫印出結果。

程式碼

```
/*  執行方式:
    在 PowerShell 輸入 cmd 按 Enter,提示字元會從 PS C:\Users\USER> 變成 C:\Users\USER>,
    之後照下面的指令打:

    cmd
    cd /d C:\Users\USER\Java
    javac -encoding UTF-8 -d . AddMax.java
    java myJava.practice.AddMax

    也可以按 Win + R,輸入 cmd 按 Enter,直接開 CMD 視窗(就不用打第一行的 cmd)。
*/

/*  檔名:AddMax.java     功能:多載 參數個數不同與型態不同的add、max    */

package myJava.practice;
import java.lang.*;

public class AddMax          //主類別
{
    public static void main(String args[])
    {
      System.out.println("add(3,5)=" + add(3,5));             //呼叫add(int,int)
      System.out.println("add(3,5,7)=" + add(3,5,7));         //呼叫add(int,int,int)
      System.out.println("add(2.5,1.2)=" + add(2.5,1.2));     //呼叫add(double,double)
      System.out.println("max(8,3)=" + max(8,3));             //呼叫max(int,int)
      System.out.println("max(4,9,6)=" + max(4,9,6));         //呼叫max(int,int,int)
      System.out.println("max(2.5,7.1)=" + max(2.5,7.1));     //呼叫max(double,double)
    }

    public static int add(int a,int b)                //兩個int
    {
        return a+b;
    }

    public static int add(int a,int b,int c)          //參數個數不同:三個int
    {
        return a+b+c;
    }
```

```
    public static double add(double a,double b)       //參數型態不同:兩個double
    {
        return a+b;
    }

    public static int max(int a,int b)
    {
        return (a>b) ? a : b;
    }

    public static int max(int a,int b,int c)
    {
        return max(max(a,b),c);       //呼叫兩個參數的max:先比a、b,再和c比
    }

    public static double max(double a,double b)
    {
        return (a>b) ? a : b;
    }
}
```

CMD 指令

```
cd /d C:\java
javac -encoding UTF-8 -d . AddMax.java
java myJava.practice.AddMax
```

執行結果

```
add(3,5)=8
add(3,5,7)=15
add(2.5,1.2)=3.7
max(8,3)=8
max(4,9,6)=9
max(2.5,7.1)=7.1
```

重點

- 多載的兩種方式：參數「個數」不同（add(int,int) 和 add(int,int,int)），或參數「型態」不同（add(int,int) 和 add(double,double)）。
- Java 依照呼叫時引數的個數和型態，自動選擇要執行哪一個版本：add(3,5) 是兩個 int，add(2.5,1.2) 是兩個 double。
- 三個數的 max 直接呼叫兩個數的 max：return max(max(a,b),c);，先找出 a、b 較大的，再和 c 比，不必重寫比較。
- (a>b) ? a : b 是條件運算子，等於 if(a>b) return a; else return b;。

40 多載2：求圓形、長方形、正方形面積 area `Area.java`

overloading　函式

題目：使用多載設計三個同名函式 area：double area(double r) 求圓面積、double area(double w,double h) 求長方形面積、int area(int side) 求正方形面積。

程式碼

```
/*  執行方式:
    在 PowerShell 輸入 cmd 按 Enter,提示字元會從 PS C:\Users\USER> 變成 C:\Users\USER>,
    之後照下面的指令打:

    cmd
    cd /d C:\Users\USER\Java
    javac -encoding UTF-8 -d . Area.java
    java myJava.practice.Area

    也可以按 Win + R,輸入 cmd 按 Enter,直接開 CMD 視窗(就不用打第一行的 cmd)。
*/

/*  檔名:Area.java     功能:多載 用同一個名稱area計算圓形、長方形、正方形面積    */

package myJava.practice;
import java.lang.*;

public class Area          //主類別
{
    public static void main(String args[])
    {
      System.out.println("半徑2.0的圓面積=" + area(2.0));           //呼叫area(double)
      System.out.println("長3.0寬4.5的長方形面積=" + area(3.0,4.5)); //呼叫area(double,doub
le)
      System.out.println("邊長5的正方形面積=" + area(5));            //呼叫area(int)
    }

    public static double area(double r)               //圓形:半徑
    {
        return Math.PI*r*r;
    }

    public static double area(double w,double h)      //長方形:長、寬
    {
        return w*h;
    }

    public static int area(int side)                  //正方形:邊長(int)
    {
        return side*side;
```

```
    }
}
```

CMD 指令

```
cd /d C:\java
javac -encoding UTF-8 -d . Area.java
java myJava.practice.Area
```

執行結果

```
半徑2.0的圓面積=12.566370614359172
長3.0寬4.5的長方形面積=13.5
邊長5的正方形面積=25
```

重點

- 同一件事（算面積）用同一個名稱，呼叫的人不用記 circleArea、rectArea 等不同名稱，這就是多載的好處。
- area(2.0) 是一個 double → 圓形；area(3.0,4.5) 是兩個 double → 長方形；area(5) 是一個 int → 正方形。
- 注意 area(2.0) 和 area(5) 參數個數一樣，靠型態 double 和 int 區分；如果寫成 area(2)，會呼叫 int 版本，印出 4，不是圓面積。
- Math.PI 是圓周率常數 3.141592653589793。

41 多載3：印出不同型態的一維陣列 printArray `PrintArray.java`

overloading　函式　陣列　for

題目：使用多載設計 void printArray(int x[])、void printArray(double x[])、void printArray(char x[])，分別印出陣列的所有元素與個數。

程式碼

```
/*  執行方式:
    在 PowerShell 輸入 cmd 按 Enter,提示字元會從 PS C:\Users\USER> 變成 C:\Users\USER>,
    之後照下面的指令打:

    cmd
    cd /d C:\Users\USER\Java
    javac -encoding UTF-8 -d . PrintArray.java
    java myJava.practice.PrintArray

    也可以按 Win + R,輸入 cmd 按 Enter,直接開 CMD 視窗(就不用打第一行的 cmd)。
*/

/*  檔名:PrintArray.java     功能:多載 印出int、double、char三種一維陣列    */

package myJava.practice;
import java.lang.*;

public class PrintArray          //主類別
{
    public static void main(String args[])
    {
      int a[]={5,3,8,1};
      double b[]={1.5,2.25,3.0};
      char c[]={'J','a','v','a'};

      printArray(a);                  //呼叫printArray(int x[])
      printArray(b);                  //呼叫printArray(double x[])
      printArray(c);                  //呼叫printArray(char x[])
    }

    public static void printArray(int x[])
    {
        System.out.print("int陣列:   ");
        for(int i=0;i<x.length;i++)
            System.out.print(x[i] + " ");
        System.out.println("(共" + x.length + "個)");
    }

    public static void printArray(double x[])
    {
        System.out.print("double陣列:");
```

```
        for(int i=0;i<x.length;i++)
            System.out.print(x[i] + " ");
        System.out.println("(共" + x.length + "個)");
    }

    public static void printArray(char x[])
    {
        System.out.print("char陣列:  ");
        for(int i=0;i<x.length;i++)
            System.out.print(x[i] + " ");
        System.out.println("(共" + x.length + "個)");
    }
}
```

CMD 指令

```
cd /d C:\java
javac -encoding UTF-8 -d . PrintArray.java
java myJava.practice.PrintArray
```

執行結果

```
int陣列:   5 3 8 1 (共4個)
double陣列:1.5 2.25 3.0 (共3個)
char陣列:  J a v a (共4個)
```

重點

- 參數是不同型態的陣列（int[]、double[]、char[]），也算參數型態不同，可以多載。
- 和 MaxEle.java（二維陣列找最大值）是同一種題型，這題換成一維陣列。
- x.length 是陣列元素個數，for 從 0 跑到 x.length-1。
- void 表示沒有回傳值，函式只負責印出；System.out.println 本身也是多載，int、double、char 都能印。

42 多載4：多載加遞迴的 power `PowerOverload.java`

overloading recursive 函式

題目：使用多載與遞迴設計 long power(int x,int y) 與 double power(double x,int y)，計算 x 的 y 次方。整數版本 y<0 時回傳 0；小數版本 y<0 時回傳 1/(x 的 -y 次方)。

程式碼

```
/*  執行方式:
    在 PowerShell 輸入 cmd 按 Enter,提示字元會從 PS C:\Users\USER> 變成 C:\Users\USER>,
    之後照下面的指令打:

    cmd
    cd /d C:\Users\USER\Java
    javac -encoding UTF-8 -d . PowerOverload.java
    java myJava.practice.PowerOverload

    也可以按 Win + R,輸入 cmd 按 Enter,直接開 CMD 視窗(就不用打第一行的 cmd)。
*/

/*  檔名:PowerOverload.java     功能:多載加遞迴 int與double兩種版本的power    */

package myJava.practice;
import java.lang.*;

public class PowerOverload          //主類別
{
    public static void main(String args[])
    {
      System.out.println("power(2,10)=" + power(2,10));         //呼叫power(int,int)
      System.out.println("power(2,40)=" + power(2,40));         //超過int範圍,所以回傳long
      System.out.println("power(2.0,10)=" + power(2.0,10));     //呼叫power(double,int)
      System.out.println("power(1.5,3)=" + power(1.5,3));
      System.out.println("power(2.0,-2)=" + power(2.0,-2));     //負次方:1/(2的2次方)
    }

    public static long power(int x,int y)             //整數版本
    {
        if(y<0)                       //整數的負次方不是整數,回傳0
            return 0;
        if(y==0)                      //邊界條件:x的0次方為1
            return 1;
        return x*power(x,y-1);        //遞迴呼叫
    }

    public static double power(double x,int y)        //小數版本
    {
        if(y<0)                       //負次方:x的-y次方 = 1/(x的y次方)
            return 1/power(x,-y);
```

```
        if(y==0)
            return 1;
        return x*power(x,y-1);          //x是double,所以呼叫的是自己(double版本)
    }
}
```

CMD 指令

```
cd /d C:\java
javac -encoding UTF-8 -d . PowerOverload.java
java myJava.practice.PowerOverload
```

執行結果

```
power(2,10)=1024
power(2,40)=1099511627776
power(2.0,10)=1024.0
power(1.5,3)=3.375
power(2.0,-2)=0.25
```

重點

- 一題同時考兩個期中範圍：多載（int 版與 double 版）+ 遞迴（return x*power(x,y-1);）。
- power(2,10) 兩個都是 int → 呼叫 long 版本，印出 1024；power(2.0,10) 第一個是 double → 呼叫 double 版本，印出 1024.0。
- 整數版本回傳 long：2 的 40 次方 = 1099511627776，超過 int 範圍（約 21 億）。x*power(x,y-1) 是 int×long，結果自動變成 long。
- 遞迴時 x 的型態不變，所以 int 版本呼叫的還是 int 版本、double 版本呼叫的還是 double 版本。
- 和 MyPower.java 比較：MyPower 只有 double 版本；這題多一個整數版本，才能看出多載的差別。

由 build.ps1 依照 problems.json 與 src 資料夾自動產生。