



JAVA 基本觀念-模組化設計

修平科技大學 資訊網路技術系
陳文敬老師 jameschen@hust.edu.tw

Agenda

1. 模組化設計的好處

2. 常用的系統內建模組

- Math, Wrapper Class (Integer, Float, Double, ...)
- String, StringBuilder, StringBuffer
- System, Arrays

3. 模組化設計

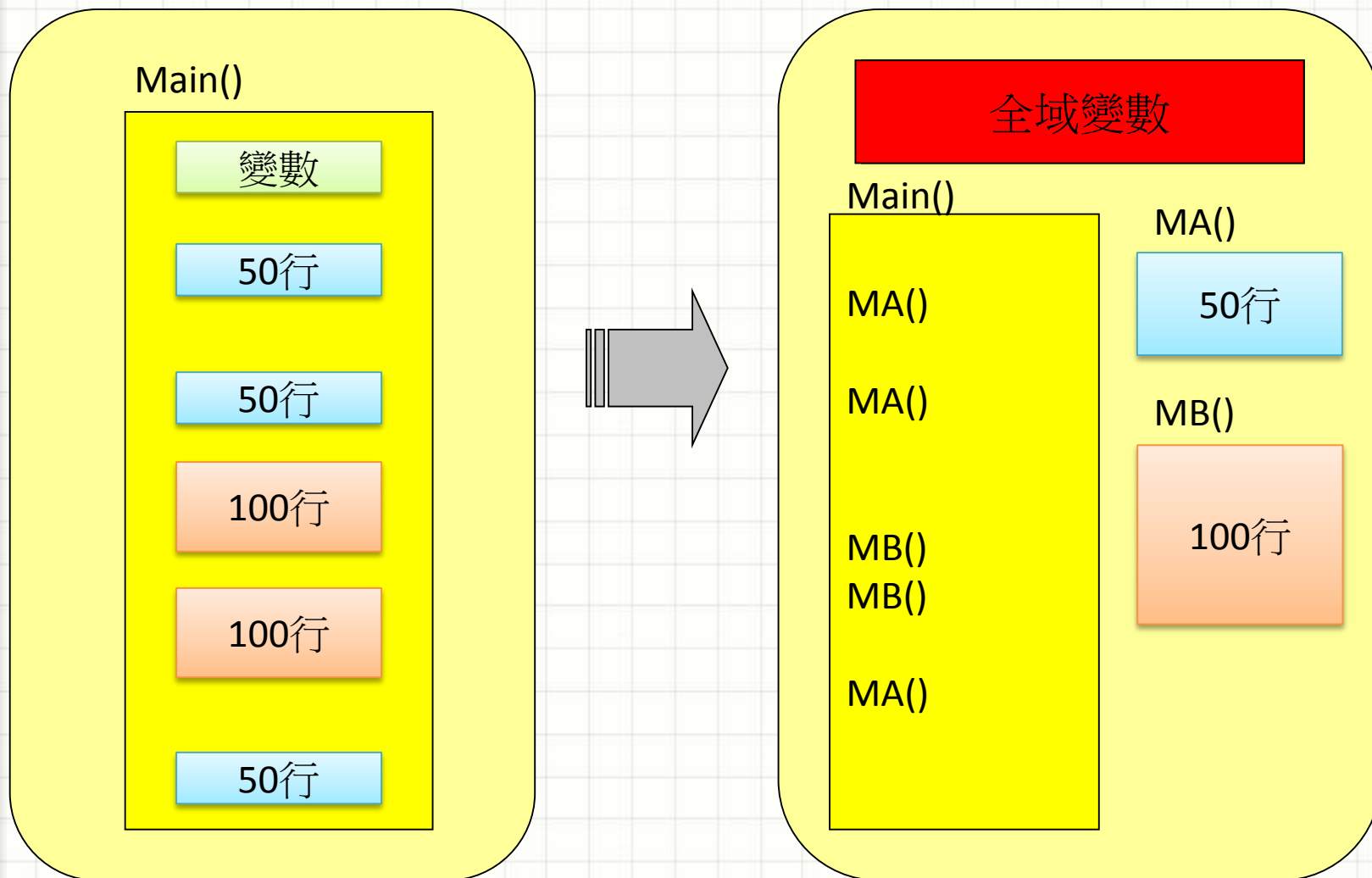
- 自訂方法(Method)、副程式(sub-routine/procedure/function)
- 自訂類別(class)、套件(package)
- 參數的傳遞 : call-by-value, call-by-reference

4. 遞迴的解題技巧

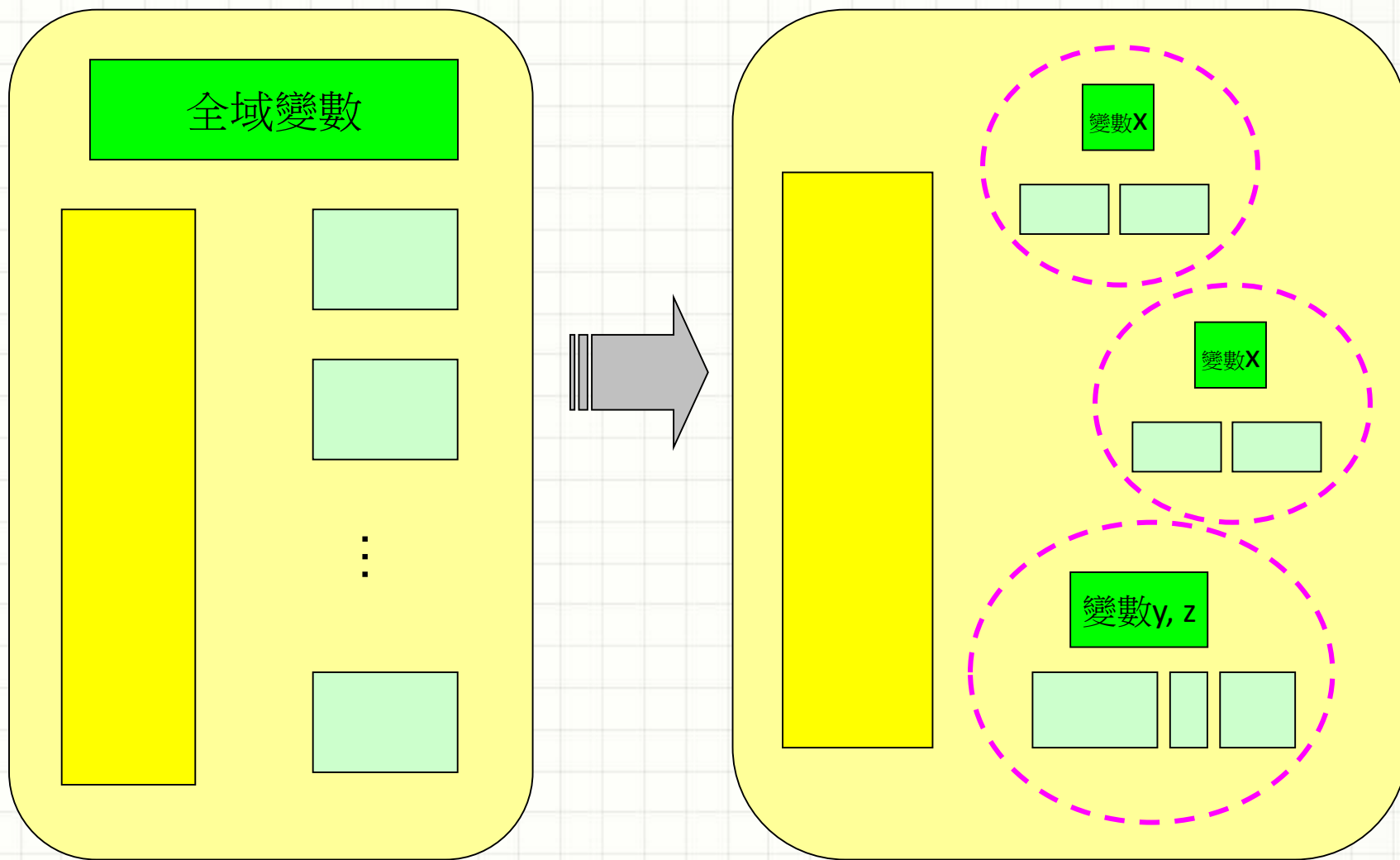
傳統系統開發的常見問題

- **維護**: 不容易修改、維護(牽一髮而動全身)
- **穩定**: 因為系統變大而導致系統不穩定 (人多手雜、誤用資料)、容易崩潰?
- **重用**: 新的專案必須重新開發、重覆使用的機率過低...
- **擴充**: 難以增加系統功能、造成疊床架屋...

程式結構的演變 - 1 (模組化)



程式結構的演變 -2 (類別/物件化)



學習物件導向技術!!

- 避免不正常的存取：資訊隱藏(Information hiding)
 - 將相關的資料與處理資料的程式(副程式)加以規範
 - 利用封裝與存取權限加以限制
- 提高程式碼的再利用率 (Reusability)
 - 元件模組化的設計思維 (component based design)
 - 以類別(物件)為最基本模組，包含相關方法(副程式)
 - 利用繼承機制來改良舊有功能或擴充新功能。
- 利用抽象化來提高系統的可擴充性 (Extensibility)
 - 利用標準介面來降低元件之間的耦合度 (loosely couple)

先學好模組化設計 ... 好處是:

1. 能夠重複使用、分享(共用)
2. 方便維護(修改)、確保程式一致性
3. 程式碼變簡潔(變小)
4. 方便閱讀程式
5.

常用的內建模組

- **數學運算 : Math**
- **外覆型別 : Integer, Float, Double, ...**
- **字串處理 : String**
- **日期 : Date, Calendar**

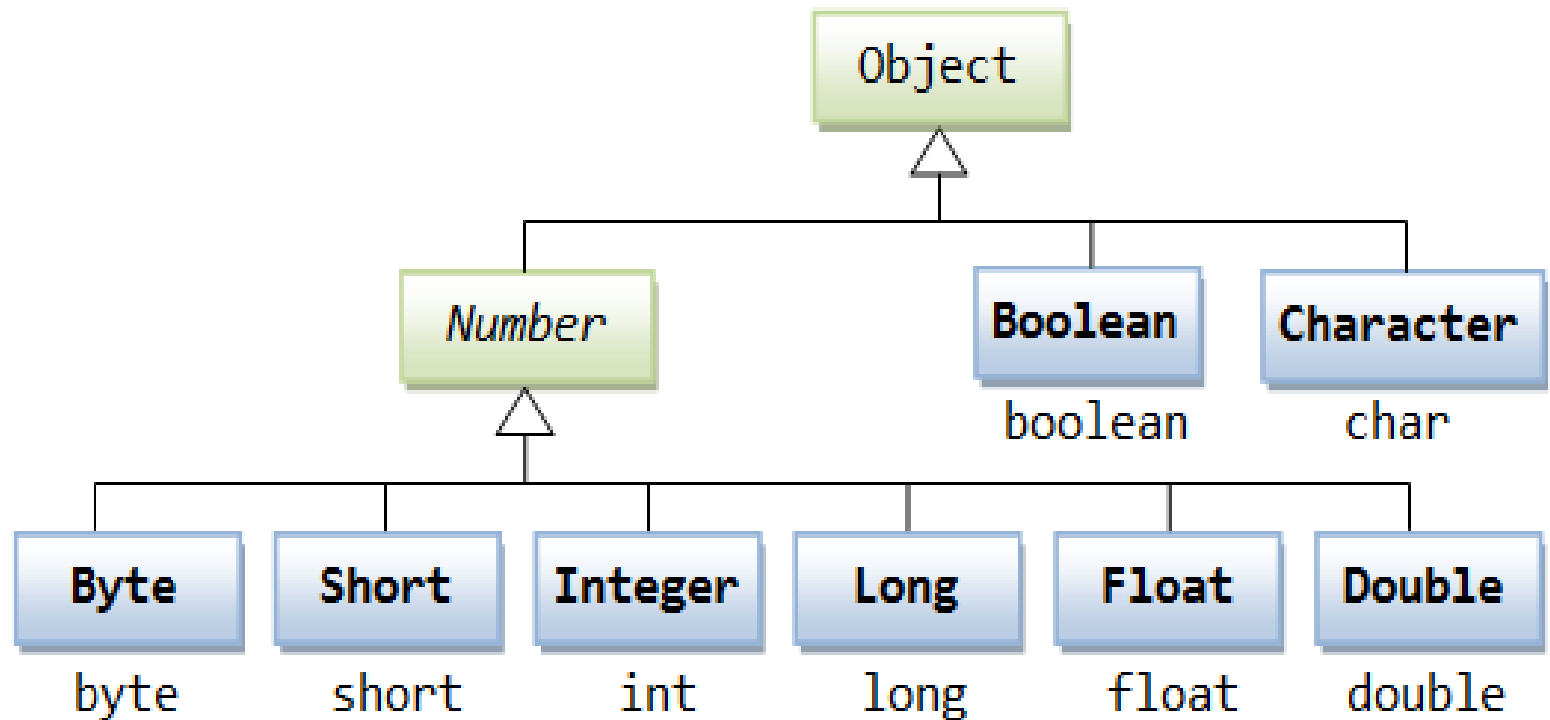
Math 數學模組

Constant	Description
Math.E	2.7182818...
Math.PI	3.1415926...

回傳值	方法名稱	說明
double	Math.abs (<i>value</i>)	absolute value
double	Math.ceil (<i>value</i>)	rounds up
double	Math.floor (<i>value</i>)	rounds down
double	Math.log10 (<i>value</i>)	logarithm, base 10
double	Math.max (<i>value1</i> , <i>value2</i>)	larger of two values
double	Math.min (<i>value1</i> , <i>value2</i>)	smaller of two values
double	Math.pow (<i>base</i> , <i>exp</i>)	<i>base</i> to the <i>exp</i> power
double	Math.random ()	random double between 0 and 1
double	Math.round (<i>value</i>)	nearest whole number
double	Math.sqrt (<i>value</i>)	square root
double double double	Math.sin (<i>value</i>) Math.cos (<i>value</i>) Math.tan (<i>value</i>)	sine/cosine/tangent of an angle in radians
double double	Math.toDegrees (<i>value</i>) Math.toRadians (<i>value</i>)	convert degrees to radians and back

外覆型別(Wrapper Classes)

- 資料轉換與解析



學習查詢 Java Docs (APIs)

The screenshot shows a web browser window displaying the Java Platform 7 API documentation for the `System` class. The browser's address bar shows the URL <https://docs.oracle.com/javase/7/docs/api/>. The page has a navigation bar with tabs for Overview, Package, Class (selected), Use, Tree, Deprecated, Index, and Help. Below the navigation bar, there are links for Prev Class, Next Class, Frames, and No Frames. The main content area shows the class hierarchy: `java.lang` > `java.lang.Object` > `java.lang.System`. The class signature is `public final class System` extending `Object`. A description states: "The `System` class contains several useful class fields and methods. It cannot be instantiated." Another paragraph mentions: "Among the facilities provided by the `System` class are standard input, standard output, and error output streams; access to externally defined properties and environment variables; a means of loading files and libraries; and a utility method for quickly copying an array." The left sidebar contains a list of packages and classes, with `System` highlighted.

System (Java Platform 7) x

安全 | <https://docs.oracle.com/javase/7/docs/api/>

java.lang
java.lang.annotation
java.lang.instrument
java.lang.invoke
java.lang.management
java.lang.ref
java.lang.reflect
java.math
java.net
java.nio
java.nio.channels
java.nio.channels.spi
RuntimePermission
SecurityManager
Short
StackTraceElement
StrictMath
String
StringBuffer
StringBuilder
System
Thread
ThreadGroup

Overview Package **Class** Use Tree Deprecated Index Help

Java™ Platform Standard Ed. 7

Prev Class Next Class Frames No Frames

Summary: Nested | Field | Constr | Method Detail: Field | Constr | Method

java.lang

Class System

java.lang.Object
java.lang.System

`public final class System`
`extends Object`

The `System` class contains several useful class fields and methods. It cannot be instantiated.

Among the facilities provided by the `System` class are standard input, standard output, and error output streams; access to externally defined properties and environment variables; a means of loading files and libraries; and a utility method for quickly copying an array.

<https://docs.oracle.com/javase/7/docs/api/java/lang/System.html>

<https://docs.oracle.com/javase/7/docs/api/>

學習查詢 Java Docs (APIs)

Oracle JDK 9 Document Overview (Java SE 9 & J...

安全 | <https://docs.oracle.com/javase/9/docs/api/index.html?overview-summary.html>

Java SE 9 & JDK 9

ALL CLASSES ALL PACKAGES

Modules

- java.activation
- java.base
- java.compiler
- java.corba
- java.datatransfer

All Classes

- AboutEvent
- AboutHandler
- AbsentInformationException
- AbstractAction
- AbstractAnnotationValueVisitor6
- AbstractAnnotationValueVisitor7
- AbstractAnnotationValueVisitor8
- AbstractAnnotationValueVisitor9
- AbstractBorder
- AbstractButton
- AbstractCellEditor
- AbstractChronology
- AbstractCollection
- AbstractColorChooserPanel
- AbstractDocument
- AbstractDocument.AttributeContext
- AbstractDocument.Content
- AbstractDocument.ElementEdit
- AbstractElementVisitor6
- AbstractElementVisitor7
- AbstractElementVisitor8
- AbstractElementVisitor9
- AbstractExecutorService
- AbstractInterruptibleChannel
- AbstractList

OVERVIEW MODULE PACKAGE CLASS USE TREE DEPRECATED INDEX HELP

PREV NEXT FRAMES NO FRAMES

SEARCH:

Java® Platform, Standard Edition & Java Development Kit Version 9 API Specification

This document is divided into three sections:

- Java SE**
The Java Platform, Standard Edition (Java SE) APIs define the core Java platform for general-purpose computing. These APIs are in modules whose names start with java.
- JDK**
The Java Development Kit (JDK) APIs are specific to the JDK and will not necessarily be available in all implementations of the Java SE Platform. These APIs are in modules whose names start with jdk.
- JavaFX**
The JavaFX APIs define a set of user-interface controls, graphics, media, and web packages for developing rich client applications. These APIs are in modules whose names start with javafx.

Java SE

Module	Description
java.activation	Defines the JavaBeans Activation Framework (JAF) API.
java.base	Defines the foundational APIs of the Java SE Platform.

<https://docs.oracle.com/javase/9/>

練習時間：樂透開獎

- 利用系統模組寫一個樂透開獎
- 要求：由 1~49 之間選取六個相異的數字
- 作法：利用系統內部模組 *Math.random()*

*(int) (Math.random() * 49 + 1)*

自訂模組(方法, Method) – 不帶參數

- 決定方法的名稱、執行參數、回傳值

```
public static void printTriangle( ) {
```

```
    for(int i=1; i<=7 ; i++) {
```

```
        for(int j=1; j<=i; j++) {
```

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

```
        }
```

```
        System.out.print("\n");
```

```
    }
```

```
    System.out.print("\n");
```

```
}
```

```
*
```

```
**
```

```
***
```

```
****
```

```
*****
```

```
*****
```

```
*****
```

自訂模組(方法, Method) – 有參數

```
1. public static void printTriangle(int n) {  
2.     for(int i=1; i<=n ; i++) {  
3.         for(int j=1; j<=i; j++) {  
4.             System.out.print("*");  
5.         }  
6.         System.out.print("\n");  
7.     }  
8.     System.out.print("\n");  
9. }
```

```
printTriangle( 3 ); printTriangle( 5 );
```

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

自訂工具類別(Class) : **MyMath**

1. 將相關、有用的方法集合在一起(封裝在一個類別之內)，方便未來可以重複使用
2. 必須注意參數的傳遞與資料回傳的方式！

```
public class MyMath {  
    public static int factorial(int x) { /*階乘*/ }  
    public static boolean isPrime(int x) { /*質數*/ }  
    public static boolean isLeapYear(int year) { /*閏年*/ }  
    public static int[] getPrimeFactor(int n) { /*質因數*/ }  
    public static int search ( int[] data, int x) { /*一般搜尋 */ }  
    public static void sort (int[] data) { /* 排序 */ }  
    public static void binarySearch (int[] data, int x) { /* 二元搜尋 */ }  
}
```

參數: Call by value ?
Call by reference?

練習時間：開發工具類別(*MyMath*)

- 下載練習專案 Ex02060X.zip
- 匯入專案：[File] -> [import...] -> [General] -> [Existing project ...] -> [Select archive file...]：選擇剛剛下載的zip
- 建立 *MyMath* 的說明文件(Javadoc → HTML)
 - 查看 *MyMath* 類別內的 JavaDoc 寫法
 - 生成 Javadoc：[Project] -> [Generate Javadoc...]
- 執行 Main.java 看看：
 - 測試已經完成的方法 *isPrime(...)*, *isLeapYear(...)*
- 嘗試編寫其他方法
 - *factorial(...)*：階乘
 - *search(...)*：搜尋資料
 - *sort(...)*：排序(由小排到大)
 - *binarySearch(...)*：另類搜尋方法(二元搜尋)

階乘 *factorial*

- 階乘定義

$$5! = 1 * 2 * 3 * 4 * 5$$

$$N! = 1 * 2 * 3 * 4 * \dots * N$$

- 完成factorial 方法設計 : 利用 *return* 回傳結果

```
public static int factorial(int n) {  
    int result = 1;  
    for (int i=1; i<= n ; i++) {  
        result = result * i;  
    }  
    //  
    return result ;  
}
```

搜尋資料 *search*

- 從一堆資料(陣列 **data**)中尋找指定的數字，若找到 **x**，則回傳**x** 的註標，否則回傳-1

```
public static int search(int[] data, int x) {  
    for (int i=0; i<data.length; i++) {  
        if (data[i] == x) { // 找到 x!  
            return i;  
        }  
    }  
    // 找不到 x  
    return -1;  
}
```

排序(Sort): 將資料由小排到大

- 排序 (Sorting) : 處理數值資料時常常使用
 - 將一堆數字由小到大依序排列!

原本： 15, 30, 7, 43, 21, 1

最終： 1, 7, 15, 21, 30, 43

縮小問題：先找出第1小的數字！

- 原本： 15, 30, 7, 43, 21, 

- 最終： 1, 

選擇排序 *sort* (需要改寫)

```
public static void sort(int[] data) {  
    // 找出最小的數，放到位置 0 (i)  
    int k = 0;  
    for (int j=k+1; j<data.length; j++) {  
        if (data[j] < data[k] ) { // 有更小的數?  
            // 交換 k, j 位置的內容  
        }  
    }  
}
```

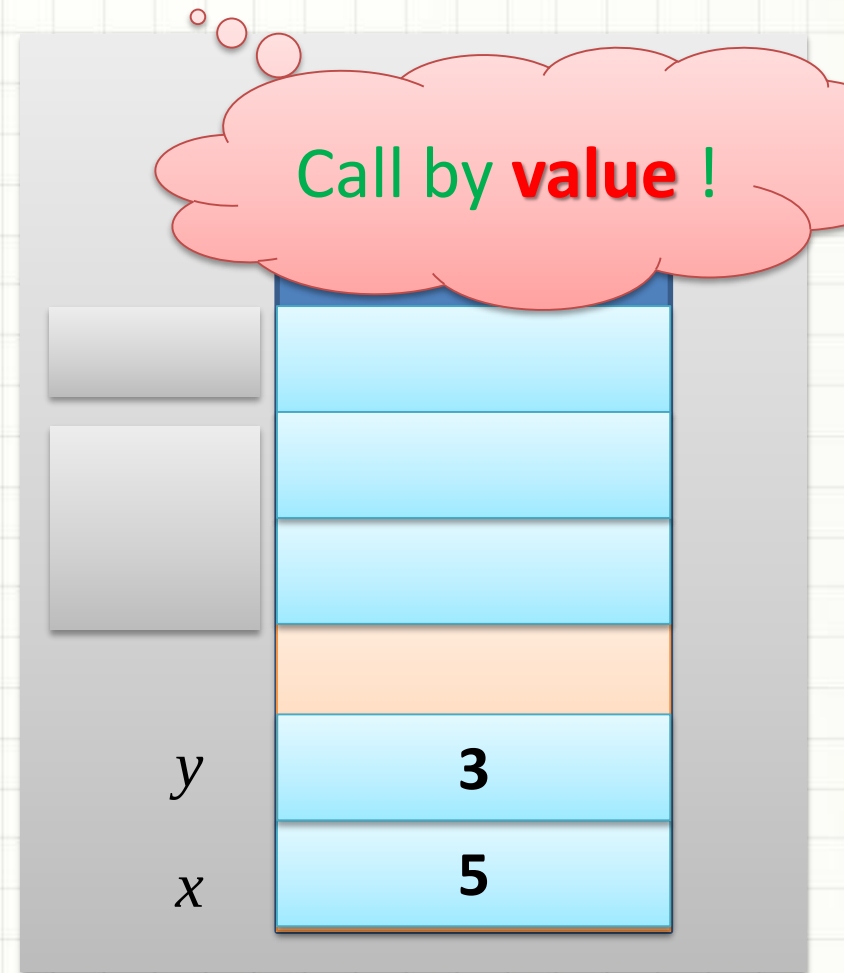
思考一個小問題：a、b內容互換

```
1. public static void swap (int a, int b) {  
2.     int temp; // 暫存  
3.     temp = a;  
4.     a = b;  
5.     b = temp;  
6. }
```

// 呼叫 swap(...) ?

int x=5, y =3;

swap(x, y);



傳遞陣列參數(變動資料大小)?

```
1. public static int search (int[ ] data, int x) {  
2.     // 從陣列data 找出 x  
3.     // 回傳 x 的位置  
4.     // 若找不到 x, 就回傳 -1  
5.     // 關鍵字: return  
6. }
```

Array:
Call by **reference**!

// 呼叫 search(...) ?

```
int[ ] list = { 2, 32, 9, 38, 39, 13, 25, 50};
```

```
int position = search( list, 13);
```

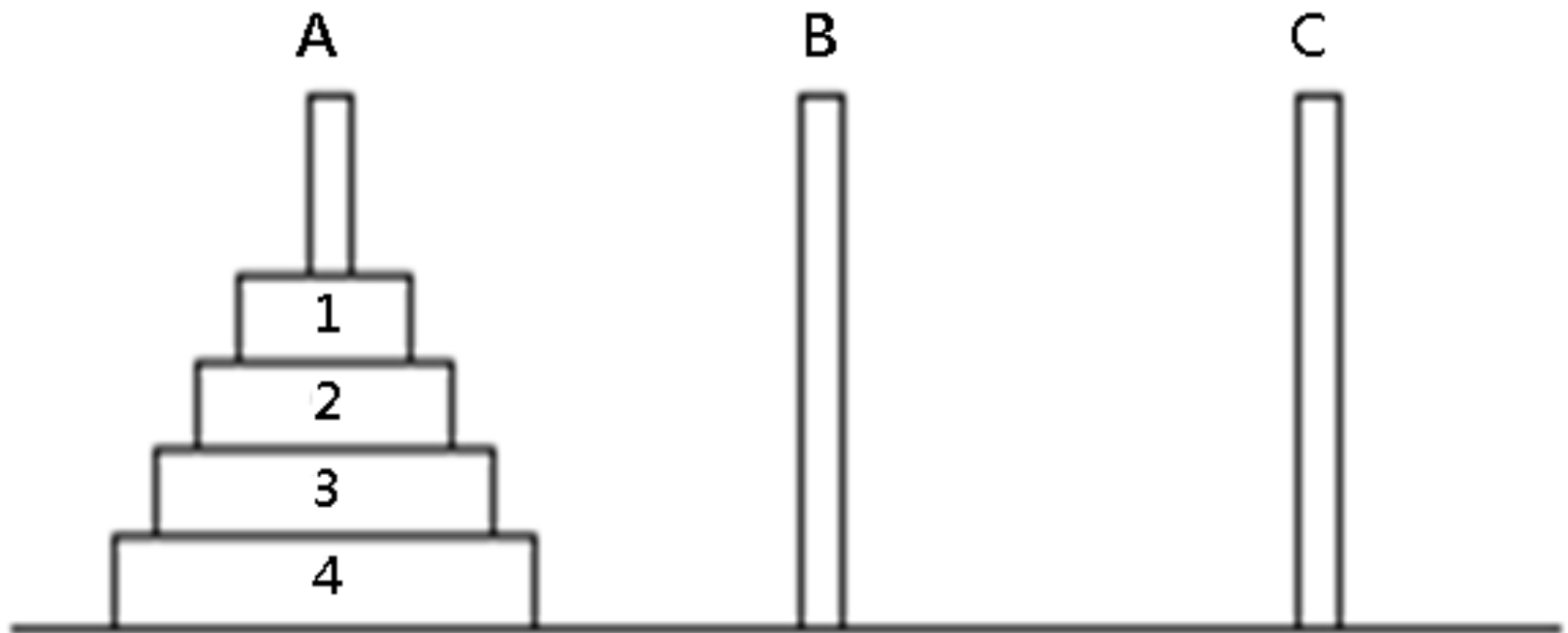
了解 Java 的程式進入點 main()

```
public class Main {  
    public static void main(String[] args) {  
        for (int i=0; i<args.length; i++) {  
            System.out.println(i+ "個參數:" + args[i]);  
        }  
    }  
}
```

執行看看: **java Main 123 456 ABC "XYZ" 789"**

遞迴的迷人之處

- **遞迴**：將大問題拆為小問題、先解出小問題，就能湊出大問題的答案；而問題夠小就一定有答案。



利用遞迴解河內之塔問題

```
public static void hanoi(int n, char from, char to, char temp) {  
    if (n == 1) { // (A)小問題:直接就有答案  
        //將盤子n (=1)從柱子(from)搬到 (to);  
    }  
    else { // (B)大問題: 利用遞迴先解較小的問題  
        hanoi(n-1, from, temp, to); // 先暫時搬走上面的 n-1個盤子  
        //將盤子n (=?)從柱子(from)搬到 (to);  
        hanoi(n-1, temp, to, from); // 再搬回上面的 n-1個盤子  
    }  
}
```

利用遞迴改寫階乘的計算?!

- 再看一次階乘的定義

$$1! = 1$$

$$2! = 1 * 2 = 1! * 1$$

$$3! = 1 * 2 * 3 = 2! * 3$$

$$4! = 1 * 2 * 3 * 4 = 3! * 4$$

$$N! = 1 * 2 * 3 * 4 * \dots * N = (N-1)! * N$$



$$\text{factorial}(n) = 1, \text{ if } n = 1$$

$$\text{factorial}(n) = \text{factorial}(n-1) * n, \text{ if } n \geq 2$$

階乘 *factorial* : 遞迴版本

- $\text{factorial}(n) = 1$, if $n = 1$
- $\text{factorial}(n) = \text{factorial}(n-1) * n$, if $n \geq 2$

```
public static int factorial(int n) {  
    if (n == 0 || n == 1)  
        return 1;  
    else if (n >= 2)  
        return factorial(n-1) * n;  
}
```

遞迴：
自己呼叫自己

練習時間： *power(x, n)*

- 迴圈(迭代)的版本

```
public static int power(int x, int n) {  
    int result = 1;  
    for (int i=0; i<n; i++) {  
        result *= x;  
    }  
    return result;  
}
```

練習時間：改寫 $power(x, n) - 1$

- 遞迴的版本1

```
public static int power(int x, int n) {
```

}

練習時間：改寫 $power(x, n) - 2$

- 遞迴的版本2 (效率比較好)

```
public static int power(int x, int n) {
```

}