基礎程式設計（Java）　講義

陣列與 ArrayList：一次裝很多筆資料

Deitel & Deitel《Java How to Program: Early Objects》11e, Ch.7

§ 1　一、名詞速查表

中文	English	一句話定義	科目	章節
陣列	array	同型別、固定長度、連續編號的一組元素	Java	CH7
元素	element	陣列裡的每一格資料	Java	CH7
索引	index / subscript	元素的編號，從 `0` 開始到 `length-1`	Java	CH7
配置	allocate	用 `new` 向系統要一塊陣列空間	Java	CH7
初始化清單	initializer list	用 `{…}` 直接列出元素值並建陣列	Java	CH7
長度	length	陣列元素個數，用 `arr.length`（不是方法）	Java	CH7
增強 for	enhanced for / for-each	`for (型別 x : arr)` 逐一取出每個元素	Java	CH7
參照型別	reference type	變數存的是物件位址；陣列就是物件	Java	CH7
多維陣列	multidimensional array	陣列的陣列，如 `int[][]`（列、行）	Java	CH7
越界	out of bounds	索引超出 `0..length-1`，丟例外	Java	CH7
動態陣列	ArrayList	可自動增減大小的串列，存物件	Java	CH7
泛型	generic `<E>`	`ArrayList<String>` 指定裝哪種元素	Java	CH7
自動裝箱	autoboxing	`int` 自動轉成 `Integer` 放進 ArrayList	Java	CH7
可變長度引數	varargs	`型別... 名` 收任意個引數，方法內當陣列用	Java	CH7
命令列引數	command-line argument	執行時傳給程式的字串，進 `main(String[] args)`	Java	CH7

§ 2 二、核心概念

要存「很多筆同型別資料」（10 個分數、5 個名字），不用宣告 10 個變數，用**陣列**一次裝一整排。陣列有三個特性：**同型別**（一個陣列只裝一種型別）、**固定長度**（建好就不能變大小）、**連續編號**（用索引 `0, 1, 2, …` 取用）。

> 索引**從 0 開始**：長度 5 的陣列，合法索引是 `0` 到 `4`。最後一格是 `arr[arr.length - 1]`，不是 `arr[arr.length]`（那會越界）。

陣列是**物件**（參照型別）：陣列變數存的是「指向那塊資料的位址」，不是資料本身。把陣列傳給方法是傳「位址」，方法內改元素，呼叫端看得到（和 CH6 的值傳遞不同）。

當「事先不知道要存幾筆、之後還會增刪」時，固定長度的陣列不夠用，改用 `ArrayList`：它會自動長大、可隨時 `add`／`remove`，代價是只能裝物件（`int` 要靠自動裝箱變 `Integer`）。

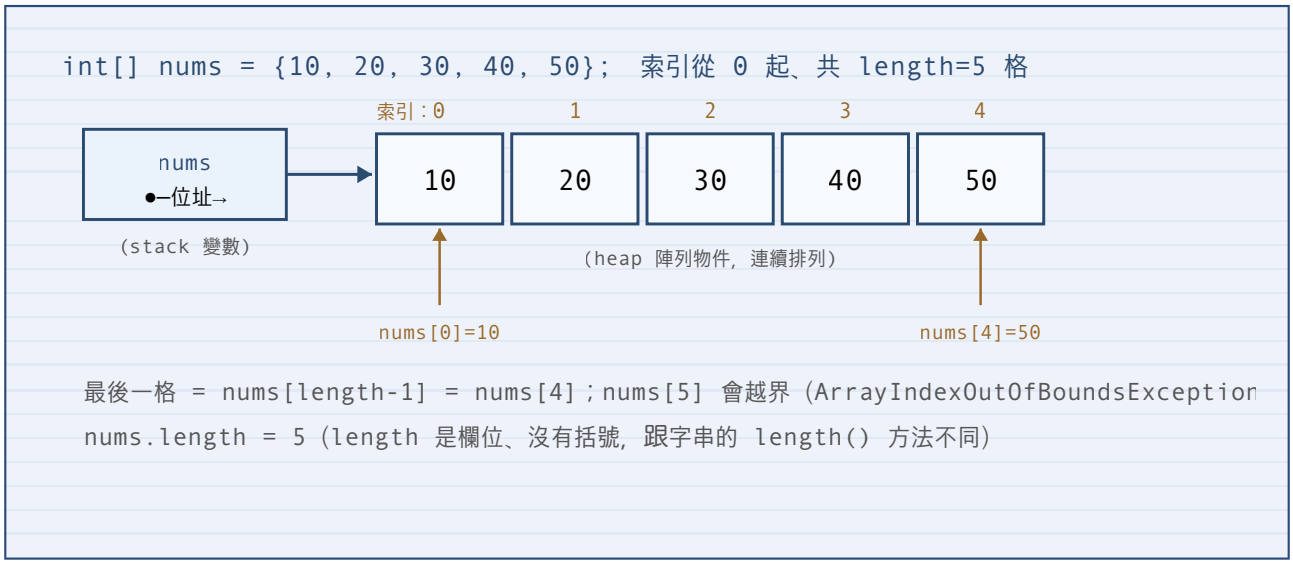
§ 3 三、主要內容

3.1 1. 宣告、配置與初始化

```
int[] grades;               // 宣告一個 int 陣列變數（還沒有實體）
grades = new int[5];        // 配置 5 格，預設值全為 0
int[] scores = new int[5]; // 宣告 + 配置一行寫完

int[] nums = {10, 20, 30, 40, 50};  // 初始化清單：直接給值，長度自動為 5
```

- `new int[5]` 的 5 格**自動填預設值**：數值型別填 `0`、`boolean` 填 `false`、參照型別填 `null`。
- 初始化清單 `{…}` 不用寫 `new`，編譯器依元素個數決定長度。

3.2 2. 陣列記憶體格子圖



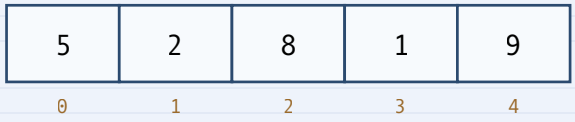
3.3 3. 走訪：傳統 for 與增強 for

```
int[] nums = {10, 20, 30, 40, 50};

for (int i = 0; i < nums.length; i++)    // 傳統 for：有索引，能改值
    System.out.println(i + ": " + nums[i]);

for (int x : nums)                       // 增強 for（for-each）：只讀、無索引
    System.out.println(x);
```

- **傳統 for**：條件用 `i < nums.length`（不是 `<=`），才不會碰到越界。需要索引、要改元素值時用它。
- **增強 for**：`x` 依序拿到每個元素的「值」，語法乾淨；但拿不到索引，也**改不了**陣列內容（`x` 是副本）。只讀時優先用。

3.4 4. 走訪步進示意（增強 for 逐格讀取求和）

```
for (int x : a) total += x;   a = {5, 2, 8, 1, 9}
```



步	x（讀到的元素）	total（前→後）
1	5	0→5
2	2	5→7
3	8	7→15
4	1	15→16
5	9	16→25 ← 走完，total=25

3.5 5. 常見陣列演算法（求和、最大、計次）

```java
int[] a = {5, 2, 8, 1, 9};

int total = 0;                      // 求和
for (int x : a) total += x;         // total = 25

int max = a[0];                     // 找最大：先假設第一個最大
for (int x : a)
    if (x > max) max = x;           // max = 9

int count = 0;                      // 計次：數大於 4 的有幾個
for (int x : a)
    if (x > 4) count++;             // count = 3 (5, 8, 9)
```

· **找最大**：`max` 初值設 `a[0]`（不要設 0，否則全負數時會錯）。

3.6 6. 陣列是物件、傳給方法

```java
int[] a = {1, 2, 3};
int[] b = a;          // b 和 a 指向「同一個」陣列（沒有複製）
b[0] = 99;            // 透過 b 改，a[0] 也變 99

static void doubleAll(int[] arr) {  // 收到的是位址
    for (int i = 0; i < arr.length; i++)
        arr[i] *= 2;                // 改的是同一個陣列
}
// 呼叫 doubleAll(a) 後，a 的元素全部加倍（呼叫端看得到）
```

陣列變數是參照；`b = a` 只複製位址。要真的複製內容得逐元素拷貝或用 `Arrays.copyOf`。

3.7 7. 多維陣列

```
int[][] m = {              // 2 列 3 行
    {1, 2, 3},
    {4, 5, 6}
};
int x = m[1][2];           // 第 1 列、第 2 行 = 6
m.length;                  // 列數 = 2
m[0].length;               // 第 0 列的行數 = 3

for (int r = 0; r < m.length; r++)        // 走訪每一格
    for (int c = 0; c < m[r].length; c++)
        System.out.print(m[r][c] + " ");
```

`int[][]` 是「陣列的陣列」：`m[r]` 本身是一個一維陣列，`m[r][c]` 才是元素。先列（row）後行（column）。

3.8 8. ArrayList：可變大小的串列

```
import java.util.ArrayList;

ArrayList<String> names = new ArrayList<>();  // 空串列，裝 String
names.add("Amy");          // 加到尾端
names.add("Ben");
names.add("Cara");
names.get(0);              // 取第 0 個 → "Amy"
names.size();              // 目前個數 → 3
names.remove("Ben");       // 移除（剩 Amy, Cara，索引自動往前補）

for (String n : names)     // 增強 for 走訪
    System.out.println(n);
```

- **動態大小**：`add` 自動長大、`remove` 自動縮，不像陣列固定。
- **只裝物件**：`ArrayList<Integer>`（不能寫 `ArrayList<int>`）；放 `int` 會**自動裝箱**成 `Integer`。
- 取個數用 `size()`（方法、有括號），跟陣列的 `length`（欄位、沒括號）不同。

3.9 9. 可變長度引數（varargs）

```
// 參數型別後加 ...，可收 0 到任意多個同型別引數
public static double average(double... numbers) {
    double total = 0.0;
    for (double d : numbers)    // numbers 在方法內就是一個 double[] 陣列
        total += d;
    return total / numbers.length;   // 用 .length 取個數
}
// 呼叫：傳幾個都可以
average(10.0, 20.0);             // 2 個
average(10.0, 20.0, 30.0, 40.0); // 4 個
```

- `型別... 名` 在方法內**等同一個陣列**，可用 `.length` 與增強 for 走訪。
- `...` 參數**只能放在參數列最後一個**，一個方法最多一個 varargs。
- `System.out.printf` 的多個引數就是靠 varargs 收進去的。

3.10 10. 命令列引數（command-line arguments）

```
public static void main(String[] args) {  // args 收命令列傳入的字串
    // 執行：java InitArray 5 0 4   → args = {"5", "0", "4"}
    int arrayLength = Integer.parseInt(args[0]);  // 字串轉 int
    int[] array = new int[arrayLength];
    int initialValue = Integer.parseInt(args[1]);
    int increment = Integer.parseInt(args[2]);
    for (int i = 0; i < array.length; i++)
        array[i] = initialValue + i * increment;
}
```

- `main` 的 `String[] args` 就是命令列引數陣列；`args.length` 是傳入的個數。
- 引數一律是**字串**，要當數字用得自己轉（`Integer.parseInt`／`Double.parseDouble`）。
- 用前先檢查 `args.length` 是否足夠，否則 `args[i]` 會越界。

§ 4 四、語法與 API 速查

```
int[] a = new int[5];           // 配置 5 格（預設值 0）
int[] b = {1, 2, 3};            // 初始化清單
a.length                        // 長度（欄位，無括號）
a[i]                            // 取/設第 i 格（i 從 0 到 length-1）
for (int x : a) { ... }         // 增強 for（只讀）
int[][] m = new int[2][3];      // 多維（2 列 3 行）

ArrayList<E> list = new ArrayList<>();  // 動態串列
list.add(e);                    // 加到尾端
list.get(i);                    // 取第 i 個
list.set(i, e);                 // 設第 i 個
list.size();                    // 個數（方法，有括號）
list.remove(i 或 物件);         // 移除

void f(int... xs) { ... }       // varargs：xs 在方法內是 int[]
public static void main(String[] args)  // args 是命令列引數（String[]）
Integer.parseInt(args[0])       // 命令列引數是字串，要自己轉數字
Arrays.copyOf(a, n)             // 複製陣列成長度 n 的新陣列（java.util.Arrays）
```

- **走訪準則**：只讀 → 增強 for；要索引或改值 → 傳統 for（`i < a.length`）。
- **長度 vs 個數**：陣列 `a.length`（無括號）；ArrayList `list.size()`（有括號）。
- **找最大**：`max = a[0]` 起步，不要設 0。

§ 5 五、常見錯誤

- **索引越界**：合法索引是 `0..length-1`；`a[a.length]` 或 `a[5]`（長度 5）丟 `ArrayIndexOutOfBoundsException`。
- `length` **加括號**：陣列是 `a.length`（欄位），**不是** `a.length()`；字串才用 `s.length()`。
- **for 條件用** `<=`：`for (int i = 0; i <= a.length; i++)` 最後一圈越界；要用 `i < a.length`。
- **增強 for 想改值**：`for (int x : a) x = 0;` 改的是副本，陣列不變；要改值用傳統 for 配索引。
- `b = a` **以為複製**：兩個變數指向同一陣列，改一個另一個跟著變；要複製用 `Arrays.copyOf`。
- **找最大把初值設 0**：資料全為負數時答案會錯成 0；初值設 `a[0]`。
- **ArrayList 用** `length`：ArrayList 取個數是 `size()`，不是 `length`；陣列才用 `length`。
- `ArrayList<int>`：泛型只能放參照型別，要寫 `ArrayList<Integer>`。
- **多維陣列列行搞反**：`m[r][c]` 是先列後行；`m.length` 是列數、`m[0].length` 是行數。

§ 6 六、練習題

例題 1：陣列走訪求和與平均

給 `int[] a = {5, 2, 8, 1, 9}`，用增強 for 求總和，再算平均（保留兩位小數）。

> 1. `total = 0`，`for (int x : a) total += x` 2. 平均 `(double) total / a.length` 3. `printf` 印總和與平均

```
public class ArraySum {
    public static void main(String[] args) {
        int[] a = {5, 2, 8, 1, 9};
        int total = 0;
        for (int x : a) total += x;
        System.out.println("Sum = " + total);                  // 25
        System.out.printf("Average = %.2f%n", (double) total / a.length);  // 5.00
    }
}
```

> 易錯：忘 `(double)` → 平均沒小數；用 `a.length()` 加括號。

例題 2：找最大值與其索引

給 `int[] a = {5, 2, 8, 1, 9}`，找出最大值與它的索引（用傳統 for 才拿得到索引）。

> 1. `max = a[0]`、`maxIdx = 0` 2. `for (int i = 1; i < a.length; i++)`：若 `a[i] > max` 更新 max 與 maxIdx 3. 印出 max 與 maxIdx

```
public class FindMax {
    public static void main(String[] args) {
        int[] a = {5, 2, 8, 1, 9};
        int max = a[0], maxIdx = 0;
        for (int i = 1; i < a.length; i++) {
            if (a[i] > max) { max = a[i]; maxIdx = i; }
        }
        System.out.println("Max = " + max + " at index " + maxIdx);  // 9 at
index 4
    }
}
```

> 易錯：max 初值設 0（全負數會錯）；要索引卻用增強 for（拿不到 i）。

例題 3：ArrayList 加入與走訪

建一個 `ArrayList`，加入三個城市名，印出個數，再用增強 for 印出每一個。

> 1. `import java.util.ArrayList` 2. `add` 三筆，`size()` 印個數 3. 增強 for 走訪印出

```
import java.util.ArrayList;
public class CityList {
    public static void main(String[] args) {
        ArrayList<String> cities = new ArrayList<>();
        cities.add("Taipei");
        cities.add("Tokyo");
        cities.add("Seoul");
        System.out.println("Count = " + cities.size());   // 3
        for (String c : cities)
            System.out.println(c);
    }
}
```

易錯：用 `length` 取個數（應 `size()`）；漏 `import java.util.ArrayList`。

§ 7 七、自我檢核

- [ ] 會宣告與配置陣列（`new int[5]`、初始化清單 `{…}`），知道索引從 0 到 `length-1`。
- [ ] 知道 `arr.length` 是欄位（無括號），會用它當 for 的邊界 `i < arr.length`。
- [ ] 分得清傳統 for（有索引、可改值）與增強 for（只讀、乾淨）何時用。
- [ ] 會寫求和、找最大（初值 `a[0]`）、計次三種常見陣列演算法。
- [ ] 知道陣列是物件、傳給方法是傳位址，方法內改元素呼叫端看得到。
- [ ] 會用 `int[][]` 多維陣列，分得清 `m.length`（列）與 `m[0].length`（行）。
- [ ] 會用 `ArrayList<E>` 的 `add`／`get`／`size`／`remove`，知道它動態大小、只裝物件、取個數用 `size()`。
- [ ] 知道 varargs（`型別...`）在方法內是陣列、只能放最後一個；知道命令列引數進 `main(String[] args)`、是字串要自己轉、用 `args.length` 取個數。