Part II 軟體設計

國立成功大學李信杰編修
取自資訊軟體人才培育推動計畫教材

抽象化1

- 抽象化(Abstraction)：由許多物件(Entities)中抽離出重要的特性(Features)來，而這些特性，足以讓被抽象化的物件，與別的物件分別開來。
  - 對一般汽車使用者而言，所感興趣的部分是車子的外觀、顏色、車子內裝、安全性、具備哪些功能等，而對於底盤的形式、油管的構造、引擎的設計等較無興趣。
  - 對於汽車研發製造者而言，其關注的焦點在於底盤、油箱、引擎的設計與結合上面，對於外觀、顏色等資訊則可能暫時不列在其關心之列。
- 抽象化階層(Levels of Abstraction)：對於不同的階層會提供不同的抽象化程度。

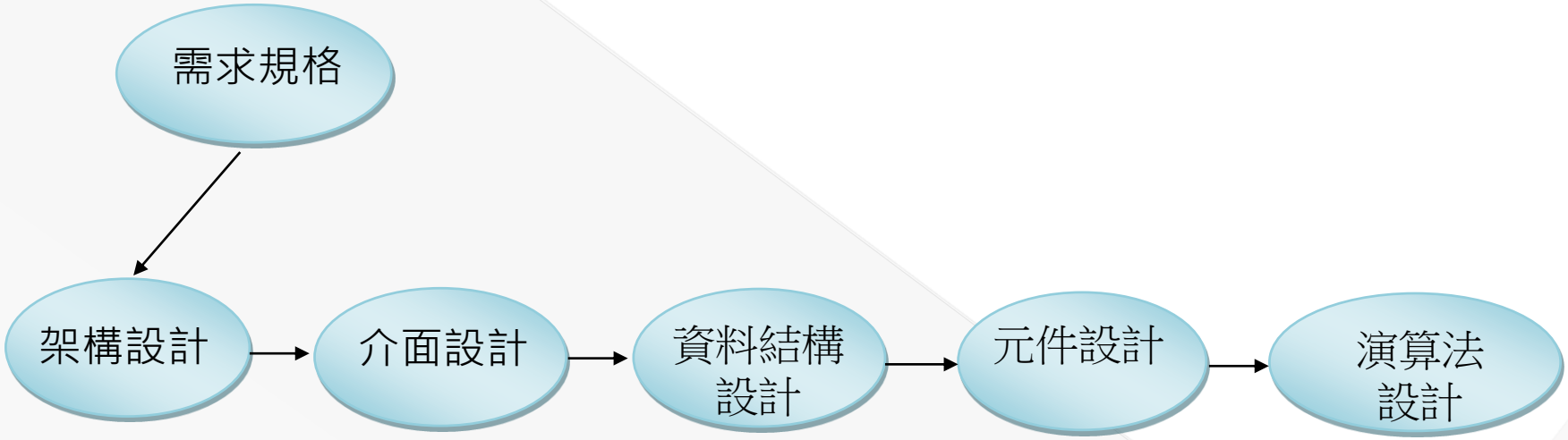
抽象化2

- 軟體系統發展的抽象化階層：
  - 軟體開發的初期（需求分析階段）：了解整個軟體系統會**提供什麼樣的功能**給使用者，而對於系統的內部運作則不是這階段的重點。
  - 設計的階段：著重了解軟體內部的設計概念，包含會使用什麼樣的**軟體架構**，以什麼**資料結構**以及**演算法**來實作軟體系統的功能等。
  - 軟體實作(Coding)階段：處理所使用程式語言的細節。

模組化

- 模組化(Modularity)：將軟體系統依其系統功能性或資料的相依性，分成數個容易了解與處理，而且可以互相溝通的單元(Module)或元件。
  - 透過軟體系統的模組化可以降低系統的複雜度，提高系統的可維護性，因此，可增加系統的穩定性以及可再利用性(Reusability)。
  - 需平衡模組的大小以及數量。

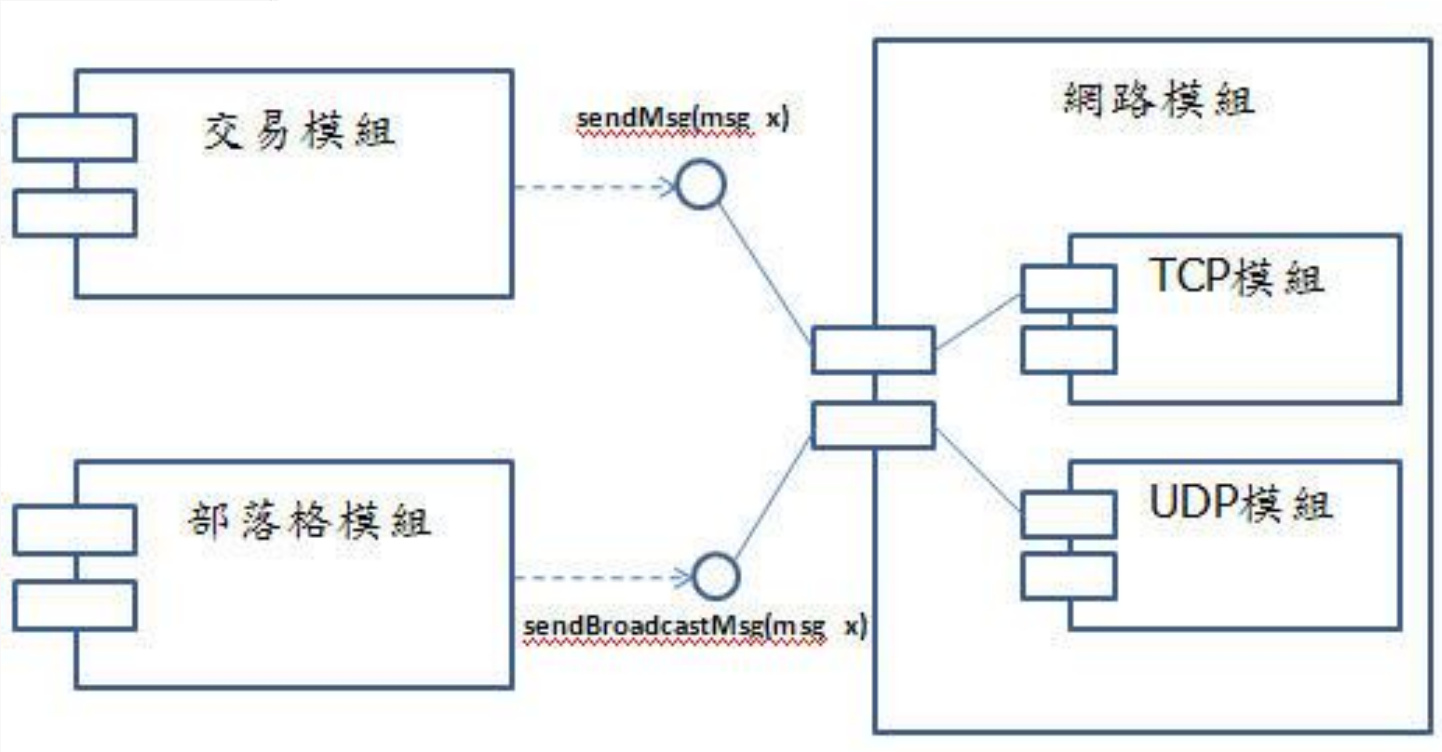
軟體設計步驟



架構設計1

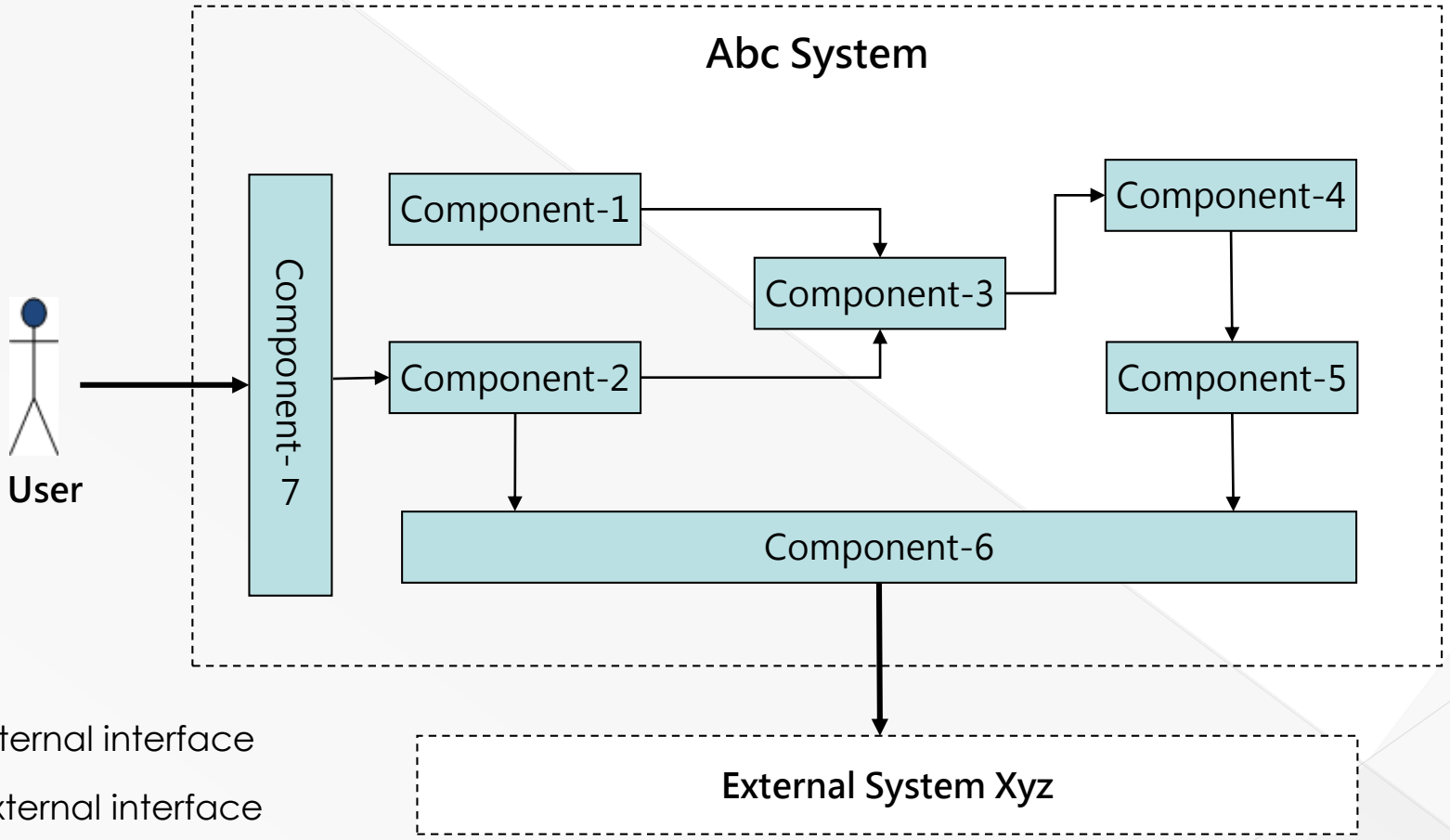
- 軟體架構設計是架構整體系統組織的過程，包含以下幾個活動：
  - (1)將系統分割成數個子系統
  - (2)決定這些子系統如何互動
  - (3)決定這些子系統的介面
- 架構圖應與需求內容一致

架構設計-範例1

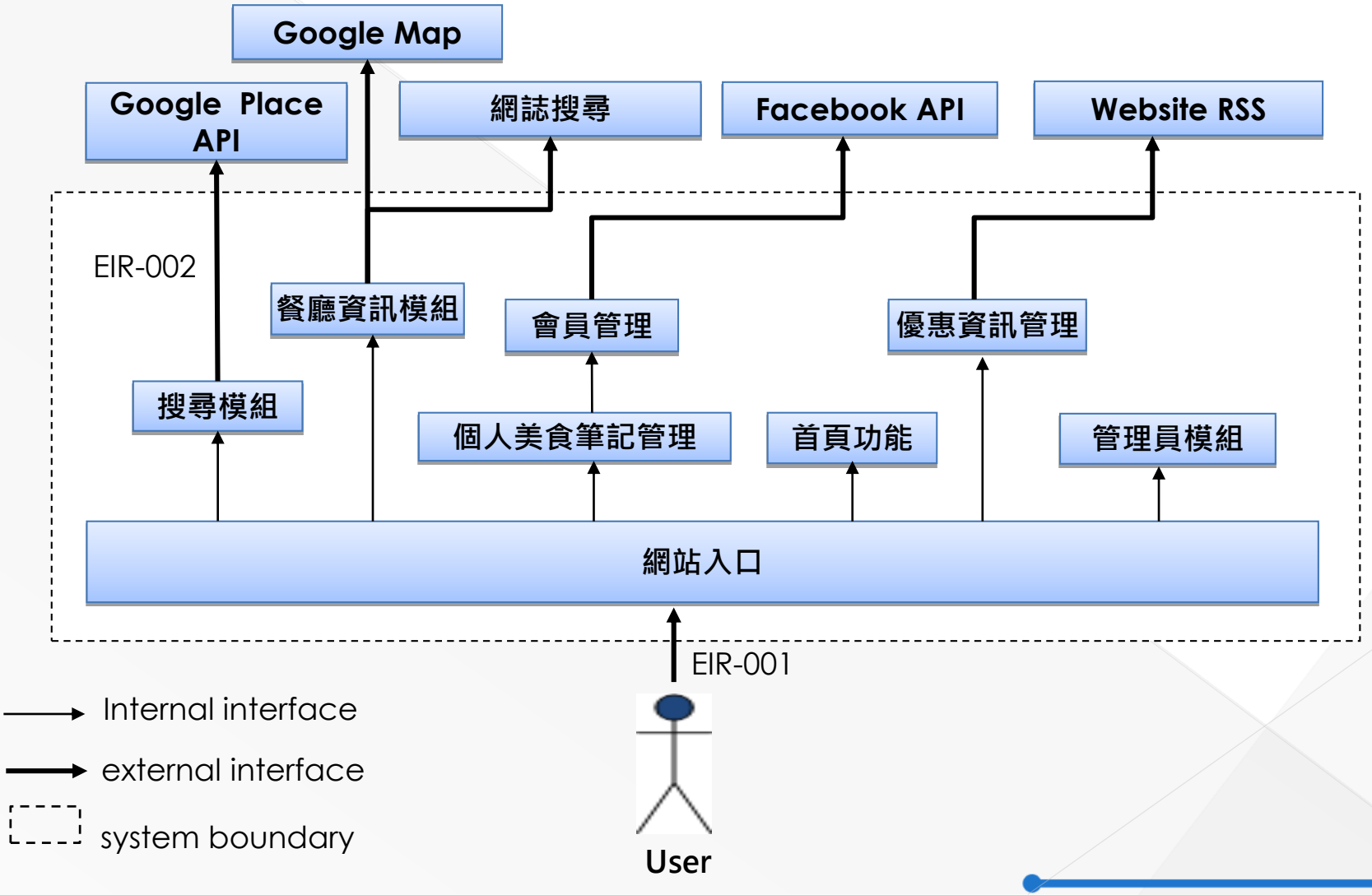


UML Component Diagram

架構設計-範例2


Abc System
Component-1
Component-2
Component-3
Component-4
Component-5
Component-6
Component-7
User
External System Xyz
Internal interface
external interface
system boundary

架構設計-範例3


Google Map
Google Place API
網誌搜尋
Facebook API
Website RSS
EIR-002
餐廳資訊模組
會員管理
優惠資訊管理
搜尋模組
個人美食筆記管理
首頁功能
管理員模組
網站入口
EIR-001
Internal interface
external interface
system boundary
User

Lab 2-1

- 延續Day1的假想系統，繪製出系統架構圖

介面設計

- 介面(Interface)不是專指User Interface，亦代表兩個軟體模組(或兩個系統)銜接時，一個模組要如何呼叫或嵌入另外一個模組
  - 舉例而言，若GUI模組要呼叫圖片管理模組，圖片管理模組開放給GUI模組的method就是介面

介面設計-範例

介面名稱(Interface Name)	介面提供者(Interface Provider)	介面使用者(Interface Consumer)
Google Place API與餐廳名稱、地址搜尋介面	Google Place API	搜尋模組
連結方式(Connection Type)	**輸入資料(Input Data)**	**輸出資料(Output Data)**
http GET	keyword、address、page	搜尋結果之XML
對應之介面需求(Interface Requirement Description)		
系統可接收外部搜尋事件，並將搜尋關鍵字傳給Google Place API執行搜尋動作，以取得搜尋結果。		

Lab 2-2

- 延續Day1的假想系統，繪製出介面設計

介面名稱(Interface Name)	介面提供者(Interface Provider)	介面使用者(Interface Consumer)
連結方式(Connection Type)	輸入資料(Input Data)	輸出資料(Output Data)
對應之介面需求(Interface Requirement Description)		

資料結構設計

- 介面設計中的資料結構
  - 例如資料庫table schema、檔案結構、XML schema、JSON - JavaScript Object Notation物件等資料結構。

介面名稱：搜尋API			
輸入資料		輸出資料	
資料名稱	型態	資料名稱	型態
關鍵字	字串	搜尋結果	字串陣列

Lab 2-3

- 延續Day1的假想系統，請針對介面提出資料結構設計

介面名稱：**XXX**			
輸入資料		**輸出資料**	
資料名稱	**型態**	**資料名稱**	**型態**

元件設計

- 元件設計分為靜態面之結構設計與動態面之行為設計
  - 結構設計 (structure design)
    - UML類別圖(Class Diagram)：明確設計系統中包含哪些類別，以及這些類別之間的關係。若遇到非一般物件類別或特殊物件類別，如HTML、JavaScript、Servlet等，可用stereo type表達，如<<JavaScript>>、<<Servlet>>。
  - 行為設計 (behavior design)
    - UML循序圖(Sequence Diagram)：設計物件間的互動關係與循序執行。
    - UML活動圖(Activity Diagram)：描述所開發的系統流程，目的是設計整個系統的完整運作，應為操作概念的細部設計。

演算法設計

- 設計草圖
- 虛擬碼/流程圖

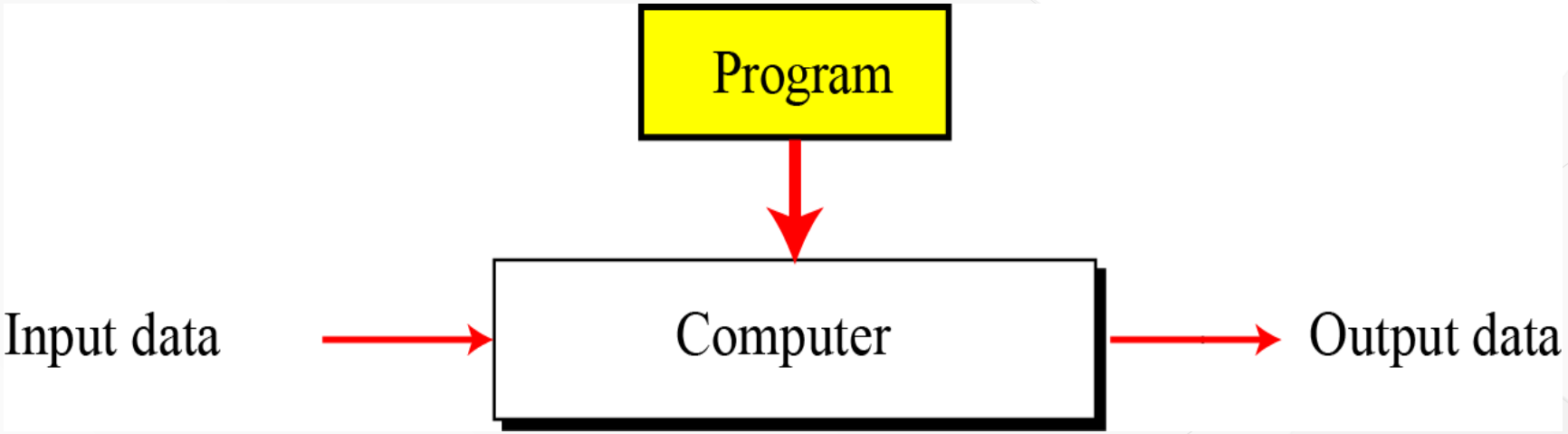
程式運作的概念

接收輸入的資料

處理輸入的資料

儲存輸入的資料

輸出處理完的資料



接收 儲存 及輸出 資料

- 請撰寫一程式由使用者處**得到二個數字**並將**相加結果輸出**

- 不要急著寫程式，先思考流程
  - 告知使用者要輸入二個數字
  - 請使用者輸入第一個數字
  - 取得使用者輸入的第一個數字並將它存起來
  - 請使用者輸入第二個數字
  - 取得使用者輸入的第二個數字並將它存起來
  - 將二個數字相加
  - 輸出相加之結果

將思維轉成適合的程式

```
Scanner output = new Scanner (System.in);
```

- //告知使用者要輸入二個數字
  - System.out.println("輸入二個數字\n");
- //請使用者輸入第一個數字
  - System.out.println("輸入第一個數字\n");
- //取得使用者輸入的第一個數字並將它存起來
  - output.nextInt(integer1);
- //請使用者輸入第二個數字
  - System.out.print("輸入第二個數字\n");
- //取得使用者輸入的第二個數字並將它存起來
  - output.nextInt(integer2);
- //將二個數字相加
  - sum = integer1 + integer2;
- //輸出相加之結果
  - System.out.printf("Sum is %d\n", sum);

程式流程越來越複雜

- 告知使用者要輸入二個數字
  若使用者輸入的不是數字該如何?
- 請使用者輸入第一個數字
  檢查使用者是否輸入數字, 若不是則要求重新輸入
- 取得使用者輸入的第一個數字並將它存起來
- 請使用者輸入第二個數字
  檢查使用者是否輸入數字, 若不是則要求重新輸入
- 取得使用者輸入的第二個數字並將它存起來
- 將二個數字相加
- 輸出相加之結果

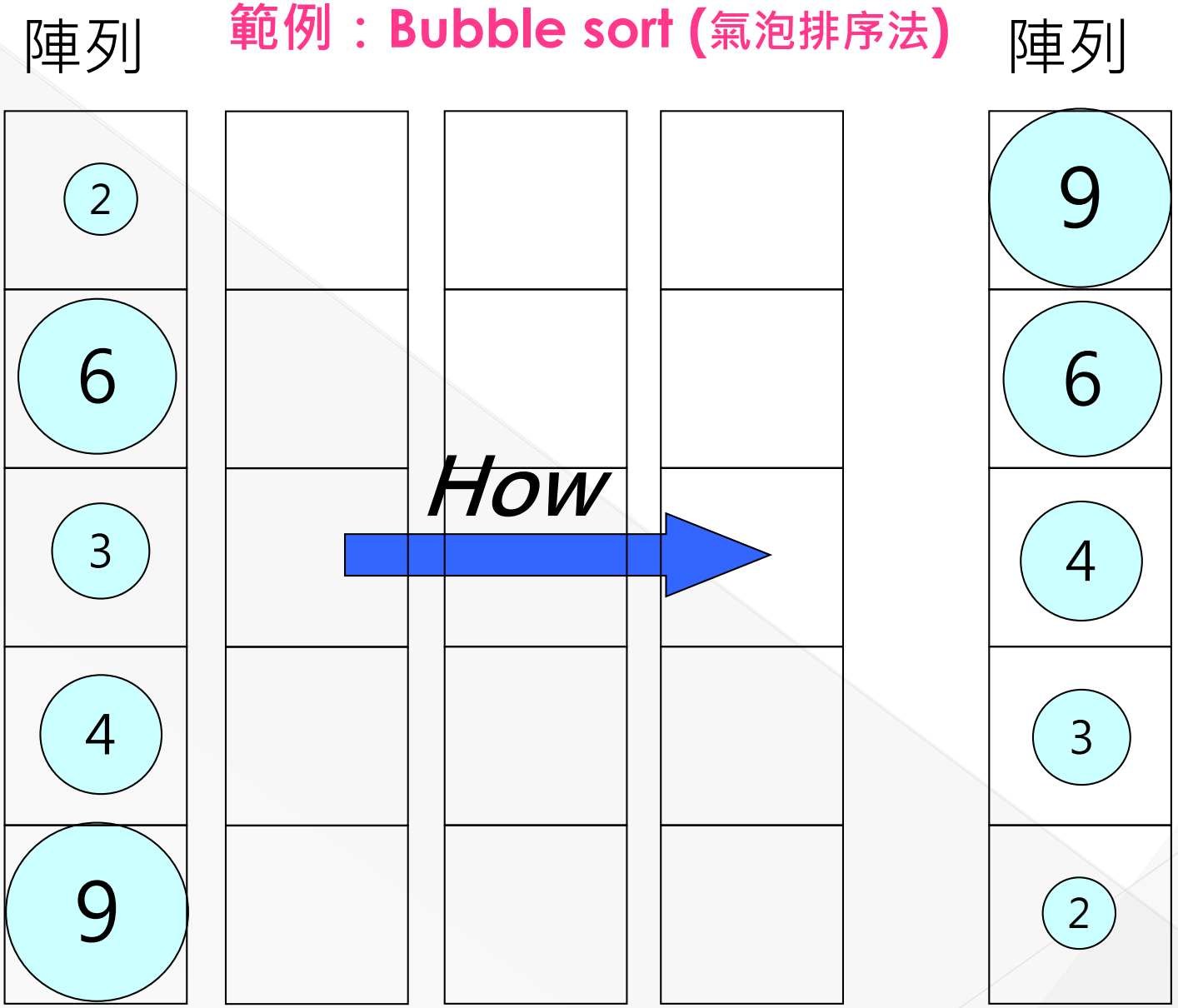
演算法設計

- 1. 設計草圖 (design sketch)
  - 寫在白板或白紙
- 2. 虛擬碼 (pseudo code) / 流程圖
  - Pseudo code 內文以英詞中句書寫

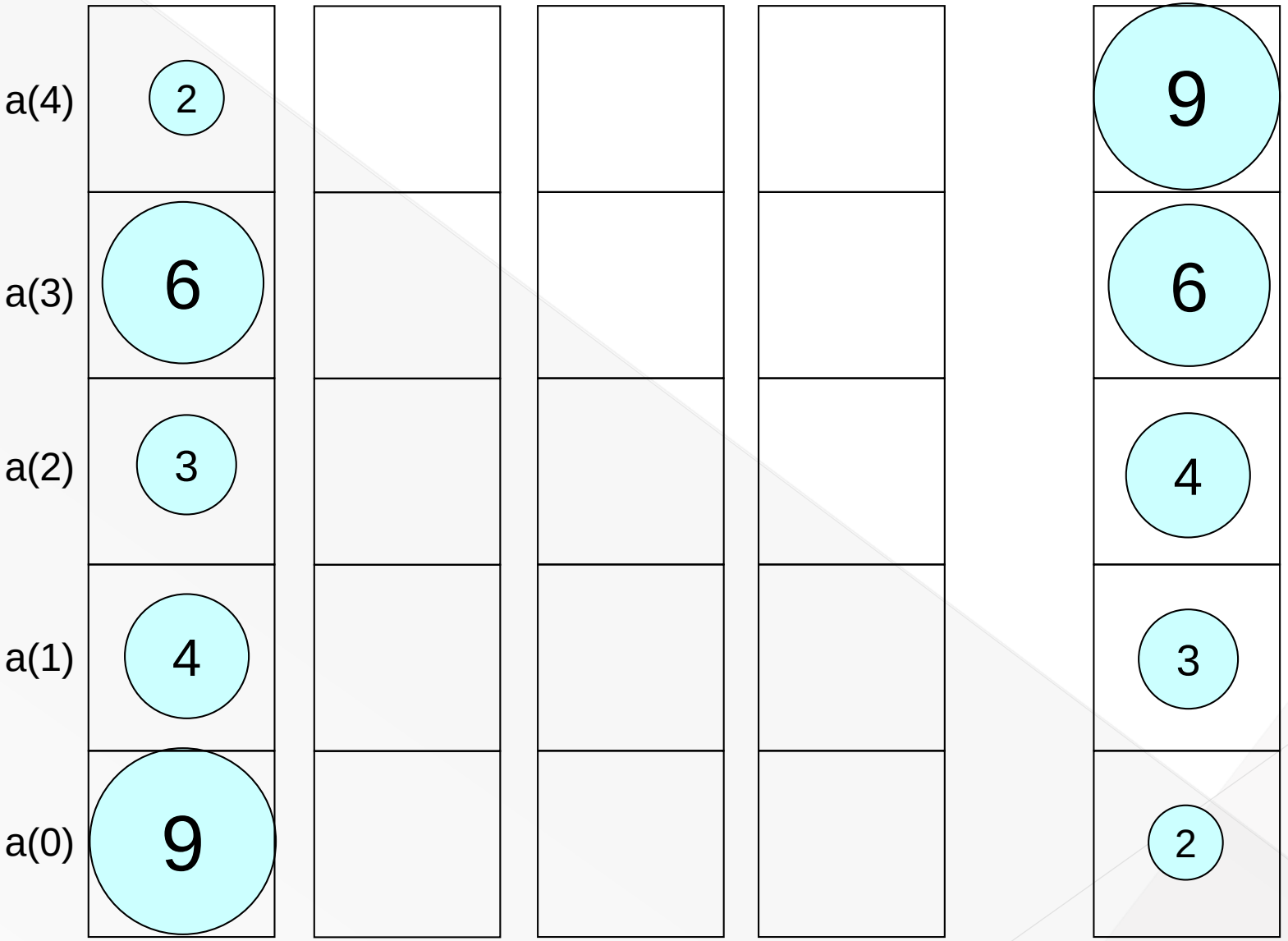
畫design sketch 後 - 看圖說故事：

- **圖**：即 design sketch 設計草圖
- **故事**：即 pseudo code / 流程圖 細部設計

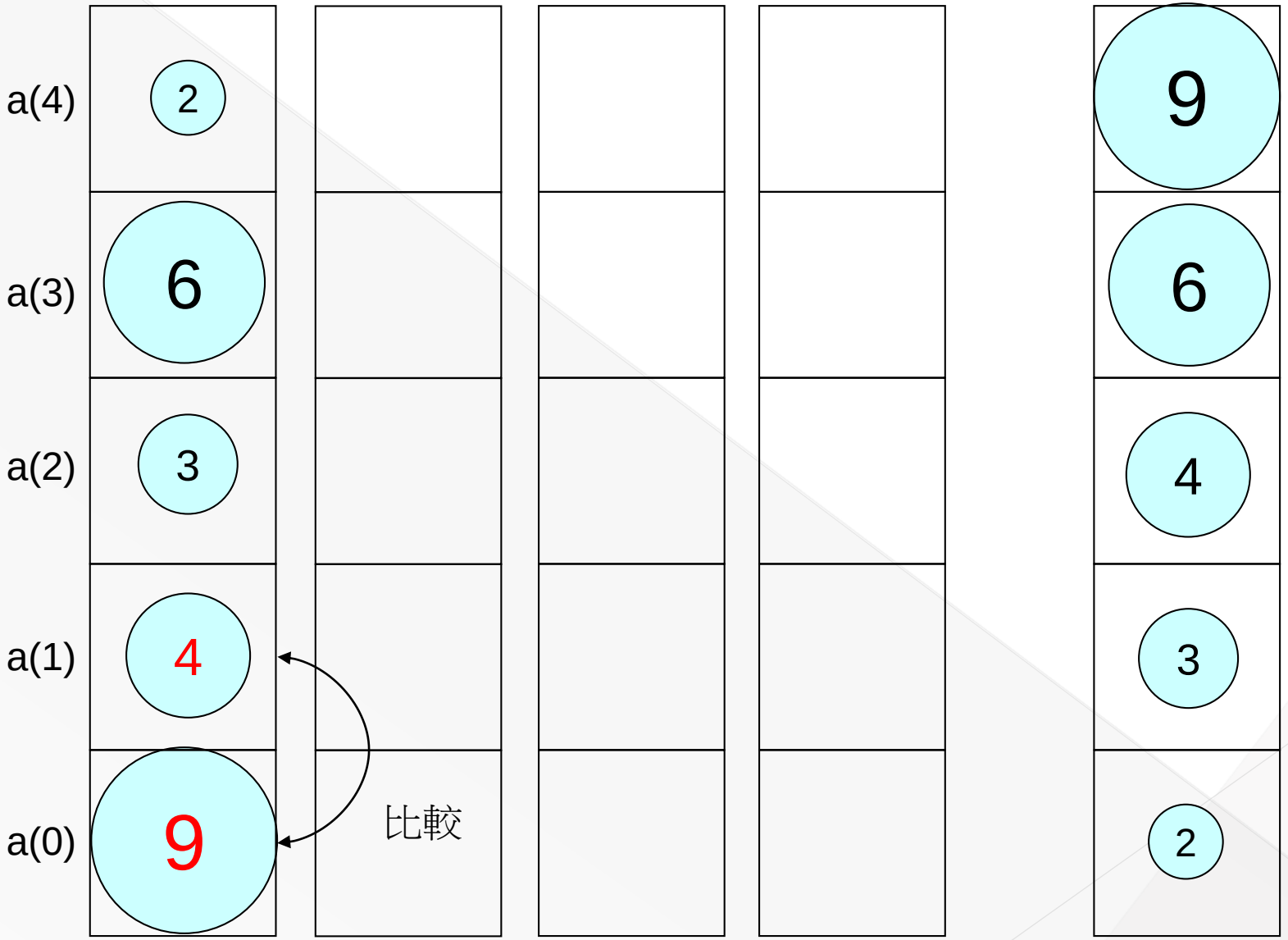
範例：Bubble sort (氣泡排序法)



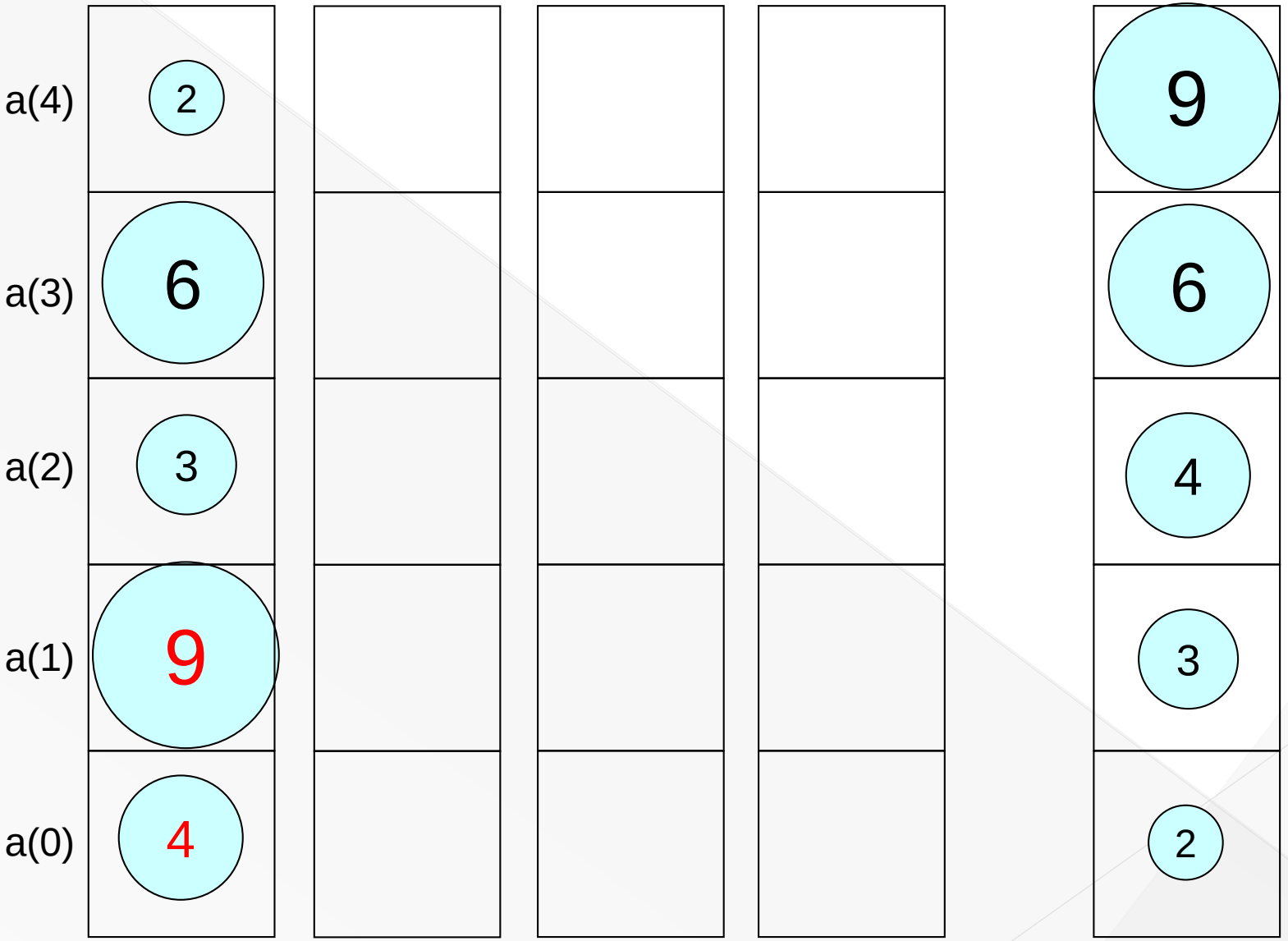
第一回合



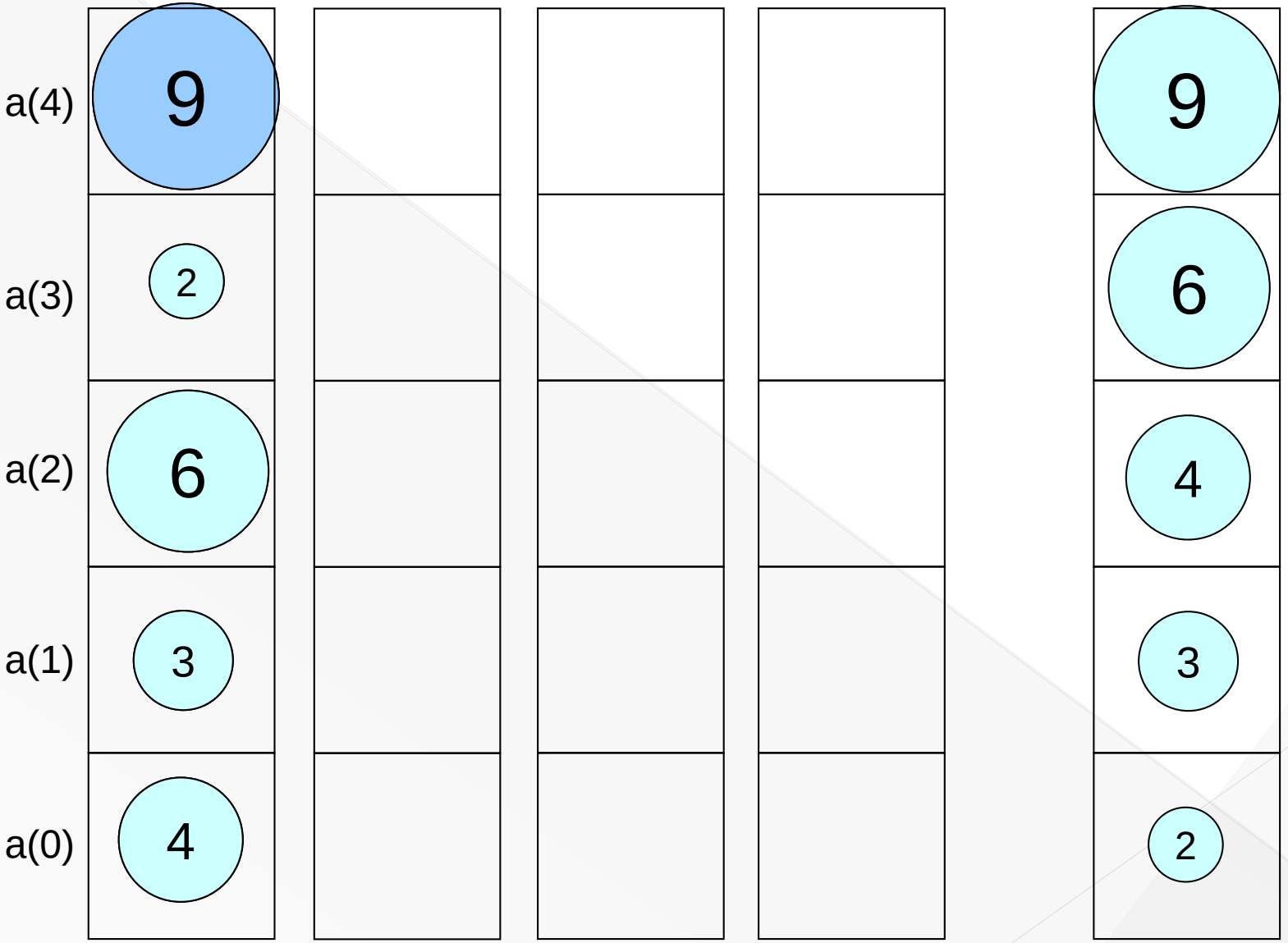
第一回合

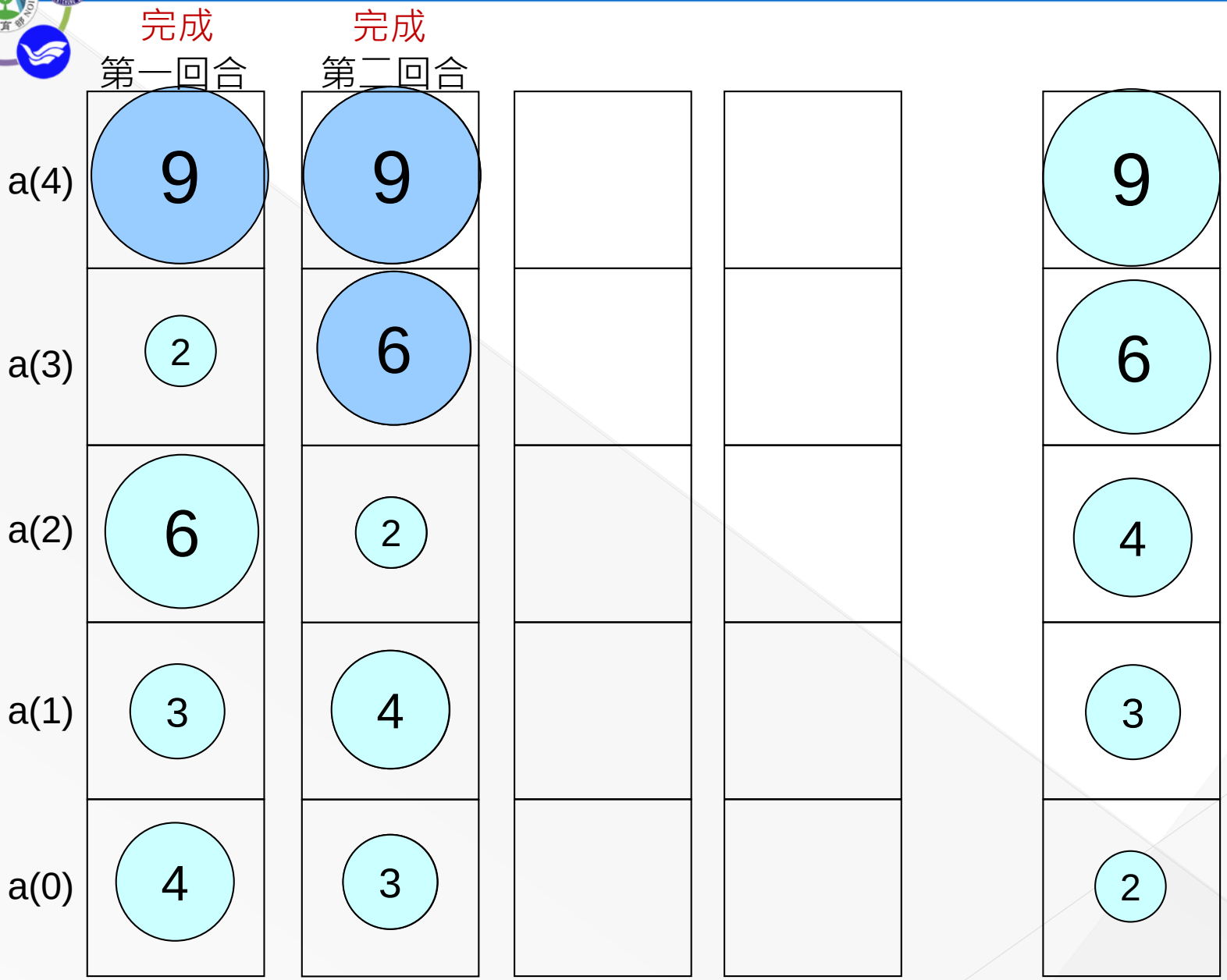


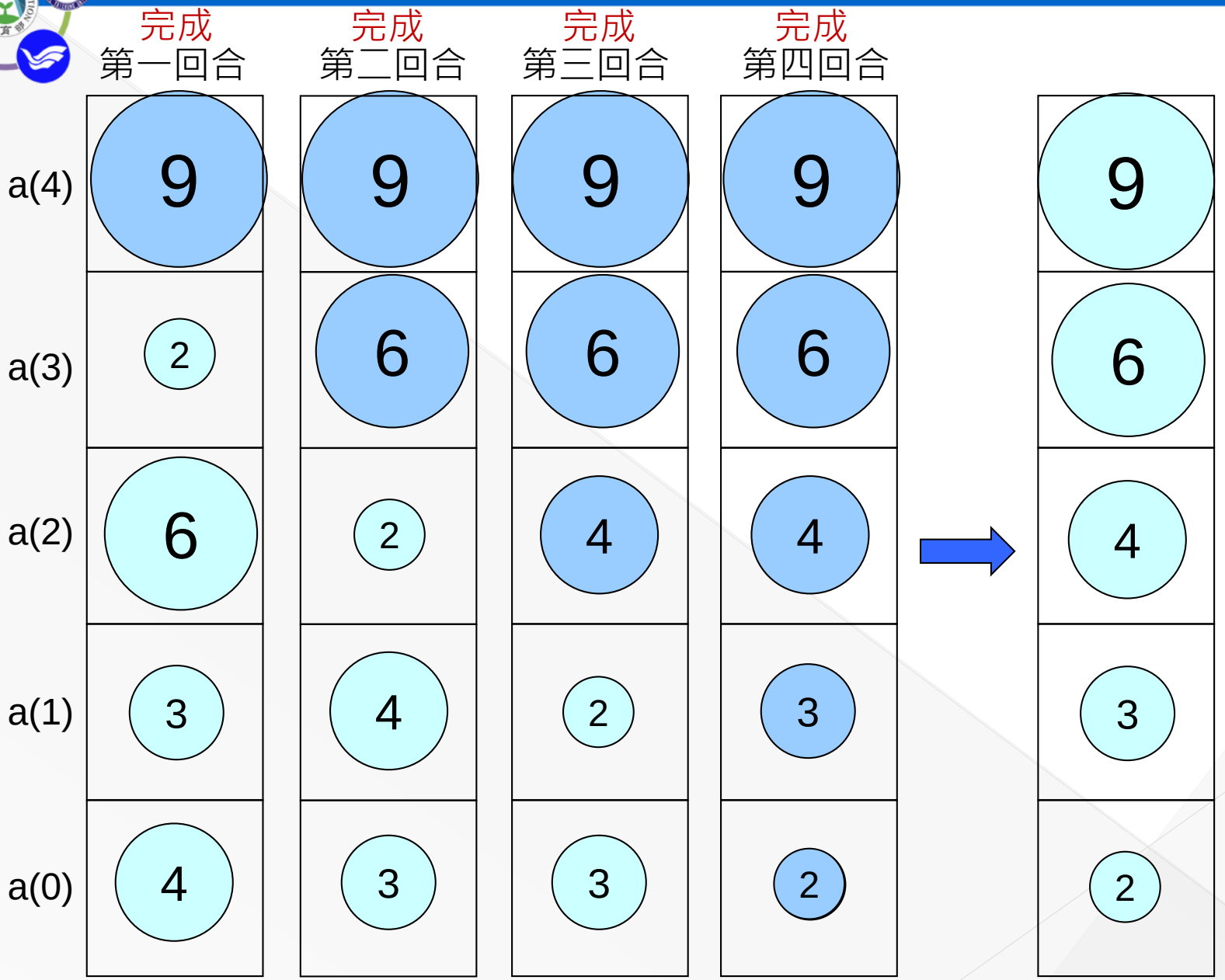
第一回合　以此類推繼續將最大的元素往上移動



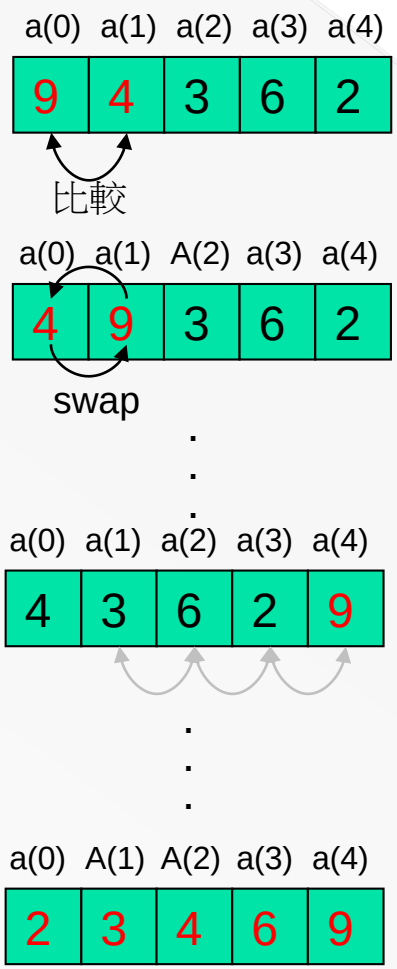
完成第一回合 以此類推繼續完成其他回合




完成
第一回合
完成
第二回合
a(4)
a(3)
a(2)
a(1)
a(0)
9
2
6
3
4
9
6
2
4
3
9
6
4
3
2


完成
第一回合
完成
第二回合
完成
第三回合
完成
第四回合
a(4)
a(3)
a(2)
a(1)
a(0)
9
9
9
9
9
2
6
6
6
6
6
2
4
4
4
3
4
2
3
3
4
3
3
2
2

Design Sketch (設計草圖)



若陣列中有n個數 (n>1)(左圖以 n=5為例)

1. 比較陣列中的 第一個氣泡(左邊的數)與
   第二個氣泡(右邊的數)

2. 如果第一個氣泡大於第二個氣泡，
   就交換兩氣泡,使大氣泡浮上來

3. 重覆1,2, 比較 第二個氣泡與第三個氣泡，
   第三個氣泡與第四個氣泡，
   …
   第 n-1個氣泡與第n個氣泡
   (這就完成了一回合)

4. 將 3重覆n-1回合，即完成排序

Pseudo Code (虛擬碼)

重覆 (3) 1到 n-1 回合

3.重覆 (1,2) 1到n-1次比較

1. 比較第一個氣泡和第二個氣泡
2. 如果第一個氣泡大於第二個氣泡，就交換兩氣泡，使大氣泡浮上來

下一次比較

下一回合

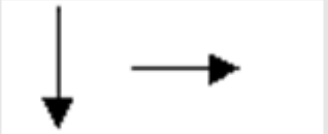
Pseudo Code with Java Source Code

```
/*重覆 (3)  1到 n-1 回合*/ for( int i = 0 ; i < n-1 ; i++){
    /*3.重覆 (1,2)  1到n-1次比較*/ for ( int j = 0 ; j < n-i-1 ; j++) {
      /*1.比較第一個氣泡和第二個氣泡*/ if (a[j] > a[j+1]){
      /*2.如果第一個氣泡大於第二個氣泡,*/
        /*就交換兩氣泡,使大氣泡浮上來*/
        int temp; temp=a[j]; a[j]=a[j+1]; a[j+1]=temp;}
    /*下一次比較*/ } // end of for j
/*下一回合*/ } // end of for i
```

Lab 2-4

- 請依以下需求繪製出Pseudo Code
  - 由使用者處得到三個數字並將三個數字的最大值輸出

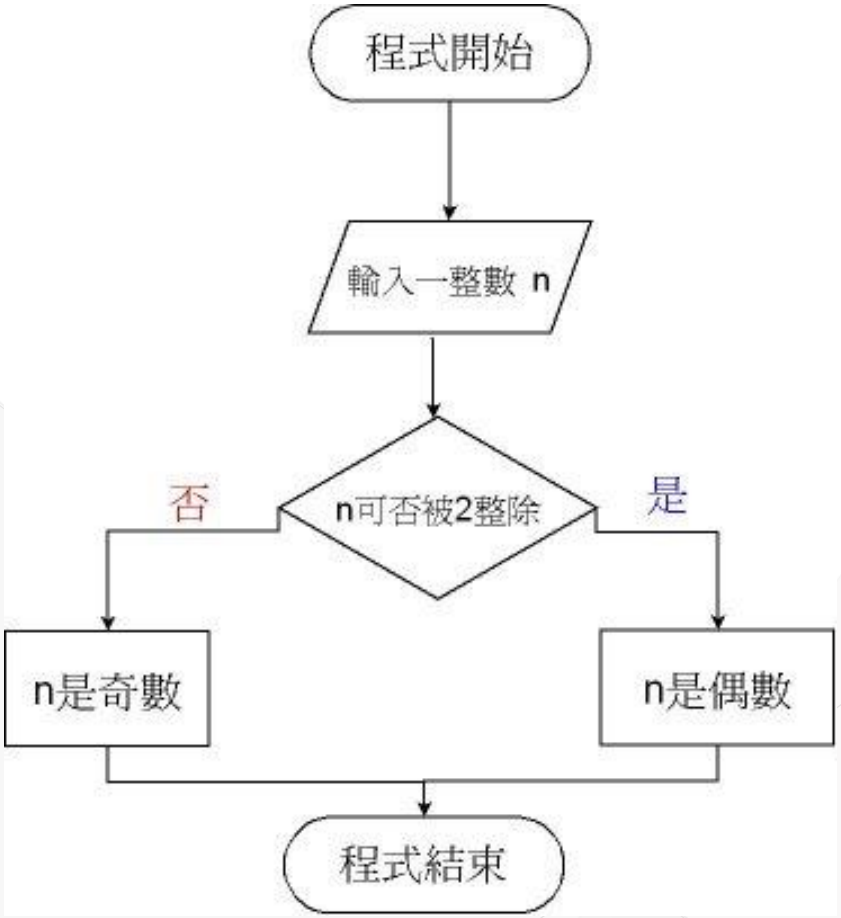
流程圖介紹

開始或結束工作的圖形	
輸入工作的圖形	
處理工作的圖形	
條件判斷的圖形	
工作流向的圖形	
連接點	

流程圖介紹

- 範例一：輸入一正整數，判斷為奇數或偶數。用流程圖方式表示。

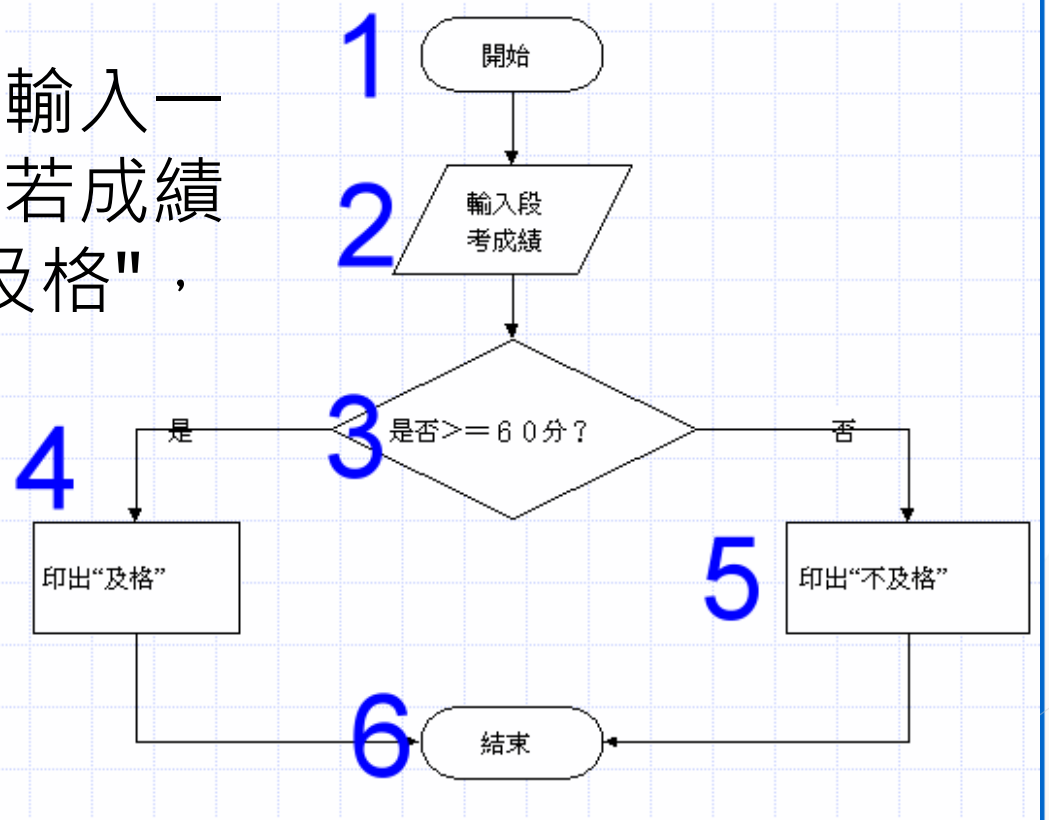
1. 由使用者處得到一整數
2. 判斷是否可以被2整除
3. 是的話輸出此數為偶數
4. 否的話輸出此數為奇數



流程圖介紹

- 範例二：設計一程式，輸入一學生成績，由程式判斷若成績在60分以上，請輸出"及格"，反之則為"不及格"。

  1、開始
  2、取得某科的段考成績
  3、判斷是否 > = ６０
  4、若是，則印出"及格"
  5、若否，則印出"不及格"
  6、結束



Lab 2-5

- 請依以下需求繪製出流程圖
  - 由使用者處得到三個數字並將三個數字的最大值輸出


Strat
輸入三個數字A、B、C
判斷何者為最大值
輸出最大值
End