2-1 演算法簡介

1. 演算法的特性
2. 演算法與電腦
3. 演算法的表達

1. 演算法的特性

什麼是演算法

- **演算法**：解決問題的方法。

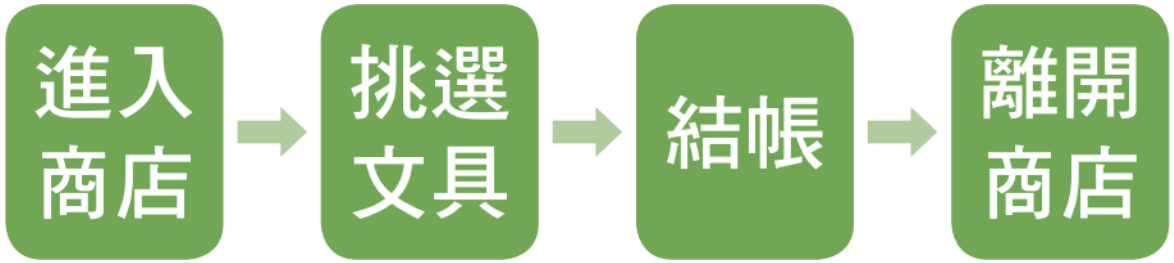
  例：食譜、數學題的解法等。

- 遵循演算法的步驟執行，將可以正確解決問題。

  →「任意變動」或「任意省略」演算法中的步驟，可能無法順利達到目的。

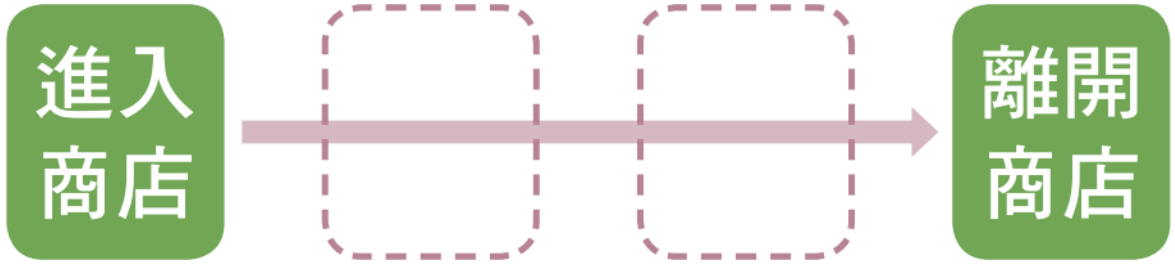
演算法的步驟與問題解決

- 「到商店購買文具」的演算法：



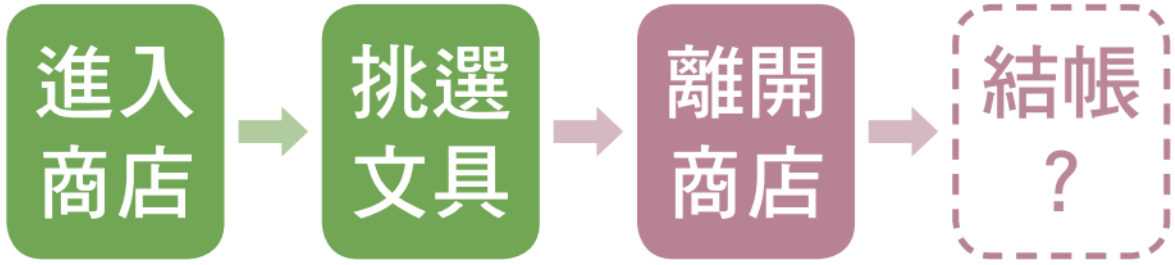
遵循演算法步驟。

可解決「購買文具」的問題。



任意省略步驟。

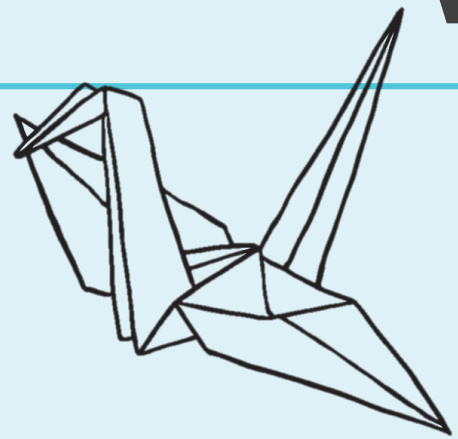
未解決「購買文具」的問題。



任意變動步驟順序。

造成犯罪行為。

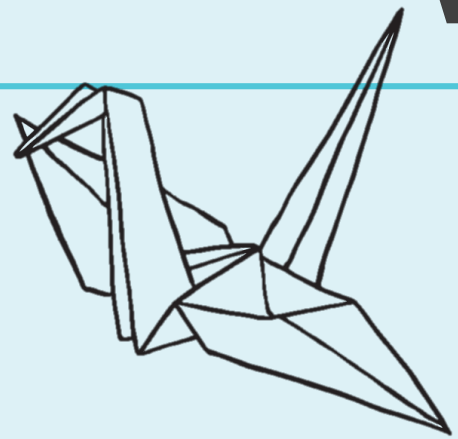
手腦並用



- **準備器材**：每人準備 1 張A4 紙。
- **活動方式**：

1. 一人背對大家，一邊摺紙，一邊敘述摺紙的步驟。
2. 其他人依敘述摺紙，不可提問、討論。
3. 數個步驟後，互相展示結果。

手腦並用



- **活動討論：**

  每個人所展示的摺紙結果都相同嗎？

  如果不同，是什麼原因造成的？

答 指令敘述不精確，導致每個人解讀不同，摺出來的結果就不一樣。

演算法的特性

- 生活中各種**解決問題的方法**，都可以視為**廣義的演算法**。但容易因為敘述不精確，導致解讀不同，而使結果有所差異。
- **資訊科技**領域中，**演算法**有嚴謹的定義，以確保能正確執行、有效解決問題。

演算法五大特性

1. **輸　入**：可有多個輸入資料，或是沒有輸入資料。
2. **輸　出**：必須至少有一個輸出結果。
3. **明確性**：每個指令必須明確，不可模稜兩可。
4. **有限性**：執行演算法，必須在有限步驟內結束。
5. **有效性**：又稱可行性，演算法中的每個命令都必須是可執行的步驟，用紙筆也能推演完畢，以確定能解決問題。

2. 演算法與電腦

演算法與電腦

- 電腦功能運作背後，都是遵循著**演算法**執行的。
- **演算法的功能：**

  明確地告訴電腦，碰到什麼狀況時，應該如何反應或執行什麼步驟。

演算法與電腦

- 由於每個人思考方式不同，針對同樣問題，會設計出不同演算法。
- 不同的演算法，會影響電腦處理問題的結果與效率。

延伸學習 如何設計出更好的演算法？

- 只要能解決問題，演算法就能成立。
- 一個問題可以有多種解決方法，我們仍會尋找更好的方法來達成目標。
- 隨著知識增長與經驗累積，就能改良、設計出更快捷的演算法。

3. 演算法的表達

演算法的表達方式

- 演算法沒有固定的呈現方式，只要能清楚表達，並符合演算法五大特性即可。

常見的表達方式

文字
用一般語言文字來描述執行步驟。

流程圖
用圖示符號來描述執行過程。

虛擬碼
用類似程式語言的方式來描述執行步驟。

①文字

- 文字表達方式最簡單，但敘述較為冗長，且容易因為每個人的表達和認知差異而造成誤解。

舉例

求甲、乙兩數相加之總和，以文字表示：

步驟1. 輸入「數字甲」。

步驟2. 輸入「數字乙」。

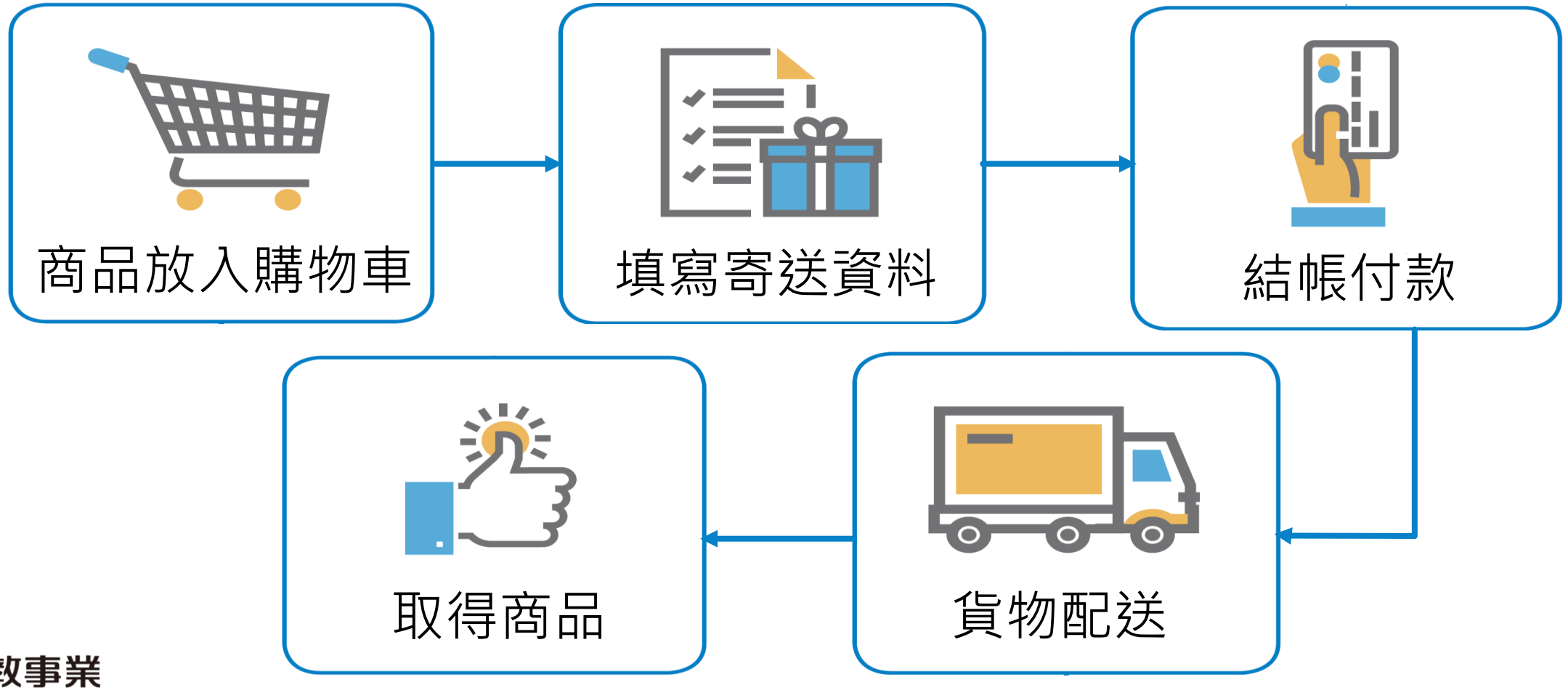
步驟3. 將「數字甲」及「數字乙」相加，所得即為「總和」。

步驟4. 輸出「總和」。

②流程圖

- 利用各種圖形與箭頭等符號，來表示執行一件事情的步驟與順序。

常見的網購流程



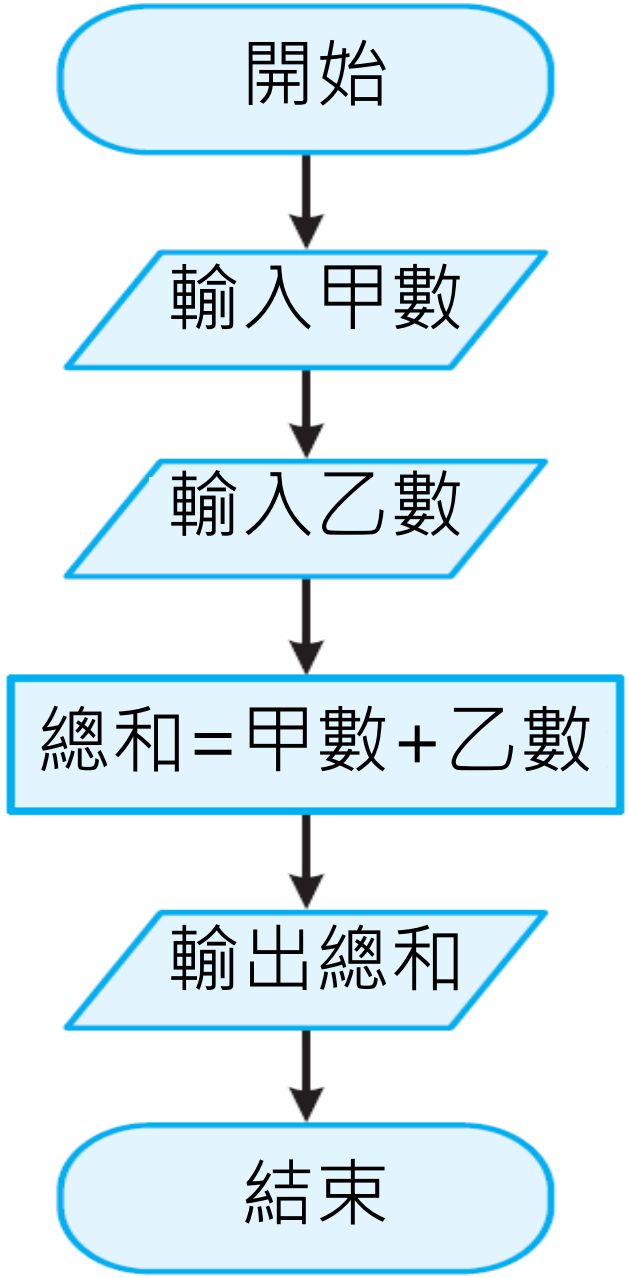
②流程圖

- 優點：
  1. 流程步驟的可讀性較高。
  2. 方便除錯。
  3. 容易修改流程。

②流程圖

舉例

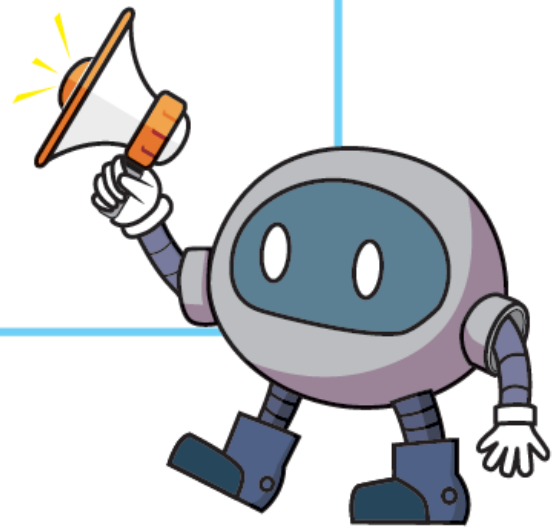
求甲、乙兩數之總和，以流程圖表示：



②流程圖

流程圖繪製原則

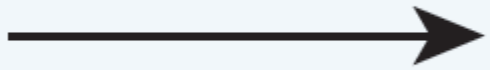
1. 使用標準符號，以便閱讀和分析。
2. 文字簡潔且明確可行。
3. 繪製方向由上而下，由左至右。
4. 流程線條避免交叉或過長。



②流程圖—符號

符號	名稱	說明
	開始 / 結束	流程的開始或結束
	輸入 / 輸出	輸入或輸出資料
	處理程序	要執行或處理的程序

②流程圖—符號

符號	名稱	說明
	決策判斷	針對條件進行判斷，以決定執行的流向
	迴圈	重複執行指定的次數
	流向線	流程進行的方向

②流程圖—符號

符號	名稱	說明
	註解	表示附註說明之用
	副程式	已定義流程的組合
	文件	輸入或輸出文件

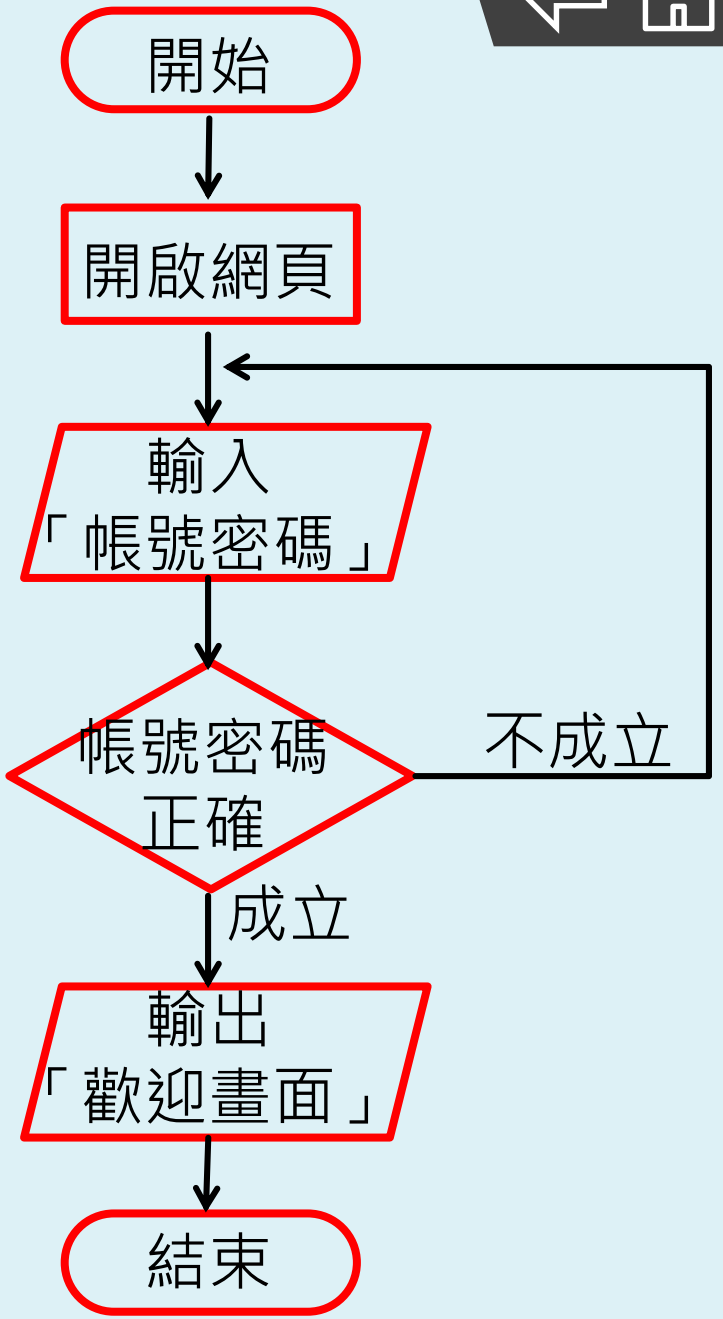
手腦並用

- 登入某網站時的操作流程：

使用者開啟網頁後要輸入帳號、密碼，若輸入正確，會顯示「歡迎畫面」；若輸入錯誤，則回到「輸入帳號、密碼」的步驟。

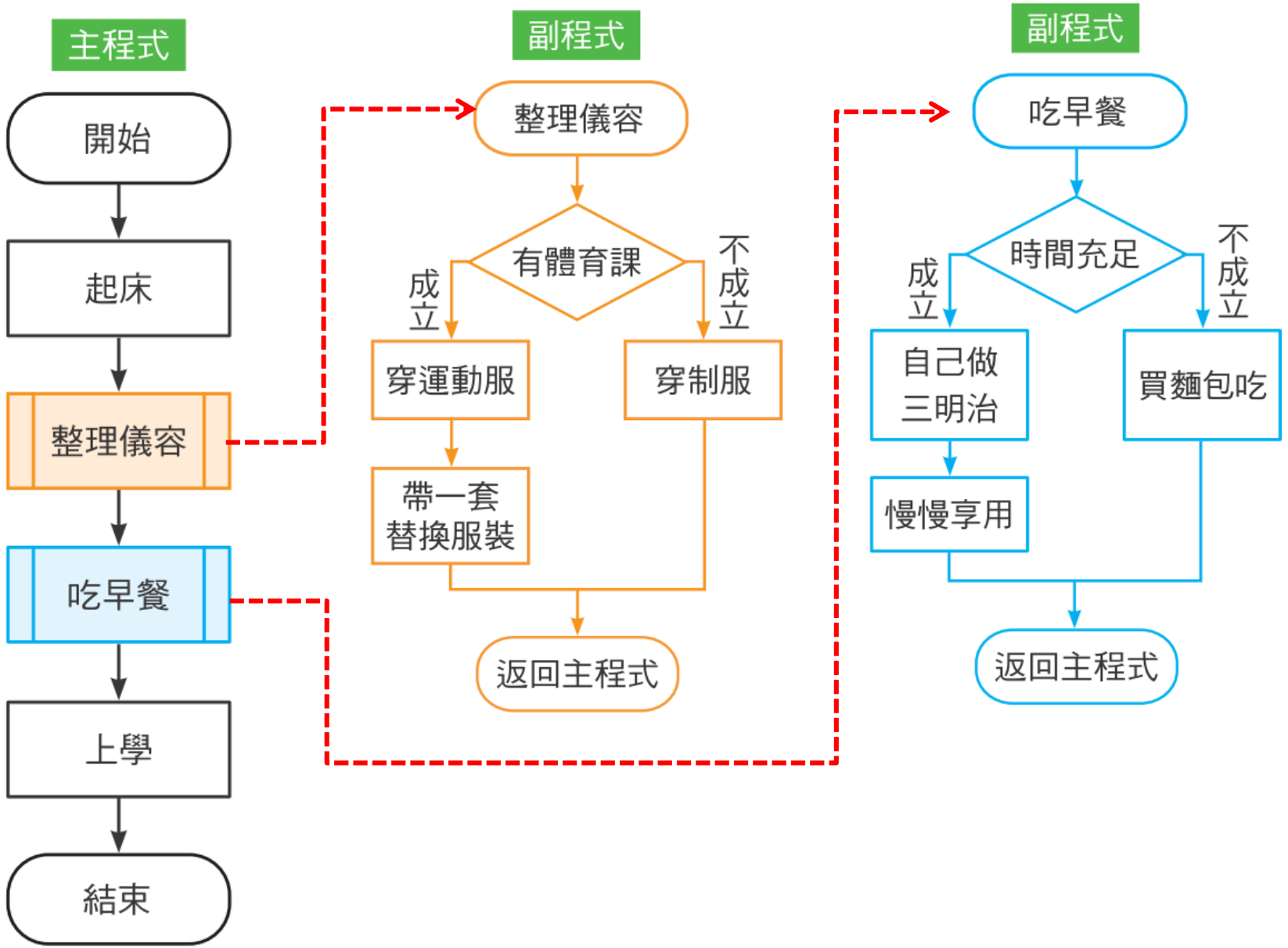
- 請為每一個步驟加上正確的流程圖符號。

答



②流程圖—模組化呈現

- 複雜流程可採用**模組化**的方式呈現，具有更高的可讀性。



③虛擬碼

- **虛擬碼**是一種介於「人類語言」與「程式語言」間的表達方式。
- 兼具文字描述容易表達，以及流程圖容易理解的優點。

三種表示法的比較

例1. 求甲、乙兩數相加之總和。

文 字

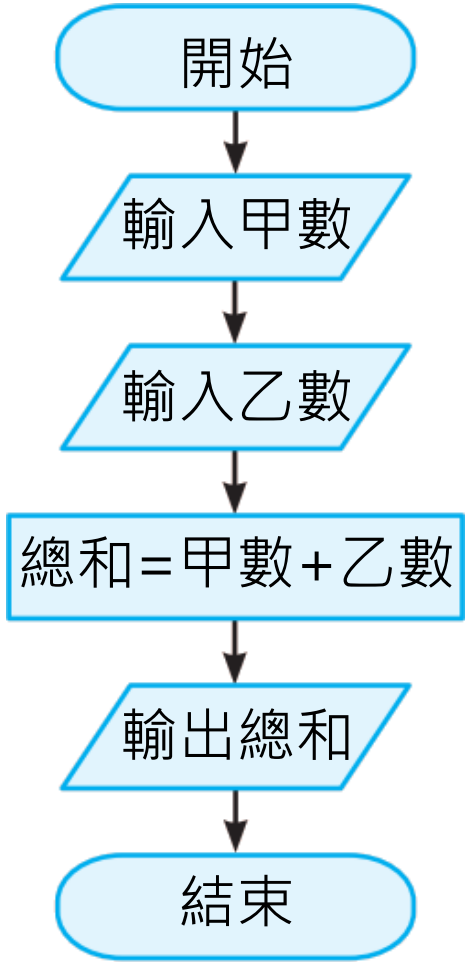
步驟1. 輸入「數字甲」。

步驟2. 輸入「數字乙」。

步驟3. 將「數字甲」及「數字乙」相加，所得即為「總和」。

步驟4. 輸出「總和」。

流程圖



虛擬碼

(1) input 甲

(2) input 乙

(3) sum = 甲 + 乙

(4) print sum

三種表示法的比較

例2. 依據「是否下雨」來決定行程。

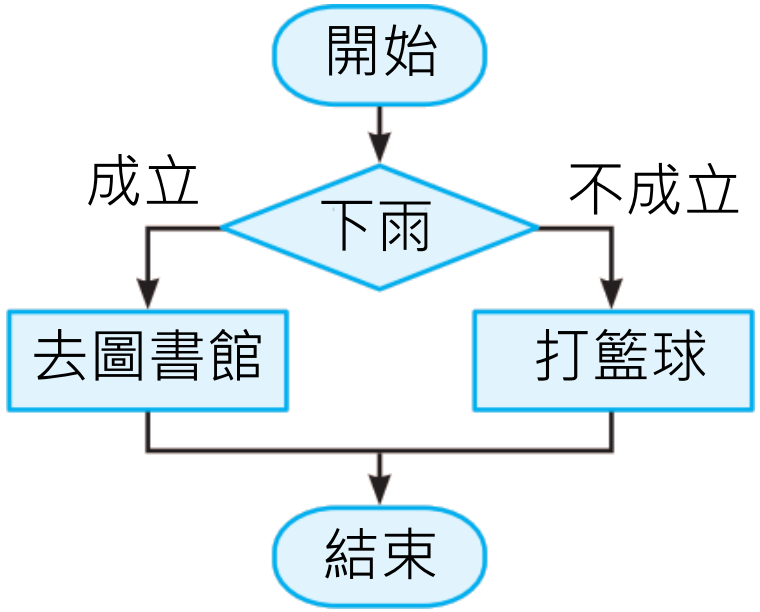
文 字

步驟1. 判斷是否下雨。

步驟2. 若下雨，就去圖書館。

步驟3. 若未下雨，就去打籃球。

流程圖



虛擬碼

(1) if 下雨

(2) then 去圖書館

(3) else 去打籃球

2．1 演算法簡介
結束