



第01章 常見的資料結構

<http://gg.gg/108IT-PlayList-YouTube>

1-1 資料結構簡介

- 將資料依照一定規律存放的方法，就稱為
資料結構（Data Structure）



圖1-1 在排列整齊的空間搜尋東西，會比在雜亂無章的空間裡搜尋更有效率

用族譜的結構描述複雜的家族關係

記錄筆記一家族關係調查

- 美枝與荷馬是夫妻
- 霸子的妹妹是點子
- 點子與小新是夫妻
- 小新是廣志的長子
- 美冴與廣志是夫妻
- 廣志的女兒是小葵
- 小新的岳父是荷馬
- 廣志的爸爸是阿銀
- 廣志的媽媽是阿鶴
- 芳治的女兒是美冴
- 美冴的妹妹是夢冴
- 夢冴的媽媽是高冴

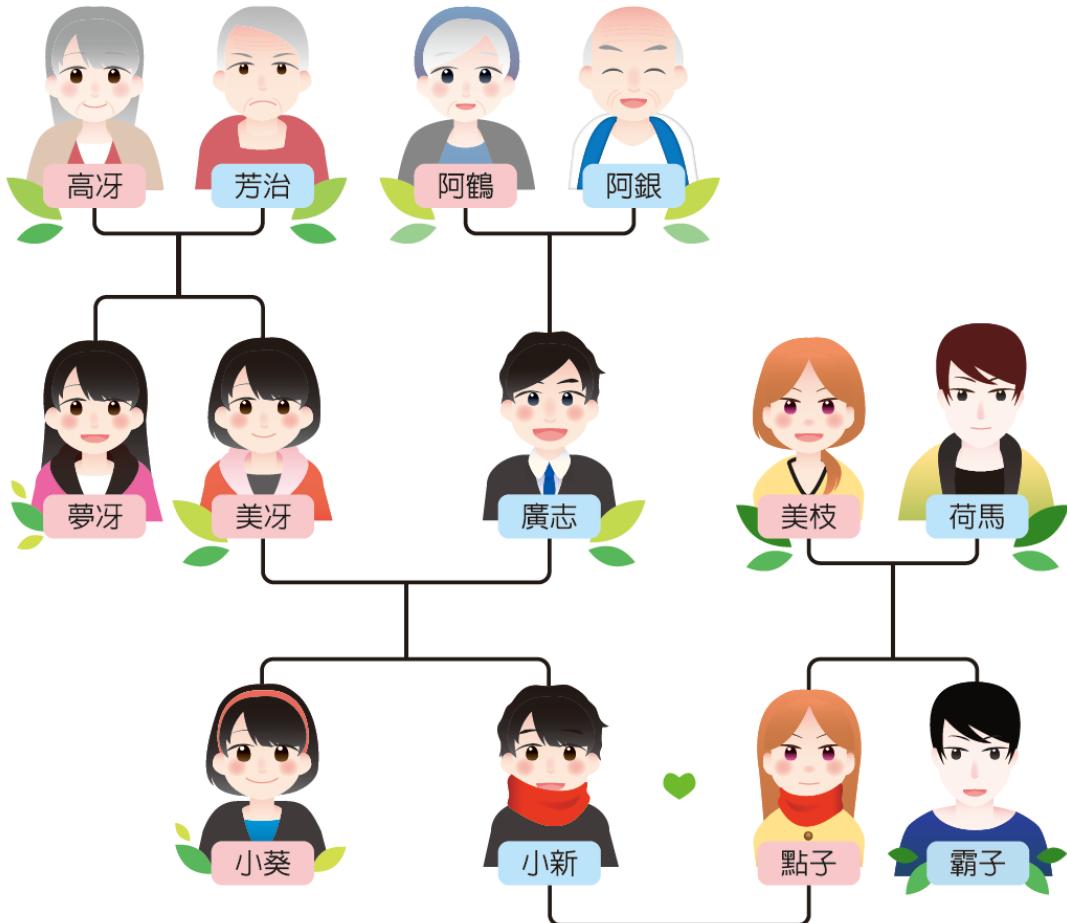


圖1-2 用族譜的結構描述複雜的家族關係，會比純文字敘述更清楚

1-2 常見的資料結構

補充

數學上的數列（例如：卡拉茲數列：10,5,16,8,4,2,1）或參加競賽名單（如 Hua, Erica, Masako, Cleopatra, Elisabeth, Sissi, Ki, Selene）在電腦科學上如何表達或存放呢？

The screenshot shows a web browser displaying a lesson from the Junyi Academy platform. The URL is <https://www.junyiacademy.org/root/junyi-math/m4nsl-/mjnsl8a/e/mjnsl8aa>. The page title is 【基礎】認識數列與數列的項. The main content area has a blue header with a star icon and the text 【基礎】認識數列與數列的項. Below it, a sub-section titled 技能進展：等差數列 is shown. A blue progress bar at the top indicates 5 items completed. The main text discusses the rule of the Kaprekar sequence: "若某項為偶數，則其下一項為其值減半。若某項為奇數，則下一項為某數數值先乘 3 再加 1。" A question below asks: "給定一個卡拉茲數列 **10, 5, 16, 8, 4, 2, 1**，則此數列有 項，末項為 。" To the right of the text are two buttons: "提交答案" (Submit Answer) in green and "解題說明" (Solution Explanation) in orange. On the left sidebar, there is a vertical list of other lessons under the heading "等差數列": B4-1-1 等差數列：教師用講義, B4-1-1 等差數列：學生用講義, 【基礎】數列與數列的項, 【基礎】認識數列與數列的項 (which is currently selected), and 【基礎】找出數列的規律.

▲ 圖 在電腦中，數列可用陣列的資料結構來表示

解

1-2 常見的資料結構

- 將同類型資料連續的放在一起，這種資料結構稱為**陣列**（Array）
- 例如：索引0可以取得數字50，這種長條狀的結構就稱為**一維陣列**。

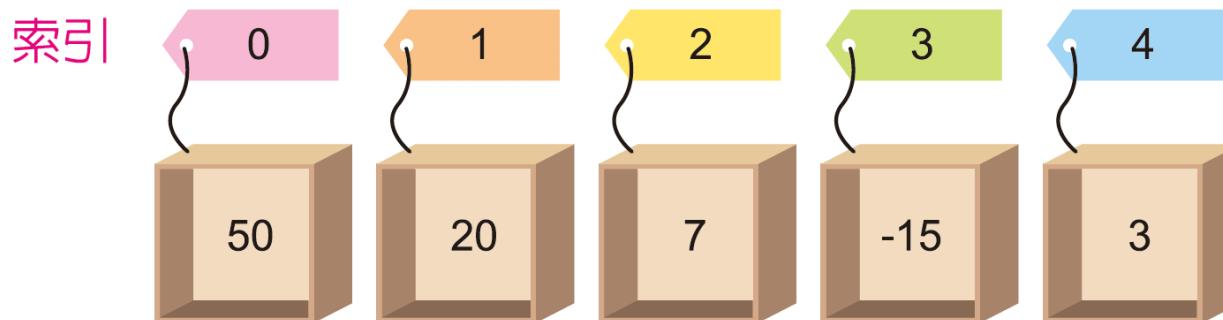


圖1-4 陣列

TIP 在電腦的世界裡，陣列的索引通常是從0開始。例如：圖1-4的陣列中有5個數字，所以索引分別是0, 1, 2, 3, 4。

二維陣列

□ 透過「一月」、「第1名」可以取得「獅子」

| 月份 名次 | 一月 | 二月 | 三月 |
|----------|-----|-----|-----|
| 1 | 獅子 | 熊 | 獅子 |
| 2 | 河馬 | 斑馬 | 猴子 |
| 3 | 長頸鹿 | 獅子 | 河馬 |
| 4 | 猴子 | 老虎 | 斑馬 |
| 5 | 熊 | 河馬 | 長頸鹿 |
| 6 | 老虎 | 猴子 | 老虎 |
| 7 | 斑馬 | 長頸鹿 | 熊 |

圖1-5 一至三月的Top 7明星動物排行榜：這個二維陣列的索引是「月份」及「名次」

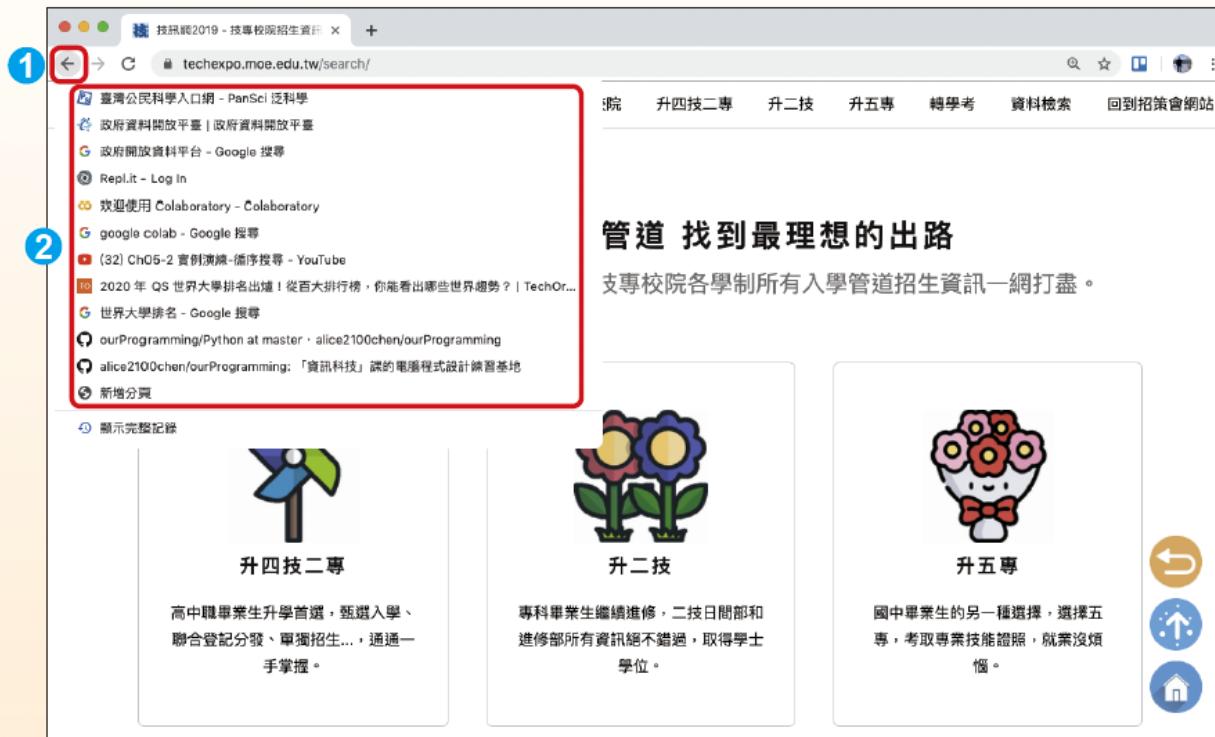


我們可以擴展到二維、三維，甚至更多維度的陣列，實用上大多使用一、二維陣列。

1-2 常見的資料結構



我們常使用應用軟體（如 WORD 等）中的「復原」（Undo）功能，或者瀏覽器（如 Chrome）的「上一頁」及「下一頁」鈕，如按住「上一頁」鈕時，會出現最近瀏覽過的網頁清單。



▲ 圖 瀏覽器的回到「上一頁」及「下一頁」功能屬於堆疊的應用

1-2 常見的資料結構

□ 後進先出特性的資料結構稱為**堆疊**（ Stack ）

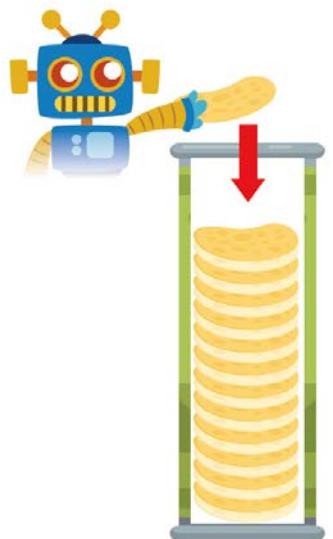


圖1-6 洋芋片裝罐與食用的順序



圖1-7 上班時間擁擠的電梯裡，最先進入電梯的人，會最後才出來



堆疊的運作原理

- 加入（Push）就好比把洋芋片裝入罐子內
- 取出（Pop）時，從頂端（Top）開始拿取

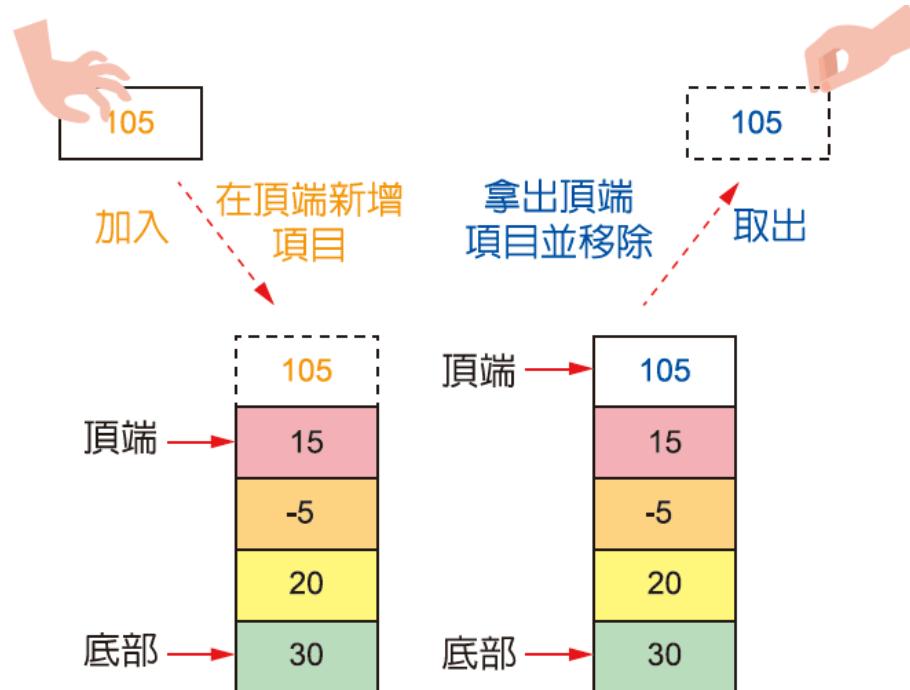


圖1-8 堆疊的運作原理

1-2 常見的資料結構

補充

當印表機接到多個列印的任務時，先送到的文件會先列印。交通顛峰時間，公車站牌前總是大排長龍，這是為了維護上車的順序，先排入隊伍的人會優先上車，這種**先進先出**（First In First Out，FIFO）的概念也常出現在現實生活中。

| hp LaserJet 2420 UPD PCL 6 | | | | | | |
|---|-----|------|-----|-----------|-----------------|----------|
| 印表機(P) 文件(D) 檢視(V) | | 狀態 | 擁有者 | 頁數 | 大小 | 已送交 |
| Microsoft Word - K104學習成果之書面報告-男女資訊科學家.docx | | user | 2 | 1.24 MB | 下午 12:18:39 ... | |
| Microsoft Word - K101基本資料之個人履歷.docx | | user | 2 | 304 KB | 下午 12:18:34 ... | |
| Microsoft Word - K103學習成果之實作作品.docx | | user | 3 | 280 KB | 下午 12:16:33 ... | |
| Microsoft Word - K105多元表現.docx | | user | 1 | 1.06 MB | 下午 12:16:29 ... | |
| Microsoft PowerPoint - K102學習成果之議題反思.pptx | 列印中 | user | 5 | 161 KB... | 下午 12:15:32 ... | DOT4_001 |
| 併列中的 5 文件 | | | | | | |

▲ 圖 1-4 印表機把接收到的印列任務，依「先送進來的先列印」就是併列的概念

1-2 常見的資料結構

- 倚列 (Queue) 是一種符合先進先出原則的資料結構



圖1-9 等待公車的隊伍：先排入隊伍的熊會最先上車，依序才是兔子與獅子

佇列的運作原理

- 加入（Enqueue）：從最尾端放入
- 移除（Dequeue）：從最前端拿取

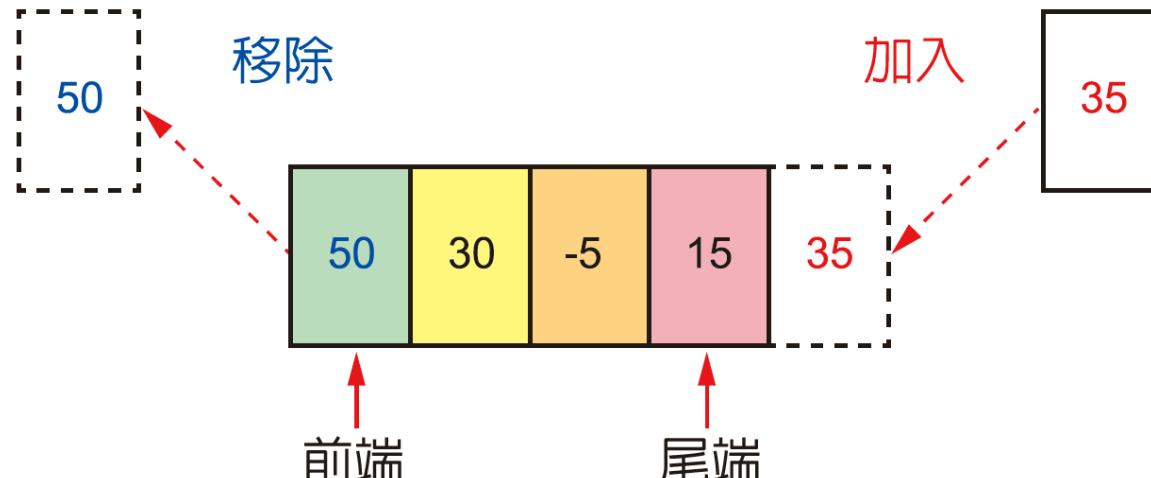


圖1-10 佇列的運作原理

1-2 常見的資料結構

補充

數學多項式 $a \times x^2 + b \times x + c$ 若以陣列結構表示，只要簡單的記錄係數即可。如

| x^2 | x^1 | x^0 |
|-------|-------|-------|
| a | b | c |

但是遇到 $2020 \times x^{109} + 3x + 2$ 時，如果仍使用陣列來表示的話，陣列的長度就會拉長且陣列充滿一堆的 0。如

| x^{109} | x^{108} | x^{107} | x^{106} | x^{105} | x^{104} | x^{103} | x^{102} | x^{101} | x^{100} | x^{99} | x^{98} | x^2 | x^1 | x^0 |
|-----------|-----------|-----------|-----------|-----------|-----------|-----------|-----------|-----------|-----------|----------|----------|-------|-------|-------|
| 2020 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 3 | 2 |

看來，使用陣列結構處理數學多項式有時候不是一個很有效的作法。以下說法，你贊成嗎？（請勾選）

- 以陣列表示多項式的好處是蠻直覺且簡單的作法
- 遇到上述的 $2020 \times x^{109} + 3x + 2$ ，使用陣列表示會佔用很大且無謂的記憶體空間
- 如果只需記錄 2020, 109, 3, 1, 2, 0 應該是不錯的作法



解

1-2 常見的資料結構

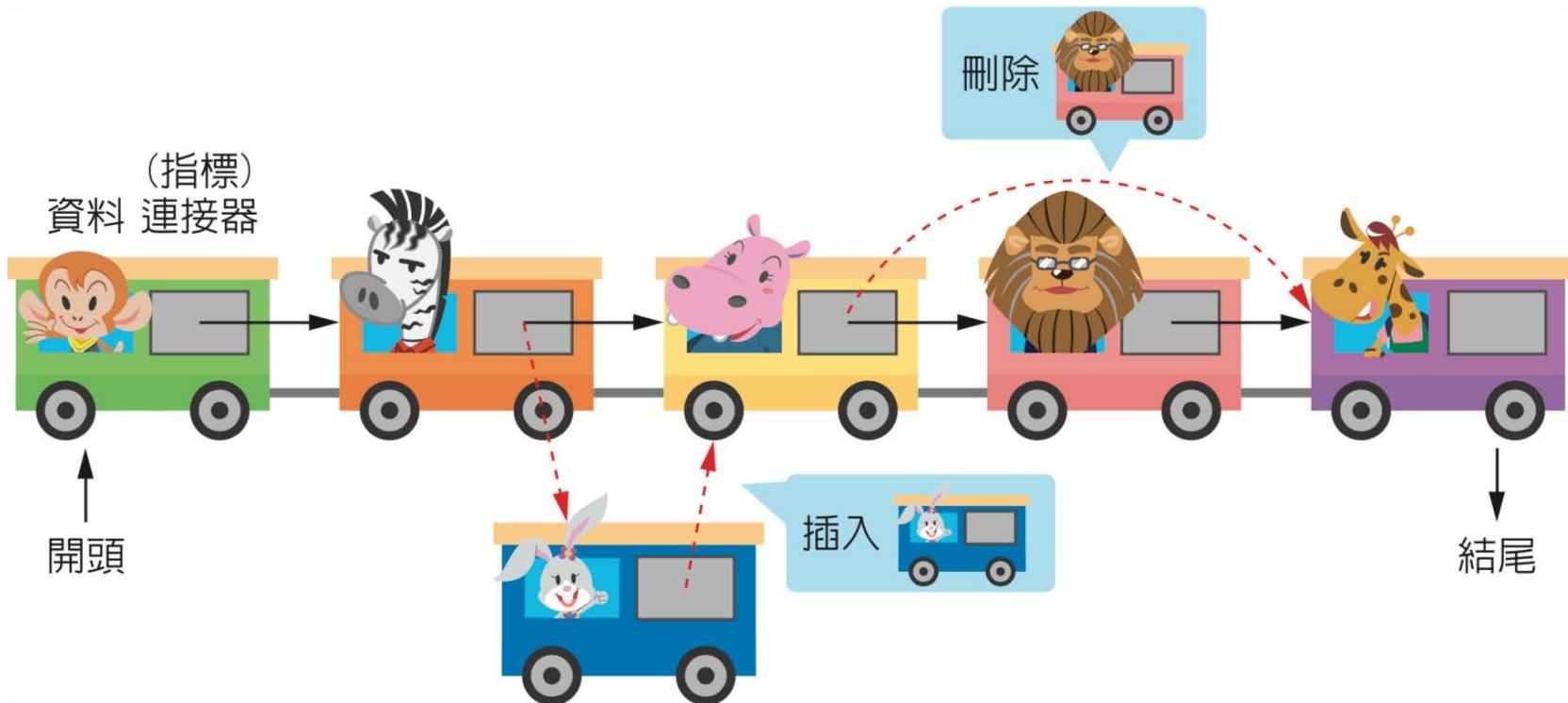
- 將物件串接在一起，方便增加或移除物件的方式，就是**鏈結串列**（Linked List）



圖1-11 火車或捷運列車由許多車廂串接在一起，方便加掛、替換或移除車廂

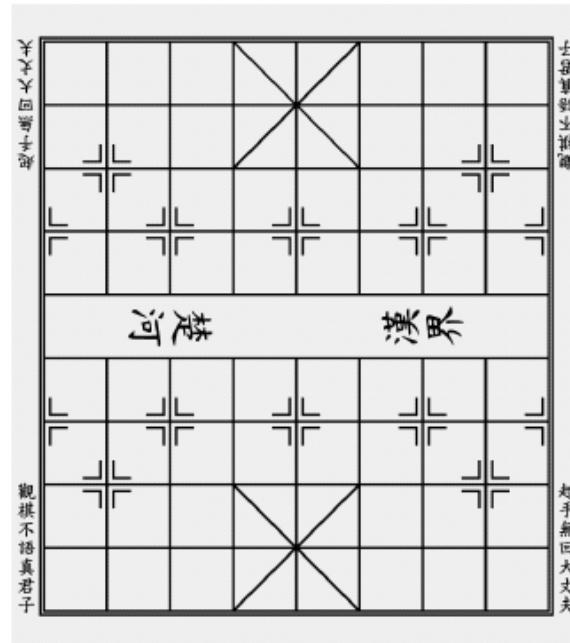
鏈結串列的運作原理

□ 插入或刪除節點，就像火車可以依需求增加或減少車廂數量





- (A) 1. 下列何者符合後進先出 (Last In First Out) 的生活實例？(A) 夜市的套圈圈遊戲 (B) 影印機的列印等候佇列 (C) 電影院買票的排隊人潮 (D) 便利商店等待結帳的人潮。
- (C) 2. 右圖中的「起手無回大丈夫」，雖然氣魄有餘，卻可能彈性不足，當我們想在下棋程式中加上「回到上一步」的功能時，適合採用下列哪種資料結構呢？(A) 陣列 (B) 鏈結串列 (C) 堆疊 (D) 佇列。



補充



假設數學多項式為 $2020 \times x^{11} + 109 \times x^2 + 5$ ，請在以下格子中填入係數 109 及 5（其餘補 0）。

1. 以陣列資料結構表示如下：

| | | | | | | | | | | | |
|------|---|---|---|---|---|---|---|---|-----|---|---|
| 2020 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 109 | 0 | 5 |
|------|---|---|---|---|---|---|---|---|-----|---|---|

2. 以鏈結串列的結構表示如下：

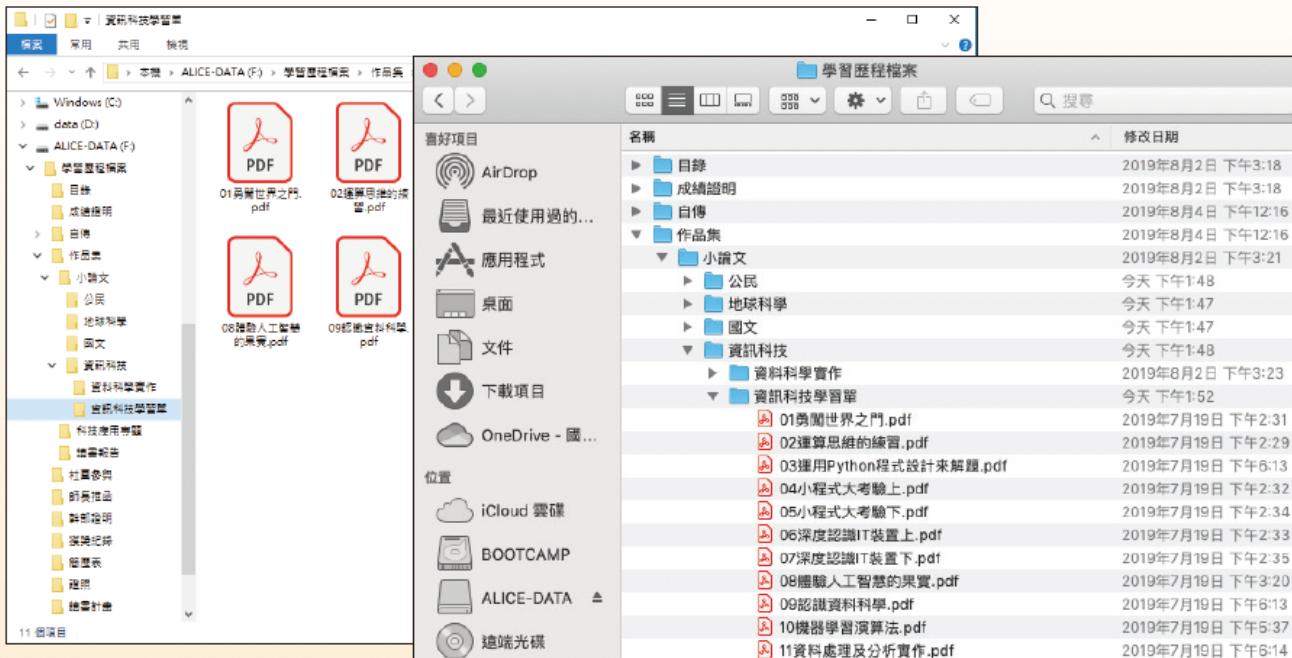


解

1-2 常見的資料結構



現今作業系統的檔案儲存結構是利用一種階層式架構的概念設計而成。常被用於分類問題的決策樹也同樣具有階層式概念。



▲ 圖 電腦的資料夾結構是屬於樹狀結構（左為 Windows，右為 macOS）



1-2 常見的資料結構

- 樹（Tree）是一種形狀類似樹木枝幹的資料結構（圖1-15），由節點（Node）與分支（Branch）組成

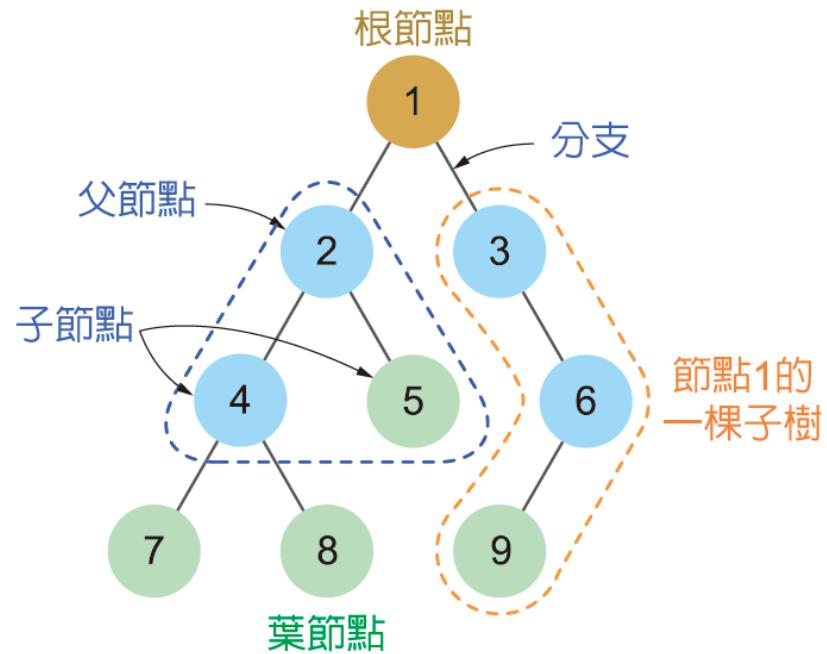


圖1-15 樹狀結構

賽程表是典型的樹狀結構

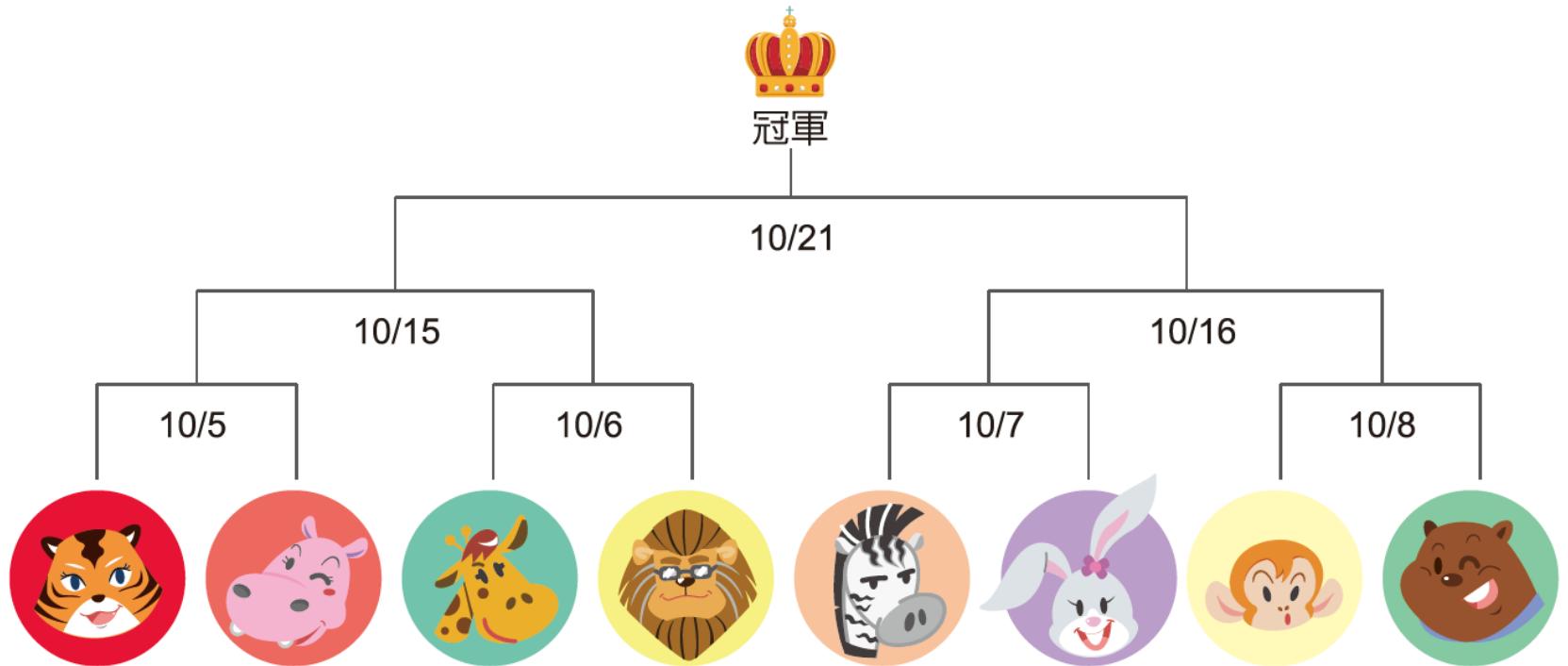


圖1-13 比賽的日期賽程表是典型的樹狀結構

電腦的資料夾結構是屬於樹狀結構

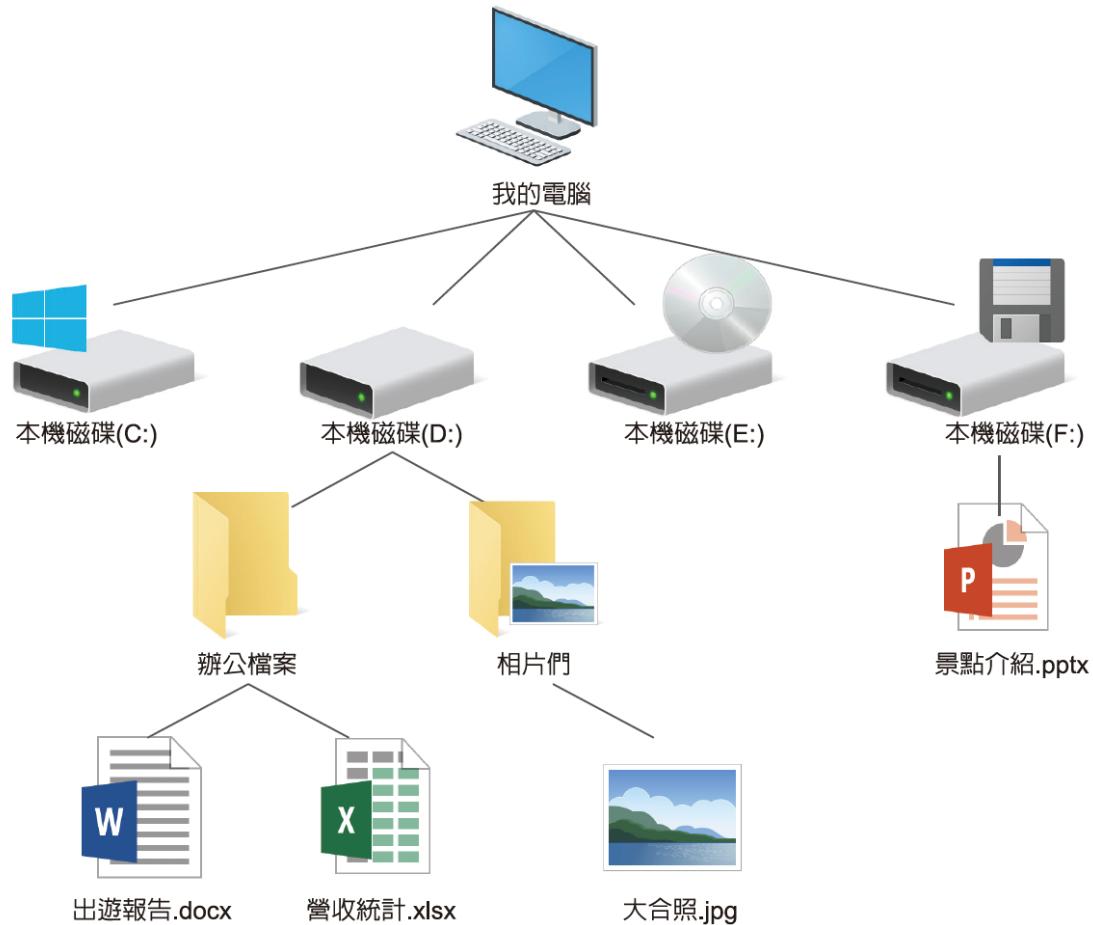


圖1-14 電腦的資料夾結構是屬於樹狀結構

二元樹

- 二元樹（Binary Tree）是指每個節點最多只有兩個子節點的樹狀結構

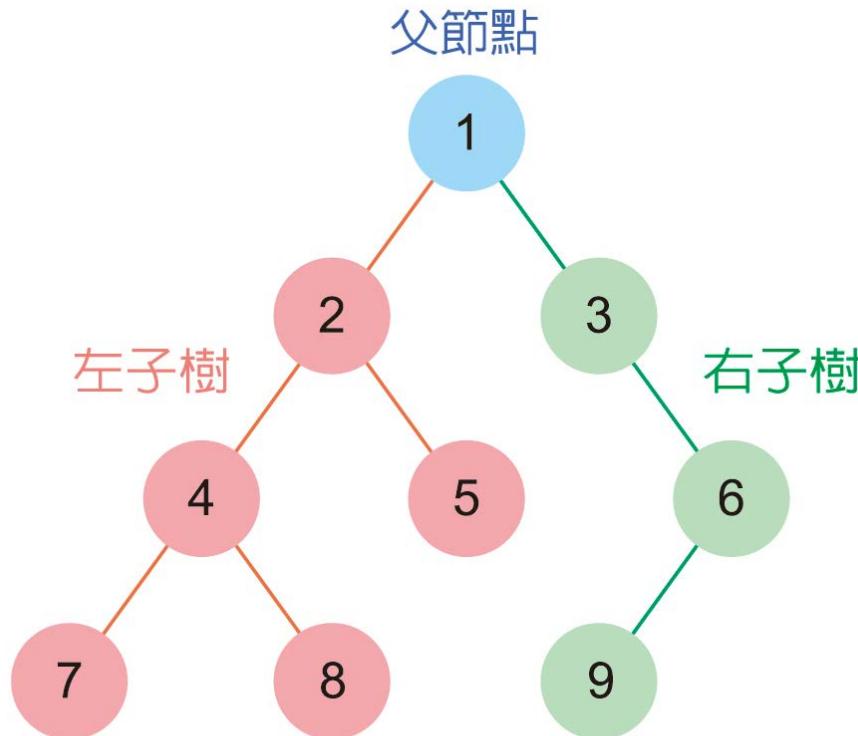


圖1-16 二元樹

二元搜尋樹

- 二元搜尋樹（Binary Search Tree）是一種專門用來搜尋的樹狀結構
- 樹中每個節點的數值，都比其右子樹裡所有節點的數值小，同時也比其左子樹裡所有節點的數值大

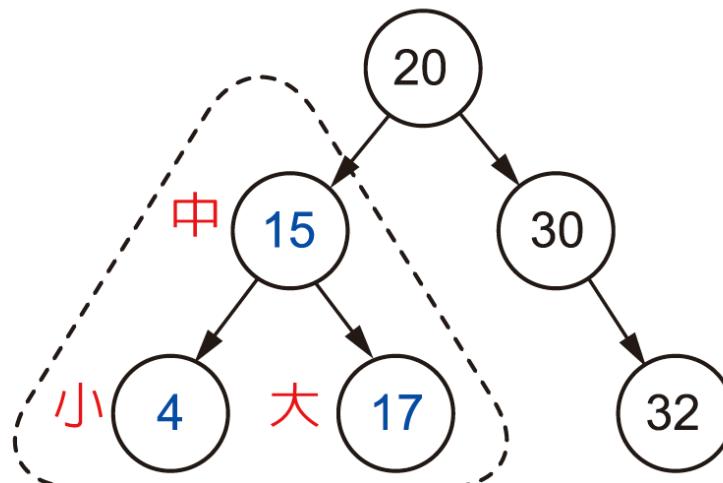
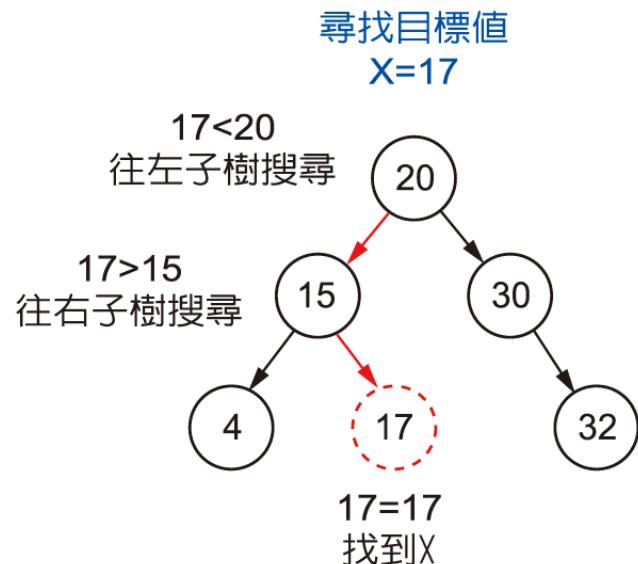


圖1-17 二元搜尋樹

在二元搜尋樹中搜尋目標值

- 尋找目標值17時，比較
「是否 $17=20$ 」、
「是否 $17=15$ 」及
「是否 $17=17$ 」
- 共比較3次



- 尋找目標值25時，比較
「是否」 $25=20$ 」及
「是否 $25=30$ 」，
□ 共比較2次

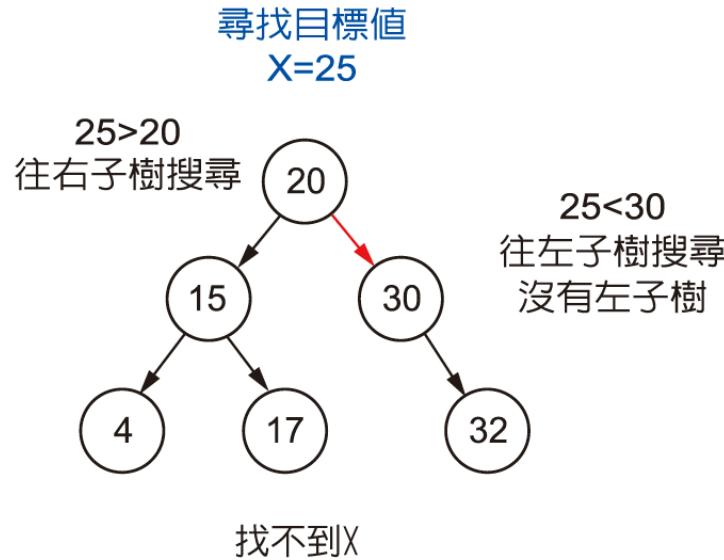
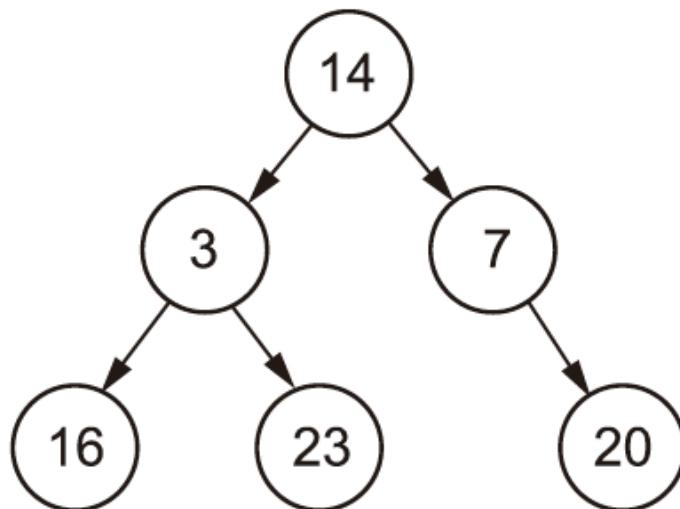


圖1-18 在二元搜尋樹中搜尋目標值的範例

前中後序走訪

- 走訪：拜訪樹上的每一個節點
- 分為前序走訪、中序走訪與後序走訪



前序走訪

14→3→16→23→7→20

中序走訪

16→3→23→14→7→20

後序走訪

16→23→3→20→7→14

圖1-19 前中後序走訪

前序走訪



前序走訪 (Pre-order)

根節點在前：先拜訪根節點，之後是左子樹，最後拜訪右子樹，如表 1-1。

表1-1 前序走訪

| 步驟 | 前序走訪 (口訣：根-左-右) | | | 新增 | 已走訪順序 |
|----|-----------------|-----------|-----------|----|-----------------|
| | 以14為根節點的樹 | 以3為根節點的子樹 | 以7為根節點的子樹 | | |
| 1 | 根節點14 | | | 14 | 14 |
| 2 | 左子樹 | 根節點3 | | 3 | 14→3 |
| 3 | | 左子樹 | | 16 | 14→3→16 |
| 4 | | 右子樹 | | 23 | 14→3→16→23 |
| 5 | 右子樹 | | 根節點7 | 7 | 14→3→16→23→7 |
| 6 | | | 左子樹 | 無 | |
| 7 | | | 右子樹 | 20 | 14→3→16→23→7→20 |

中序走訪



中序走訪 (In-order)

根節點在中間：先拜訪左子樹，接著是根節點，最後拜訪右子樹，如表 1-2。

表1-2 中序走訪

| 步驟 | 中序走訪（口訣：左-根-右） | | | 新增 | 已走訪順序 |
|----|----------------|-----------|-----------|----|-----------------|
| | 以14為根節點的樹 | 以3為根節點的子樹 | 以7為根節點的子樹 | | |
| 1 | 左子樹 | 左子樹 | | 16 | 16 |
| 2 | | 根節點3 | | 3 | 16→3 |
| 3 | | 右子樹 | | 23 | 16→3→23 |
| 4 | 根節點14 | | | 14 | 16→3→23→14 |
| 5 | 右子樹 | | 左子樹 | 無 | |
| 6 | | | 根節點7 | 7 | 16→3→23→14→7 |
| 7 | | | 右子樹 | 20 | 16→3→23→14→7→20 |

後序走訪



後序走訪 (Post-order)

根節點在後：先拜訪左子樹，接著是右子樹，最後拜訪根節點，如表 1-3。

表1-3 後序走訪

| 步驟 | 後序走訪 (口訣：左-右-根) | | | 新增 | 已走訪順序 |
|----|-----------------|----------|-----------|----|-----------------|
| | 以14為根節點的子樹 | 以3為根節點的樹 | 以7為根節點的子樹 | | |
| 1 | 左子樹 | 左子樹 | | 16 | 16 |
| 2 | | 右子樹 | | 23 | 16→23 |
| 3 | | 根節點3 | | 3 | 16→23→3 |
| 4 | 右子樹 | | 左子樹 | 無 | |
| 5 | | | 右子樹 | 20 | 16→23→3→20 |
| 6 | | | 根節點7 | 7 | 16→23→3→20→7 |
| 7 | 根節點14 | | | 14 | 16→23→3→20→7→14 |



前中後序走訪示意圖

從上面的前中後序走訪範例中，我們能不能觀察出什麼規則呢？參考圖1-20，如果沿著二元樹上標示的箭頭方向走，凡是遇到（紅、藍、紫色的）點，就印出旁邊節點的數值，如此一來，就能得到前中後序走訪的結果。

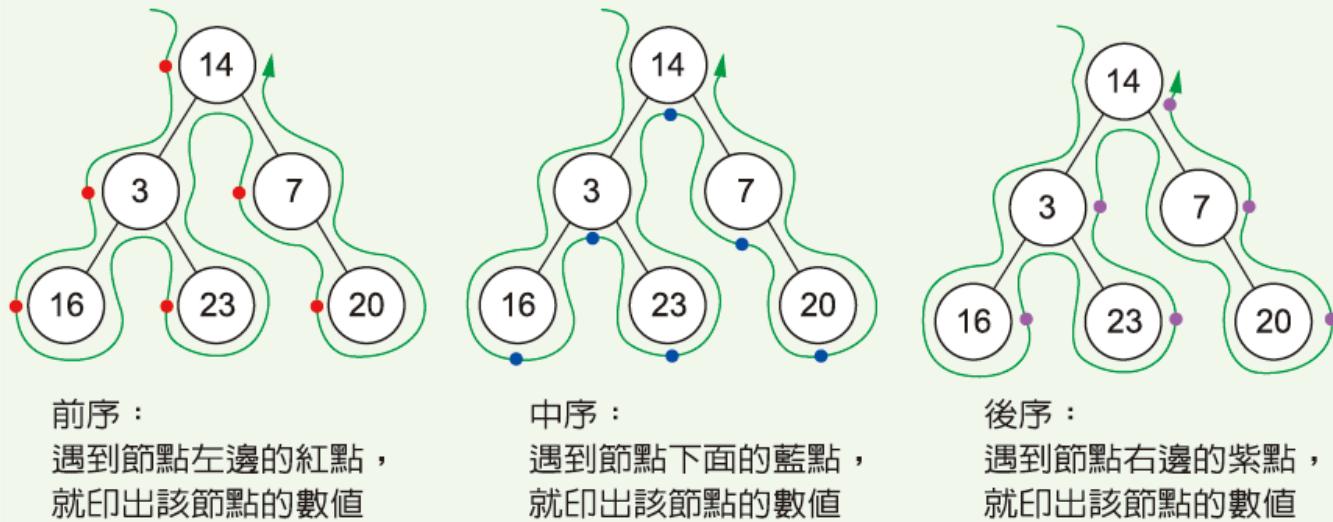


圖1-20 前中後序走訪示意圖

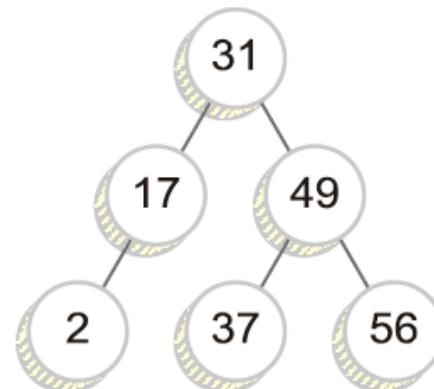
補充



1. 右圖是一個二元搜尋樹，請問在樹中搜尋「37」需要進行多少次數值大小的比較（如 $37 > 31$ 便是一次比較）？ 3
2. 如果使用中序走訪圖 4-27 的二元搜尋樹，會得到什麼樣的序列？

提示：利用二元搜尋樹的特性「樹根居中，左子樹較小或相等，右子樹較大」，具有排序效果。

2, 17, 31, 37, 49, 56

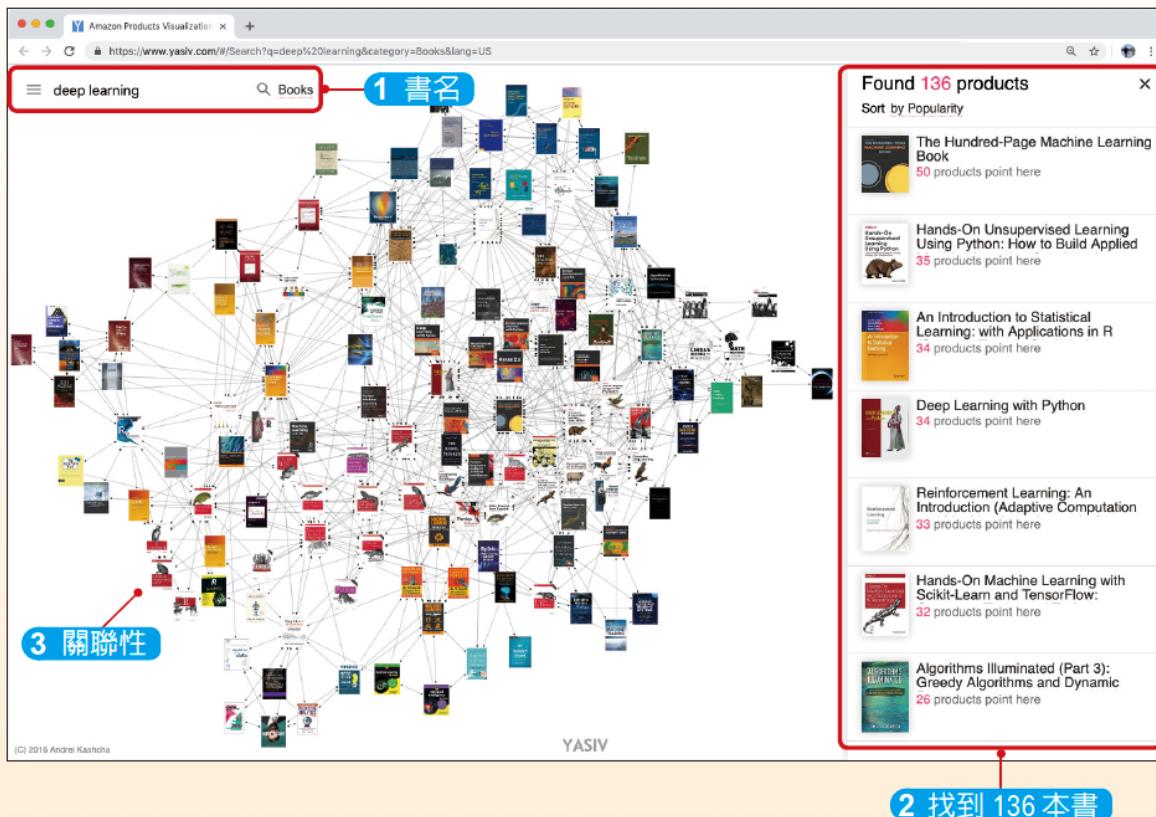


解

1-2 常見的資料結構

補充

從前我們常利用關鍵字進行「搜尋」，現在可以進一步找「關聯性」，例如：利用書名找到所有相關的書。



▲ 圖 在 yasiv 網站 (www.yasiv.com) 輸入「deep learning」尋找書及書之間的關聯

1-2 常見的資料結構

- 圖 (Graph) : 由數個節點 (Vertex) , 以及連接著這些節點的邊 (Edge) 所組成的資料結構

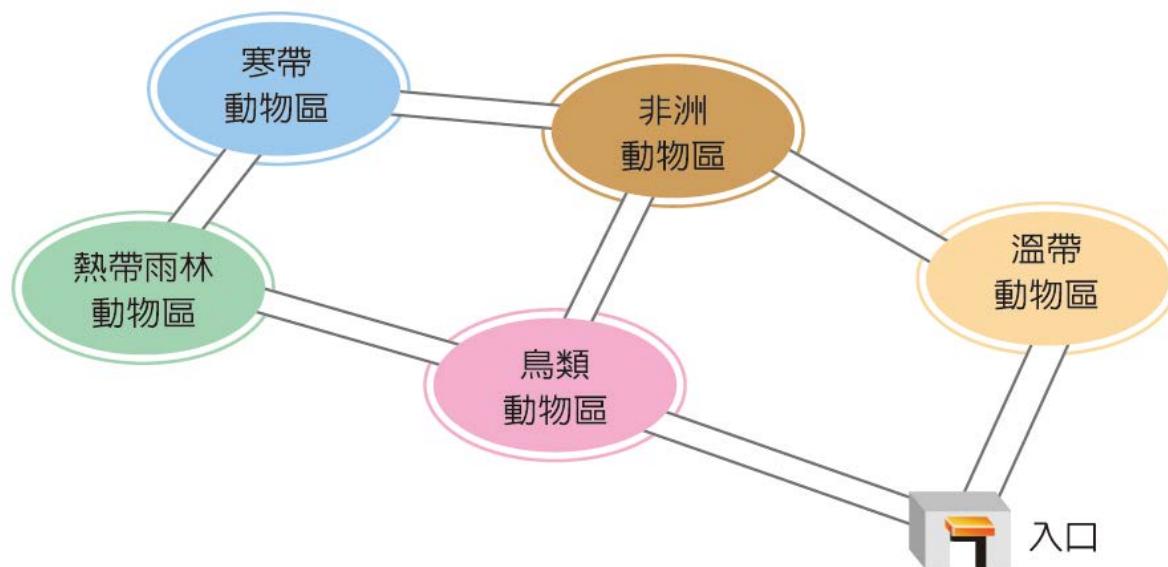
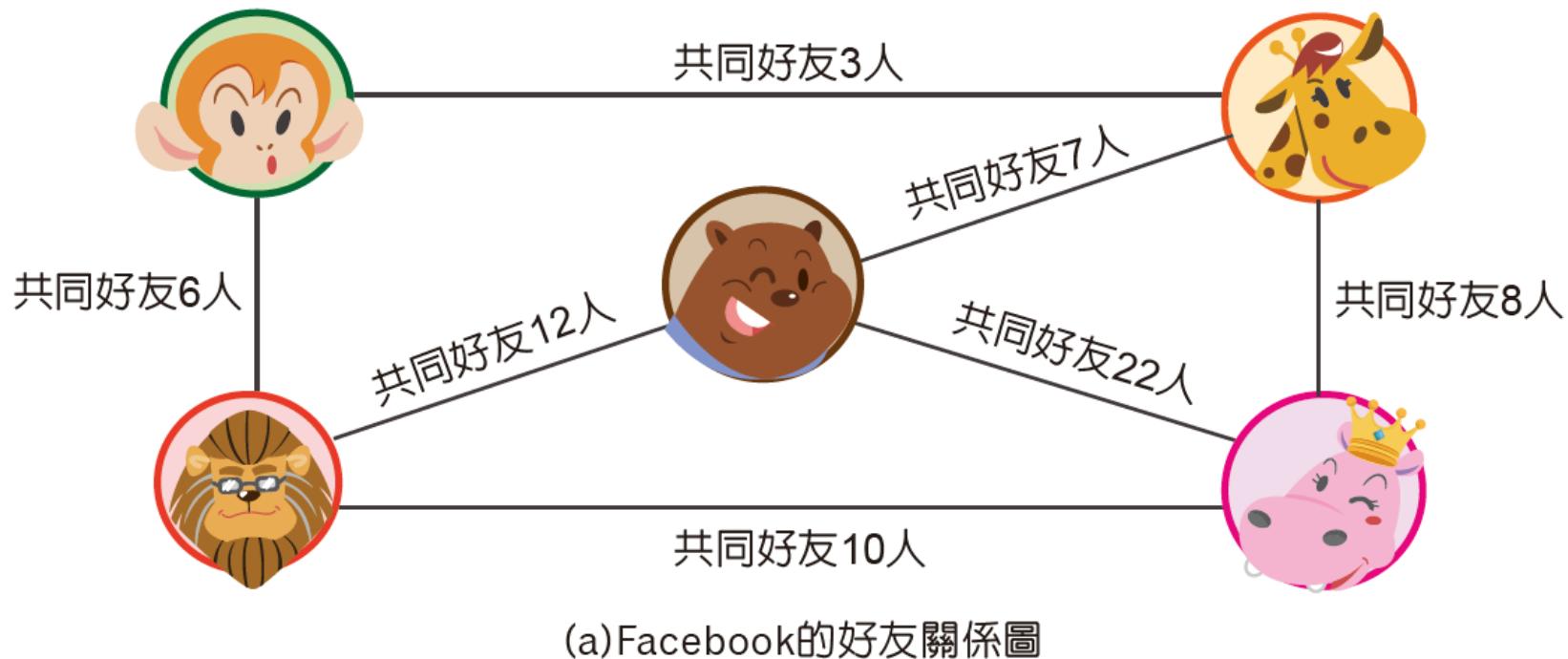


圖1-21 動物園地圖描述各個地區之間的地理關係

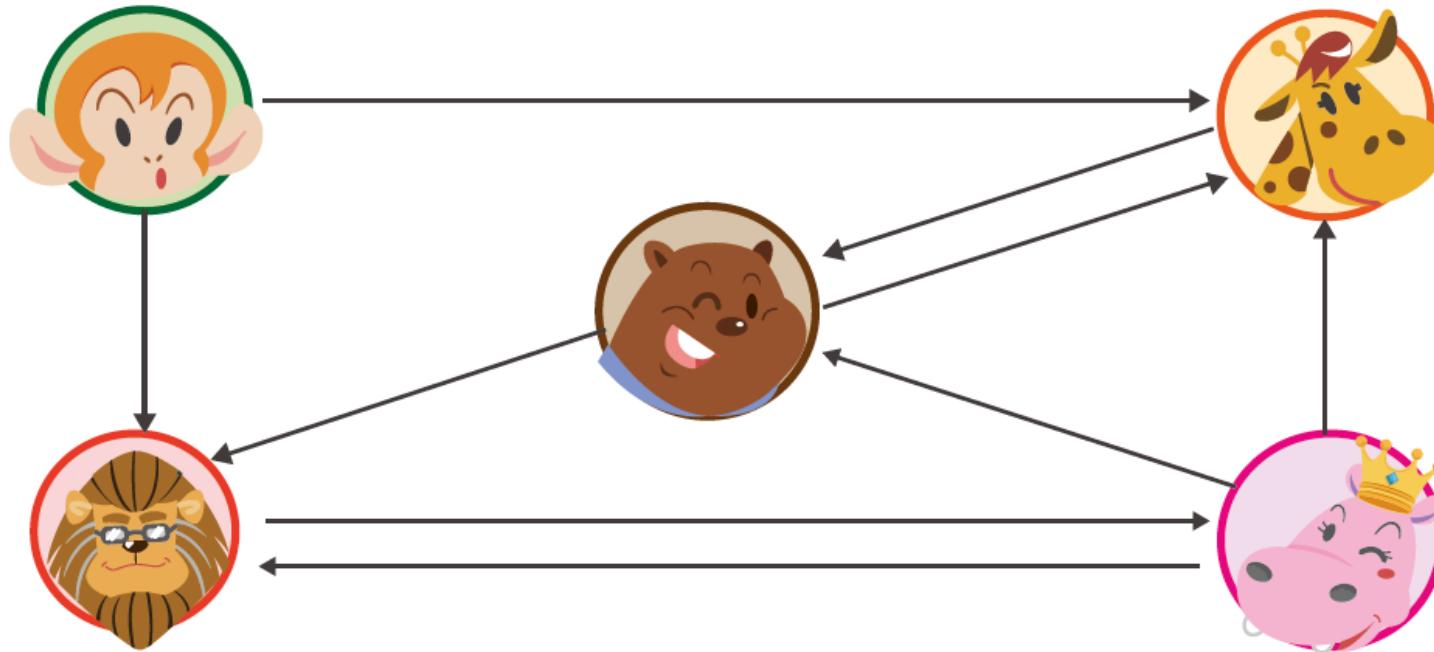
無向圖 (Undirected Graph)

- 無向圖的邊沒有方向性
- Facebook的好友機制中，成為好友後就可以看到彼此的貼文



有向圖 (Directed Graph)

- 有向圖的邊有方向性
- Instagram需要追蹤對方才能看到對方的貼文，就算被追蹤，也看不到追蹤者的貼文



(b)Instagram的好友關係圖

圖的表示法

- 圖這種資料結構主要用來描述節點與節點之間的關係。

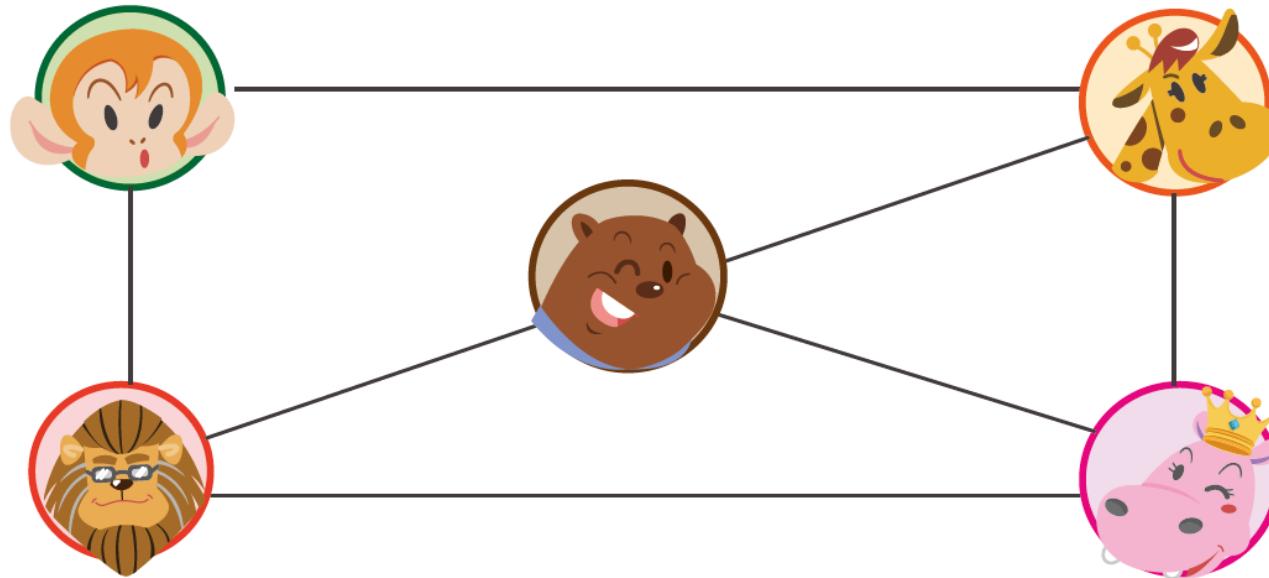


圖1-23 用圖的結構來表達動物之間的朋友關係

相鄰矩陣

□ 相鄰矩陣（Adjacency Matrix）：兩兩節點間如果有邊連接，則相鄰矩陣（屬於二維陣列）中對應的元素就是1，反之則為0。

表1-4 圖的表示法：利用相鄰矩陣表示動物之間的朋友關係圖

| | 猴子 | 獅子 | 長頸鹿 | 河馬 | 熊 |
|-----|----|----|-----|----|---|
| 猴子 | 0 | 1 | 1 | 0 | 0 |
| 獅子 | 1 | 0 | 0 | 1 | 1 |
| 長頸鹿 | 1 | 0 | 0 | 1 | 1 |
| 河馬 | 0 | 1 | 1 | 0 | 1 |
| 熊 | 0 | 1 | 1 | 1 | 0 |

補充



- (B) 1. 下列何者不是構成圖的元素？(A) 節點 (B) 父節點 (C) 邊 (D) 權重。
- (D) 2. 右圖表示五個景點之間路線及預計交通時間（小時），如果以「圖」（Graph）做為資料結構，則數字「2,2,1,3,6,7」代表下列何者？
(A) 圖的複雜度 (B) 圖的方向 (C) 節點的數量 (D) 邊的權重。



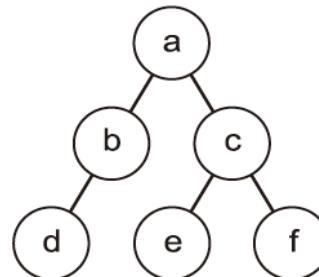
▲ 圖 台灣五個知名景點旅遊圖

解

實力挑戰

選擇題

- () 1. 將資料依照特定的形式與方法排列並組織在一起，稱為？
(A)資料結構 (B)資料組織 (C)資料排列 (D)資料構成。
- () 2. 請問先進先出 (First In First Out) 是哪一種資料結構的特性？
(A)陣列 (B)堆疊 (C)佇列 (D)樹。
- () 3. 下列何者符合後進先出 (Last In First Out) 的生活實例？
(A)電影院買票的排隊人潮 (B)影印機的列印等候佇列
(C)夜市的套圈圈遊戲 (D)便利商店等待結帳的人潮。
- () 4. 下列何者不是鏈結串列相對於陣列的優點？
(A)插入資料的速度較快 (B)刪除資料的速度較快
(C)讀取一筆資料的速度較快 (D)資料可以分散在不連續的記憶體。
- () 5. 下列的樹狀結構，使用後序走訪，會得到何者序列？
(A)abcdef (B)bcdefa (C)fedcba (D)dbefca。



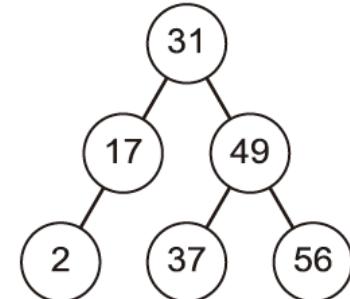
實力挑戰

() 6. 下列何者不是構成圖的元素？

- (A)節點 (B)父節點 (C)邊 (D)權重。

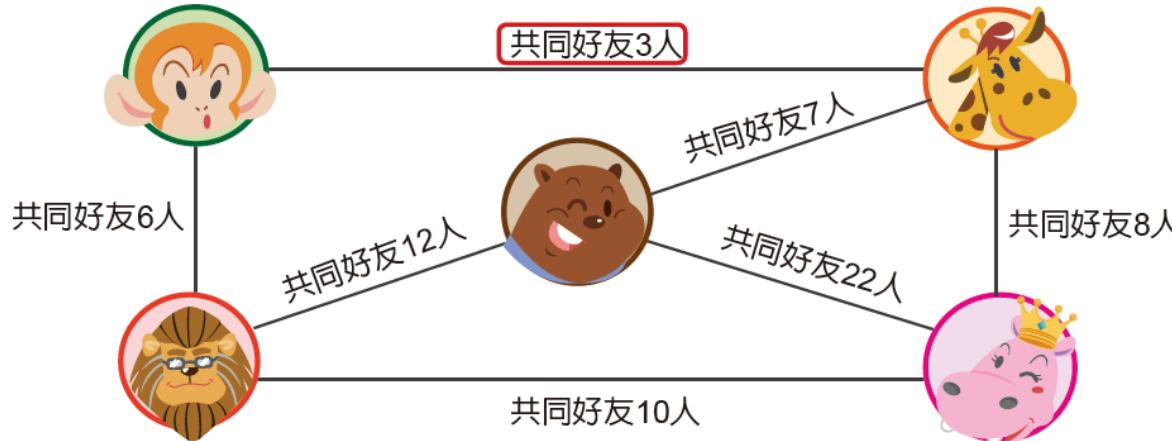
() 7. 右圖是一個二元搜尋樹，請問在樹中搜尋「37」需要進行多少次數值大小的比較（如 $37 > 31$ 便是一次比較）？

(A)1次 (B)3次 (C)5次 (D)7次。



() 8. 下圖是一張描述朋友關係的圖，請問紅色框起的地方為下列何者？

- (A)邊 (B)邊的方向 (C)節點 (D)邊的權重。



實力挑戰

() 9. 下列關於資料結構的說明，何者正確？

- (A)佇列是符合後進先出（LIFO）特性的資料結構
- (B)相鄰矩陣可以用來表達圖上的相鄰關係
- (C)二元搜尋樹每個節點可以有任意數量的子節點
- (D)Instagram的追蹤關係適合使用無向圖來表示。

多元練習

- ◆ 請將下列的數字，建立成一棵二元搜尋樹： $<3, 15, 24, 28, 39, 57, 61>$ 。建立出來的二元搜尋樹可能有許多不同的排列方式，請嘗試讓任意節點的左、右子樹節點數盡量接近。