



2022 國際運算思維挑戰賽

編輯團隊：李忠謀、柯佳伶、林子立
黃子容、葉宜珊、梅宜琇

ebras 際運算思維挑戰賽

ational Challenge
ormatics and Computational Thinking



2022任務解析



Benjamin 組

易	中	難
樹林照片 7	生日派對 23	小美的社區 13
漢堡食譜 37	挑禮物 27	連接海島 15
搭建房屋 87	石頭篩選機 101	草莓 51
蜂巢之旅 99	服飾推薦系統 103	海狸餐廳 71

Cadet 組

易	中	難
生日派對 23	小美的社區 13	排數字遊戲 25
泳池置物櫃 89	連接海島 15	地鐵路線圖 57
教室座位 91	派對後大掃除 47	重疊的村莊 63
石頭篩選機 101	草莓 51	奧赫里德珍珠 81
服飾推薦系統 103	海狸餐廳 71	貪婪的妖怪 - 題組二 85

Junior 組

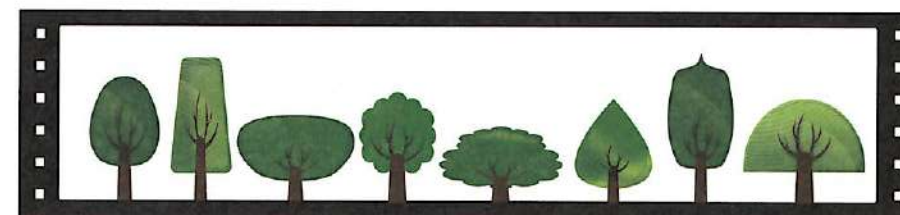
易	中	難
小美的社區 13	海狸水壩 21	飛機棚轉盤 61
滑雪道地圖 19	排數字遊戲 25	重疊的村莊 63
派對後大掃除 47	地鐵路線圖 57	給總統的禮物 65
串列 77	奧赫里德珍珠 81	貪婪的妖怪 - 題組二 85
螺帽和螺柱 111	井字遊戲 105	切線遊戲 95

Senior 組

易	中	難
海狸水壩 21	海狸資料庫 31	迷宮移動 17
拔河 55	海狸運動會 39	生成字串 33
奧赫里德珍珠 81	給總統的禮物 65	石頭搬運 41
電影之夜 93	貪婪的妖怪 - 題組一 84	最喜愛的寶石 67
井字遊戲 101	切線遊戲 95	無人機路徑 - 題組一 107

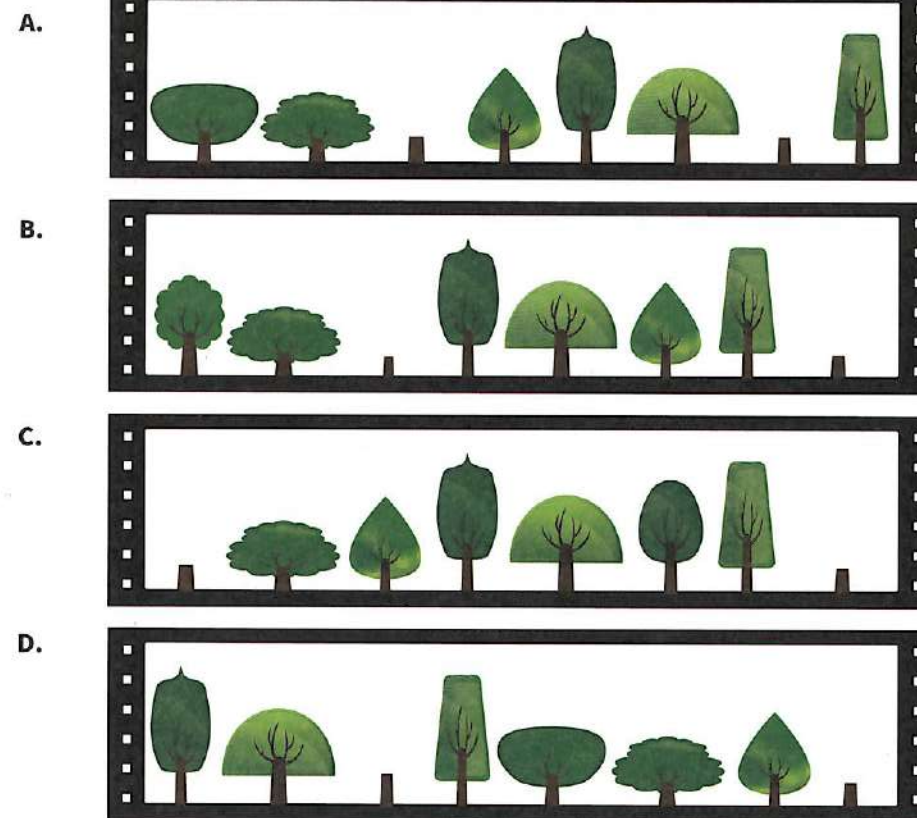
1. 樹林照片

樹林中有八棵圍成圓形的樹木，小海狸站在圓中間，拍了一張 360 度環景照片，如下圖。



幾天後，小海狸回到樹林中的同一地點，他發現有兩棵樹被砍倒了，又拍了一張照片。

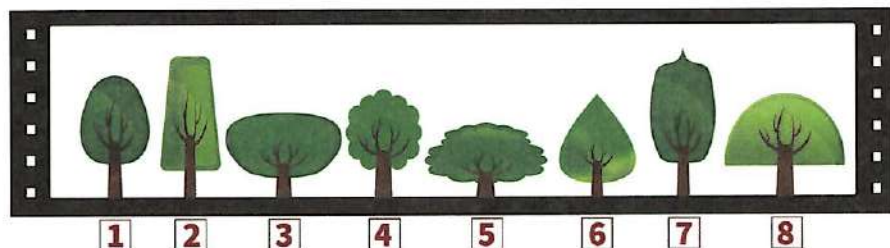
下列哪張相片是小海狸這次拍的呢？





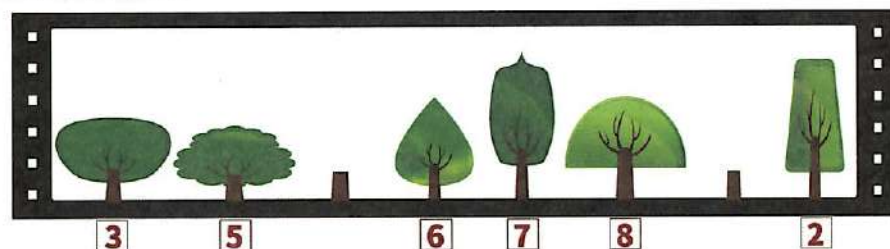
正確答案是：C

讓我們先對第一張照片中的樹木進行編號，會得到以下結果：

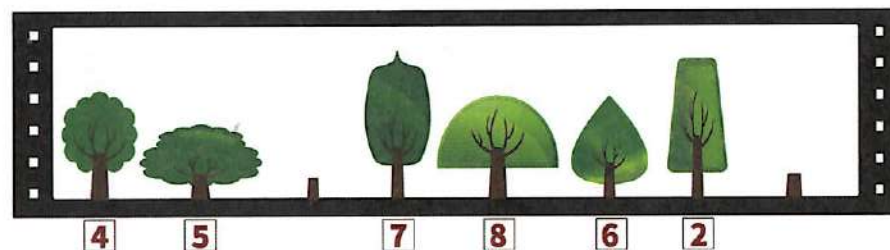


因為每棵樹都長得不一樣，以第一張照片中的樹木來比對選項 (A) 照片，可得到下圖編號結果。可發現 3 號與 5 號樹木中，缺了 4 號樹木，且 5 號與 6 號之間多了 1 棵樹木。

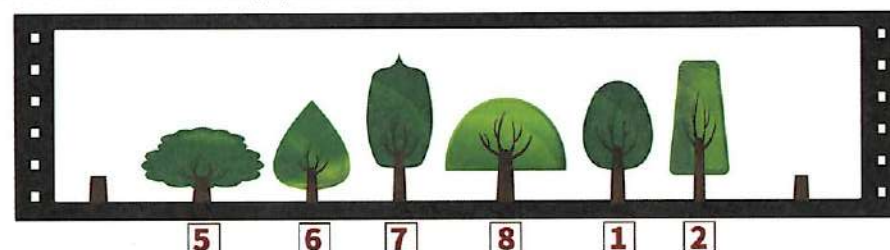
所以選項 (A) 是錯誤的。



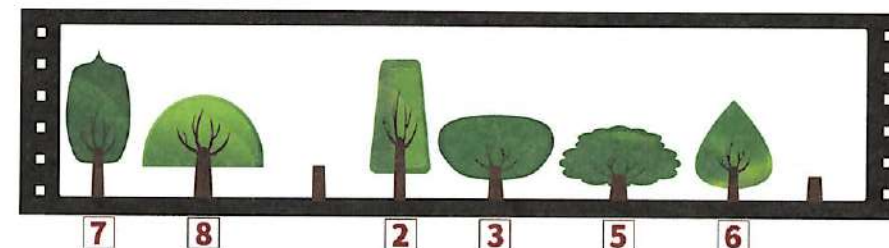
以第一張照片中的樹木來比對選項 (B) 的照片，可得到下圖編號結果。因為樹木不會任意移動，6 號樹木不應該出現在 8 號與 2 號樹木中間，所以選項 (B) 是錯誤的。



以第一張照片中的樹木來比對選項 (C) 的照片，可得到下圖編號結果。2 號與 5 號樹木之間，正好缺了 3 號及 4 號，所以選項 (C) 是正確答案。



以第一張照片中的樹木來比對選項 (D) 的照片，可得到下圖編號結果，可發現 3 號與 5 號樹木中，缺了 4 號樹木，而 6 號與 7 號樹木中，多了一棵樹木。所以選項 (D) 是錯誤的。



資訊科學上的意義

本任務中我們可以由原本的樹林照片取得 8 棵樹木的位置順序關係，而且知道每棵樹的前後樹木順序是不會改變的。例如 1 號樹木的前一棵樹木是 8 號樹木，而 1 號樹木的後一棵應該是 2 號樹木。在這個任務中，答案選項中樹木的編號可由樹林照片比對得到，再透過邏輯推理來比對樹木相對順序是否可能發生，當我們發現樹的順序有問題時，就可以判定是錯誤答案。

在運用電腦解題的過程中會需要用到 **邏輯推理 logical inferenc** 及 **除錯 debugging** 的技巧。邏輯推理是學習資訊科學領域重要的能力，程式中經常會運用條件的邏輯組合，來判斷狀況並決定對應的執行指令。除錯是程式設計師經常做的工作，透過判斷執行結果不符合推斷結果，找出程式中的錯誤指令。當面對此任務，因部分資料不確定（不知道哪棵樹木被砍倒），所以先比對已知資料（比對未砍倒樹木的編號），再用邏輯推論判斷該順序是否可能存在。

在日常生活中，可將解決某項問題的檢查項目整理成清單，再依清單中的每個項目逐一進行檢查。例如：當汽車無法啟動時，會逐一檢查是電池沒電了還是燃料不足，或是有某些零件損壞影響汽車的運作，一一檢查進行故障排除，找出問題讓功能恢復正常，這就是除錯的過程。



關鍵字

模式辨識、邏輯推理、除錯



2. 機器人工廠

海狸市有一家工廠生產可以執行家務的機器人。

機器人由 3 個部位（頭部、身體和腿部）組合而成。

頭部有 3 種造型、身體有 4 種造型、腿部有 3 種造型，依下表順序輪流出現。

頭部	 藍	 綠	 橘	(再次重複)	
身體	 圓形	 方形	 三角形	 五角形	(再次重複)
腿部	 兩條腿	 四條腿	 一條腿	(再次重複)	

工廠生產機器人時，第一個機器人會從每個部位的第一種造型：

頭（）、身體（）和腿（）組合而成。



接下來生產的機器人，頭部和腿部會跳過一種造型，身體會跳過兩種造型。

因此，生產的第二個機器人是長這樣的：



請問按照以上規則，工廠組合出的第 10 個機器人會長什麼樣子？

- A.  B.  C.  D. 



正確答案是：B

依任務規則，頭部與腿部的生產模式每 3 次會重複，而身體的生產模式為每 4 次會重複。所以機器人工廠所生產的不同機器人樣式有 12 種，生產順序如下圖。每生產 12 個機器人後（3 與 4 的最小公倍數），又會回到第一種機器人，依此規則繼續生產。



由上圖可發現第 10 個生產的機器人為選項 B，為正確答案。

選項 A、C 與 D 並未出現在機器人工廠所生產的樣式中，所以為錯誤答案。



資訊科學上的意義

在此任務中，辨識機器人頭部、身體與腿部的生產模式，是我們要用來解題的關鍵。

依任務要求，頭部和腿部會跳過一種造型，身體會跳過兩種造型。可獲得各部位的生產順序如右表。

頭部				(再次重複)
身體				 (再次重複)
腿部				(再次重複)

我們可發現頭部及腿部的造型，每生產 3 次機器人就重新循環 1 次，所以第 10 個生產的機器人，頭部及腿部應是第 1 種造型（、）；而身體部分，每生產 4 次機器人就重新循環 1 次，所以第 10 個生產的機器人，身體部份應該是第 2 種造型（）。

模式辨識 pattern recognition 是指對表徵事物或現象等不同形式的訊息進行處理和分析，描述、辨認、分類並解釋其中的規則性。透過由訊息中辨識出的規律模式，可用來自動辨別並預測接續出現的事物，進而採用適合的方法來解決問題。

在生活中，我們可以觀察到很多模式存在，因而可依模式的規律現象進行預測。例如：月球環繞地球公轉時，地球、月球、太陽的相對位置會規律性地移動，地球上同一位置看到月球被太陽照明的部分會隨之改變，因此月相盈虧圓缺的規律變化可被預測。而海水的水位則受月球及太陽的引力影響，因此可推估出每天漲潮及退潮的時間。澎湖的雙心石滬就是利用海水的潮汐來捕魚，當漲潮時，魚會游進石滬裡覓食，退潮後魚就會受困在裡面，這時漁民就可以趁機捕撈漁獲。



關鍵字

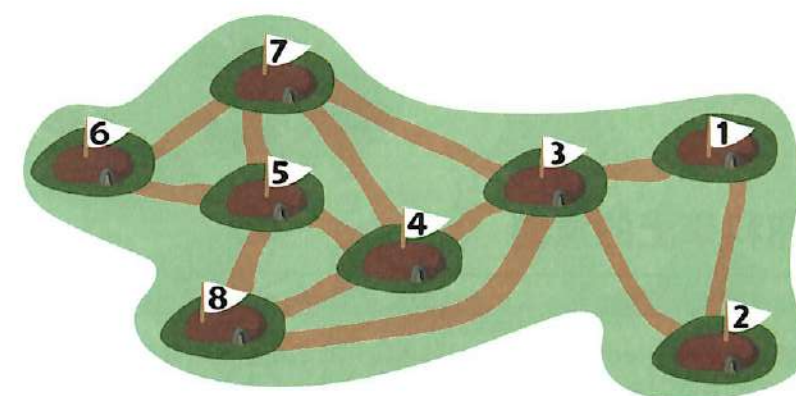
模式辨識、程式追蹤

3. 小美的社區

下面是一張海狸社區地圖，每一間海狸屋 上都有編號，兩屋之間若有一條直接相連的道路 ，彼此就算鄰居。

有一天小海想去拜訪小美，但不知道她住在幾號，只知道下面資訊：

- 小美、小札、小潘各有四個鄰居。
- 小尼的鄰居只有小札及小潘。



根據以上資訊，請問小美住在幾號呢？（範圍為 [1~8] 的整數）

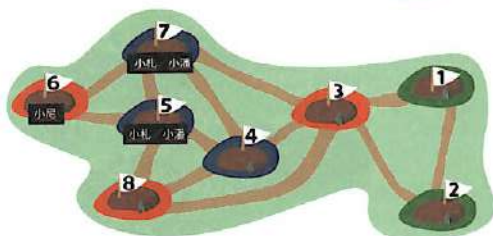
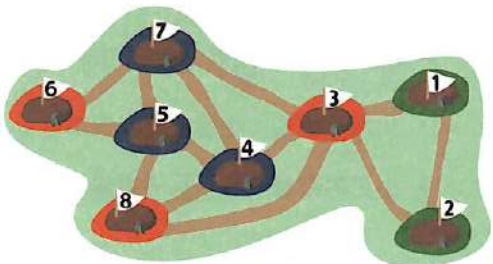
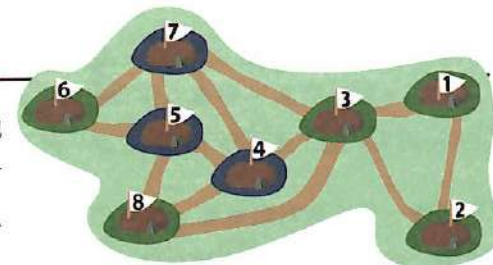


正確答案是：4

根據第一個資訊，我們找出有四個鄰居的海狸屋，也就是連接四條道路的海狸屋。從地圖上可以看出只有4、5、7號屋符合；但此時我們還不能確定哪間是小美、小札、小潘的家。

根據第二個資訊，我們知道小尼是4、5、7號屋中某兩戶的鄰居，所以小尼的有可能住在3、6或8號屋。此外，根據第二個訊息小尼只有兩個鄰居，而3、6、8號屋中只有6號屋連出兩條道路，所以我們可以確定小尼住在6號屋。

確定小尼家在哪後，我們可以推論出5、7號屋分別是小札及小潘家，因此剩下4號屋便是小美住的地方。

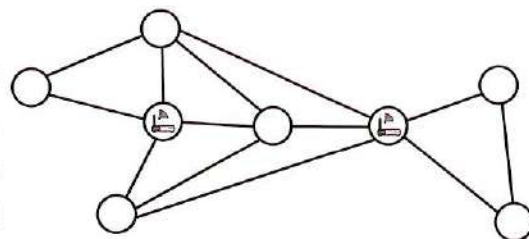


資訊科學上的意義

圖 graph 用於表示物件與物件之間的關係，是圖論的基本研究對象。圖由節點 node（通常以點表示），和一組連接這些節點的邊 edge（通常以線段表示）而組成；若兩個節點之間有邊相連，就表示這兩個節點之間存在某種關聯。在此任務中海狸屋即是節點，屋子間連接的道路即是邊，而鄰居關係即是這些邊所代表的關聯。許多現實生活中的問題以圖表示後，就能透過程式解決。

資訊科學家常運用圖來解決網路訊號相關的問題，例如：在何處架設行動網路基地台、在室內何處擺放無線網路 AP，才能將網路訊號達到最大化利用；還可以利用化簡後的圖論模型找出伺服器流量的瓶頸在哪裡、也可以設計出當某些機器過載的時候資料該如何繞道而行。

在規劃公共設施時，也會運用圖來尋找適當的設施位置，來增進公共設施的使用效率。像是規劃醫院地點時，會希望城市內的每個居民都能在較短時間內抵達醫院；或是規劃學校的區位時，希望能用較少的建設成本來照顧到城市內的所有學生。而在日常生活中，也常看到有圖來表示人事物之間的關聯；例如人與人之間的社交網絡、或是不同地點間的交通網絡。



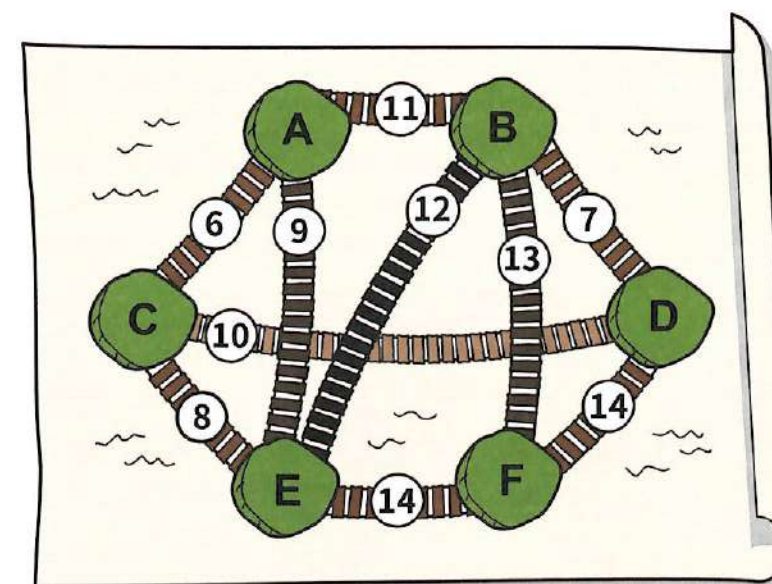
關鍵字

圖論

4. 連接海島

海狸國共有六座島，居民為了方便往來打算建造橋樑連接島嶼。下圖顯示建橋的規劃，每座橋僅連接兩座海島，而橋上的數字表示建造這座橋的推估成本。

由於材料珍貴，海狸國決定用最少的成本連接六座海島，就算要繞路經過別的島嶼才能抵達目的地也沒關係。

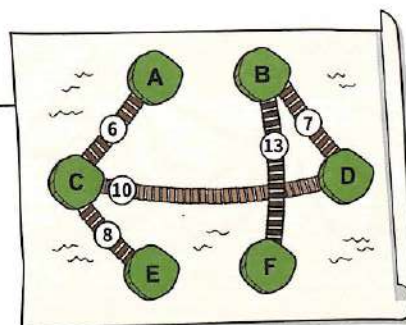


根據這張圖，請問建橋連接六座島的最少總成本是多少？（範圍 [0~100] 的整數。）



正確答案是：44

為了要建橋連接六座島，就算繞路經過別的島嶼才能抵達目的地也沒關係；我們可以從任一座海島開始向外連接，連接的過程中，從可連接到尚未相連的海島之中，逐一選擇成本最少的那座橋來連接海島。其中一種連接方式如下所述：



1. 從 A 島開始。
2. A 島與其他島之間共規劃三座橋，推估成本分別是：6、9、11；選成本最少 (6) 的橋連接 A 島與 C 島。
3. 已相連的 A、C 島與其他島之間共規劃四座橋，推估成本為：8、9、10、11；選成本最少 (8) 的橋來連接 E 島。
4. 已相連的三座島與其他島之間共規劃四座橋，推估成本為：10、11、12、14；選成本最少 (10) 的橋來連接 D 島。
5. 已相連的四座島與其他島之間共規劃五座橋，選擇成本最少 (7) 的橋來連接 B 島。
6. 已相連的四座島與 F 島之間共規劃三座橋，選擇成本最少 (13) 的橋來連接 F 島。
7. 至此所有海島都已相連。

將這五座橋的推估成本加總後，可以得到最少成本 $6+8+10+7+13=44$ 。



資訊科學上的意義

從圖中選擇可以連接所有節點的邊，當這些邊不形成循環時，就會產生樹的結構，我們稱這樣的樹狀結構為**生成樹**。當圖中的邊含有權重，且某個生成樹中所有邊的權重總和是所有生成樹中最小的，該生成樹即稱作**最小生成樹 Minimum Spanning Tree**。這個任務其實就是在給定的圖（橋樑規劃成本圖）中，找到最小生成樹（成本最少的建橋方法）。而題解中找尋最小生成樹的方式運用到**普林演算法 Prim's Algorithm**：

1. 從任一節點開始，將它放入清單當中。
2. 接著按照以下方法找出下一個放入清單的節點，直到所有節點都在清單當中：
 - a. 從清單中的節點與未放到清單的節點，找出連接兩組節點間權重最低的邊。
 - b. 將這個邊連接的節點放入清單中。

此外，Kruskal 演算法也是常備採用的最小生成樹演算法。在日常生活中，人們常常利用最小生成樹演算法來解決電信網路、交通網路或供水路徑建構等等的問題。

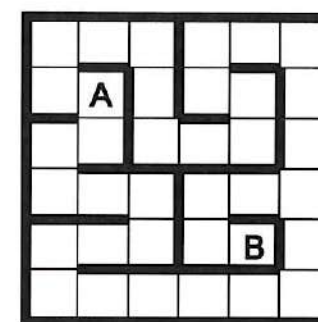


關鍵字

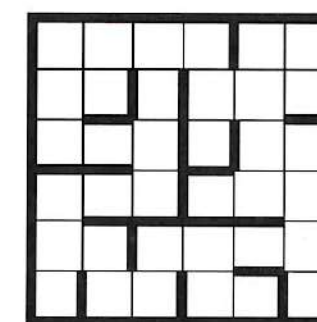
最小生成樹、普林演算法、圖

5. 迷宮移動

小海狸困在迷宮裡了。這個迷宮有兩層，每一層內部都有自己的阻隔牆（黑色粗線）。



第1層



第2層

如果兩個相鄰的格子之間沒有阻隔牆，小海狸可以在兩個相鄰格子之間移動一格，這樣要花 1 秒。

小海狸也可以選擇使用魔法移動到迷宮另一層相同位置的格子，但這樣要花 5 秒。

例如，如果小海狸位在迷宮中位置 A，有三種移動的方式：

1. 花 1 秒在同一層往左移動。
2. 花 1 秒在同一層往下移動。
3. 花 5 秒用魔法移動到迷宮中另一層相同位置。

小海狸想從迷宮中的位置 A 盡快移動到位置 B。

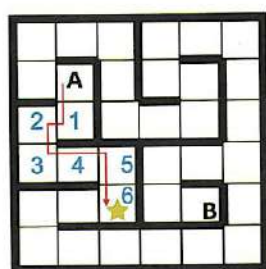
請問小海狸從迷宮中位置 A 出發，到達位置 B 所需要的最短時間是多少秒？（範圍 [1~30] 的整數）



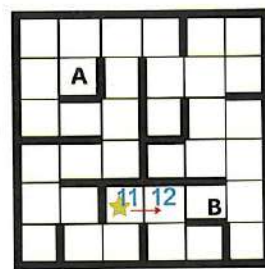
正確答案是：18

我們可以分別計算迷宮同一層移動及使用魔法移動到迷宮另一層所耗費的時間，只要選擇最短所需時間即可。將位置 A 到位置 B 之間在迷宮同一層移動或配合魔法移動所需要的最短秒數，可以產生如右邊的數據圖。

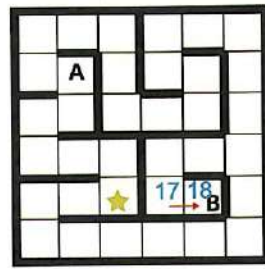
我們可以發現，當我們依據下面紅色的箭頭路徑，走到下圖第 1 層標記★位置（已經走了 6 秒），可以選擇魔法移動到第 2 層（5 秒）、右移一步（1 秒）後再魔法移動回到第 1 層（5 秒），總共花費 17 秒（6+5+1+5=17），再花 1 秒向右即可到 B 點，共花 18 秒。



第1層

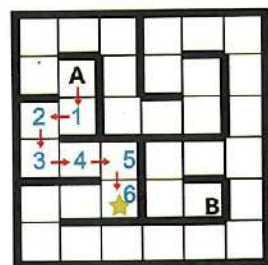


第2層

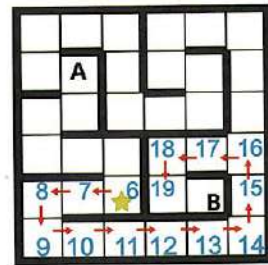


第1層

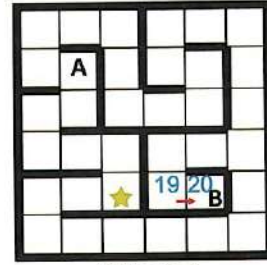
若是選擇不使用魔法移動，在同樣的第 1 層繞過下方的阻隔牆花 13 秒（下方中間圖上紅色箭頭路徑）到同樣位置（6+13=19），再花 1 秒向右即可到 B 點，共花 20 秒。很明顯地，第一個方法魔法移動兩層迷宮的方式比較省時間。



第1層



第1層



第1層

由上面的說明可以發現，如果本任務設定在兩層迷宮間使用魔法移動所需時間如果改為 6 秒（比原來多 1 秒），從 A 點到 B 點使用魔法移動花費的時間，就會與不使用魔法移動一樣都需要 20 秒，因此，如果魔法移動所需要的秒數太多，選擇使用就不見得划算。



資訊科學上的意義

最短路徑問題 **shortest path problem** 是一個很經典的圖論問題，其應用包括在計算機網路及城市地圖中找到最佳路徑。格狀圖 **grid graph** 上的最短路徑問題的實例則有超大型積體電路（VLSI）設計中的佈線。就像在多層 VLSI 電路中，跨層佈線比層內佈線更昂貴，但在同一層佈線也許成本較低卻又會因為繞路而增加傳輸效能的考量。

如同本任務在給定的不同層之間魔法移動需要 5 秒，而在同一層內移動，只需 1 秒。常見的 VLSI 電路放置在 10 層矽晶體上，本題只簡化成 2 層的問題，此外，VLSI 多層的佈線需要考慮的影響因素又更為複雜。



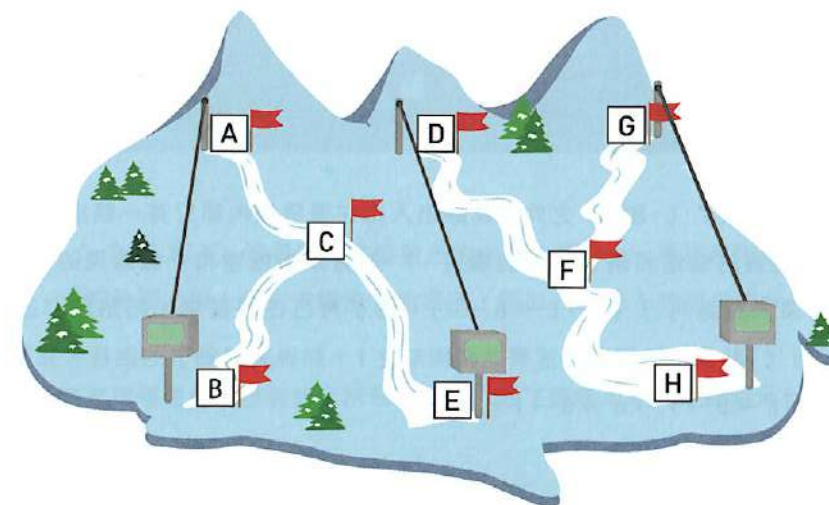
關鍵字

最短路徑問題、格狀圖



6. 滑雪道地圖

下圖是某個滑雪場的滑雪道和纜車路線，其中白色是滑雪道，灰色直線是纜車路線，標記字母 A 到 H 的紅色旗子處皆是滑雪場的出入口。



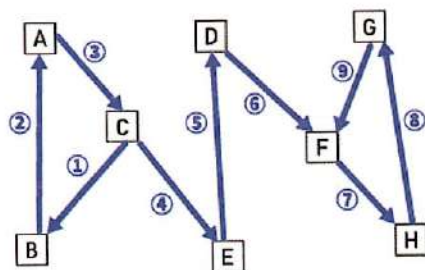
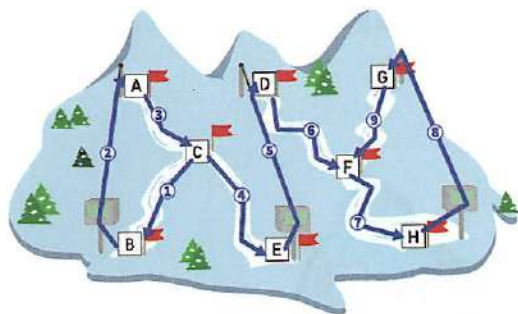
滑雪客若要上山只能搭乘纜車，若要下山只能循滑雪道滑下來，故分別有：A → C, C → B, C → E, D → F, F → H, 以及 G → F 共九條移動路線。

以上九條移動路線，小智想要各走過恰好一次，請問應該要從哪一個出入口開始？
(範圍為英文字母 [A, B, C, D, E, F, G 或 H])



正確答案是：C

先將滑雪場的出入口，以及可以行進的方向標示出來，再將其他不重要的資訊移除：



觀察上圖可以發現，除了 C 和 F 之外，其餘出入口在滑雪場內都只有一條抵達與一條離開的路徑；C 有一條抵達的雪道和兩條離開的雪道，F 則有兩條抵達和一條離開的雪道，而 C 前往 F 只有一條單向路線可行（C → E → D → F）。為了符合任務要求，起始點必須在 C，先經過 B 及 A 回到 C 點後，再經過上述單向路線前往 F，然後完成剩下的路程，完整的順序為：C → B → A → C → E → D → F → H → G → F。



資訊科學上的意義

資訊科學中的圖 graph 是種具有節點和邊的資料結構，此任務中的各個出入口代表節點，滑雪道和纜車路線代表邊，比較特別的是滑雪道只能由上往下行進，滑雪纜車只能由下往上行進。當一個圖的邊有特定的方向，就被稱作有向圖 directed graphs。

在日常中活中，運用有向圖可以清楚呈現以「方向」為主軸的資訊，例如城市單行道地圖、路線規劃圖、公車路線圖…等。在資訊科學中，這一類的圖，從一個節點移動到另一個節點的方向是很重要的；若除去方向，這一類的圖可能會變得毫無意義。



關鍵字

有向圖

7. 海狸水壩



六隻海狸（A, B, C, D, E 和 F）沿著河流建造自己的水壩。然而，一場暴風雨導致部分建造用的木材被沖往下游，幸好每根木材上都刻有建造者的名字，例如：海狸 A 建造用的木材上面都刻著 A。

暴風雨過後，所有海狸帶著自己撿到的木材聚在一起（如下圖所示），準備將木材歸還給建造者。



根據每隻海狸撿到的木材判斷，以下何者可能是他們的水壩位置關係（由河的上游到下游）？

- A. A → B → C → D → E → F
- B. C → B → F → A → D → E
- C. C → F → B → D → A → E
- D. E → C → F → B → A → D



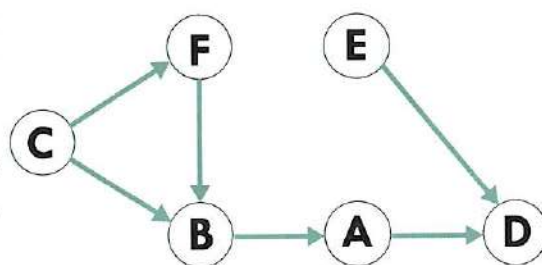
正確答案是：D

正確答案是選項 (D) $E \rightarrow C \rightarrow F \rightarrow B \rightarrow A \rightarrow D$ 。

建造用的木材從上游被沖往下游，因此若海狸甲撿到海狸乙的木材，我們就可以推斷海狸乙的水壩在海狸甲的上游。

為了能夠更輕鬆地判斷水壩間的位置關係，可以用箭頭標示木材漂流的方向（水壩的上下游相對位置），如右圖所示。

可以注意到右圖中沒有任何箭頭指向 C 和 E，換言之，只有 C 和 E 的水壩可能在整條河流最上游的位置。



因此，我們可以知道選項 (A) 是錯的。

選項 (B) 是錯的，圖中有一個箭頭從 F 指向 B，因此 F 的水壩應該在 B 的上游才對。

選項 (C) 是錯的，圖中有一個箭頭從 A 指向 D 以及一個箭頭從 E 指向 D，因此 D 的水壩應該在 A 和 E 的下游才對。

只有選項 (D) 與上圖相符，因此選項 (D) 是正確答案。



資訊科學上的意義

有向無環圖 Directed Acyclic Graph DAG 是由節點與有向邊組成的圖，且圖中不存在有迴路。在此題敘述中，我們可以將每隻海狸的水壩視為一個節點，木材漂流的方向視為有向邊，此問題即為拓撲排序 topological sort 的問題：將節點根據彼此的方向關係排序。

一個有向無環圖可能存在不只一組拓撲順序，其中一種尋找拓撲順序的方式是使用卡恩 (Kahn) 演算法：首先，先選擇一個入分度為 0（沒有邊指向它）的節點 S 加入序列 T 的結尾，並刪除 S 及所有由 S 出發的邊，接下來重複前述的步驟，直到圖中所有節點都加入到序列 T 中，T 即形成一組拓撲順序。

在資訊科學中，拓撲排序常用於處理排程的問題。現實生活中也可以找到許多工作相依的例子，例如選課時，會有必須先修完基礎課程才能修進階課程的限制，拓撲排序可用於找到一組合理的修課順序。



關鍵字

有向圖、拓撲排序

8. 生日派對

小海狸想舉辦一場生日派對，他列出了五項待辦工作，如下表：

待辦工作	確定人數	購買點心	決定日期	預估花費	租借場地
要先完成的工作	10	10	無		

為了規劃工作順序，他在表中還列出做每項待辦工作前要先完成的其他工作。

例如：在做 （確定人數）之前，必須先完成 10（決定日期）。

請問下列哪個工作順序規劃，能夠順利完成生日派對所有的待辦工作？

- A. → → 10 → →
- B. → → → → 10
- C. 10 → → → →
- D. 10 → → → →



正確答案是：C

我們可以從待辦工作的表格中發現，在 「決定日期」之前沒有要先完成的工作，所以第一件要完成的工作就是 ，因此可以排除選項 A 和 B。

完成後，對照表格的第一欄，知道下一件可以做的作品是 「確定人數」；

都完成後，有兩種選擇：

1. 對照表格第二欄，下一件可以做的作品是 「購買點心」；

都完成後，對照表格第五欄，完成 「租借場地」，最後完成 「預估花費」，得到 → → → → 的工作順序。

2. 對照表格第五欄，下一件可以做的作品是 「租借場地」；

都完成後，對照表格第二欄，完成 「購買點心」，最後完成 「預估花費」，得到 → → → → 的工作順序。

此外，因為要先確定人數，才能預估花費，所以選項 D 也是不對的。



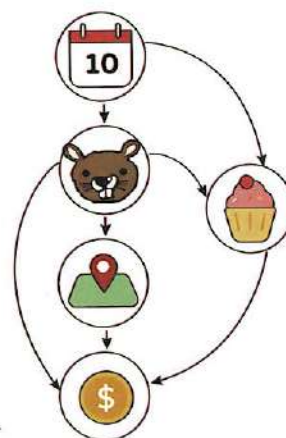
資訊科學上的意義

給定一些工作間的先後條件，要安排一系列工作先後順序的任務稱為 **拓撲排序 topological sort**。

給定一些數字，按照大小排序的順序結果只有一種，但是當需要排序的項目不是數字，或是項目間不是根據其特性數值的大小決定先後順序時，可以運用由點和邊組成的「圖」，用有箭頭方向的邊來表示項目間必須符合的先後關係。而由圖指定的先後關係，符合條件的項目順序有可能找出多組答案。

在本任務中，因為工作項目較少，所以我們可以從表格依序推導出工作順序。當工作增加，或是先後順序關係更多時，我們可以用圖圈表示工作，有箭頭方向的邊表示完成工作間的先後順序，將本任務的表格以右圖表示，再透過電腦用拓撲排序演算法，從「圖」的表示法，有效率地找出符合條件的工作排序。

在我們的日常生活中，拓撲排序可應用於安排一系列的工作或任務。例如：在玩遊戲時，我們必須通過特定的關卡才能玩下一個關卡；在準備上床睡覺前，我們也會有許多任務要先完成，例如要先刷牙、洗澡、整理書包，才能關燈，然後睡覺。



關鍵字

拓撲排序

9. 排數字遊戲

排數字遊戲的玩家須用 1、2、3、4、5、6 排成一個六位數，每個數字只能出現一次，而且排列的順序必須符合關主訂下的 $a \rightarrow b$ 條件；其中 $a \rightarrow b$ 表示數字 a 必須位於數字 b 的左側， a 與 b 不一定要相鄰。例如：若關主設定兩個條件： $2 \rightarrow 1$ 和 $6 \rightarrow 2$ ，那麼 625143 就符合條件，但 123456 不符合條件。

現在關主設定的條件如下：

$4 \rightarrow 1$ 、 $1 \rightarrow 2$ 、 $4 \rightarrow 5$ 、 $2 \rightarrow 3$ 、 $5 \rightarrow 2$ 、 $3 \rightarrow 6$

請寫出符合關主設定的條件中，數值最小的六位數。（範圍 [123456~654321] 的整數）

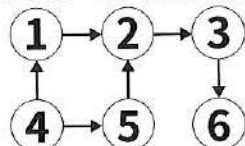


正確答案是：415236

首先逐一按照給定的條件，把可能的數字排列寫出來；如果條件涵蓋許多可能的數字排列方式，就把所有可能性都寫出來。找出符合條件排列的六位數有兩個：451236 及 415236，其中數值最小的是 415236，因此 415236 為正確答案。

規則	依據規則產生的組合	目前可能的組合
4 → 1	4 1	4 1
4 → 1, 1 → 2	4 1 2	4 1 2
4 → 1, 1 → 2, 4 → 5	4 1 2 4 5	4 5 1 2 或 4 1 5 2 或 4 1 2 5
4 → 1, 1 → 2, 4 → 5, 2 → 3	4 1 2 3 4 5	4 5 1 2 3 或 4 1 5 2 3 或 4 1 2 5 3 或 4 1 2 3 5
4 → 1, 1 → 2, 4 → 5, 2 → 3, 5 → 2	4 1 2 3 4 5 2	4 5 1 2 3 或 4 1 5 2 3
4 → 1, 1 → 2, 4 → 5, 2 → 3, 5 → 2, 3 → 6	4 1 2 3 6 4 5 2	4 5 1 2 3 6 或 4 1 5 2 3 6

要找到正確答案，除了逐一寫出符合條件的數字，也可以依照關主給的條件，把數字用 → 串連在一起畫成如右圖：



圖中完全沒有 → 指向它的數字，就該成為數字組合中最左邊的位數。若將上圖拉直成一列，就能清楚明瞭的看出符合關主條件的數字組合從左到右可能的排列順序：



因為沒有任何 → 指向 4，所以 4 是第一個數字；而 4 指向兩個數字 1 和 5，其中 1 比較小，所以第二個數字挑選 1，才會形成較小的數值，故最後答案就是 415236。



資訊科學上的意義

若一個圖中連接節點的邊有方向性，而且圖中沒有形成迴路，就稱作有向無環圖；而 **拓撲排序 topological sort** 就是將這些節點依照邊的方向來排序。在此任務中每一個數字代表圖中的各個節點，而關主的每一項條件代表著圖中連接兩個節點的邊和其方向，因此依據關主給的條件來排序數字的過程就是拓撲排序。現實生活中有許多工作需要依照特定的先後順序來完成，這時候就很適合運用拓撲排序來確保工作執行順利。例如教室佈置時，要先決定風格主題，才能開始製作佈置的道具等其他工作；又如在組合傢俱時，必須先組合某個零件，才能進行其他零件的組合；拓撲排序即可用來找到一個合理的執行順序。而在資訊科學中，拓撲排序常用於處理工作排程的問題。



關鍵字

拓撲排序

10. 挑禮物

海狸爸爸準備了 4 個盒子，裡面各裝了一份禮物，分別是：手環、筆記本、戒指和香水。

如下圖所示，每個盒子上面都貼了一句話，提示裡面裝著什麼禮物，海狸妹妹可以根據這 4 句話選擇一盒禮物。



請問若海狸妹妹想得到戒指，她應該挑選哪一個盒子？

A.



B.



C.



D.





正確答案是：D



由於每個盒子只裝 1 個禮物，故可以根據提示來排除或確認盒中禮物。以下是其中一種思考步驟：

1. 選項 A 和選項 B 的盒子提示相同，故手環與筆記本一定分別放在這兩個盒子中。
2. 根據第 1 點可以推論出選項 D 的盒子裝著戒指。故正確答案是選項 D。
3. 根據第 2 點可以推論出選項 C 的盒子裝著香水。

由於題目沒有其他線索，因此無法確認手環與筆記本確切是在選項 A 或是選項 B 的盒子中。



資訊科學上的意義

邏輯推論是從一系列事證中推斷出新的資訊，邏輯運算 logical operators 則是推理過程中使用的運算式。邏輯運算通常有 真 True 與 假 False 兩種值，稱為 布林值 boolean，常見的邏輯運算符號包括 AND、OR、NOT、XOR。在本任務中，由於每個盒子中都只有一份禮物，因此「裡面手環 或 筆記本」中指的兩個禮物的存在關係是 XOR：兩個禮物中一定是一個在盒子中 (True)，另一個不在盒子中 (False)。透過邏輯運算式逐步推論，就能得出解答。

我們日常玩的數獨、1A2B 猜數字等，都是運用邏輯運算的遊戲：已知某些條件，再使用邏輯運算式去推論出更多的訊息，並從這些訊息中找出正確的解答或可能解決問題的方法。邏輯推理與邏輯運算是電腦科學的基礎，也是資訊科學家透過程式解決問題時，經常用來檢驗程式正確性的方法。

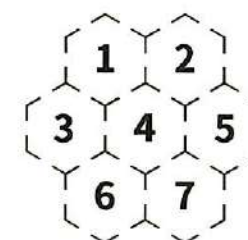


關鍵字

邏輯運算、拆解問題、抽象化

11. 繽紛之塔

山姆的六角形拼圖有 綠、黃、藍 三種顏色，當他將三片排成  的形狀時，這三片拼圖的顏色必須完全相同或完全不同。



舉例來說上圖中的 <1, 3, 4>、<2, 4, 5>、<4, 6, 7> 即需符合上述的規則。

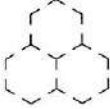
如下圖，山姆將拼圖排成尖塔狀，最底層的顏色安排如圖所示，請問最上面的拼圖  是甚麼顏色？



- A. 
- B. 
- C. 
- D. 不只一種可能



正確答案是：A

從任務描述得知，只要確定  中任兩格顏色，就能推斷第三格是什麼顏色。
從底層最左邊開始：

- 1、2 都是綠色，因此 7 需為綠色
- 2、3 分別是綠色和黃色，因此 8 需為藍色
- 3、4 分別是黃色和藍色，因此 9 需為綠色
- 4、5 分別是藍色和綠色，因此 10 需為黃色
- 5、6 分別是綠色和藍色，因此 11 需為黃色

如此便能拼完第二層。依此類推，依序完成第三層、第四層…後，就可以知道所有拼圖的顏色，如下圖所示，因此最上面拼圖是黃色。



資訊科學上的意義

演算法 algorithm 是完成任務的一連串指令或一組規則，而這些指令或規則可以用多種方式表示，包括自然語言、虛擬碼、流程圖、程式語言等。由於演算法是一連串明確步驟，因此指令的順序很重要；相同的指令依照不同的順序執行後，可能會形成不一樣的結果。

演算法中使用三種基本控制結構來描述指令：循序、條件敘述 conditional statement（選擇結構）和迴圈 loop（重複結構）。此任務逐一完成拼圖的過程中，運用條件敘述來選擇擺放的拼圖顏色，並且運用迴圈來重複擺放拼圖，直到拼完所有拼圖為止。

在日常生活中，我們也常常需要制定解決問題的步驟，且重複執行直到問題解決或任務完成為止，這樣的例子有很多：食譜就是流通最廣的演算法，烹飪時按照食譜的步驟可完成菜餚。



關鍵字

演算法、條件敘述、迴圈

12. 海狸資料庫



海狸村有若干家庭居住其中，身為資訊科學家的海狸小傑幫忙建置了一套村民資料庫，所有村民的資料依照以下規則轉換為 16 位元的序列，由左至右分別為第 15 位元～第 0 位元：

$b_{15} b_{14} b_{13} b_{12} b_{11} b_{10} b_9 b_8 b_7 b_6 b_5 b_4 b_3 b_2 b_1 b_0$

$b_{15} \sim b_{12}$	此四個位元用於表示家庭的編號
b_{11}	此位元用於表示性別 (0 = 女性, 1 = 男性)
$b_{10} \sim b_4$	此七個位元用於表示重量 (為一自然數，其單位為磅)
$b_3 \sim b_2$	此二個位元用於表示工作 (00 = 小木屋建造, 01 = 水壩建造, 10 = 糧食儲藏, 11 = 海狸教育訓練)
$b_1 \sim b_0$	此二個位元用於表示最喜愛的食物 (00 = 樹皮, 01 = 水生植物, 10 = 青草, 11 = 莎草)

舉例來說，序列 0100 0 0100101 10 01 代表一隻來自家庭 4 的海狸，這隻海狸是女性，重量為 37 磅，工作為儲藏村裡的糧食，最喜歡的食物則是水生植物。

海狸小傑使用布林運算式 (0 = 假, 1 = 真) 來查詢資料庫的資料，以下運算式可以找到哪一些村民的資料？

$$b_{11} \text{ 且 非 } (b_{10}) \text{ 且 } b_9 \text{ 且 } b_7 \text{ 且 非 } (b_3 \text{ 且 } b_2)$$

- 女性且重量至少 16 磅，工作為儲藏村裡的糧食。
- 男性且重量至少 64 磅，工作為建造小木屋或水壩。
- 男性且重量至少 40 但不高於 63 磅，工作為建造東西或儲藏糧食。
- 男性且重量至多 39 磅，工作為建造水壩。



正確答案是：C

查詢的運算式中 $b_{11}=1$ 代表男性， $b_{10}=0$ 代表重量至多 63 磅， $b_9=1$ 且 $b_7=1$ 代表重量至少 $(32+8)$ 磅，最後 $b_3=0$ 或 $b_2=0$ 則只能推斷工作不是海狸教育訓練 ($b_3=1$ 且 $b_2=1$)，根據以上資訊推斷選項 C 是正確的。

要特別注意，根據 **笛摩根定律 De Morgan's laws**，非 (b_3 且 b_2) 等價於非 (b_3) 或非 (b_2)。

由於 $b_{11}=1$ 代表男性，選項 A 是錯的，選項 A 的運算式應為非 (b_{11}) 且 b_9 且 b_3 且 非 (b_2)。

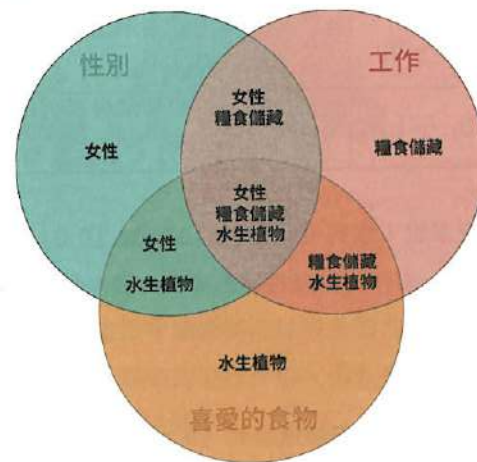
由於 $b_2=0$ 也可以代表工作是糧食儲藏，選項 B 是錯的，選項 B 的運算式應為 b_{11} 且 b_{10} 且 非 (b_3)。

由於 $b_3=0$ 也可以代表工作是小木屋建造，選項 D 是錯的，選項 D 的運算式應為 b_{11} 且 非 (b_{10}) 且 非 (b_3) 且 b_2 。



資訊科學上的意義

如本任務敘述示範，布林運算式 **Boolean operators** 可用於簡易的資料庫查詢。此外，布林運算式可用於標示資料中的子集合，這些子集合間的關係亦可用文氏圖 **Venn diagrams** 呈現，如右圖。



多數現代的程式語言可對資料進行條件檢查，並以一元布林運算子 非 (not)、二元布林運算子 且 (and) 及或 (or) 進行邏輯運算。布林運算式還可用在電子電路中，使用基本的邏輯閘，例如：且閘 (and gate)、或閘 (or gate)、及非閘 (not gate) 來控制電路，組成電腦計算所需的基本元件，例如：加法器、乘法器、邏輯運算單元等。

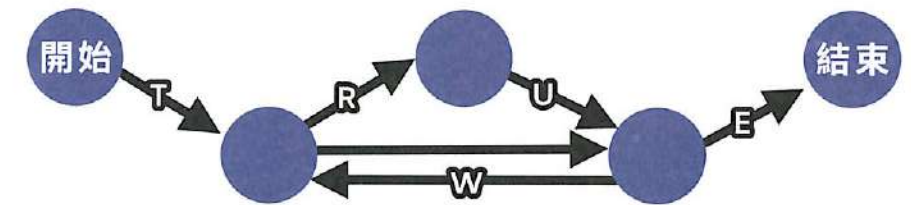


關鍵字

布林運算、笛摩根定律

13. 生成字串

貝姬想利用下面的路徑圖產生隨機的英文字串。她從「開始」的圓圈，依箭頭方向走到下一個圓圈，直到走到「結束」的圓圈為止，並把路徑上的英文字母（也可能沒有字母）依序記錄成為一個字串。



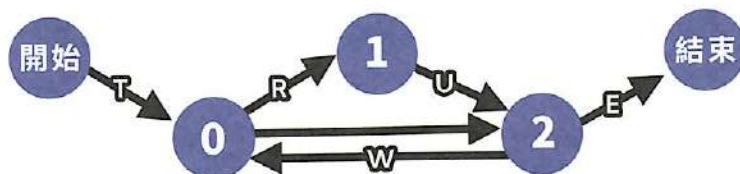
例如：她可以產出字串 “TWWWE” 或 “TRUE”。

請問貝姬可以產出幾個由 8 個字母組成的相異字串？（範圍 [1~20] 的整數）



正確答案是：9

首先，她永遠都必須使用字母 T 開頭，而以字母 E 結尾。所以我們要搜尋的目標就是由字母 R、U 及 W 組合成長度為 6 個字母的字串。



讓我們將此問題轉化成比較小的問題：給定指定長度，以上圖節點 0 為起點，節點 2 為終點，可以組合出多少個符合長度的字串。

長度 1：1 種

1. W $0 \rightarrow 2 \rightarrow 0 \rightarrow 2$

長度 2：2 種

1. RU $0 \rightarrow 1 \rightarrow 2$
 2. WW $0 \rightarrow 2 \rightarrow 0 \rightarrow 2 \rightarrow 0 \rightarrow 2$

長度 3：3 種

1. RUW $0 \rightarrow 1 \rightarrow 2 \rightarrow 0 \rightarrow 2$
 2. WRU $0 \rightarrow 2 \rightarrow 0 \rightarrow 1 \rightarrow 2$
 3. WWW $0 \rightarrow 2 \rightarrow 0 \rightarrow 2 \rightarrow 0 \rightarrow 2 \rightarrow 0 \rightarrow 2$

長度 4：4 種

1. RUWW $0 \rightarrow 1 \rightarrow 2 \rightarrow 0 \rightarrow 2 \rightarrow 0 \rightarrow 2$
 2. WRUW $0 \rightarrow 2 \rightarrow 0 \rightarrow 1 \rightarrow 2 \rightarrow 0 \rightarrow 2$
 3. WWRU $0 \rightarrow 2 \rightarrow 0 \rightarrow 2 \rightarrow 0 \rightarrow 1 \rightarrow 2$
 4. WWWW $0 \rightarrow 2 \rightarrow 0 \rightarrow 2 \rightarrow 0 \rightarrow 2 \rightarrow 0 \rightarrow 2 \rightarrow 0 \rightarrow 2$

長度 5：6 種

1. RUWRU $0 \rightarrow 1 \rightarrow 2 \rightarrow 0 \rightarrow 1 \rightarrow 2$
 2. RUWWW $0 \rightarrow 1 \rightarrow 2 \rightarrow 0 \rightarrow 2 \rightarrow 0 \rightarrow 2 \rightarrow 0 \rightarrow 2$
 3. WRUWW $0 \rightarrow 2 \rightarrow 0 \rightarrow 1 \rightarrow 2 \rightarrow 0 \rightarrow 2 \rightarrow 0 \rightarrow 2$
 4. WWRUW $0 \rightarrow 2 \rightarrow 0 \rightarrow 2 \rightarrow 0 \rightarrow 1 \rightarrow 2 \rightarrow 0 \rightarrow 2$
 5. WWWRU $0 \rightarrow 2 \rightarrow 0 \rightarrow 2 \rightarrow 0 \rightarrow 2 \rightarrow 0 \rightarrow 1 \rightarrow 2$
 6. WWWW $0 \rightarrow 2 \rightarrow 0 \rightarrow 2 \rightarrow 0 \rightarrow 2 \rightarrow 0 \rightarrow 2 \rightarrow 0 \rightarrow 2 \rightarrow 0 \rightarrow 2$

長度 6：9 種

1. RUWRUW $0 \rightarrow 1 \rightarrow 2 \rightarrow 0 \rightarrow 1 \rightarrow 2 \rightarrow 0 \rightarrow 2$
 2. RUWWRU $0 \rightarrow 1 \rightarrow 2 \rightarrow 0 \rightarrow 2 \rightarrow 0 \rightarrow 1 \rightarrow 2$
 3. RUWWWW $0 \rightarrow 1 \rightarrow 2 \rightarrow 0 \rightarrow 2 \rightarrow 0 \rightarrow 2 \rightarrow 0 \rightarrow 2 \rightarrow 0 \rightarrow 2$
 4. WRUWRU $0 \rightarrow 2 \rightarrow 0 \rightarrow 1 \rightarrow 2 \rightarrow 0 \rightarrow 1 \rightarrow 2$
 5. WRUWW $0 \rightarrow 2 \rightarrow 0 \rightarrow 1 \rightarrow 2 \rightarrow 0 \rightarrow 2 \rightarrow 0 \rightarrow 2 \rightarrow 0 \rightarrow 2$
 6. WWRUWW $0 \rightarrow 2 \rightarrow 0 \rightarrow 2 \rightarrow 0 \rightarrow 1 \rightarrow 2 \rightarrow 0 \rightarrow 2 \rightarrow 0 \rightarrow 2$
 7. WWRUW $0 \rightarrow 2 \rightarrow 0 \rightarrow 2 \rightarrow 0 \rightarrow 2 \rightarrow 0 \rightarrow 1 \rightarrow 2 \rightarrow 0 \rightarrow 2$
 8. WWWWRU $0 \rightarrow 2 \rightarrow 0 \rightarrow 2 \rightarrow 0 \rightarrow 2 \rightarrow 0 \rightarrow 2 \rightarrow 0 \rightarrow 1 \rightarrow 2$
 9. WWWW $0 \rightarrow 2 \rightarrow 0 \rightarrow 2 \rightarrow 0 \rightarrow 2 \rightarrow 0 \rightarrow 2 \rightarrow 0 \rightarrow 2 \rightarrow 0 \rightarrow 2 \rightarrow 0 \rightarrow 2$



資訊科學上的意義

在此任務中，我們嘗試枚舉 enumeration 所有字串，並排除出現過的重複字串。

在大多數的情況下，人工進行窮舉搜尋是耗時的，但是電腦很擅長將所有可能的結果都列出來，逐一檢查它們的值，然後從中選擇符合我們需要的結果。這個方法被稱作暴力搜尋法 brute-force search 或窮舉法 exhaustive search。

此任務中的路徑圖就是依照限制條件產生字串的過程。此種方式可用於有系統的編列出識別碼，例如學號或是汽車車牌號碼等，以產生遵循指定格式但獨特不重複之符號序列。



關鍵字

窮舉法



14. 漢堡食譜

海狸潔西卡正按照食譜製作漢堡。

漢堡食譜

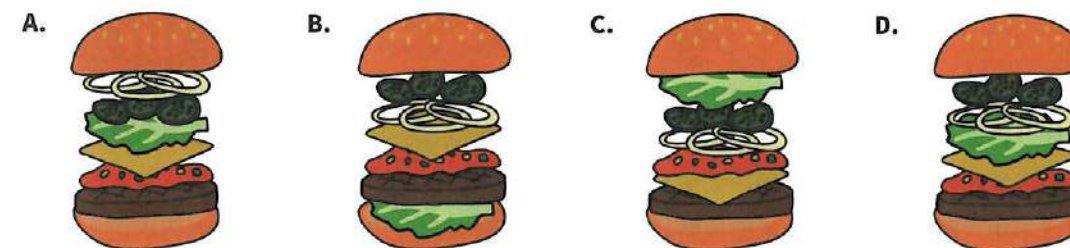
• 食材：

漢堡麵包	肉	醬料	酸黃瓜	萵苣	洋蔥	起司
						

• 製作規則：

1. 醬料要淋在肉上。
2. 肉和起司都要放在酸黃瓜、萵苣和洋蔥的下面。
3. 洋蔥不能接觸到麵包。
4. 所有食材都要夾在麵包中間。

下列哪個漢堡有按照以上規則製作？





正確答案是：D

選項 A 的漢堡符合製作規則 1 和規則 2。但洋蔥接觸到了上面的漢堡麵包，所以違反規則 3。

選項 B 的漢堡符合製作規則 1，但肉和起司放在萵苣的上面，違反規則 2。

選項 C 的漢堡，起司隔在肉和醬料之間，違反規則 1。

選項 D 的漢堡符合所有規則，所以是正確答案。

順帶一提，選項中所有的漢堡都符合規則 4。



資訊科學上的意義

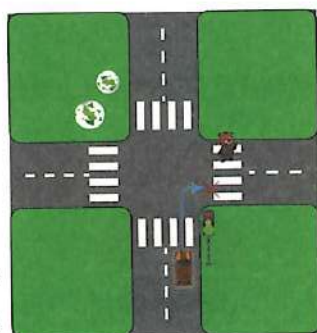
在資訊科學中，這類型的問題被稱為 **限制滿足問題 Constraint Satisfaction Problems CSP**。為了解決這類的問題，如果有多個變數（如本題中的多種食材），我們會選定一個具有最多條件限制的變數，稱為最大限制變數（如本題中的「肉」在製作漢堡時就有多項的條件限制），列舉此變數可能的值，然後再陸續找出限制下的其他變數值。而問題的答案就是滿足指定限制條件的所有變數值組合。

所謂的「限制」指的是必須遵守的規則，而「滿足」指的是符合所有規則。在這個漢堡食譜中提到的製作規則就是上述「限制」，而你必須確認製作的漢堡是否符合製作規則，即為檢查是否「滿足」。常見的填字遊戲、數獨及其他一些邏輯益智遊戲，都是一種限制滿足問題，要找出符合遊戲規則的一組答案。

在以電腦進行自動控制時，通常會制定一些規則避免出錯，檢查某個控制決定是否符合規則的過程稱作「限制檢查 constraints checking」。

例如右圖中的自動駕駛車在右轉前必須先檢查是否符合兩個規則：

1. 紅綠燈必須是亮綠燈。
2. 右前方不能有正在過馬路的行人。



確定一組可能答案是否符合規則是一件事，但是要列出各種可能答案進行檢查又是另一回事了（這就是所謂的 限制滿足問題）。絕大部分的現實問題，若問題中的變數太多，要一一列出各種可能的答案是很花時間的，在有限時間下可能找不完。例如在地圖上的兩個地點間可能有成千上萬條路徑可以選擇，但我們可以加上尋找最短路徑的目標，如 GPS 導航系統採用良好的路徑規劃演算法，就可有效排除許多繞遠路的路徑，找出最短路徑。



關鍵字

限制滿足問題



Benjamin/

難

Cadet/

難

Junior/

難

Senior/

中

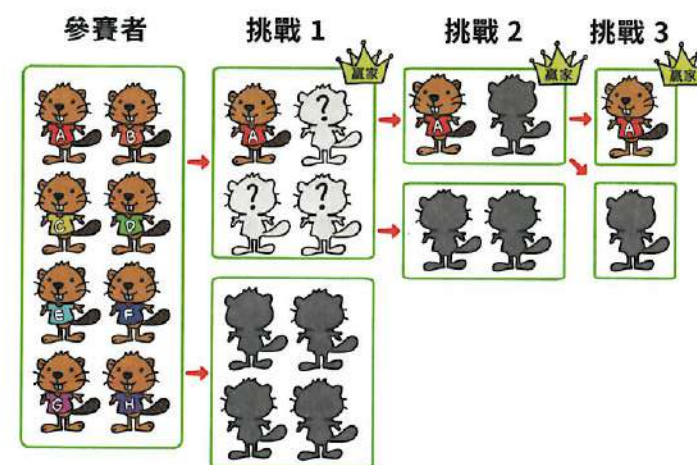
15. 海狸運動會

海狸運動會是一項由挑戰 1 → 挑戰 2 → 挑戰 3 依序組成的年度比賽，挑戰 1 和挑戰 2 以團隊形式進行，團隊成員由抽籤決定，得分總和較高的團隊獲勝並晉級下個挑戰，挑戰 3 是個人比賽。如果連續贏得三個挑戰就是贏家，海狸 A  是今年海狸運動會的幸運贏家，

下表列出了三個挑戰每個人的分數：

	A	B	C	D	E	F	G	H
參賽者								
挑戰 1	15	16	19	18	17	20	19	19
挑戰 2	20	27	30	24	28	24	30	30
挑戰 3	10	14	11	15	16	13	9	12

下圖中挑戰 1 的四隻海狸，除了編號 A 外，請依照字母順序寫下另三隻的編號。



請依照字母順序填入 3 個 [B~H] 的大寫英文字母。



正確答案是：DFG

海狸 A 團隊的其他三名成員是海狸 D 、F  和 G 。

挑戰 3 是個人賽，因為海狸 G 是唯一一個得分比海狸 A 較少的成員，他們在挑戰 2 中一定是同隊。他們在挑戰 2 中的總分是 50，必須比另一個兩人團隊的總分高，只有海狸 D 和 F 這兩個參賽者的總分低於 50 分，因此他們在挑戰 1 一定與海狸 A 和 G 為同一個團隊。

我們已經知道海狸 A、D、F 和 G 為同一團隊，所以，剩下的海狸 B、C、E 和 H 為另一團隊。這意味著在挑戰 1 中（海狸 A、D、F、G）的團隊得 72 分，而（海狸 B、C、E、H）得 71 分，所以海狸 A 的團隊獲勝。在挑戰 2 中（海狸 A、G）得 50 分，（海狸 D、F）得 48 分。在挑戰 3 中海狸 A 得 10 分贏過海狸 G 得 9 分，所以海狸 A 是總冠軍。



資訊科學上的意義

許多資訊科學的問題都會給定一些限制條件，嘗試找到一組滿足所有限制條件的解，對於這類問題，通常會使用搜尋法搜尋解答。如果從挑戰 1 開始尋找答案解決這個任務，需嘗試所有 3 名成員的組合，即 $C(7,3)=35$ ，並對每一個組合評估是否符合挑戰的結果，這需要花很多時間。因此，資訊科學家會尋找更有效的方法來解決任務，如答案解釋中所述，從結果往前可快速推導出正確的解答，這就是 **回溯法 backtracking**，用來尋找滿足某些限制條件的解答。在某些情況下，須結合向前搜尋和向後搜尋才能找到解答。

反向搜尋法是許多演算法中使用的技術，用於解決解答空間上的搜尋，例如：數字方塊滑動遊戲、路由器決定封包來源到目的間的路徑。



關鍵字

回溯法、限制滿足問題

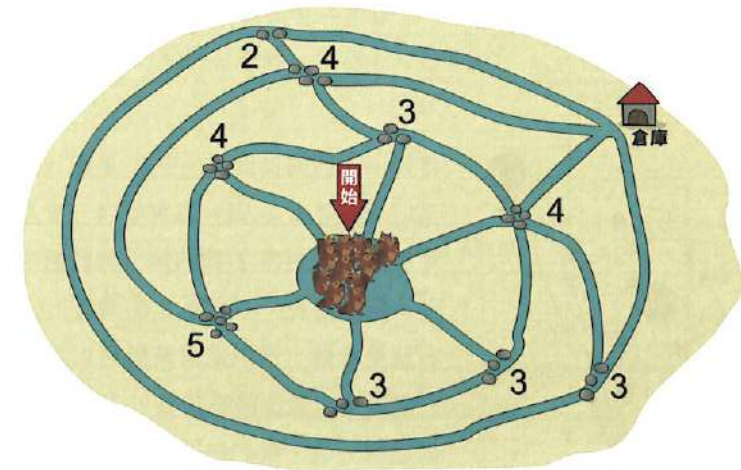
16. 石頭搬運

海狸家族居住地有 21 條河，海狸從河流交叉口游到下一個交叉口，不管河流長短，海狸都能剛好花 1 小時游到。

現在河裡有 31 塊石頭需被搬運到倉庫。

一隻海狸一次只能搬運 1-2 顆石頭，海狸將石頭搬運到倉庫後，可以再回頭搬運其他石頭。

現在，海狸家族同時由中心點出發，必須在 4 小時內完成石頭搬運任務。



請問海狸家族至少需幾隻海狸才能完成任務？

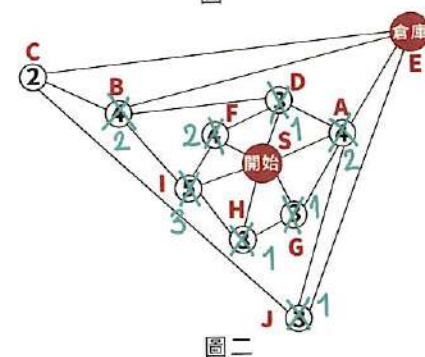
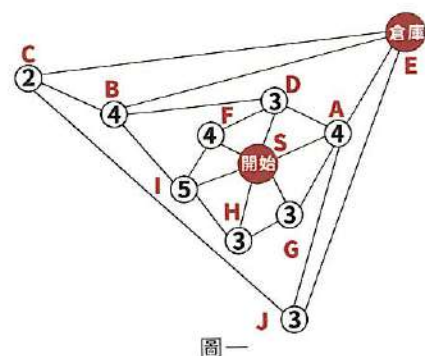
- A. 13 隻海狸
- B. 14 隻海狸
- C. 15 隻海狸
- D. 16 隻海狸



正確答案是：B. 14 隻海狸

我們把海狸家族居住地畫成圖一來說明，所有的海狸一開始都在 S 點，E 點用來表示倉庫，我們用直線代表河流，圖片中的數字代表各個河流交叉口的石頭數量。

我們先計算海狸從 S 點到倉庫 E 點之間必須游過的河流數量，當作起點 S 和倉庫 E 之間的河流路徑長度，我們從圖一發現：

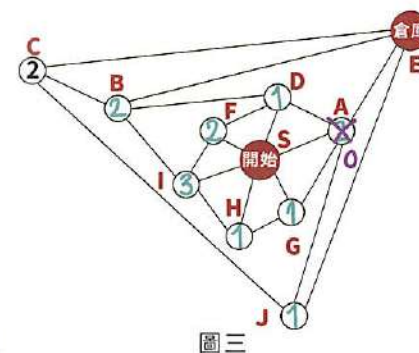


1. 沒有直接連接 S 點到 E 點的河流（即沒有長度為 1 小時的路徑）。
2. 有長度為 2 小時、3 小時和 4 小時的河流路徑。
3. 海狸在 4 小時內必須搬運石頭至倉庫，所以不需要進一步考慮河流路徑長度超過 4 小時者。

首先，我們先找出圖一中有 3 顆（含）以上石頭的交叉口，總共有 8 個。如果各派一隻海狸去搬運，每隻海狸最多可搬運 2 顆石頭，我們把所有海狸搬運的路徑，記錄在表一。編號 8 的海狸，只花了 2 小時搬運石頭至倉庫，所以還有時間可以回來重覆搬運，其他 7 隻海狸，至少都得花 3 小時才能把石頭搬至倉庫。經過這一輪，我們修改圖一為圖二，藍綠色數字即為剩下的石頭。

表一 海狸搬運紀錄

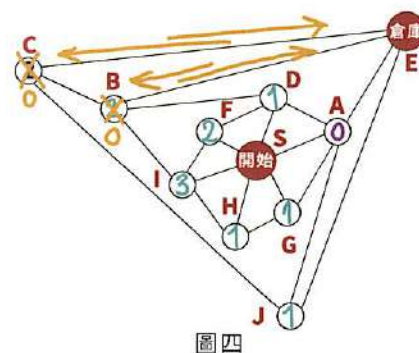
海狸編號	搬運路徑（搬離的石頭數）（剩下的石頭數）	花費時間	已清空石頭的交叉口
1	S → D (2)(1) → B → E	3	
2	S → F (2)(2) → D → B → E	4	
3	S → I (2)(3) → B → E	3	
4	S → H (2)(1) → G → A → E	4	
5	S → G (2)(1) → A → E	3	
6	S → A → J (2)(1) → E	3	
7	S → D → B (2)(2) → E	3	
8	S → A (2)(2) → E	2	



接著再從 S 點派編號 9 海狸，搬運清空圖三 A 點交叉口的石頭，我們標示為 0，我們更新搬運紀錄如表二，編號 9 海狸也只花了 2 小時，故還有時間可以再回頭搬運。

表二 海狸搬運紀錄

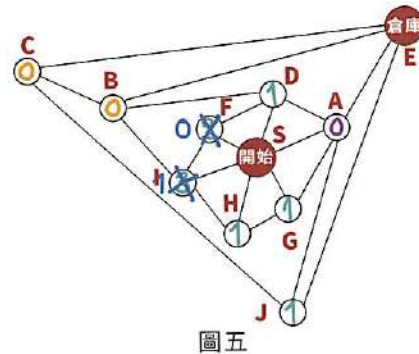
海狸編號	搬運路徑（搬離的石頭數）（剩下的石頭數）	花費時間	已清空石頭的交叉口
1	S → D (2)(1) → B → E	3	
2	S → F (2)(2) → D → B → E	4	
3	S → I (2)(3) → B → E	3	
4	S → H (2)(1) → G → A → E	4	
5	S → G (2)(1) → A → E	3	
6	S → A → J (2)(1) → E	3	
7	S → D → B (2)(2) → E	3	
8	S → A (2)(2) → E	2	
9	S → A (2)(0) → E	2	A



而圖四中的 B 點和 C 點，只要請已經在倉庫 E 點的 2 隻海狸（編號 8 及編號 9），分別前往 B、C 點搬運石頭回倉庫，即可清空 B、C 點的石頭。故目前還是只派出 9 隻海狸，更新海狸搬運紀錄表如表三。

表三 海狸搬運紀錄

海狸編號	搬運路徑 (搬離的石頭數) (剩下的石頭數)	花費時間	已清空石頭的交叉口
1	S → D (2)(1) → B → E	3	
2	S → F (2)(2) → D → B → E	4	
3	S → I (2)(3) → B → E	3	
4	S → H (2)(1) → G → A → E	4	
5	S → G (2)(1) → A → E	3	
6	S → A → J (2)(1) → E	3	
7	S → D → B (2)(2) → E	3	
8	S → A (2)(2) → E → B → E	2+2	B
9	S → A (2)(0) → E → C → E	2+2	A,C

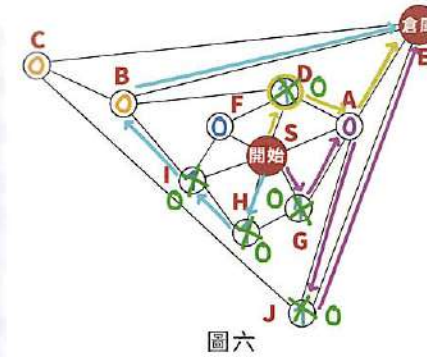


圖五

觀察圖四中 F 點還有 2 顆石頭，I 點還有 3 顆石頭，可從開始 S 點再派出 2 隻海狸搬運，但 I 點經過海狸搬運，仍留下一顆石頭待搬運，目前已派出 9+2=11 隻海狸，剩餘石頭數如圖五，更新海狸搬運紀錄表如表四。

表四 海狸搬運紀錄

海狸編號	搬運路徑 (搬離的石頭數) (剩下的石頭數)	花費時間	已清空石頭的交叉口
1	S → D (2)(1) → B → E	3	
2	S → F (2)(2) → D → B → E	4	
3	S → I (2)(3) → B → E	3	
4	S → H (2)(1) → G → A → E	4	
5	S → G (2)(1) → A → E	3	
6	S → A → J (2)(1) → E	3	
7	S → D → B (2)(2) → E	3	
8	S → A (2)(2) → E → B → E	2+2	B
9	S → A (2)(0) → E → C → E	2+2	A,C
10	S → F (2)(0) → D → B → E	4	F
11	S → I (2)(1) → B → E	3	



圖六

從圖五觀察，只剩五個交叉口 (D、I、J、H、G) 仍有 1 顆石頭需搬運，但只要再派 3 隻海狸就可以完成任務，如圖六所示，分別沿著粉紅色路線 (沿途陸續搬走 2 顆石頭)、黃色路線及藍色路線 (沿途陸續搬走 2 顆石頭)，將石頭搬運至倉庫，所以本題任務至少需要派出 11+3 = 14 隻海狸才能完成任務。最終海狸搬運紀錄表如表五。

表五 海狸搬運紀錄

海狸編號	搬運路徑 (搬離的石頭數) (剩下的石頭數)	花費時間	已清空石頭的交叉口的交叉口
1	S → D (2)(1) → B → E	3	
2	S → F (2)(2) → D → B → E	4	
3	S → I (2)(3) → B → E	3	
4	S → H (2)(1) → G → A → E	4	
5	S → G (2)(1) → A → E	3	
6	S → A → J (2)(1) → E	3	
7	S → D → B (2)(2) → E	3	
8	S → A (2)(2) → E → B → E	2+2	B
9	S → A (2)(0) → E → C → E	2+2	A,C
10	S → F (2)(0) → D → B → E	4	F
11	S → I (2)(1) → B → E	3	
12	S → D (1)(0) → A → E	3	D
13	S → H (1)(0) → I (1)(0) → B → E	4	H,I
14	S → G (1)(0) → A → J (1)(0) → E	4	G,J



資訊科學上的意義

圖 graph 用於表示物件與物件之間的關係，是圖論的基本研究對象。圖由頂點或節點 node (通常以點表示) 和一組連接這些節點的邊 edge (通常以線段表示) 組成。在此題中，海狸的出發點、倉庫，以及河流交叉口的石頭數即是「點」，而每條河流即為「邊」。許多現實生活中的問題以圖表示後，就能透過電腦程式解決。例如，導航系統，就是一個眾所周知的應用實例。



最佳化 optimization 是尋求一個問題最佳解答的過程，也就是在已知的條件限制下，將問題以數學方式表示，並透過運算或搜尋的方法求出最佳解 optimal solution 的步驟。以石頭搬運這一題來說，在數值限制的條件下：海狸從河流交叉口游到下一個交叉口正好 1 小時；海狸必須在 4 小時內完成石頭搬運；河流交叉口的石頭數共 31 顆；一隻海狸一次只能搬運 1-2 顆石頭，要求找尋使用最少的海狸數量來完成搬運任務的最佳解答，就是最佳化的問題。

最佳化在科學、工程、數學界都是非常重要的研究課題。在真實世界中，每一種需要最佳解的問題性質都不同，例如：最短的路徑、最少的花費、利用人工智慧尋找最符合特徵的人臉，或判斷投資的最佳時機等，因此找不到唯一的最佳化方法來對應所有問題。在科學領域也發展出許多演算法來找尋最佳解，有些雖然無法保證能找到最佳解，但是可以幫助我們找到最接近最佳解的結果。

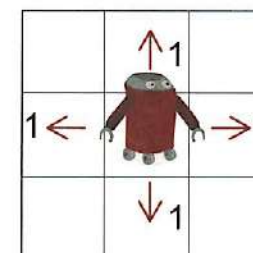


關鍵字

圖、最佳化

17. 派對後大掃除

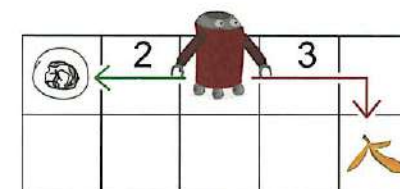
安娜有一台掃地機器人布奇會上下左右移動，每移動一格算一步。



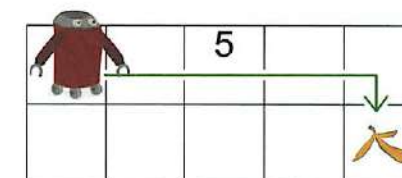
布奇工作時會重複以下步驟，直到所有垃圾撿完為止：

1. 偵測並計算所有垃圾的距離
2. 移動到最接近（步數最少）的垃圾
3. 撿起垃圾

舉例來說，布奇會先移動兩步去撿比較近的紙團（如圖一綠色線），再移動五步去撿香蕉皮（如下圖二）。



圖一

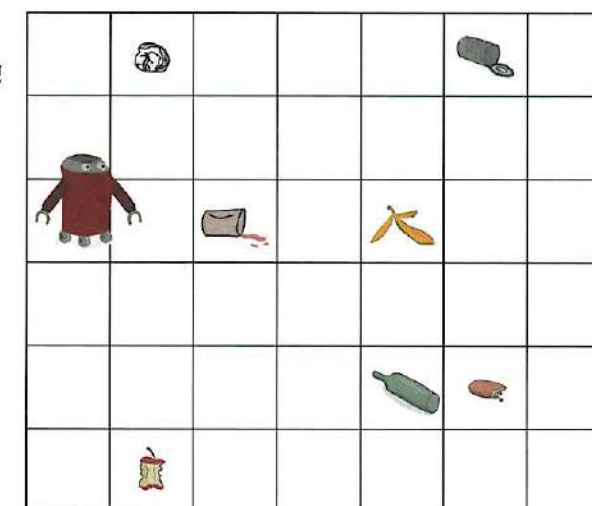


圖二

某次派對結束後房內呈現右圖的狀態。

安娜啟動機器人布奇打掃，請問布奇最後撿起來的垃圾是什麼？

- A.
- B.
- C.
- D.





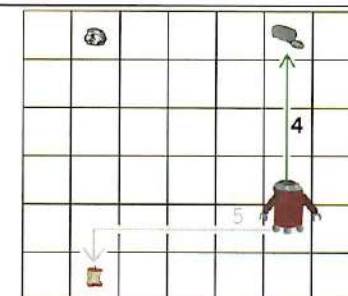
正確答案是：C

布奇在啟動後會偵測與所有垃圾的位置，再選擇前往距離最近的垃圾；每次撿完垃圾後便會重新偵測與垃圾的距離並做選擇。

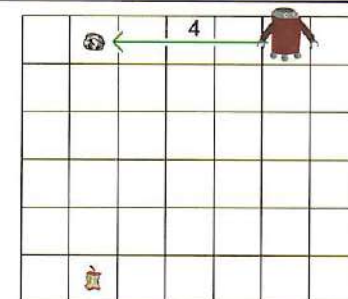
布奇選擇移動 2 步去撿最近的水杯。 此時紙團距離 3 步、蘋果核距離 4 步。	
選擇移動 2 步去撿香蕉皮。 此時紙團距離 3 步、蘋果核及酒瓶距離 4 步。	
選擇移動 2 步撿酒瓶。 此時空罐頭及餅乾距離 3 步。	
選擇移動 1 步撿拾餅乾。 此時空罐頭距離 5 步、蘋果核距離 4 步。	

選擇移動 4 步撿拾空罐頭。

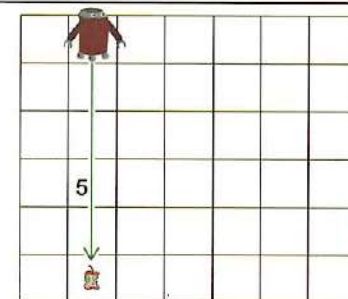
此時蘋果核距離 5 步。



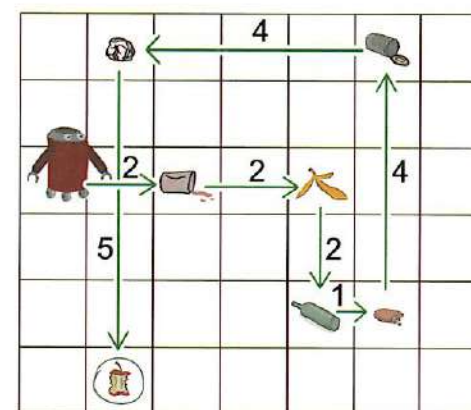
移動 4 步撿拾紙團。



最後移動 5 步撿拾蘋果核。



故布奇撿拾垃圾的順序為：水杯 → 香蕉皮 → 酒瓶 → 餅乾 → 空罐頭 → 紙團 → 蘋果核。如下圖：





資訊科學上的意義

在此任務中，布奇找出自己位置（起點）到垃圾（目標）的方法稱為 **路徑搜尋演算法**。

它的策略是選擇偵測當時距離最短的垃圾先撿拾，無法考慮這個選擇是否為撿完所有垃圾的最短路徑。因此，按照它的工作步驟，撿完所有垃圾所需的總步數可能不是最少的。此種搜尋法稱為 **貪心演算法 greedy algorithm**，它只做出在當前看來是最好的選擇 local optimization，不考慮是否為整體的最佳解 global optimization。雖然如此，對範圍相當廣泛的問題仍能產生整體最優解或者是近似最優解。

路徑搜尋在生活上經常出現，例如我們會使用導航系統規劃到達目的地的路線，因為乘坐交通工具不同，以及設定變數不同（例如避開收費站），而出現多條建議路線。導航系統可能採用最短路徑、最快路徑，或由中間經過地點等限制來搜尋路徑。



關鍵字

路徑搜尋演算法、貪心演算法

18. 草莓

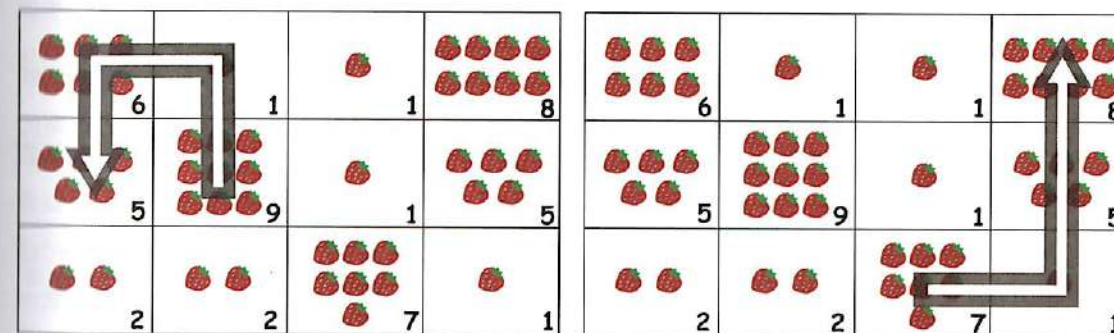
小海狸們很喜歡玩「草莓採集」的遊戲！

這個遊戲是這樣玩的：在場地上的每個格子裡會放入不同數量的草莓，小海狸可以隨意挑選一個格子當成起點，然後移動 3 步。每次要移動的時候可選擇往上、下、左、右的相鄰格子移動 1 格。小海狸可以吃掉移動路徑所經過的 4 個格子裡全部的草莓。

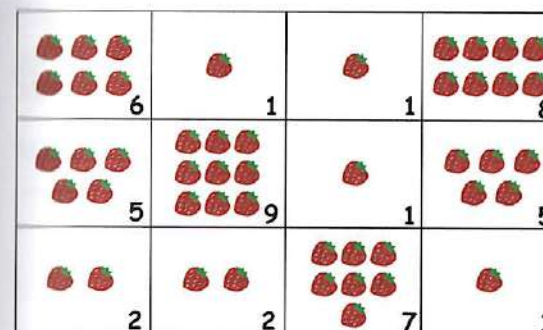
下面有 2 個同一場地上的範例：

選擇左方的路徑，小海狸可以吃到 $9+1+6+5=21$ 顆草莓；

如果選擇右方的路徑，小海狸也可以吃到 $7+1+5+8=21$ 顆草莓。



小海狸這次在下面的場地進行遊戲：



小海狸最多可以吃到幾顆草莓呢？（範圍 [1~48] 的整數。）



正確答案是：23

雖然在給定的場地上總共有 51 種移動方式可以完成遊戲（詳見下圖），但是我們不需要把這 51 種可吃到的草莓數量都一一計算出來。

每隻小海狸會經過 4 個連續且相鄰的格子，這 4 個格子只會形成下表這幾種形狀：

形狀	正方形	直線	垂直 L 形	水平 L 形	水平 Z 形	垂直 Z 形
圖樣						
移動方式	6	3	12	16	8	6

接著只要檢查遊戲場上，有包含 9 顆或 8 顆草莓的格子的形狀就好（同時有 9 與 8 顆草莓的形狀也要）。在檢查 9 或 8 顆草莓的過程中也會涵蓋有 5、6 或 7 顆草莓的格子，因此有比較多草莓的格子都會被檢查到。然後就可以從中找到總量最多的吃法：

	小海狸在路徑上可以吃到的草莓數量		
	包含 9 顆的格子	包含 8 顆的格子	左邊兩種較大者
正方形 	9+5+6+1=21	8+5+1+1=15	21
直線 	5+9+1+5=20	6+1+1+8=16	20
垂直 L 形 	7+2+9+1=19	8+5+1+7=21	21
水平 L 形 	9+1+5+8=23	9+1+5+8=23	23
水平 Z 形 	5+9+2+7=23	8+1+1+9=19	23
垂直 Z 形 	6+5+9+2=22	8+5+1+7=21	22

如果去除 9 或 8 顆草莓的格子不算，任意 4 個其它格子的組合最多草莓總數是 $7+6+5+5=23$ 。也就是說，若不經過 9 和 8 顆草莓的格子，頂多也只能吃到 23 顆（而且還不一定按照遊戲規則）。因此，小海狸在遊戲中能吃到的最多草莓數量是 23 顆。



資訊科學上的意義

在資訊科學領域中，有許多問題的目標都是要找出最大值或最小值，像是執行最多任務，花最少錢，或是這個任務裡要吃到最多草莓。在這些執行任務、消費、吃草莓等的問題裡，過程中每個步驟都有很多選項可以選擇。如何在這些選擇中找到的最大值或最小值，被稱之為 **最佳化 optimization**。

通常有許多方式可以求得最佳解。其中一種方式就是將所有可能的選擇都列出來，一一計算它們的值，然後從中選擇最好的結果。這個方法被稱作 **暴力搜尋法 Brute-force search** 或 **窮舉法 Exhaustive search**。然而在選項數量眾多的情況下，這個方式可能需要耗費巨量的計算時間。因此分析問題並找出可以減少選項數量的約束或限制條件，就可以更快找到最佳解。有許多問題，資訊科學家已經知道如何有效率地找出它們的最佳解，但仍有些問題已知無法被有效率地解決。



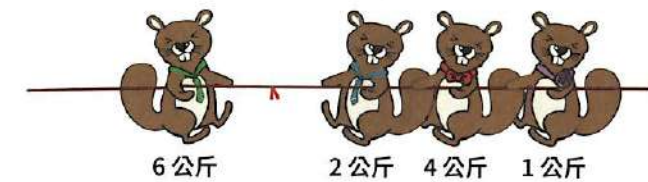
關鍵字

窮舉法、最佳化



19. 拔河

海狸學校正在舉行一場拔河比賽，為了公平競賽，學校將所有參賽的海狸按體重分成兩隊，目標是使每隊的總體重盡可能接近。以下圖為例：這兩隊的總體重相差 1 公斤。



現在老師想要把下圖中的五隻海狸分成兩隊，使每隊的總重量盡可能接近。



這兩隊的體重最少相差多少公斤？（範圍 [0~23] 的整數。）



正確答案是：3

五隻海狸的總體重以 A 來表示為 $1+4+9+4+5=23$ ，理想情況下，兩隊的總體重皆應恰好為 $A/2$ ，即 11.5。然而，每隻海狸的體重是整數，每隊的總體重也應該是整數，在這個限制條件下，一個隊的理想總體重是 11，另一個隊是 12，可以使總重量最接近。但是，五隻海狸中找不到任何組合的總體重為 11 或 12。所以，我們嘗試每隊的總體重分別為 10 (11-1) 和 13 (12-1)。在這種情況下，10 公斤的隊可以是 (1,4,5) 或 (1,9)，另一隊可以是 (4,9) 或 (4,4,5)。因此，兩隊體重相差最小值為 $13-10=3$ 。



資訊科學上的意義

此任務運用資訊科學的 **背包問題 knapsack problem** 概念，它是一種組合 **最佳化 optimization** 的問題。背包問題的目標是選擇不同的物品組合，在有限的總重量限制內使總價值最大化。如果限定每種物品只能選擇 不放入 (0) 或是放入 1 個 (1)，則此問題稱為 0/1 背包問題。

我們可以把這個任務視為 0/1 背包問題：將一個隊伍視為背包，將海狸視為物品，並將海狸的體重視為物品重量，同時也代表這項物品的價值。目標是選擇將某些海狸放入一個限重為所有海狸總體重一半的背包，並使總價值最高。

類似的問題經常出現在生活中，在有限資源下找出一個最佳的解決方案，例如：在限定時間內玩最多遊樂設施，或在有限的預算內旅行最多地方。這類問題的最佳解，可以透過動態規劃的方法得到答案。

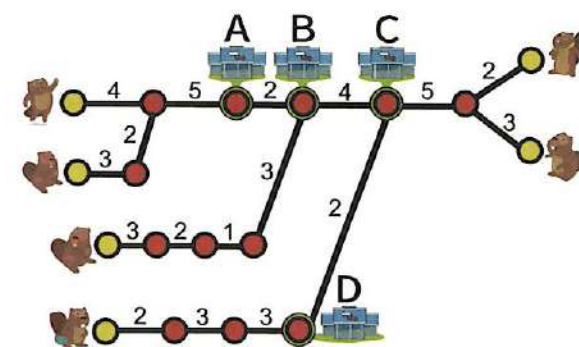


關鍵字

背包問題、最佳化

20. 地鐵路線圖

下圖呈現海狸城的地鐵路線，圖中圓圈表示車站位置，路線上的數字表示車站間的行駛時間；此外，列車於每個車站停留時間皆為一分鐘，足以供所有乘客上下車或轉乘其他路線。



有六隻海狸朋友正同時從不同的車站上車（圖中標示六處 ●），希望能在短時間內到達地鐵的 A、B、C、D 其中一個車站會合。

請問海狸朋友們應該約在哪個車站見面才能最快到齊呢？

- A. A 車站
- B. B 車站
- C. C 車站
- D. D 車站



正確答案是：B. B 車站

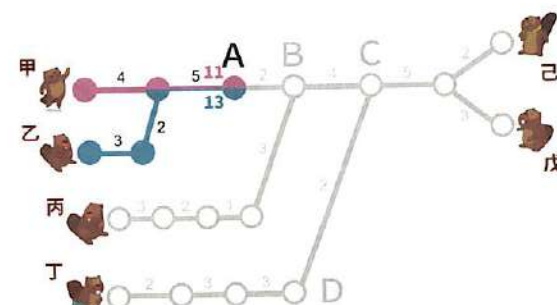
為方便說明，將圖中六隻海狸分別命名為甲、乙、丙、丁、戊和己。

我們可以計算每隻海狸抵達各個集合車站（A、B、C、D）所花費的時間，再找出約在各個車站時，所有海狸都到齊需要多少時間，最後所需時間最少的車站就是應該相約會合的車站，但這個過程非常的繁複。

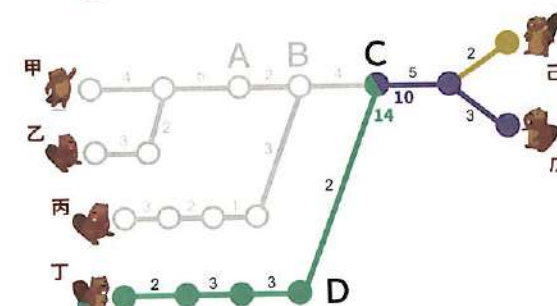
其實只要將這個問題分解成幾個子問題，就能簡化解決問題的過程；其中一種拆解問題的方法，就是先找出海狸們直達最近的集合車站所花費的最長時間，再考慮從最近集合車站到其他集合車站所需的時間，最後取最小的時間總合就是答案了。

拆解完問題後，還可以用子問題的抽象化來找出幾條對最後所需時間有決定性的路線，而不需計算每隻海狸抵達各個車站的時間。而因為海狸是同時上車，故要計算海狸到齊的時間，只要考慮最後抵達的海狸所需時間，也就是旅行時間最長的路線。

以 A 車站為例（如右圖），甲、乙會從 A 車站的左側前來，包含每一站的列車停靠時間，甲到 A 車站需要 11 分鐘（ $1+4+1+5$ ），乙需要 13 分鐘（ $1+3+1+2+1+5$ ）；所以在 A 車站的左側，只需要考量旅行時間較長的乙路線。

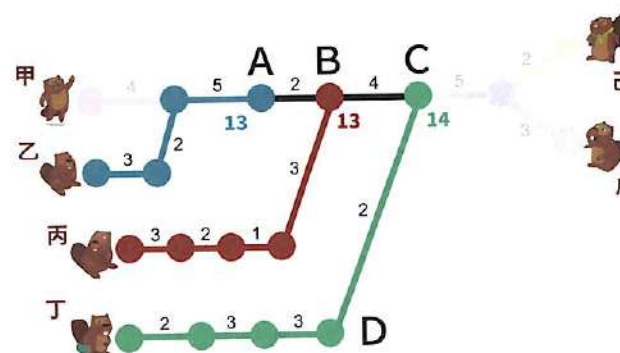


同樣的，右圖的戊、己會在 10 分鐘後於 C 車站相遇（ $1+3+1+5$ ），但丁到 C 車站需要 14 分鐘（ $11+1+2$ ），所以在 C 車站的右側和下方，只要考量丁路線。



經簡化後，右圖顯示各方向的海狸抵達 ABC 各車站時，旅行時間最長的路線及所需時間：

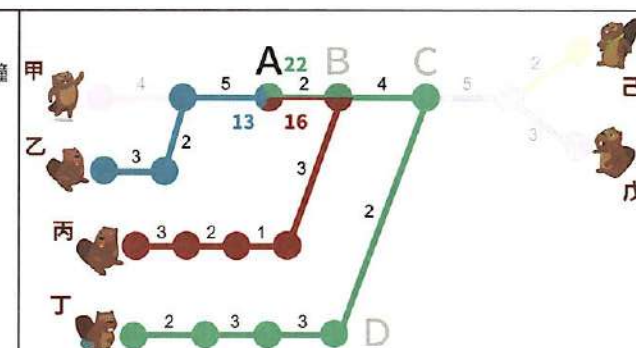
1. 由左側抵達 A 車站，最多需 13 分鐘
2. 由下方抵達 B 車站，需 13 分鐘
3. 由右側或下方抵達 C 車站，需 14 分鐘



若約在 A 車站見面：

1. 由 A 車站左側前來，最多需 13 分鐘
2. 由 B 車站下方前來，需 $13+1+2=16$ 分鐘
3. 由 C 車站右側或下方前來，最多需 $14+1+4+1+2=22$ 分鐘

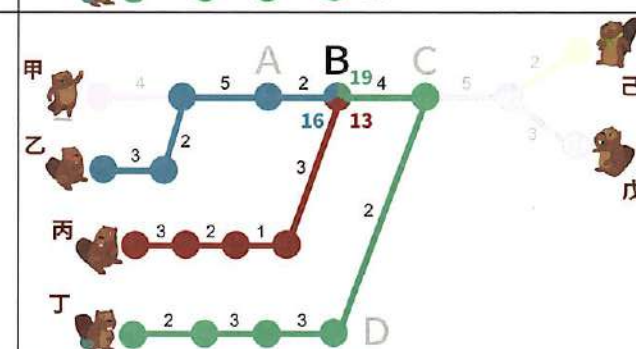
故海狸們最多需要 22 分鐘才會到齊。



若約在 B 車站見面：

1. 由 A 車站左側前來，最多需 $13+1+2=16$ 分鐘
2. 由 B 車站下方前來，需 13 分鐘。
3. 由 C 車站右側或下方前來，最多需 $14+1+4=19$ 分鐘

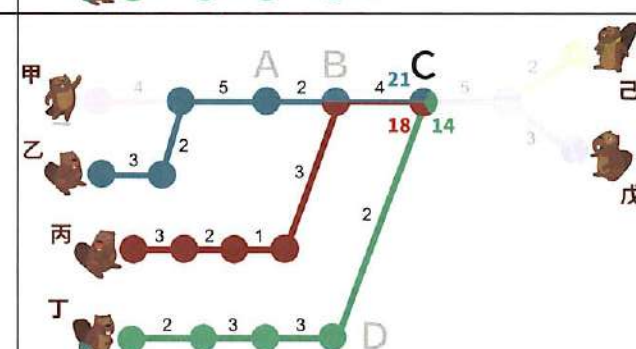
故海狸們最多需要 19 分鐘才會到齊。



若約在 C 車站見面：

1. 由 A 車站左側前來，最多需 $13+1+2+1+4=21$ 分鐘
2. 由 B 車站下方前來，需 $13+1+4=18$ 分鐘
3. 由 C 車站右側或下方前來，最多需 14 分鐘

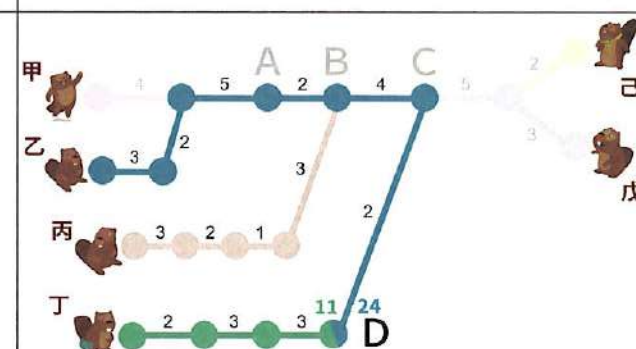
故海狸們最多需要 21 分鐘才會到齊。



若約在 D 車站見面：

1. 由 C 車站方向前來，最多需 $21+1+2=24$ 分鐘
2. 丁前往 D 車站需花 $1+2+1+3+1+3=11$ 分鐘

故海狸們最多需要 24 分鐘才會到齊。



依上表所示，海狸們約在時間 B 車站見面能最快到齊，只需要 19 分鐘（ $19 < 21 < 22 < 24$ ）就能見到彼此。



資訊科學上的意義

動態規劃 dynamic programming (簡稱 DP) 是將複雜問題分解為較簡單的子問題，先分別求解後，再根據各子問題的解來找出原問題的解決辦法。動態規劃每解決一個子問題後，就會將此該子問題的解儲存於陣列中，當該子問題重覆出現時，即可直接查詢陣列取得其解，如此一來就能節省重覆處理同樣子問題的時間。

利用動態規劃來解決此任務，將完整的路線圖拆解成從各方向前往 A、B、C、D 車站的子路線，並利用子路線的抽象化找出應該考量哪些路線的時間計算、哪些路線是可被忽略的，並記錄各子路線應該考量的旅行時間，以供後續查詢。

我們在日常生活中也可以運用到動態規劃來完成複雜的任務，像是規劃旅遊行程時，考慮如何在最短的時間內前往所有想去的景點，或是規劃交通及住宿的搭配以達到最少預算。



關鍵字

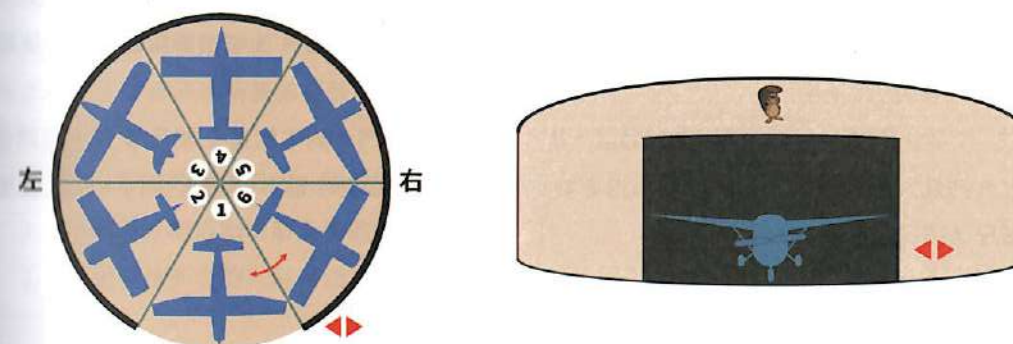
動態規劃



Benjamin/ 難 Cadet/ 難 Junior/ 難 Senior/ 中

21. 飛機棚轉盤

飛機場內有一個停機棚，裡面有六架飛機停放在一個轉盤上。停機棚門口的控制面板上有兩個方向鍵 ◀▶，可以控制裡面的轉盤向左或是向右。每按一次左鍵或右鍵，轉盤就會向左或向右旋轉一個間距，讓一架飛機可以從停機棚大門駛離。由於轉盤轉動的速度非常慢，所以如果可以減少按方向鍵的次數就能節省時間，每次駛離一架飛機都會用最少的轉動方式。



舉例：以目前的狀況要駛離 5 號飛機，可以選擇左二 ◀◀ 或右四 ▶▶▶▶，我們會選擇左二 ◀◀。

每天早上，停機棚大門都是對準了轉盤 1 號位置。轉動最省時的飛機出勤排班順序，只需要按五次控制面板的方向鍵就能讓所有飛機從停機棚大門駛離。例如出勤排班順序 1, 2, 3, 4, 5, 6 位置的飛機依序駛離，只需按五次向右的方向鍵 ▶，或是出勤排班順序 1, 6, 5, 4, 3, 2 位置的飛機，只需按五次向左的方向鍵 ◀。

想一想，怎樣的飛機出勤排班順序，會是最差的狀況，也就是需要按最多次的方向鍵（包含向左或向右）。假設 1 號位置正對停機棚大門，目前飛機停放方式如上圖所示，請問怎樣的排班順序是最差情況？（因為左右對稱，所以答案應該有兩組，請回答排班順序中最後一台飛機編號較大的那一組答案，答案為六個範圍 [1~6] 的整數。）



正確答案是：413625

透過觀察任務的設定，我們可以知道最差的飛機出勤排班順序，就是每次出勤的飛機，當時都位於停機棚裡面離大門最遠的位置，需透過轉盤轉 3 次才將飛機轉到大門。

因為每天早上停機棚大門都是對準了旋轉盤 1 號位置，因此最差的飛機出勤排班順序，是一開始就要把最裡面的 4 號位置飛機旋轉到大門（不論是左轉或右轉都是要按三次方向鍵）。待 4 號位置飛機轉到大門，變成原來 1 號位置的飛機離大門最遠，因此第二架出勤的飛機如果是 1 號，又需要再按三次方向鍵（左轉或右轉同向三次）把 1 號位置的飛機轉到大門。

當 4 號及 1 號飛機都出勤了，大門口是 1 號位置，這時候離大門最遠的飛機就是 3 號或 5 號。因為題目要求最後一台離開大門的飛機編號要較大的，所以我們選擇先將 3 號飛機轉到大門，這時離大門最遠的飛機就是 6 號。

當 4 號、1 號、3 號、6 號飛機都離開轉盤，而 6 號位置在大門，剩下的是 2 號及 5 號。這時候 2 號是離大門最遠的，因此我們應該選擇先把 2 號轉至大門，最後才是 5 號轉至大門。所以最後答案應該是 413625。



資訊科學上的意義

在本任務中，出勤飛機的順序及其在轉盤上的位置，需要移動到大門的轉動次數就是我們要評估的效率。以搭電梯為例，最有效率的方式是一趟電梯向上或向下，能載運最多的人到希望到達的樓層，但實際上人們在不同樓層出現的時間無法事先預定。因此，如何依實際搭電梯的需求，經過計算來安排多個電梯先到哪個樓層，以減少電梯移動需要的時間，就是可以利用資訊科學解決的問題。

在資訊科學中，評估一個工作流程的效率是非常重要的問題。我們通常要考量用某個方法解決一個問題的**最佳情況 best case**及**最差情況 worst case**，以評估該方法的效能，其中考慮的情況可能是需要的時間、能源或存儲空間等資源。

我們以電商平台的倉儲送貨舉例：為了有效率地獲取指定的商品快速出貨，就可能要設計一個演算法安排送貨時間，根據該商品要出貨的指定時段、地點、以及送貨地點間的距離。這些因素都會影響電商平台出貨的效率。



關鍵字

演算法效率、最佳情況、最差情況



22. 重疊的村莊



海狸市有紅菇村、綠草村與蘿蔔村三個村莊，同個村莊的房屋形狀相同。上圖為海狸市的房屋距離圖，圖下方一個刻度代表 1 單位距離。隨著新屋變多，村莊範圍開始重疊。市鎮制定以下步驟來決定新屋該屬於哪個村莊：

1. 找出距離新屋最近的 N 間房屋，計算這些房屋屬於哪個村莊（ N 為整數）。
2. 新屋將分配給出現最多次的村莊。
3. 若出現最多次的村莊不止一個，將新屋分配給「出現最多次村莊中，離新屋最近的那個村莊」。
4. 數字 N 決定後不再變動。

現在海狸市要在下圖 P 點蓋一間新屋（簡稱房屋 P），並分配房屋 P 所屬村莊。



接著在 Q 點蓋一間新屋（簡稱房屋 Q），並分配房屋 Q 所屬村莊（此時需將房屋 P 的村莊納入計算）。



如果想讓房屋 Q 分配至紅菇村，那麼 N 最小應該選擇哪個數字呢？（範圍 [1~8] 的整數。）



正確答案是：5

為了找到能將房屋 Q 分配至紅菇村  的最小 N，可以從 N=1 開始，依序驗證 N 是否能讓房屋 Q 分配至紅菇村 ：

- 假設 N=1，最靠近房屋 P 與房屋 Q 的房屋都屬於蘿蔔村 ，所以房屋 P 與房屋 Q 都會屬於蘿蔔村 。
- 假設 N=2，則房屋 P 屬於蘿蔔村 ；最靠近房屋 Q 的 2 棟房屋都是蘿蔔村  的房屋，故房屋 Q 會屬於蘿蔔村 。
- 假設 N=3，則房屋 P 屬於綠草村 ；最靠近房屋 Q 的 3 棟房屋分別屬於蘿蔔村 、綠草村  與紅菇村 ，其中最靠近的房屋屬於蘿蔔村 ，故房屋 Q 會屬於蘿蔔村 。
- 假設 N=4，最靠近房屋 P 的 4 棟房屋有 2 棟屬於蘿蔔村 ，2 棟屬於綠草村 ，因此房屋 P 與最靠近的房屋一樣屬於蘿蔔村 。而最靠近房屋 Q 的 4 棟房屋有 2 棟屬於蘿蔔村 ，2 棟屬於紅菇村 ，因此房屋 Q 與最靠近的房屋一樣屬於蘿蔔村 。
- 假設 N=5，最靠近房屋 P 的 5 棟房屋有 2 棟屬於蘿蔔村 ，2 棟屬於綠草村 ，1 棟屬於紅菇村 ，因此房屋 P 屬於最靠近的蘿蔔村 。而最靠近房屋 Q 的 5 棟房屋有 2 棟屬於蘿蔔村 ，3 棟屬於紅菇村 ，因此房屋 Q 屬於數量最多的紅菇村 。

由上述驗證可知，N 最小應該選擇 5。



資訊科學上的意義

當資訊科學家要分析大量資料時，常會利用分群 clustering 或分類 classification 的方法來區分資料。分群 clustering 是先蒐集資料，並依據現有資料的相似性來將資料分成多個小群體；分類 classification 則是找出區分資料的規則，將取得的資料依規則分類。

KNN 演算法 K-Nearest-Neighbors 又稱 K 近鄰演算法，是資料分類 classification 的一種方法；古人所說的「近朱者赤，近墨者黑」很傳神的說明了 KNN 演算法的概念。在 KNN 演算法中，一筆資料最接近的 K 個鄰居最多數屬於哪一類，那麼這筆資料就會被歸入那一個類別。本任務中的新屋將分配給最近的 5 間房屋中出現最多次的村莊，就是採用 KNN 演算法的概念來幫新屋分類。KNN 演算法採用資料特徵空間的距離來找出最相似的 K 筆資料，稱為 K 個鄰居；它是一種基於實體區分的分類學習。由於它簡單易懂，適用範圍廣，是機器學習領域中常見的演算法。

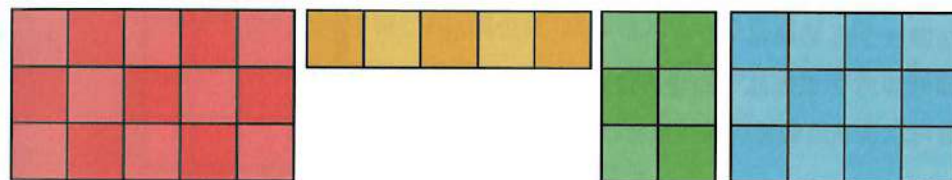


關鍵字

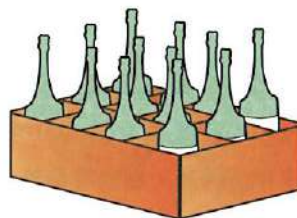
KNN 演算法

23. 給總統的禮物

一家工廠生產四種果汁，使用不同尺寸的特殊板條箱包裝。下圖圖一的四張圖片展示從上方看這些板條箱的樣子，其中最大的板條箱可放 15 瓶。（圖二則展示一個裝 12 瓶的箱子，從側面看的樣子。）



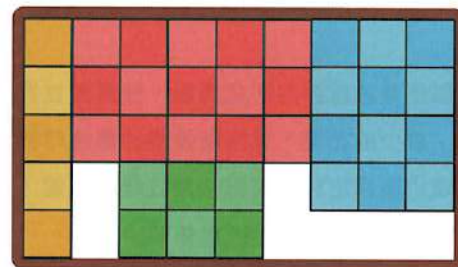
圖一



圖二

你必須為總統準備一份特別尺寸的禮物箱，擺放四種果汁的板條箱後（不能疊在一起），剩下的空隙愈小愈好，再添加單瓶填滿。

例如，如果使用下圖尺寸禮物箱擺放四種板條箱，須添加 7 個單瓶才能填滿。



若重新選擇適合尺寸的禮物箱，擺放好四種板條箱後，最少須添加多少個單瓶才能裝滿？(範圍 [1~99] 的整數。)



正確答案是：2

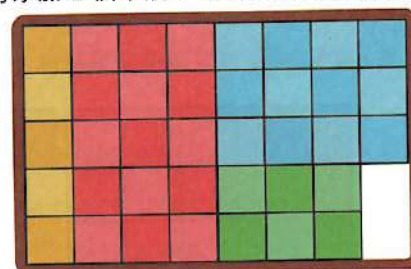
四個板條箱中的瓶子數量為 $12+15+6+5=38$ 。

如果禮物箱剛好容納 38 個瓶子，尺寸必須為 1×38 或 2×19 。但是，這樣的尺寸下， 3×5 板條箱（或 3×4 板條箱）皆無法順利擺放至禮物箱。

如果禮物箱容納 39 個瓶子，也就是擺放四個板條箱後，須再添加 1 個單瓶，則尺寸有兩種可能：

1×39 （不可能擺放）或 3×13 。而兩個最大的板條箱 3×5 及 3×4 ，在這樣的禮物箱中需佔據 9 排，剩餘的 3×4 空間，無法擺放 1×5 板條箱。

因此，額外添加 2 個單瓶，四種板條箱的排列方式如右圖所示：是可以裝滿這種特別尺寸禮物箱的最小值。



資訊科學上的意義

最佳化 optimization 是尋求一個問題最佳解答的過程，也就是在已知的條件限制下，將問題以數學方式表示，並透過運算或搜尋的方法求出最佳解 optimal solution 的步驟。裝箱問題 bin packing 是把一些物品打包到有限箱子中的問題，屬於經典的組合最佳化問題。

以本任務來說，禮物箱必須先擺放好四種板條箱後，計算裝滿的最少添加單瓶數。在限制的條件下，找尋最佳解答，就是最佳化的問題。本任務同時也是裝箱問題，我們希望能快速找到一個最佳解決方案的演算法。然而，即使我們使用功能非常強大的電腦，當不同尺寸的板條箱種類很多時，也很難找到最佳的裝箱解決方案。所以，我們只能盡可能找到接近最佳的解。

裝箱問題是現實生活中常常會遇到的情況，例如：貨車司機想盡可能多載運一些箱子；停車場管理人想規劃出最多的停車位；裁縫師想用最少的總面積在布料上裁剪出不同形狀的布片；設計師想在最小面積的方形木板上切割出組合元件；工程師將用到的電子元件安排到最省面積的麵包板或主機板上。

最佳化在科學、工程、數學界都是非常重要的研究課題。在真實世界中，每個需要最佳解的問題性質都不同，例如：最短的路徑、最少的花費，或判斷投資的最佳時機等，因此沒有唯一的最佳化方法來對應所有問題。在科學領域也發展出許多演算法來找尋最佳解，有些雖然無法保證能找到最佳解，但是可以由資料特性的分析幫助我們找到接近最佳解的結果。



關鍵字

最佳化、裝箱問題

24. 最喜愛的寶石

小伊收集許多寶石，在他心中，有寶石喜愛程度的排列順序。他的朋友莎拉，知道小伊收集寶石，但她不知道小伊心中的寶石排名。因此，她計劃想找出小伊心中最喜愛的寶石，方法如下：

- 第一輪，莎拉從全部的寶石中，先選了四顆，問小伊：這四顆寶石你最喜歡哪一顆？
- 第二輪，莎拉再從全部的寶石中選四顆，問小伊：這四顆寶石你最喜歡哪一顆？
- 最後一輪，莎拉也是從全部的寶石中選四顆，問小伊：這四顆寶石你最喜歡哪一顆？

提示：

在第二輪及最後一輪中，莎拉可以選擇先前已挑選過的寶石。



假如莎拉從這三輪中，必定能找到小伊最喜愛的寶石，請問小伊收藏的寶石數量最多為何？（範圍 $[1 \sim 16]$ 的整數。）



正確答案是：10

為了找到小伊收藏的寶石最數量，莎拉可以透過前兩輪向小伊詢問 8 顆不同的寶石，小伊在這兩輪的答案都可能是他最喜歡的寶石。因此，最後一輪她可以選擇前兩輪的答案與另外 2 顆還沒選的寶石，小伊最後一輪的答案一定是他最喜歡的寶石。在這種情況下，莎拉的策略就會有效，小伊收藏的寶石數量為 $8 + 2 = 10$ 顆。（請注意，還有其他適用於 10 顆寶石的正確策略，例如：莎拉可以在前兩輪選擇至少 1 顆同樣的寶石。）

接著我們來看看如果小伊有 $10 + 1 = 11$ 顆寶石，莎拉的策略是否會有效。莎拉可以在前兩輪選擇至少 1 顆同樣的寶石，那麼在前兩輪之後，至少有 4 ($11 - 4 - 3$) 顆寶石沒有選過。在這種情況下，最後一輪莎拉必須選擇這 4 顆剩下的寶石來詢問，因為如果是這麼選擇，最後 1 顆沒有被選到的寶石可能是小伊最喜歡的。然而，她也因此沒有機會將前面 7 顆中最喜歡的寶石與最後 4 顆寶石中最喜歡的相比，在這種情況下，她的策略無法確定小伊最喜歡哪顆寶石。

如果莎拉在前兩輪都選擇不一樣的寶石，那麼小伊可能最喜歡的寶石包括前兩輪的答案和其餘 3 個未選擇的寶石，小伊最喜歡的寶石共有 5 種可能性，而莎拉無法知道哪顆是小伊最喜歡的，因此她的策略無法有效。

由此可知，如果寶石超過 10 顆，莎拉的策略就不會有效，小伊收藏的寶石數量最多為 10 顆。



資訊科學上的意義

在資訊科學的領域，我們經常需要找出解決問題的最佳演算法 algorithm，它是解決問題的一系列步驟。然而，可解決該問題的演算法可能有很多個，我們必須訂定演算法的評估標準，用來比較各個演算法的好壞。

在本任務中要解決的問題是從多個寶石中找出小伊最喜愛的寶石，而設計解決此問題的演算法有以下限制：

1. 只能從 4 顆寶石中詢問其中最喜歡的。
2. 只能透過三輪，每次選擇 4 顆寶石詢問。

評估演算法的方式是其可以運作的寶石最大數量。

對演算法設計加上限制條件可能有不同的原因，例如：要求結果必須在一定的時間內完成，這在即時作業系統很重要。另一個原因可能是有些運算需要額外的成本。演算法在某些問題的狀況會失效，工程師有責任避免這些情況發生，例如：此任務不能用於包含大於 10 的元素的集合。



關鍵字

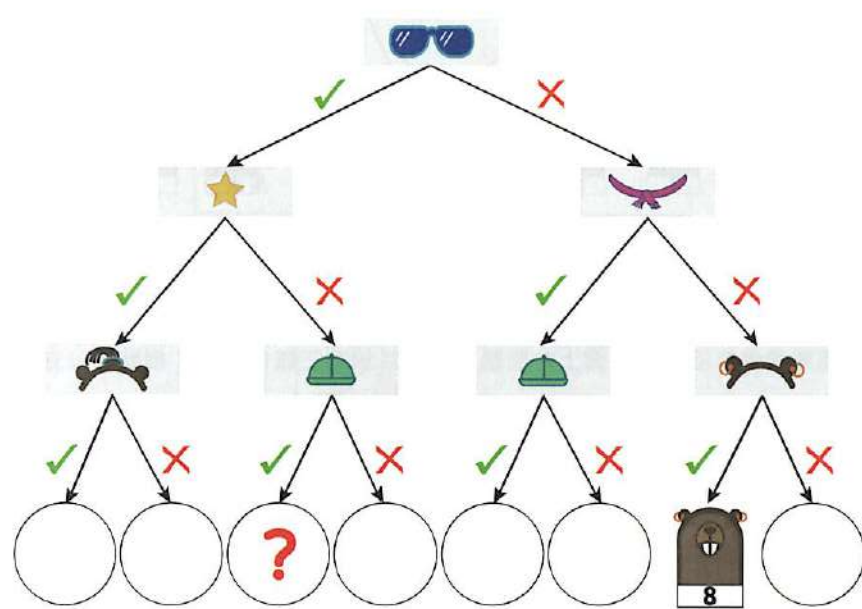
演算法、演算法評估



25. 狸臉辨識



大家要和同學們初次見面了！上圖是 8 位新同學的相片和座號。以下是區分新同學的方式：



請問上圖 ? 的同學座號應該是幾號？（範圍 [1~7] 的整數）

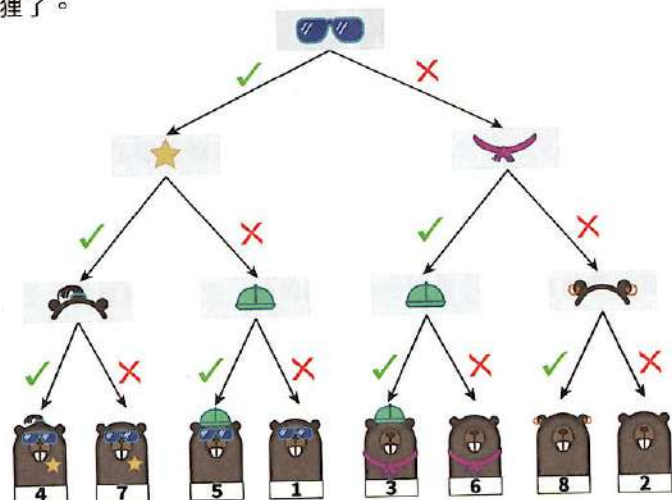




正確答案是：5

本題目有多種解法，其一的解法就是從選項中選一隻海狸，順著圖表由上到下對應的穿戴的特徵，找出每隻海狸的正確位置，所以是座號 5 的海狸會對應到圖中 ? 的位置。

另一種解法是依決策樹中的條件，找到有戴眼鏡和帽子但沒有星星胸章的海狸，對應圖中 ? 的答案就是座號 5 的海狸了。



資訊科學上的意義

這個任務我們能靠著依序回答樹狀圖上的問題來將照片放到正確的位置。樹狀圖分支上的每個問題都只有「是」與「否」兩種答案，然後據此向右或向左分支繼續往下作答，直到找到位置。

在資訊科學中，可以透過 **決策樹 decision tree** 的建立來將物品進行分類，決策樹是採用一個樹狀結構，最上層的根節點表示涵蓋所有資料，在每個節點挑出資料的某一個特徵，經由分支上的特徵值條件進行決策將資料分組，再繼續下一層條件判斷，直到區分出物品的類別。

另外在資訊科學中的 **條件判斷 conditional statements**，在程式撰寫的語法是：「如果 $\times\times$ ，那麼 $\triangle\triangle$ ，否則 $\diamond\diamond$ (英文式常用 If $\times\times$, then $\triangle\triangle$, else $\diamond\diamond$.)。這個語法的意思是，當條件式 $\times\times$ 成立時 (真)，則執行陳述句一 $\triangle\triangle$ ，要不然就執行陳述句二 $\diamond\diamond$ ；如果條件式不成立時不需要作任何事，則 else 及接續的敘述可以省略。以本任務舉例說明，樹狀圖上顯示的每個條件判斷描述如下：「如果海狸有戴星星胸章，那麼就往左邊分支，否則往右邊分支。」。

在生活中也常用到條件式的判斷結果將事物分門別類，例如我們在玩猜詞語遊戲時，出題者只用「對」與「錯」來回答挑戰者的問題，而挑戰者則要設法問有效的特徵條件來得到線索，猜出答案。



關鍵字

決策樹、條件判斷



26. 海狸餐廳

海狸餐廳的每項餐點是由不同的葉子 和小花 兩種圖案的組合來表示。

例如下圖，漢堡用 表示；炒飯用 表示。

菜單	圖案組合
漢堡	
炒飯	

服務生收到客人點餐後，會將餐點圖案組合用訊號燈傳到廚房。但老闆發現一個問題：若有人點炒飯，廚師看到 就會開始準備漢堡，不會等到 訊號，這樣就會造成錯誤。

為了避免錯誤，老闆更新了代表餐點的圖案組合，如下圖：

菜單	圖案組合
漢堡	
炒飯	
三明治	
披薩	
餡餅	

老闆想新增販售炸薯條，請問炸薯條應該用下列哪種圖案組合才能避免錯誤呢？

- A.
- B.
- C.
- D.
- E.



正確答案是：A

某一項餐點的完整圖案組合，不能和另一項餐點圖案組合的前面幾個相同，否則廚師只接收部分的訊號燈，誤以為已經接收完全部，便開始製作錯誤的餐點。

選項 A 正確是因為它完整的圖案組合，與其他現有餐點的前幾個圖案都不同；而其他現有餐點的完整圖案組合，也和選項 A 的前幾個圖案不同。

選項 B 錯誤，因為當廚師收到   便開始製作漢堡，永遠不會製作炸薯條。

菜單	圖案組合
漢堡	 
選項 B (炸薯條)	  

選項 C 錯誤，因為當廚師收到    便開始製作選項 C 代表的炸薯條，永遠不會製作炒飯。

菜單	圖案組合
炒飯	   
選項 C (炸薯條)	  

選項 D 錯誤，因為當廚師收到    便開始製作餡餅，永遠不會製作炸薯條。

菜單	圖案組合
餡餅	  
選項 D (炸薯條)	   

選項 E 錯誤，因為當廚師收到    便開始製作披薩，永遠不會製作選項炸薯條。

菜單	圖案組合
披薩	  
選項 E (炸薯條)	   



資訊科學上的意義

編碼 **encoding** 是指將訊息從某種形式轉換成另一種形式。在資訊科學中，我們會利用適當的編碼，將文字、音訊、圖片等各種不同形式的資料，轉換成可以透過電腦處理的形式。在此任務中就是透過一套編碼規則來點餐，所以為了避免錯誤，每種餐點需要獨一無二的代碼，並且代碼不可以與任一另種餐點代碼的前面部分重複；這種代碼在科學上稱為 **前置碼 prefix code**（或稱前綴碼）。前置碼系統通常是可變長度碼，在編碼中的每個碼字 **code word**，都不能被其他碼字當成前置代碼 **prefix**。並可以讓代碼變得獨一無二，不容易產生錯誤。

在資訊科學中，前置碼常應用於資料壓縮：比如 ASCII 編碼中，每個字母由 7 位元的 0 和 1 組合而成，但如果將前置碼的概念應用到字母的編碼，字母可以使用不同長度的位元序列進行編碼，只要確定每個字母的位元序列編碼，與另一個字母的位元序列開頭不相同，就可以將編碼重複部分壓縮得更短，用更少空間來儲存。例如，"e" 是最常見的字母，且後面常跟著 "a" 和 "t"，我們可以使用 0 作為 "e" 的編碼，10 表示 "a"，110 表示 "t"。當我們對單字 "eat" 進行編碼時，會得到 010110。原本 ASCII 中，編碼 "eat" 時需要 $3 \times 7 = 21$ 位元儲存，現在只要 6 位元空間。這種編碼方式稱為 **霍夫曼編碼 Huffman code**（前置碼的一種），其概念是把出現頻率越高的字母用越短的編碼來表示；相反地，字母出現頻率越低的字母則可以用長一點的編碼來表示，如此即可大幅節省編碼的長度，以達到資料壓縮的目的。"A"

我們在生活中會將冬被中的空氣壓扁，以節省空間；另外，以文字或符號詮釋各種訊息，也會進行縮減。例如：現代人會以 3Q 表示 **Thank you**、200000000 美金 縮寫成 兩億鎊、**Doctor** 縮寫成 **Dr.**、公共汽車 簡稱為 公車，甚至簡潔的文言文，都是將資料壓縮的例子。



關鍵字

前置碼、編碼、霍夫曼編碼



27. 彩色蠟燭 - 題組一

賽門有數字 0 到 9 形狀的蠟燭，每種蠟燭的數量不限，但是同一種數字的顏色是固定的，比方說數字 0 的顏色都是橘色的；各數字對應的顏色如下表所示。每年過生日的時候他就會把表示自己年紀的蠟燭插在生日蛋糕上。

0	1	2	3	4	5	6	7	8	9
橘色	紅色	藍色	橘色	紅色	藍色	橘色	紅色	藍色	橘色

比方說如下圖，今年賽門 47 歲，蛋糕上便插著一根數字 4 和一根數字 7 的蠟燭。



他發現今年蛋糕上所有蠟燭的顏色相同。

請問再過幾年，賽門生日蛋糕上所有蠟燭的顏色又會相同？

- A. 3 年
- B. 4 年
- C. 5 年
- D. 6 年



28. 彩色蠟燭 - 題組二

賽門有數字 0 到 9 形狀的蠟燭，每種蠟燭都恰好兩個；同一種數字的顏色是固定的，比方說數字 0 的顏色都是橘色的；各數字對應的顏色如下表所示。每年賽門生日的時候他就會把表示自己年紀的蠟燭插在生日蛋糕上。今天是賽門 11 歲生日，他發現蛋糕上的兩根蠟燭顏色相同，他的家人決定往後每當生日的兩根蠟燭顏色相同，他們就多送賽門一份禮物。因此三年後等到賽門 14 歲時，他會多獲得一份禮物；再三年後賽門 17 歲時，也會多獲得一份禮物；再來則是過五年後當他 22 歲時。

0	1	2	3	4	5	6	7	8	9
橘色	紅色	藍色	橘色	紅色	藍色	橘色	紅色	藍色	橘色



直到賽門 99 歲前，可獲得額外禮物的兩次生日之間最長間隔幾年？

- A. 5 年
- B. 6 年
- C. 7 年
- D. 8 年



27. 正確答案是：C.5 年

28. 正確答案是：A.5 年

賽門 47 歲生日的時候使用了兩根同樣是紅色的蠟燭，若要找出接下來哪一年會再使用兩根同樣顏色的蠟燭，可以先假設賽門年齡的十位數字是 X ，個位數字是 Y 。如果 X 依然是 4（紅色），那 Y 只可能是 8 或 9，但這兩個數字的蠟燭都不是紅色，因此 X 不會是 4。接著假設 X 是 5（藍色），同樣是藍色蠟燭的數字為 2、5、8，所以說當賽門 52 歲、55 歲和 58 歲時，都會使用兩根同樣顏色的蠟燭；而其中最接近 47 歲的年齡是 52 歲，也就是 5 年後。

若以 X 和 Y 分別表示年齡的十位數和個位數，並依題意直接假設 X 和 Y 為同色，可歸納：

- 當 $Y \in \{0, 1, 2, 3, 4, 5, 6\}$ 時，需要等待 3 年，直到年齡變為 $10X + (Y+3)$ 時，兩根蠟燭才會是同色的。
- 當 $Y=7$ 時，兩根蠟燭要同樣是紅色，則 $X \in \{1, 4, 7\}$ ，此時需等待 5 年，直到年齡變為 $10(X+1)+2$ 時，兩根蠟燭才會是同色的。
- 當 $Y=8$ 時，兩根蠟燭要同樣是藍色，則 $X \in \{2, 5, 8\}$ ，此時需等待 2 年，直到年齡變為 $10(X+1)+0$ 時，兩根蠟燭才會是同色的。
- 當 $Y=9$ 時，兩根蠟燭要同樣是橘色，則 $X \in \{0, 3, 6, 9\}$ ，此時需等待 2 年，直到年齡變為 $10(X+1)+1$ 時，兩根蠟燭才會是同色的。

因此可獲得額外禮物的兩次生日最長間隔 5 年。



資訊科學上的意義

序列 **sequence** 是指被排成一系列有順序的資料，序列中的元素可稱為 **項**。在資訊科學中，我們經常被告知某個序列的生成規則，並需要依此規則模擬序列的生成。在此任務中，給定年齡的任一位數，透過一個給定的函數對應到一個顏色集合。因此由年齡的兩位數可對應到兩個顏色組成的項。依年齡由小到大排列，對應的兩根蠟燭顏色組合項即成為一個序列，具在此序列中可推估出符合某些規則的項，發生在第幾項。

在解決一個問題時，我們可以透過 **模式辨識 pattern recognition** 問題中的特定 **模式**，也就是其共通點或是關聯，進而從這些模式中找到解決問題的方法。例如若要完成此任務，我們必須找出蠟燭的數字與顏色之間的關係；此關係就是我們需要辨識出的模式。模式辨識在我們日常生活中十分常見，例如影片串流平台會透過模式辨識，找到觀影者偏愛的影片與節目特性，進而推薦觀影者可能會有興趣的影片與節目。



關鍵字

序列、模式辨識



29. 串列

我們將數字 3, 5, 2, 4, 1 以串列的形式表示，如下圖所示，此串列名稱為 X ，上面較小的紅色數字代表串列中位置的編號。

	1	2	3	4	5
X	3	5	2	4	1

我們以 $(X\ 2)$ 來表示在 X 串列中位置編號為 2 的數字，因此 $(X\ 2)$ 是 5， $(X\ 5)$ 是 1。

也可以運用間接方式表示位置；例如 $(X\ (X\ 3))$ 是 5，因為 $(X\ 3)$ 是 2，所以 $(X\ (X\ 3))$ 可置換為 $(X\ 2)$ ，而 $(X\ 2)$ 是 5。

現在有 A、B 和 C 三個串列，如下圖所示：

A	3	2	4	1	5
B	6	4	1	3	2
C	2	5	4	3	1

請問依上圖所示， $(A\ (B\ (C\ 3)))$ 所表示的數字是多少？（範圍 $[0\sim9]$ 的整數）



正確答案是：4

若要知道 (A (B (C 3))) 所表示的數字，我們要由內而外找出各串列中特定位置的數字。

依圖示可知 (C 3) 是 4，(B 4) 是 3，(A 3) 是 4。故 (A (B (C 3))) 可被取代為 (A (B 4))，而 (A (B 4)) 再被取代為 (A 3)，所以答案為 4。

A	3	2	4	1	5
B	6	4	1	3	2
C	2	5	4	3	1



資訊科學上的意義

資料結構是程式設計非常重要的元素；而眾多資料型態 data types 中，串列 lists 可以用來儲存一連串有順序性的元素。

一個串列的元素中會紀錄下一個元素的位置，如本任務中 X(X3) 的範例；這種以間接方式來取得下一筆資料位置的概念，依取用規則逐步處理後，就會得到一個資料序列。若跨越不同的表，例如此任務中的 (A (B (C 3)))，就是用存放在 (C 3) 的資料（即數字 4）來描述在 B 中的位置，再取用存放在 (B (C 3)) 的資料來描述 A 中的位置（即數字 3）；在經過依序逐步處理後，即可取得 (A (B (C 3))) 所代表的資料值 4。

在資料庫中，我們也常將用索引值來關聯多個資料表間的關係，用以節省資料儲存量。以下圖為例，若要紀錄某班級短跑比賽的結果，可用表 A 以名次為索引紀錄比賽結果，每個索引對應的位置紀錄得到該名次的學生座號；再另外利用表 B 以學生座號為索引紀錄學生姓名，每個索引對應位置紀錄該座號的學生姓名。若要知道哪位學生獲得第 5 名，可使用 (B (A 5)) 查找：(A 5) 為 13 故 (B (A 5)) 即為 (B 13)，也就是「林小茗」。

	1 st	2 nd	3 rd	4 th	5 th	6 th	7 th	8 th	9 th	10 th
A	25	10	1	3	13	2	16	5	31	20
	1	2	3	4	5	13	16	20	25	31
B	王一明	李大同	陳三頂	劉四全	張五良	林小茗	周大明	林春天	趙夏季	孫秋山

爾後若有其他比賽，都可以利用不同表紀錄比賽中獲得各名次的學生座號，透過查找紀錄比賽結果的表，找出該比賽獲得特定名次的學生座號，即是在表 B 中的位置，再找到對應的學生姓名。只紀錄座號的資料量比紀錄姓名小（2 位數字的座號占 2 bytes，而當姓名為 3 個中文字時則占 6 bytes），所以上述將學生姓名獨立存放於表 B 的紀錄方式，可節省儲存的資料量。

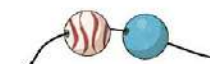


關鍵字

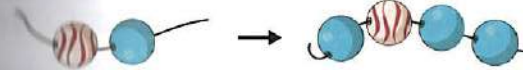
資料型態、串列、陣列

30. 水手項鍊

水手項鍊是由紅條紋珠子  和藍色珠子  透過以下規則製作。首先，將一顆紅條紋珠子和一顆藍色珠子依下圖的順序串在繩子上。

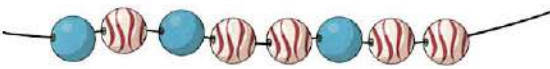
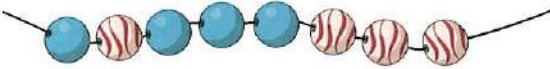


接著，只能透過以下兩種動作加長水手項鍊：

(X) 從繩子的兩端各加一顆藍色珠子	(Y) 從繩子的最右端加兩顆紅條紋珠子
	

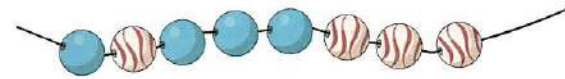
(X)、(Y) 兩種動作可以任意搭配重複多次。

下列哪條項鍊不是水手項鍊？

- A. 
- B. 
- C. 
- D. 



正確答案是：D



一種完成任務的方法，是先找出兩顆起始珠子，再模擬執行一系列 (X) 和 (Y) 動作後，是否能串出選項中的項鍊。

- 選項 A 起始珠子在第二、三顆，依序執行 (X)、(Y)、(Y) 動作即可串出。
- 選項 B 起始珠子在第三、四顆，依序執行 (X)、(X)、(Y) 動作即可串出。
- 選項 C 起始珠子在第二、三顆，依序執行 (Y)、(X)、(Y) 動作即可串出。
- 選項 D 起始珠子在第二、三顆，雖然可能執行一次 (X) 動作，但之後沒有辦法串成圖中項鍊。

上述的方法在項鍊很長、起始珠子的可能性很多時，無法有效率的完成任務。在這種情況下，我們可以利用解構串珠的方式：反覆移除 (X) 或 (Y) 動作加入的珠子，直到最後只剩下兩顆起始珠子。此外，根據水手項鍊說明，不管使用哪種方式串項鍊，串完後一定會有奇數的藍色珠子和奇數的紅條紋珠子。但選項 D 中兩種顏色珠子的數量都是偶數，因此不會是水手項鍊。



資訊科學上的意義

佇列 queue 是一種先進先出的有序串列，它的資料處理方式是將資料由儲存結構的後端加入、由前端刪除；就像人們在排隊進電影院一樣，加入隊伍的人會從後端加入、從前端離開隊伍入場。雙向佇列 double-ended queue 則在前端或後端皆可添加與刪除資料，但不能從中間添加或刪除資料。如同此任務中，只能在項鍊的兩端添加珠子，而無法將珠子插入項鍊中間任一位置；同樣的，若想取出珠子，無法從項鍊中間直接取出任一顆珠子，而只能從兩端依序取出珠子。日常生活中，只能從兩端置入及取出物品的容器，像是串珠飾品，就具有雙向佇列的特性。



關鍵字

雙向佇列

31. 奧赫里德珍珠

小莫和小維到馬其頓度假，他們買了有名的奧赫里德珍珠串成項鍊，兩人項鍊的串珠圖如下圖所示：

小莫	
小維	

他們決定依照下列步驟更換項鍊中的珍珠：

步驟 1：小莫和小維分別從自己項鍊的右側取出一顆珍珠。

步驟 2：如果珍珠是黃色（方形）或紅色（三角形），則把這顆珍珠加到對方項鍊的左側。

如果是其他顏色，他們會把這顆珍珠加到自己項鍊的左側。

每次都是小莫先更換，接著小維再更換。

步驟 3：如果二個人都已經給了對方三顆珍珠，則停止更換，否則繼續從步驟 1 執行更換。

請問經過上述步驟後，最後小莫和小維的項鍊串珠圖，為下列何者？

A.

B.

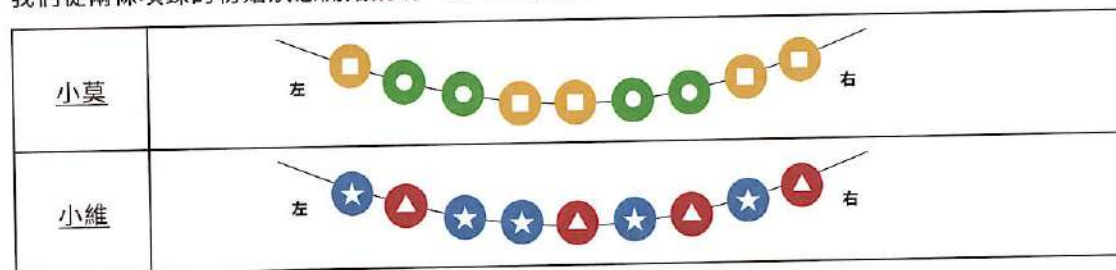
C.

D.

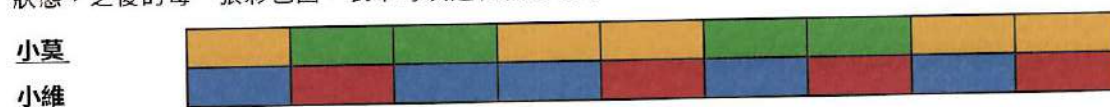


正確答案是：A

我們從兩條項鍊的初始狀態開始說明，如下圖所示：



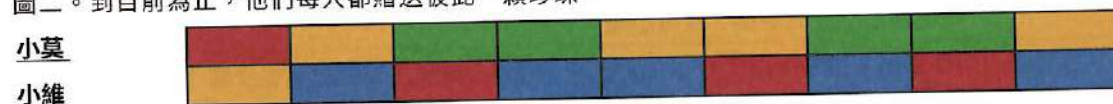
我們依照步驟從每條項鍊的右側取出珍珠，一次一顆，從小莫開始。以下的說明，我們將使用專門用語「迭代」來表示完成步驟 1 到 3。每張彩色圖用來表示每次迭代的結果，圖一表示項鍊的起始狀態，之後的每一張彩色圖，表示每次迭代後的項鍊。



圖一

【迭代 1】

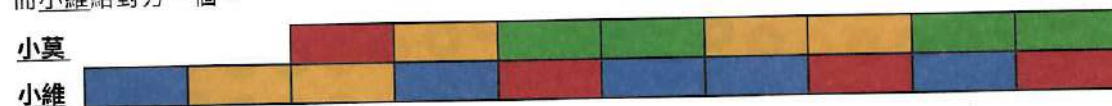
由於兩個人初始圖右側的珍珠都是黃色或紅色的，因此將它們交換並添加到對方左邊的項鍊上，如圖二。到目前為止，他們每人都贈送彼此一顆珍珠。



圖二

【迭代 2】

由於小莫的項鍊右側有一顆黃色珍珠，他將這顆珍珠加到小維項鍊的左側。小維右側則是一顆藍色珍珠（不會交換），所以它被加到了他自己項鍊的左邊，如圖三。到目前為止，小莫給了對方兩個，而小維給對方一個。



圖三

【迭代 3】

小莫的下一顆珍珠是綠色的，所以他把這顆珍珠添加到自己項鍊的左側。小維的下一顆珍珠是紅色的，所以他將它交換，加到小莫項鍊的左側，如圖四。到目前為止，每個人都給了對方兩個。



圖四

【迭代 4】

小莫的下一顆珍珠是綠色的，所以他把這顆珍珠添加到自己項鍊的左側。小維的下一顆珍珠是藍色的，所以他把這顆珍珠添加到自己項鍊的左側。這次迭代，彼此都沒有交換，如圖五。到目前為止，每個人還是只給了對方兩個。



圖五

【迭代 5】

兩個人下一顆珍珠是黃色和紅色的，所以它們被交換並添加到對方項鍊的左邊，如圖六。現在，每個人都給了對方三個。



圖六

因為他們各自給了對方三顆珍珠，所以他們停止迭代，對應圖六，最後的項鍊看起來如下圖。即答案是選項 A。



如上述說明，若小莫和小維每人需給對方三顆珍珠，則需迭代交換五次，所以選項 B、選項 C、選項 D，都是不正確的。



資訊科學上的意義

本任務可以使用資訊科學中的佇列 queue 概念來說明。佇列是一種資料結構的抽象型別，其中資料新增是從尾端 rear 添加，刪除是從前端 front 移除，進出的原則是先進先出 First In First Out FIFO。這是一種常用的資料結構，因為它會依照順序，將新的元素添加到佇列的最後。小莫和小維的項鍊串珍珠的方式算是佇列，因為本題規定項鍊的左側是添加新珍珠的一端，而另一端則僅能去除珍珠。

日常生活中類似佇列的例子很多，例如排隊等公車、排隊買電影票及印表機的工作排程等。在資訊科學中，佇列先進先出的特性，經常使用在資料儲存和處理的排程上。



關鍵字

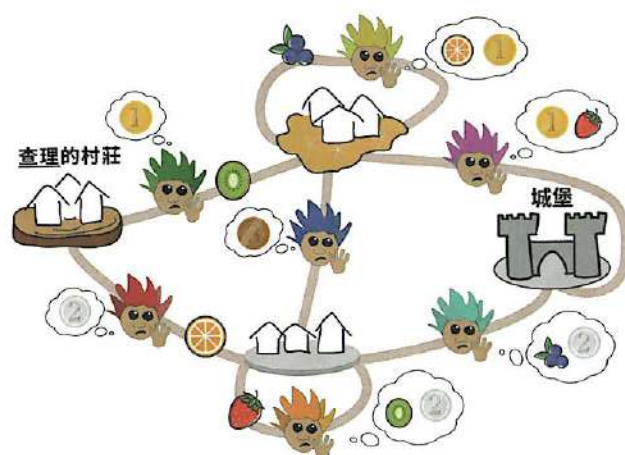
佇列、先進先出



32. 貪婪的妖怪 - 題組一

查理想要從他的村莊出發到城堡，但在村莊之間的樹林裡，有貪婪的妖怪在路上等著。（如下圖）妖怪們會向經過的人要求過路費，除了一枚特定的硬幣（1, 2, 3），有些妖怪還會額外要求一種特定的水果（蘋果, 香蕉, 橘子, 葡萄）。幸運的是，這些水果可以在某些小路找到，但必須先向這條路上的妖怪支付過路費才能拿到。

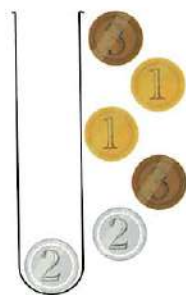
提示：不用走過每一條路，若是經過重複的路，要再支付過路費才可通過。



查理帶一個剛好只能容納五個硬幣的竹筒，只能從上面一個接一個拿硬幣，在他出發之前，必須將硬幣依正確的順序放進竹筒裡。

查理還帶著一個空的小背包來存放路上找到的水果，這個小背包很窄，每次只能拿到最上面的一種水果。

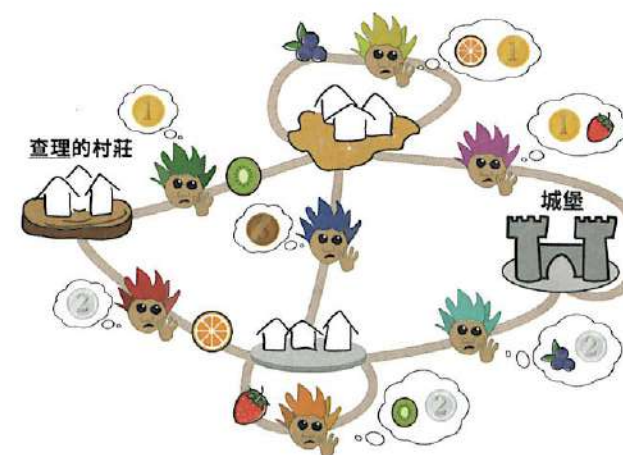
目前查理先裝了一枚硬幣（如下圖），請依照竹筒中由上到下的順序填寫四枚硬幣上的數字，這樣他就可以在前往城堡的路上向貪婪的妖怪支付過路費，不要忘記他還需要收集一些妖怪需要的水果。（四個範圍 [1~3] 的整數）



33. 貪婪的妖怪 - 題組二

查理想要從他的村莊出發到城堡，但在村莊之間的樹林裡，有貪婪的妖怪在路上等著。（如下圖）妖怪們會向經過的人要求過路費，除了一枚特定的硬幣（1, 2, 3），有些妖怪還會額外要求一種特定的水果（蘋果, 香蕉, 橘子, 葡萄）。幸運的是，這些水果可以在某些小路找到，但必須先向這條路上的妖怪支付過路費才能拿到。

提示：不用走過每一條路，若是經過重複的路，要再支付過路費才可通過。



查理帶一個剛好只能容納五個硬幣的竹筒，只能從上面一個接一個拿硬幣，在他出發之前，必須將硬幣依正確的順序放進竹筒裡。

查理還帶著一個空的小背包來存放路上找到的水果，這個小背包很窄，每次只能拿到最上面的一種水果。

請以正確的順序將五枚硬幣填滿查理的竹筒，這樣他就可以在前往城堡的路上向貪婪的妖怪支付過路費，不要忘記他還需要收集一些妖怪需要的水果。請選出竹筒中硬幣由上到下的排列順序。



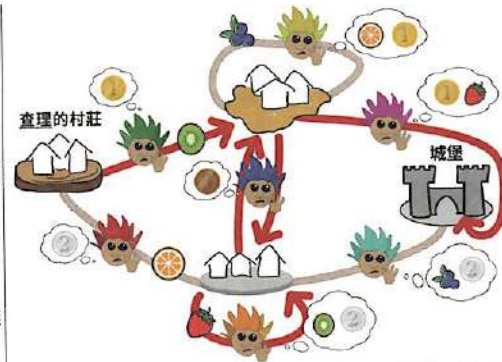


32. 正確答案是：2313

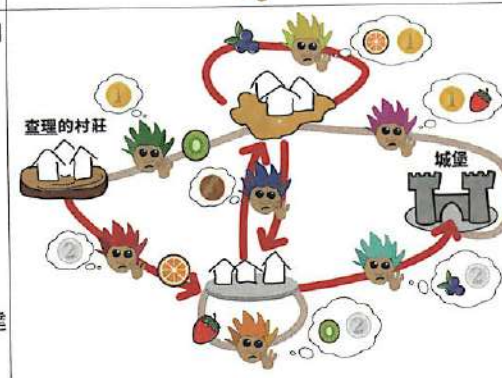
33. 正確答案是：D

根據任務敘述，查理應該有兩種移動順序：

1. 支付一枚金幣 ①，採摘奇異果 ⑥ 後，移動上方金色村莊
2. 支付一枚銅幣 ③，移動到下方銀色村莊
3. 支付一枚銀幣 ② 和奇異果 ⑥，並採摘草莓 ⑦ 後，回到銀色村莊
4. 再次支付一枚銅幣 ③，移動回上方金色村莊
5. 最後支付一枚金幣 ① 及草莓 ⑦，即可抵達城堡



1. 支付一枚銀幣 ②，採摘柳橙 ④ 後，移動到下方銀色村莊
2. 支付一枚銅幣 ③，移動到上方金色村莊。
3. 支付一枚金幣 ① 和柳橙 ④，採摘藍莓 ⑤，回到金色村莊
4. 再次支付一枚銅幣 ③，移動回下方銀色村莊
5. 最後支付一枚銀幣 ② 和藍莓 ⑤，即可抵達城堡



查理若想移動到城堡，就必須要支付草莓 ⑦ 或 藍莓 ⑤ 之中的一種水果：

- 若想取得草莓 ⑦ 就要先想辦法取得奇異果 ⑥，加上竹筒中只能容納五枚硬幣的限制，就只有上方第一種移動方式。
- 若想取得藍莓 ⑤ 就要先想辦法取得柳橙 ④，就只有第二種移動方式。

其餘的移動方式，都需要超過五枚硬幣。



資訊科學上的意義

圖 graph 是由若干節點 vertex 與連接節點間的邊 edge 所組成的資料結構，在此任務中村莊和城堡可視為圖中的節點、道路為邊。而裝有沿途所需硬幣的竹筒，以及可裝入水果的背包，運作方式則與堆疊 stack 相同。堆疊是一種具有後進先出 Last In First Out, LIFO 特性的資料結構，資料會從堆疊頂端 top 推入 push，並從堆疊頂端取出 pop 資料，如同此題中硬幣和水果都是由同一端放入和取出。在使用瀏覽器時「回到上一頁」和「前進下一頁」的功能，以及各種軟體中「復原」與「取消復原」的功能也都是堆疊的應用。

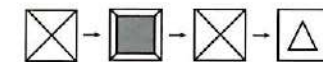


關鍵字

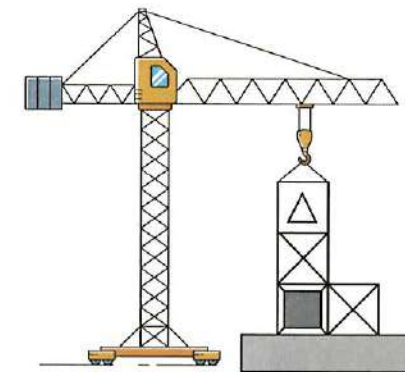
堆疊、有向圖

34. 搭建房屋

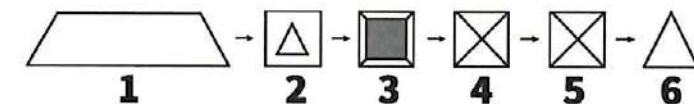
海狸國的房屋是用各種磚塊搭建的。海狸操作起重機，按指定的順序一次舉起一個磚塊，並放到地基上或是其他磚塊上面。



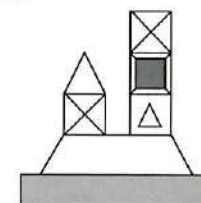
以上圖舉起磚塊的順序為例，可以蓋出如下圖的房屋。



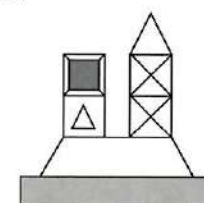
下圖是起重機舉起磚塊的順序，請問不可能蓋出哪棟房屋？



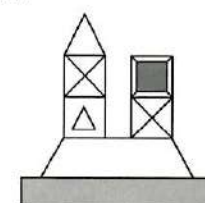
A.



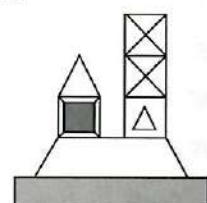
B.



C.



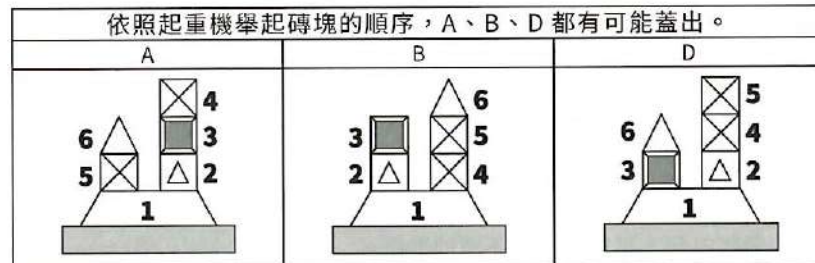
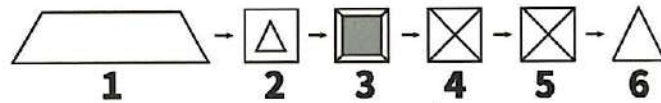
D.



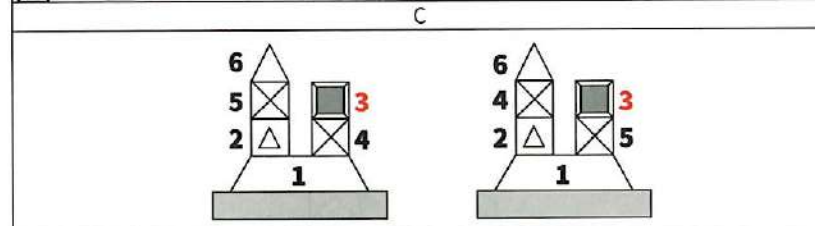


正確答案是：C

依照起重機舉起磚塊的順序編號如下：



C 不可能蓋出。因為 是第三個舉起的磚塊，若要蓋在圖中的位置， 下面 要先被舉起（編號順序要小於 3）才行。



資訊科學上的意義

堆疊 **stack** 是資訊科學領域一種常見的資料結構，它是一種抽象的資料型態，用來限制資料進出線性儲存結構的方式。在堆疊中，資料只能從一個端點（top）加入與取出，因此進出的順序特性是後進先出 Last In First Out, LIFO。

堆疊後進先出的特性經常使用在儲存資料和處理資料時，生活中對應的實例如同本題：將磚塊依序舉起，並一個一個蓋成房屋，先舉起的磚塊會被疊在下層。此外，瀏覽器的返回功能 也具有堆疊的特性，當按返回 時，最先回到頁面就是先前最後被瀏覽過的網頁。

餐袋中先放入便當盒 ，再放入餐具盒 ，最後放入水果盒 ，則拿出來的順序就是：水果盒、餐具盒、便當盒。



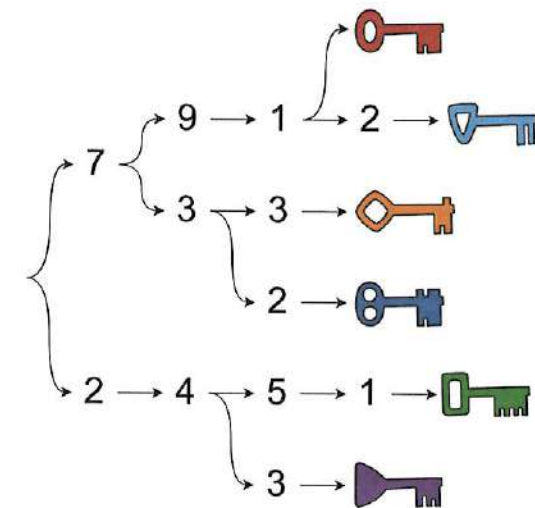
關鍵字

指令、堆疊

35. 泳池置物櫃

海狸社區泳池的置物櫃由一位老海狸看管，他將每隻海狸的貴重物品放入置物櫃後，會將鑰匙保管好，並在一張紙條上寫下置物櫃號碼交給物品主人。當海狸要離開泳池時可以拿著紙條領取置物櫃中的物品。

老海狸將置物櫃號碼畫成下方的數字和鑰匙關聯圖，方便他找尋鑰匙。例如，拿到 243 號置物櫃的紙條 時，老海狸會分別從紙條由左到右的 2、4 和 3 三個數字找到紫色鑰匙 。



傍晚時，海狸們發現數字卡片上沾了水，弄髒了其中一些數字。

老海狸依序拿到 、、 三張紙條；請問依照此順序的鑰匙組合為何？

- A.
- B.
- C.
- D.



正確答案是：A

第三張紙條為 7 開頭的 4 位數字，因此第三張紙條只會對應到 (7912)。

而第二張紙條是 2 為開頭的 4 位數字，因此第二張紙條只會對應到 (2451)。

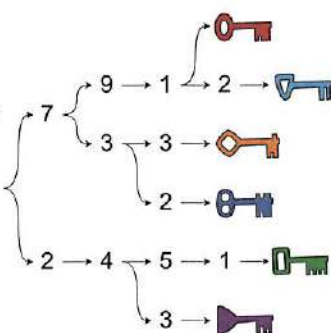
而第一張紙條可能對應到 (733) 或 (732)，因為這兩把鑰匙對應的數字前面兩碼都是 73。

由上述推論可知，正確答案可能 是 或 ，後者選項並未出現在選項中，所以選項 A 為正解。

選項 B 是錯的，因為 會對應到 3 位數字，而非 4 位數字。

選項 C 是錯的，因為 會對應到 2 開頭的數字，而不是 7。

選項 D 是錯的，因為 對應的數字前兩碼是 79，而不是 73。



資訊科學上的意義

老海狸繪製的圖形即所謂的 **字典樹 trie**，它被建構成一個有向樹，其中每條邊都指向鍵值中的一個字母或數字，透過逐字查找的方式找到對應的資料；如同此任務中，以數字找到對應的鑰匙。字典樹允許使用 **字首 prefix** 由前往後搜索，並找到所有可能的匹配項。除了用字首搜尋外，字典樹結構還可以擴展成以字尾由後往前搜尋。

字典樹常用於儲存字典，或是大量的字串，還可被搜尋引擎系統，用來統計文本中的詞頻。它的優點是可利用字串的字首比對，排除不需要的字串比較，提高查詢效率。字典樹也可以應用在搜索引擎中，在輸入關鍵詞時出現字首匹配的關鍵詞提示。



關鍵字

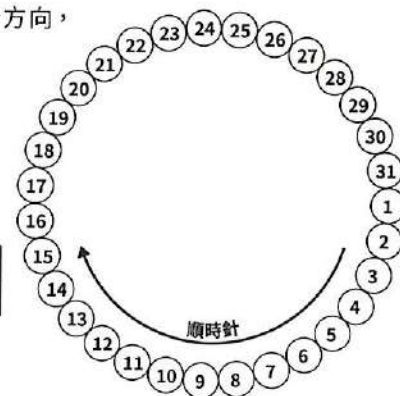
字典樹、字首、字尾

36. 教室座位

教室裡有 31 張椅子，編號 1 到 31，如下圖所示依序排成一個圓圈。

孩子們遵循以下規則找座位：

- 一次一個人進入教室
- 尋找自己生日日期號碼的椅子坐下
- 如果自己生日日期號碼的椅子已有人坐了，就沿順時針方向，找到下一個空位坐下



下表為四個孩子的生日：

姓名	吉塔	西塔	阿倫	阿賓
生日	4 月 20 日	4 月 20 日	1 月 21 日	9 月 22 日

如果進入教室的順序是吉塔、西塔、阿賓、阿倫，那麼吉塔會坐在 20 號椅子上，西塔坐在 21 號，阿賓坐在 22 號，阿倫坐在 23 號。

如果進入教室的順序是西塔、阿倫、阿賓、吉塔，那麼西塔坐在 20 號，阿倫坐在 21 號，阿賓坐在 22 號，吉塔坐在 23 號。

下表為六個孩子的座位資訊：

姓名	大衛	安安	小艾	依依	阿偉	小志
生日	8 月 11 日	2 月 12 日	5 月 11 日	9 月 14 日	4 月 13 日	7 月 12 日
椅子編號	11	12	13	14	15	16

下列哪項敘述絕對不可能發生？

- 依依是第一個坐下
- 小志是最後一個坐下
- 阿偉在小艾前先坐下
- 安安在大衛前先坐下



正確答案是：C. 阿偉在小艾前先坐下

選項 A 有可能發生，因為依依坐在與自己生日日期相同的椅子上，她有可能第一個坐下。

選項 B 有可能發生，小艾和大衛都出生於 11 日，而小艾坐在 13 號椅子，所以小艾比較晚坐下，而小志出生日期是 12，但卻坐在 16 號椅子，所以小志一定比坐在 12 到 15 號椅子上的人晚坐下，是最後一個坐下的人。

選項 C 不可能發生，如果阿偉在小艾之前坐下，那麼阿偉就會坐在 13 號椅子而不是 15 號椅子。

選項 D 有可能發生，安安和大衛都坐在與他們生日日期相同數字的椅子上，所以他們坐椅子的順序可以互換。



資訊科學上的意義

在處理資料時，人們為了方便日後能有迅速的找到正確的資訊，常常會運用 **雜湊函數 hash function**，將資料項目和與用來檢索的索引值建立關聯，形成一個容易搜尋的對應表，稱為 **雜湊表 hash table**。此任務中，每一個孩子代表一筆資料，而孩子的生日的日期被當做索引值。

由於雜湊函數是一個多對一函數，可能發生多筆資料項目映射到同一索引值的狀況，稱作發生 **碰撞 collision**；在此任務中不同孩子的生日日期相同時，就是發生碰撞。雜湊函數應該要提供處理碰撞的方式，讓雜湊表中的每一個索引都只對應到一筆資料；就像孩子們如果看到自己生日日期的椅子被別人坐著，就要順時針找到下一個空椅子，每一張椅子只會坐著一個孩子。

日常生活中有時也會找到雜湊函數的應用，像是有些學校在為學生建立學生電子信箱的帳號時，就會用學生姓名的英文縮寫；當不同學生的姓名縮寫相同時，相當於雜湊函數發生碰撞，必須在縮寫後面加上數字來區別不同的學生。資訊科學家為了方便找尋資料庫中的資料，常用雜湊表來建立可有效率搜尋到資料的索引結構。



關鍵字

雜湊函數



Benjamin/

中

Cadet/

中

Junior/

易

Senior/

易

37. 電影之夜

娜亞老師想為七位學生安排電影之夜的圓桌會議室（如下圖）座位，設計規則如下：

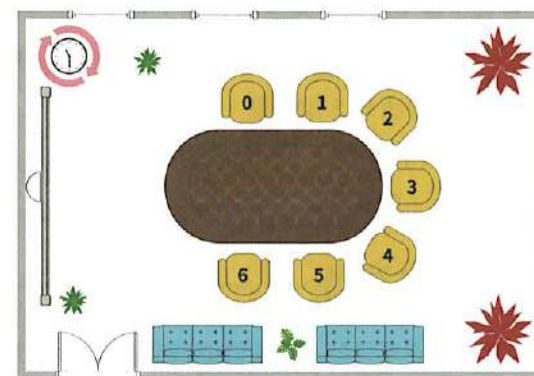
1. 生日規則：每位學生的座位號碼由生日的月與日相加，如果超過 7，則除以 7 取餘數。

例如：朱朱的座位號碼是 1，因為 $(12+24)$ 除以 7 取餘數為 1。

2. 重複規則：如果某位學生的座位號碼與別的同學重複，則移到下一個空的座位。

例如：座位號碼 2 已經有人，則移到座位 3，如果座位 2 跟 3 都有人，則移到座位 4。如果座位 6 有人，則移到座位 0。

姓名	朱朱	瑪莉	哈利	莫莫	喬恩	莎拉	艾蜜莉
生日 (月份 / 日期)	12/24	01/05	02/08	04/16	09/09	12/02	01/30



依照下列學生的順序來安排，莎拉會安排在那個座位？

瑪莉 → 哈利 → 艾蜜莉 → 喬恩 → 朱朱 → 莫莫 → 莎拉

- A. 座位號碼 0
- B. 座位號碼 4
- C. 座位號碼 2
- D. 座位號碼 5



正確答案是：C. 座位號碼 2

當我們依照題目所指示的規則，依序為七個同學安排座位，就會得到如右方的座位圖。

黃色的座位代表安排時可以直接入座，沒有發生同位置已經有人的碰撞情況；而紅色的座位就是發生位置重複，需要往後面編號找空位才入座的情況。

讓我們依規則將七位同學的生月月份及日期相加，除以 7 取餘數轉換成 0 到 6 的數字，並參考入座順序：

瑪莉 → 哈利 → 艾蜜莉 → 喬恩 → 朱朱 → 莫莫 → 莎拉

就會得到下面的表格資料

姓名	瑪莉	哈利	艾蜜莉	喬恩	朱朱	莫莫	莎拉
生日 (月份/日期)	01/05	02/08	01/30	09/09	12/24	04/16	12/02
相加結果 (MM+DD)	01+05 = 6	02+08 = 10	01+30 = 31	09+09 = 18	12+24 = 36	04+16 = 20	12+02 = 14
除以 7 取餘數	6	3	3	4	1	6	0
實際座位	6	3	3 → 4	4 → 5	1	6 → 0	0 → 1 → 2

同學依序入座，瑪莉坐入 6 號位置，哈利坐入 3 號位置，到了第三位同學艾蜜莉要坐入 3 號位置時就會發現已經坐了哈利，因此艾蜜莉就往後坐到 4 號位置，第四位同學喬恩本來要坐 4 號位置，因為已經有坐了艾蜜莉，所以就往後坐到 5 號位置。

接下來朱朱坐到 1 號位置，莫莫要坐的 6 號位置已經坐了瑪莉，就改坐 0 號位置，最後入座的莎拉因為 0 號、1 號位置都已經有人，就坐到 2 號位置。



資訊科學上的意義

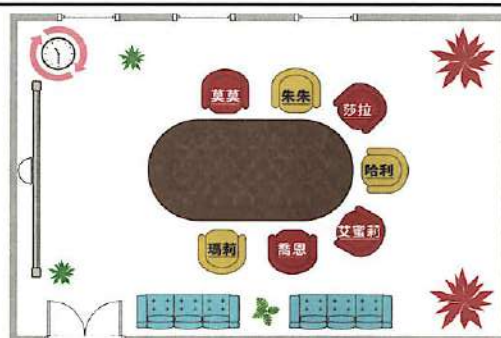
在資訊科學中，將資料存放到資料庫或是儲存裝置的磁碟空間時，有效率的安排儲存位置才能快速的存放及提取。良好的 **雜湊函數 hash function** 可以讓系統快速計算出資料儲存位置，並且讓資料盡量平均地分散在儲存設備中。

有很多方法可以對資料以數學運算的方式找到對應存放位置。本任務中使用 模算數 modular arithmetic 作為 雜湊函數 hash function，利用取除法餘數計算出存放位置的索引值，是一個簡單而常見的方式。但因為可能會碰到兩筆或多筆資料的取除法餘數相同而產生 **碰撞 collision**，這時候如何安排碰撞的資料到另一個存放位置，需要設計發生碰撞的處理規則。以本任務來說，因為座位與學生人數相同，因此只要將座位編號增加或歸零，最後還是可以找到空的位置。如何設計索引值較平均散佈的雜湊函數並有效率解決碰撞情況，是資訊科學家們探討的議題之一。



關鍵字

雜湊函數、碰撞



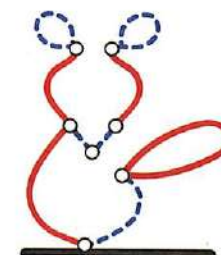
38. 切線遊戲

大狸與小狸在玩「切線遊戲」。

遊戲起始如右圖所示，有 1 條粗線（——），稱為「地面」；在地面上方則有虛線與實線組成 1 隻老鼠的形狀。

遊戲規則如下：

1. 大狸和小狸輪流切掉 1 條線段；大狸只能切虛線線段，小狸只能切實線線段。
2. 切掉選定線段後，留下的線段如果無法連接到地面，也在同回合被切掉。
3. 切下最後線段的人勝利。

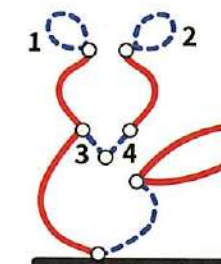


下面是某次遊戲的過程：

第 1 回合：大狸	第 2 回合：小狸	第 3 回合：大狸	第 4 回合：小狸
切掉身體右側虛線，無法連接地面的尾巴實線也被切掉	切掉頭部左側實線，左側耳朵虛線也被切掉	以此類推	切掉最後 1 條實線，小狸獲勝

假設遊戲從大狸開始，在大狸每回合都做出最佳決策的前提下，第 1 回合選擇切除哪條虛線可以保證大狸獲勝？

- 虛線 1
- 虛線 2
- 虛線 3
- 虛線 4
- 大狸不可能贏

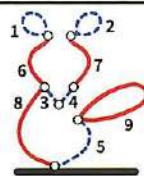




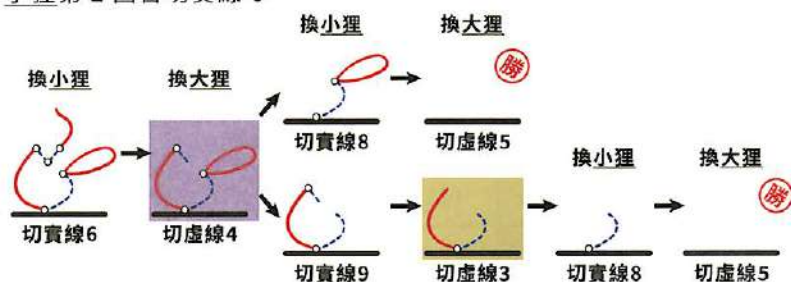
正確答案是：B. 虛線 2

在大狸每回合都做出最佳決策的前提下，只有大狸在第 1 回合切虛線 2 時，不管小狸怎麼切線，都可以保證大狸獲勝。

為了證明這一點，要分別檢視第 2 回合小狸切實線 6、實線 7、實線 8、實線 9 的四種不同可能狀態下，大狸做出最佳決策的後續發展。因為大狸無法控制小狸會切哪一條線，但一定會做出對自己最有利的決定；所以小狸切線的所有可能狀態都需要檢視，但對於大狸切線的可能狀態中，只要檢視最佳的策略即可。



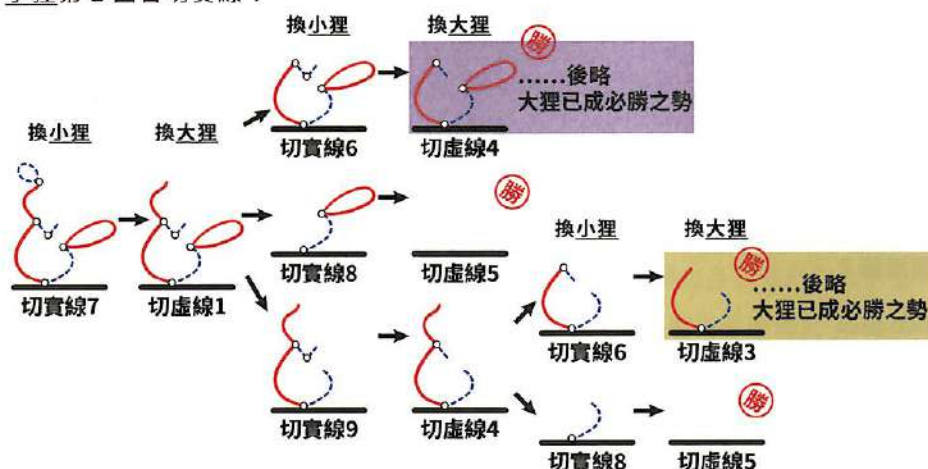
• 小狸第 2 回合切實線 6



上圖為小狸在第 2 回合切掉實線 6 之後，大狸做出最佳決策的後續發展。雖然大狸在第 3 回合還能選擇切掉虛線 3 或虛線 5，但是若是大狸切掉虛線 3 或虛線 5，就有可能會輸，所以切這兩條虛線都不是最佳決策。由此可以看出，只要大狸做出最佳決策，必能獲勝。

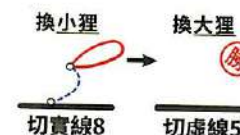
此外，觀察上圖可以發現若大狸切完線後，呈現紫色或黃色的狀態時，只要大狸做出最佳決策，就必定會獲勝。因此在分析後續的可能性時，只要大狸在切完線後呈現這兩個狀態，就表示大狸必勝。

• 小狸第 2 回合切實線 7



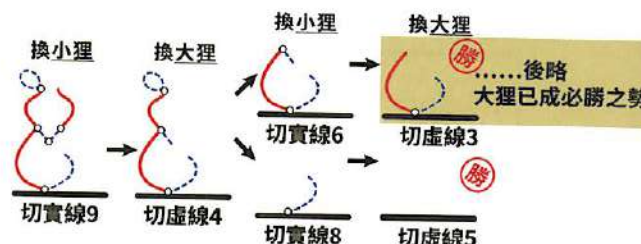
上圖為小狸在第 2 回合切掉實線 7 之後，大狸做出最佳決策的後續發展：只要大狸做出最佳決策，必能獲勝。此外，觀察上圖可以發現若大狸切完線的必勝狀態，除了紫色和黃色外，還包含橘色和綠色。

• 小狸第 2 回合切實線 8



上圖為小狸在第 2 回合切掉實線 8 之後，大狸必能獲勝。

• 小狸第 2 回合切實線 9

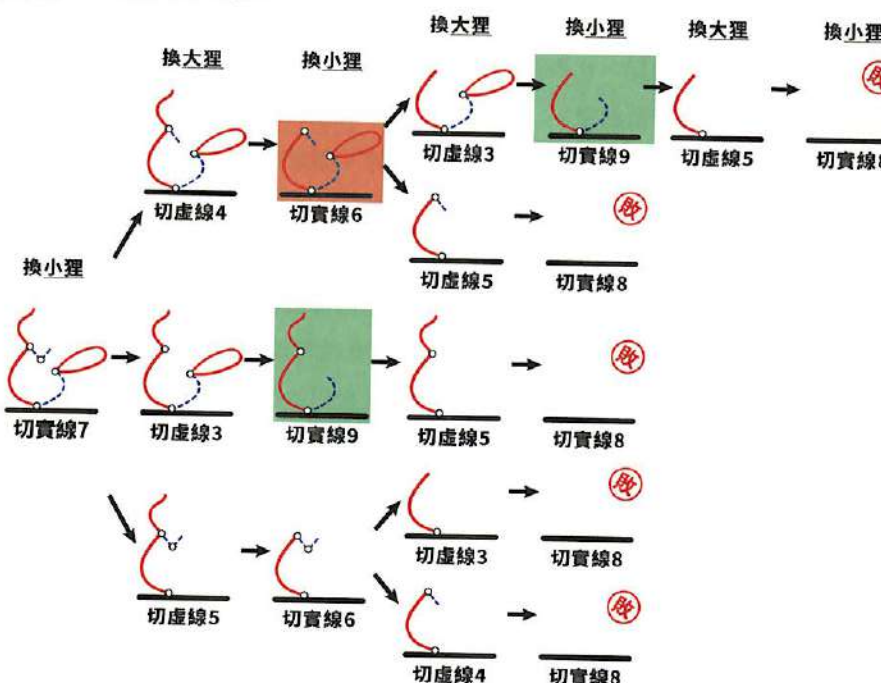


上圖為小狸在第 2 回合切掉實線 9 之後，大狸做出最佳決策的後續發展：只要大狸做出最佳決策，必能獲勝。

至此我們窮舉了大狸在第 1 回合切掉虛線 2 之後，大狸做出最佳決策之下的所有可能性；由此可以看出大狸在第 1 回合切掉虛線 2 之後就一定能獲勝。

為了證明這是唯一讓大狸確保勝利的方法，要再依序檢視大狸在第 1 回合分別切掉虛線 1、虛線 3、虛線 4 的後續可能狀態。而因為大狸無法控制小狸會切哪一條線，所以不用檢視小狸切線的所有可能狀態，只要找到小狸第 2 回合切哪一條線，會讓大狸不管如何都無法獲勝，就能證明大狸在第 1 回合切這幾條虛線，有可能會輸。

• 大狸第 1 回合切虛線 1





上圖為大狸在第 1 回合切掉虛線 1 之後，小狸在第 2 回合切掉實線 7 之後的可能狀態。接下來不管大狸在第 3 回合怎麼選擇，都有可能輸給小狸，因此大狸在第 1 回合切掉虛線 1 的話，可能會輸。

此外，觀察上圖可以發現，如果小狸切完線之後呈現橘色或綠色的狀態，接下來不管如何大狸都有可能輸。

- 大狸第 1 回合切虛線 3



如上圖所示，大狸在第 1 回合切虛線 3 之後，若小狸在第 2 回合切掉實線 6，大狸便有可能輸，因此大狸在第 1 回合切掉虛線 3 的話，可能會輸。

- 大狸第 1 回合切虛線 4



如上圖所示，大狸在第 1 回合切虛線 4 之後，若小狸在第 2 回合切掉實線 6，大狸便有可能輸，因此大狸在第 1 回合切掉虛線 3 的話，可能會輸。

由以上分析可以看出，在大狸每回合都做出最佳決策的前提下，只有大狸在第 1 回合切虛線 2 時，不管小狸怎麼切線，都可以保證大狸獲勝。



資訊科學上的意義

遊戲樹 game tree 又稱競賽樹，是「組合賽局理論」中，用來表達一個賽局中各種後續可能性的樹。一個完整的競賽樹會有一個起始節點，代表賽局中某一個情形，接著下一層的子節點是下一步的各種可能性，依照這規則擴展直到賽局結束。

以切線遊戲這一題來看，每一回合大狸或小狸切下任一條線，都會造成後續不同的發展，因此解答中列出了第 2 回合（也就是小狸的第 1 刀）所有的可能性。這些可能性發展出來，就形成了一棵競賽樹。如果我們假設大狸切下的第 1 刀是 A 選項虛線 1，則又會長出另一棵不同的競賽樹。

競賽樹在人工智慧的應用相當重要，若想贏得某場賽局，利用電腦快速運算的性能畫出競賽樹並且搜尋最佳解，是最有效的做法，例如井字圈叉遊戲及切線遊戲可以利用競賽樹預測賽局未來的發展。然而某些更複雜的比賽例如西洋棋、圍棋等，電腦無法畫出從頭到尾完整的可能性，此時通常會採用部分的遊戲樹 partial game tree 來進行搜尋，例如畫出從現在開始 10 步以內競賽樹，並且從中選擇最佳解來決定接下來的戰略，接著再下 10 步...以此類推直到結束。

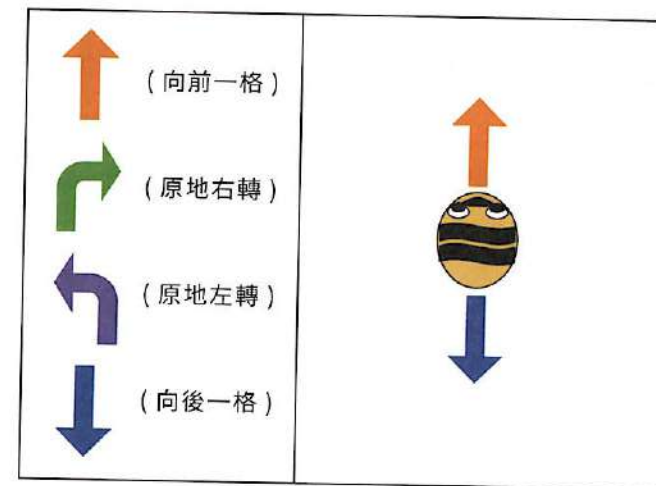


關鍵字

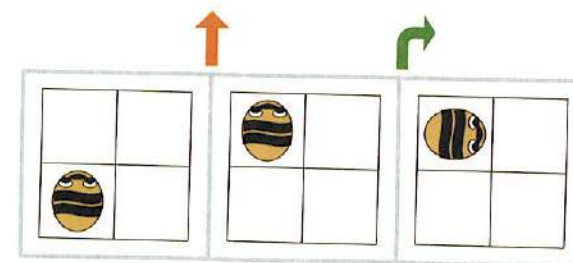
遊戲樹、刪邊遊戲、組合博弈理論、超現實數

39. 蜂巢之旅

有四種指令能控制機器蜜蜂移動位置或改變方向。



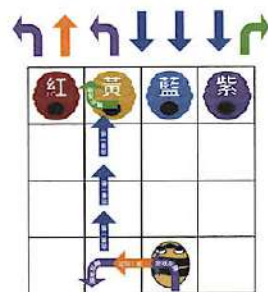
下圖顯示機器蜜蜂收到一組指令（向前一格、原地右轉）後的停留位置及方向。



下列哪組指令能讓機器蜜蜂停留在黃色的蜂巢？

- A. 向前一格, 原地右轉, 向前一格, 原地右轉, 向前一格, 原地右轉, 向前一格, 原地右轉
- B. 原地左轉, 向前一格, 原地左轉, 向前一格, 原地左轉, 向前一格, 原地左轉, 向前一格
- C. 原地右轉, 向前一格, 原地右轉, 向前一格, 原地右轉, 向前一格, 原地右轉, 向前一格
- D. 向前一格, 原地左轉, 向前一格, 原地左轉, 向前一格, 原地左轉, 向前一格, 原地左轉





在撰寫程式時，我們會寫下明確的指令來告訴電腦我們希望它完成的事情。當我們發現程式執行的結果跟預想結果不同時，我們會進行程式 **除錯 debugging**，以追蹤指令的方式將程式碼從頭到尾一步一步執行，驗證每一步的執行結果是否符合我們所設想的結果，從這個過程中找出是哪個指令錯誤。我們在生活中也經常運用到除錯的概念，例如當多位整數加法的計算結果錯誤時，可以利用直式加法逐位相加及進位的過程一步一步驗算，找出發生錯誤的步驟。



40. 石頭篩選機

奇數機器 會把奇數位置的石頭保留並送出，移除掉其他位置的石頭，如下圖所示：



如果把下圖的石頭依序送入這組機器，最後送出的石頭依序為何？



- A. 
- B. 
- C. 
- D. 



正確答案是：B

下圖為一開始輸入石頭的排列順序。



經過奇數機器  後，只留下奇數位置的石頭，排列順序如下圖：



最後經過裁切機器  時，機器上設定為 2，所以將最右邊 2 個及最左邊 2 個石頭去除，剩下的石頭依序如下圖所示：



所以，正確答案為選項 B。

選項 A 是先經過裁切機器  再經過奇數機器  後的結果，故不正確。

選項 C 是先將偶數位置的石頭留下來後，再去掉最右邊 2 個及最左邊 2 個石頭的結果，故不正確。

選項 D 是先將奇數位置的石頭留下來後，再去掉最右邊 3 個及最左邊 3 個石頭的結果，故不正確。

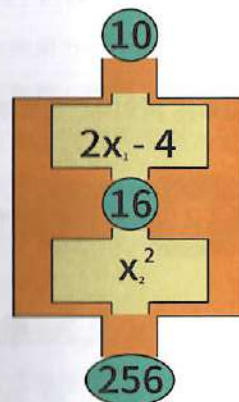


資訊科學上的意義

在程式設計中，**模組化**表示將一串可完成特定功能處理的指令形成一個元件。

函數 function 是程式設計中很常用的一種模組化工具，一組輸入值經過函數依定義的內容運算後即產生一組輸出值。任務中的兩種機器就像是程式中兩個不同的函數，石頭就像是待處理的輸入值，以及處理過後的輸出值。一個函數的輸出值可當作另一個函數的輸入值，再得到另一個輸出值。

舉例來說，若要撰寫程式來求 $x = 10$ 時 $(2x-4)^2$ 的答案，可依運算順序將 $(2x-4)^2$ 分解為兩個函數並依序執行：將 10 輸入第一個函數 $2x-4$ 得到輸出值 16，此值成為第二個函數 x^2 的輸入值，再執行第二個函數後得到輸出值 256 即為答案。在資訊科學中，我們常會把複雜的問題拆解成數個小問題，分別制定解決小問題的函數，藉此找到解決複雜問題的辦法；且函數模組可依輸入的 x 值不同，得到對應的輸出值。

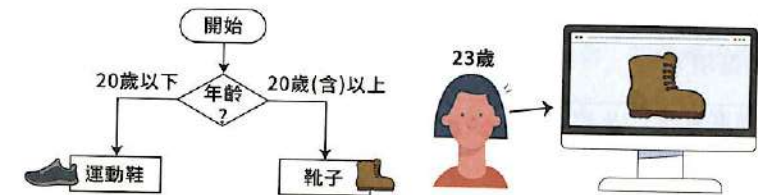


關鍵字

函數

41. 服飾推薦系統

服飾店引進了最新的服飾推薦系統，只要輸入顧客的資訊，它就會依照給定的規則推薦對應的服飾。例如，若一位 23 歲的顧客想要購買鞋子，系統就會依照下圖鞋子推薦的規則，根據顧客符合「年齡」→「20 歲(含)以上」的條件，推薦靴子給這位顧客。



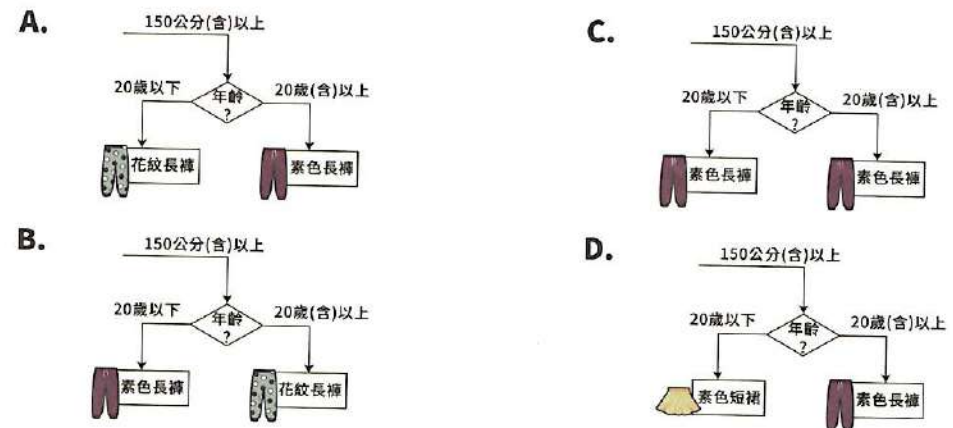
下圖是服飾推薦的規則，但有一部分不小心遺失了。



幸好，服飾店還留有舊的推薦記錄，如下表：

顧客資訊	18 歲 140 公分	32 歲 145 公分	28 歲 155 公分	15 歲 160 公分	10 歲 152 公分
推薦服飾	素色短裙 	花紋短裙 	花紋長褲 	素色長褲 	素色長褲 

請你依照推薦記錄推測，哪個圖可能是遺失的部分規則？





正確答案是：B

從任務中的服飾推薦系統的左半部規則可以看出，第一層採用的條件是顧客的身高，所以可以推斷右半部的規則為「身高 150 公分（含）以上」，且所有符合條件的顧客有三位，其資訊跟推薦服飾如下表。從中可以發現 150 公分（含）以上的顧客都被推薦長褲，所以可排除選項 D。

第二層採用的條件是顧客的年齡，因此可將這三位顧客再區分為「20 歲以下」和「20 歲（含）以上」，發現「20 歲以下」的兩位顧客被推薦素色長褲，「20 歲（含）以上」的顧客被推薦花紋長褲，所以遺失的規則是選項 B。

顧客資訊	28 歲 155 公分	15 歲 160 公分	10 歲 152 公分
推薦服飾	花紋長褲	素色長褲	素色長褲



資訊科學上的意義

此任務中用來判斷推薦系統依據規則的圖稱為 **流程圖 flowchart**，在資訊科學中，流程圖可以用來表示演算法、工作流程或程序的處理過程。流程圖是由特定的形狀和箭頭組成，各種形狀代表不同的意義，例如 代表條件判斷。在本任務表示服飾推薦規則的流程圖中，由兩層的條件判斷組合而成，第一個條件判斷後又有另一個條件判斷後才決定如何處理，這種敘述稱為 **巢狀條件敘述 nested conditionals**。

在生活中有許多巢狀條件敘述的情況，例如：要決定當天的衣著，可能要先考慮天氣，再考慮喜好；或是在判定體重是否符合標準時，要先根據性別，再依照 BMI 值的範圍來判定。



關鍵字

巢狀條件敘述、流程圖

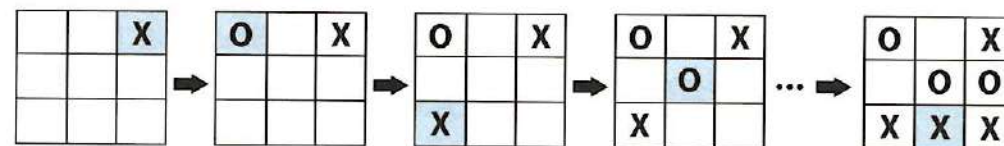


42. 井字遊戲

井字遊戲是兩位玩家的紙筆遊戲，其規則為：

二位玩家分別以 O 與 X 為代表標記，輪流填入 3×3 的井字棋格中。當某玩家的代表標記於棋格連線成功時（含水平、垂直或對角線），該玩家獲勝、遊戲隨即結束；若所有棋格都被填滿卻無玩家連線成功時，遊戲以平手結束。

下列各圖為某局遊戲過程，藍底顯示該次填入標記的棋格：



在最右側的圖是此局遊戲結束的棋格狀態，由 X 標記玩家獲勝。

以下哪一張圖是符合規則的遊戲結束棋格狀態呢？

- A.

X	O	X
O	X	O
O	O	X
- B.

X	O	X
O	X	
O	X	X
- C.

X	X	O
	O	X
O	O	X
- D.

X	O	X
O	X	O
O	X	



正確答案是：C

從井字遊戲的遊戲規則中，可以歸納出結束的棋格狀態具有以下特性：

1. O 標記和 X 標記的數量之間只會相差 0 或 1。
2. 如果一場遊戲一直沒有玩家獲勝，就會持續直到填滿所有棋格為止；意即以平手收場。
3. 一個遊戲結束的棋格狀態，只會有唯一一個獲勝順序。

選項 A 不正確，玩家 O 不可能會有 5 個標記。假設玩家 X 先下，在第 4 步時連成一線並停止遊戲，此時玩家 O 只會有 3 個標記；假設玩家 O 先下，玩家 X 在第 4 步時連成一線並停止遊戲，此時玩家 O 也只會 4 個標記。

選項 B 不正確。因為有 5 個 X 標記但只有 3 個 O 標記，兩者數量相差 2，所以不可能發生。

選項 C 是正確的，當玩家 X 先開始，在玩家 O 連成一線獲勝後遊戲立即停止，並留下 1 個空棋格。

選項 D 不正確，因為尚未有玩家獲勝，且棋格未填滿，所以不是遊戲結束的棋格狀態。



資訊科學上的意義

有效性檢查 validity check 是指確認資料沒有錯誤，而且符合所有標準或規範，其為處理資料時的重要步驟，可以確保資料的強健及一致性。

在進行有效性檢查時，規則是非常重要的一環。例如此任務描述中，要符合上述的三個特性，才是遊戲結束的合理棋格狀態；透過檢驗每個棋格狀態是否符合這三個特性，就能確認該棋格狀態是否為「有效的遊戲結束棋格狀態」。

在資訊科技中，最基本的有效性檢查常應用於檢驗使用者輸入的資料格式是否正確；例如不同的圖片檔案格式有不同的規則定義，當我們透過圖片編輯程式打開圖檔時，編輯程式就會先做有效性檢查，若檔案格式符合規則，才能讀取檔案進行圖片編輯。

日常生活中，也常常用到有效性檢查的概念，例如我們的身分證號、銀行帳號中都有檢核碼，透過特定的計算規則可以確認檢核碼是否正確，進而檢核身分證號或銀行帳號是否合理。



關鍵字

有效性檢查

43. 螺帽和螺栓

在海狸建材工廠中，小強負責螺帽和螺栓的組裝。



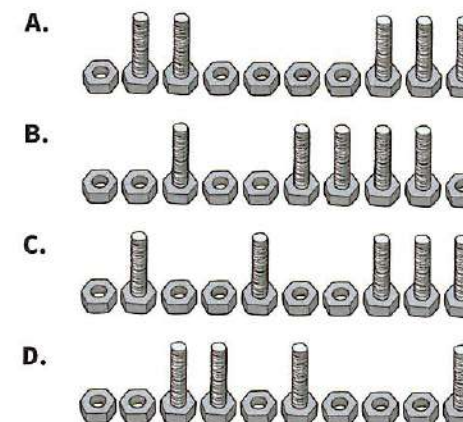
小強站在輸送帶的末端，輸送帶會送來螺帽或螺栓，他的工作內容如下：

- 把輸送帶送來的螺帽或螺栓拿起來。
- 如果拿到的是螺帽，就把螺帽放進桶子中。
- 如果拿到的是螺栓，就從桶中拿起一個螺帽，將螺栓跟螺帽組合在一起後放到旁邊的箱子上。

然而，小強可能遇到下列兩種無法順利處理的狀況：

1. 當拿到螺栓時，桶子中沒有螺帽讓他組裝。
2. 輸送帶上已經沒有螺栓或螺帽了，但桶子中還有螺帽。

下列從左到右是輸送帶上螺栓、螺帽的排列順序，請問哪個選項小強可以順利處理？





正確答案是：C

小強處理選項 C 的流程如下表，表中顯示小強每拿起一個螺帽及螺栓處理後，桶子和輸送帶中螺帽及螺栓出現的狀態變化。當輸送帶空了，桶子中的螺帽也刚好用完。

桶子	輸送帶
空的	
空的	
空的	空的

讓我們看看其他選項：

- 選項 A 的排列順序：
在序列中出現第 2 個 時就無法順利處理，因為桶子中沒有螺帽可用。

- 選項 B 的排列順序：

在序列中出現第 5 個 時會無法順利處理，因為桶中已沒有螺帽可用。可以注意到，這是在第 5 個 之前，輸送帶上只出現過 4 個 。

- 選項 D 的排列順序：

當輸送帶空了，組裝過程結束後，才會發現剩兩個螺帽在桶中的問題。這是因為這個序列總共出現 6 個 和 4 個 。



資訊科學上的意義

編譯器 **compiler** 是一種電腦程式，它會將某種程式語言寫成的原始碼（原始語言）轉換成另一種程式語言（目標語言）。它主要的目的是將便於人編寫、閱讀、及維護的高階電腦語言程式碼，翻譯為電腦能解讀並執行的低階機器語言程式碼，也就是執行檔。

括號匹配 **bracket matching** 問題是符號匹配任務中的一種例子，就是要對輸入的一串括號進行左右括號一組一組的對應，當每個左括號都可以找到對應的右括號，就輸出「正確」，否則輸出「錯誤」。完整成對的括號在程式語言中是表達程式區段執行控制的主要方式，因此偵測括號是否成對在編譯器中是基本且重要的。括號匹配的處理過程需要一個容器來暫存輸入的左括號，然後讓它們匹配輸入的右括號，必須遵循先前最後出現的左括號匹配最近出現的右括號。所以可以運用堆疊 **stack** 容器後進先出的特性來存放左括號，輸入時遇到左括號先放進堆疊，當遇到右括號時取出先前出現最近的左括號（即容器頂端的左括號）進行匹配，若可取出一個左括號匹配則繼續進行，若堆疊已經空了無法取出一個左括號匹配則輸出「錯誤」。若全部括號已輸入結束但堆疊中仍有左括號，亦要輸出「錯誤」。

在本題中，在輸送帶末端取到的螺栓或螺帽為「輸入」，裝螺帽的桶子對應到「容器」。我們可以將螺栓與螺帽對應到一組左右括號，例如「 $((((($)」和「 $((((($)」的序列就不是完整成對的括號。在數學四則運算中也經常用到括號來對應表示計算式中加減乘除的先後順序，例如： $(5-3) * (4-2) -4$ 有成對的左右括號，但 $(5-3) * ((4-2) -4$ 中第 2 個左括號便無法正確配對到右括號，因此是錯誤的計算式。



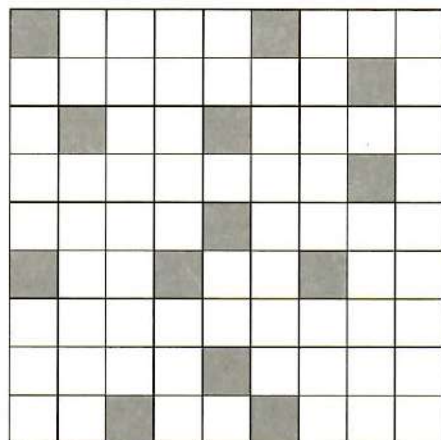
關鍵字

括號匹配、編譯器



44. 無人機路徑 - 題組一

一臺無人機以下方黑白棋盤的某個白色格子當做起點，機頭朝向上下左右任一方向。



接著它開始以下列方式巡航經過棋盤上其他 8 個白色格子：

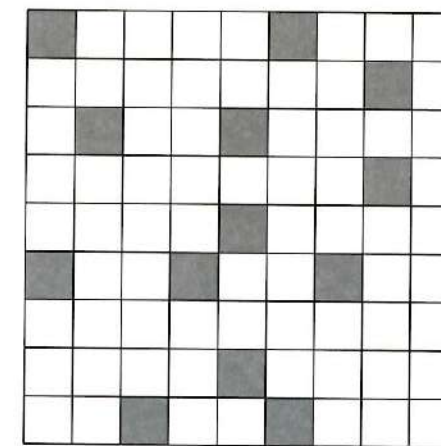
1. 往前移動 2 格
2. 在同一格內 左轉 90 度
3. 向前移動 4 格
4. 在同一格內 右轉 90 度
5. 再往前移動 2 格

棋盤上有幾個可做為起點的白色格子？（範圍為 [1~68] 的整數）



45. 無人機路徑 - 題組二

一臺無人機以下方黑白棋盤的某個白色格子當做起點，機頭朝向上下左右任一方向。



接著它開始以下列方式巡航經過棋盤上其他 8 個白色格子：

1. 往前移動 2 格
2. 在同一格內 左轉 90 度
3. 向前移動 4 格
4. 在同一格內 右轉 90 度
5. 再往前移動 2 格

棋盤上有幾個可做為起點的白色格子？

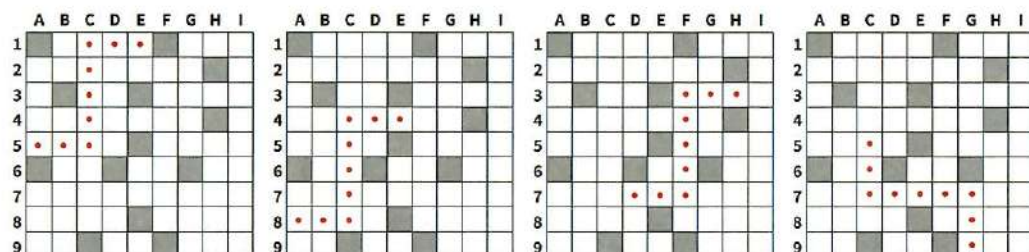
- A. 4 個
- B. 6 個
- C. 8 個
- D. 10 個



44. 正確答案是：8

45. 正確答案是：C. 8 個

棋盤上符合任務的移動方式，只會經過白色格子的路徑，共有以下 4 條：



要找出這些答案，最簡單的方式是先找出 5 個連續的白色格子，然後嘗試在兩個端點以分別以相反的垂直方向各延伸 2 個白色格子。為了確保答案正確，必須以系統性的方式來操作這個過程，例如：從左上方一路嘗試到右下方的格子，直向、橫向都要找一遍。

另外，本題中無人機經過的路徑具有對稱性，所以上面 4 條路徑的兩端都可當成起點，因此共有 8 個相異的起點。



資訊科學上的意義

本題為固定影像（或稱模板）旋轉匹配任務的簡單範例，這是資訊科學領域裡 **影像處理 image processing** 的其中一項工作。題目中定義了欲搜尋的模板（步驟 1 - 5），而整個棋盤就是要從中比對模板的影像。像是「威利在哪裡」這系列的繪本中，我們得在一張張不同場景的圖畫裡尋找主角「威利」的身影一樣，在一張包含許多內容的圖片裡尋找我們感興趣的圖樣 Pattern 即稱為 **模板匹配 template matching**，它有許多應用，在實際生活上也會用在像是人臉辨識以及醫學影像處理上。在一些模板匹配的方法中，為了能夠處理不同大小的影像，也常會以一種向量表示的方式來描述要搜尋的樣式（模板）。



關鍵字

影像處理、模板匹配