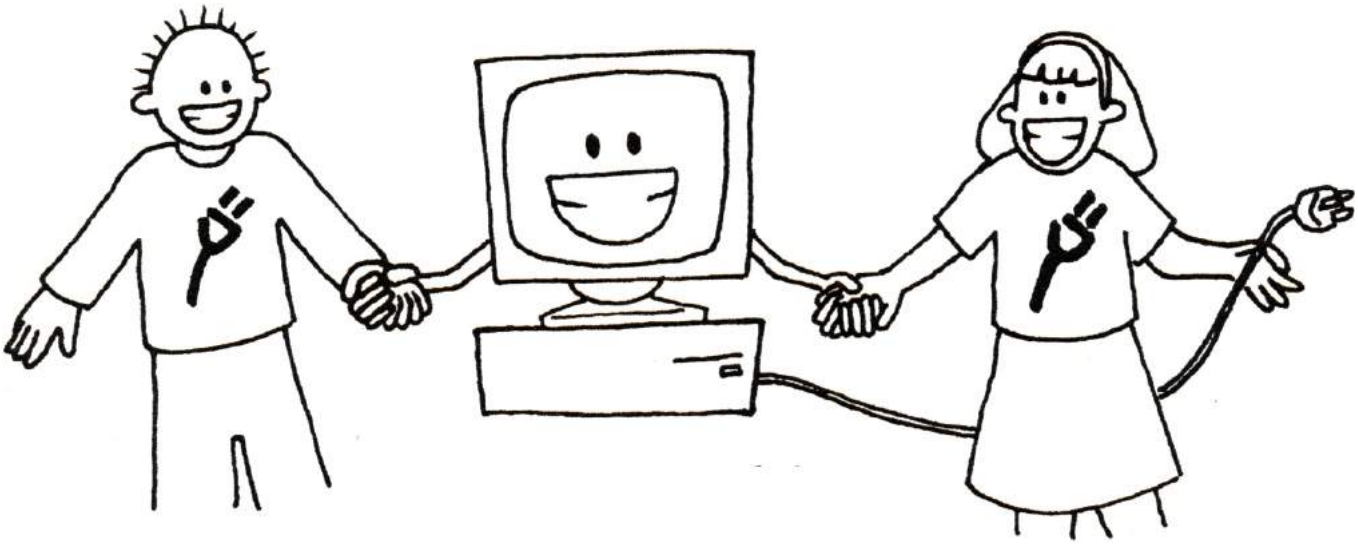
CS UNPLUGGED



不插電的資訊科學

原著
Tim Bell, Ian H. Witten, Mike Fellows

活動設計
Robyn Adams, Jane McKenzie

插畫
Matt Powell

2015 年修訂版
Sam Jarman

中華民國軟體自由協會
編譯

本書介紹

電腦無所不在。我們都要學習如何使用，而我們之中一定有人天天使用電腦。不過，電腦是怎麼運作的呢？又是怎麼思考的？還有，人們是如何寫出快速且容易使用的軟體呢？資訊科學是一個很吸引人的學科，可以探索上述這些問題。這本書設計給各年齡層的學生，利用簡單有趣的活動來介紹電腦的運作 -- 但完全不透過電腦！

這本書可以有效率的運用在充實與拓展課程內容，即使在一般普通的教室裡也沒問題。老師們不需要是電腦專家，也可以跟學生一起學習這些主題。這本書包含一系列的活動、簡單的背景資料說明、提供每個問題的解答，並且在每一個活動的結尾都有「這個活動在說什麼？」講述此活動的相關知識。

很多活動以數學為基礎，例如探索二進位數字、地圖和圖表、樣式和排序問題，以及密碼學。其他的部份與相關技術課程、瞭解電腦如何使用的知識等都有很好的銜接。在精心設計的內容中，學生可以積極的參與討論，學習解決問題、創造力以及思維的技能。這些活動也提供一個非常吸引人的方法，來探索所謂的「運算思維」，這在現代教育中已經是不可或缺的主題。

除了這本書之外，「不插電」的計畫還有很多免費線上資源。在 csunplugged.org 中可以找到很多影片、照片以及額外的補充等等。

隨著 2015年校正了這本書，我們也建立一個新網站，有著更多的資源、更好取得開源的素材，以及更強的資訊科學與運算思維的課程銜接。

這本書的作者是三位資訊科學的講師，以及兩位學校老師，以我們現場授課經驗，還有二十幾年間數百名老師的回饋所撰寫而成。我們發現許多重要的觀念可以不用透過電腦來學習。事實上，有時電腦是學習分心的主因。通常資訊科學的教學多半從程式開始，不過不是所有學生都能夠一下就上手，進而變成瞭解資訊科學有趣的部分的一大障礙。所以拔掉電腦的插頭，讓我們來學習資訊科學這門有趣的學問吧！

我們鼓勵大家在教育方面使用這本書，也歡迎你列印下來分送給學生。有任何問題與建議要反應給作者，請到 csunplugged.org。

這本書被翻譯成許多不同的語言。請到網站上找找相關譯本。

謝誌

許多孩子與老師們幫助我們的想法更完善。在 South Park School（英國維多利亞地區）、Shirley 小學 , Ilam 小學以及 Westburn 小學（紐西蘭基督城）的老師與學生們試過這些活動 。我們尤其感謝 Linda Picciotto, Karen Able, Bryon Porteous, Paul Cathro, Tracy Harrold, Simone Tanoa, Lorraine Woodfield, 和 Lynn Atkinson 讓我們到他們的教室參觀，並提供活動的建議及精緻化。

Gwenda Bensemann 已試作一些活動，並提供建議修改。Richard Lynders 與 Sumant Murugesh 協助課堂的試驗。部分的加密活動是由 Ken Noblitz 所開發。有些活動是在維多利亞 “Mathmania” 集團的協助下進行，並得到 Kathy Beveridge 的大力幫忙。較早版本的插圖是由 Malcolm Robinson 與 Gail Williams 所繪製，我們也受益於 Hans Knutson 的建議。Matt Powell 在「不插電」計畫中提供許多實質的協助。我們非常感謝 Brian Mason Scientific and Technical Trust 在此書發展的早期提供大手筆的贊助。

特別感謝 Paul and Ruth Ellen Howard 測試了許多活動，並提供大量有幫助的建議。Peter Henderson, Bruce McKenzie, Joan Mitchell, Nancy Walker-Mitchell, Gwen Stark, Tony Smith, Tim A. H. Bell[1] , Mike Hallett, 和 Harold Thimbleby 也提供許多有用的評論。

當然，我們也欠了我們家人很多：感謝 Bruce, Fran, Grant, Judith 和 Pam 的支持，以及 Andrew, Anna, Hannah, Max, Michael[2] 和 Nikki 給我們許多啟發。他們也經常是第一個接受測試的孩子們。

我們特別感謝 Google 公司資助不插電計畫，以及讓這本書能免費下載。

我們歡迎有關活動的評論以及建議。若要聯繫作者可透過 csunplugged.org。

[1] 與本書作者只是同名同姓而已。

[2] 事實上，文字壓縮的活動是由 Michael 發明的。

謝誌（正體中文版）

這本書的正體中文版，從起心動念要翻譯到完成電子書，僅僅四個多月的時間。這要感謝許多人的努力奉獻，一起合作在很短的時間內完成包括翻譯、校稿、編排等工作，呈現在大家的眼前。

首先要感謝成功大學資訊工程系的楊中平教授與吳鳳科技大學資訊管理系的邱垂鎮教授，在得知筆者有翻譯此書的想法時伸出援手，分別指導學生進行翻譯、排版等工作。

再來要感謝所有參與翻譯、協助排版、提供資訊的同學們。這些同學包括了選修成功大學資訊工程系楊中平教授所開的的「自由軟體開發與社群發展」課程的同學們，還有邱垂鎮教授所指導的學生，用上課時間與期末專題的方式讓這本書能順利呈現。所有參與的同學名單包括：

參與翻譯者：

王晉鋒	王郁強	王昶鈞	王韋傑	王冠鈞	平震傑	方鈞麒
毛詩堯	石家瑋	李泓哲	李偉綸	李俊德	呂伯駿	何昕叡
林彥亨	林揚勳	林偉哲	林京樺	周子軒	邱子恆	青光恆
柯欣彤	徐佳筠	翁瑞陽	翁介誠	唐鼎鈞	陳劭傑	陳侑廷
陳昀聖	陳川儒	陳彥安	陳亭翰	張矽晶	張友誠	張瑜真
許家偉	黃承鴻	黃暐程	黃勇霖	郭子瑋	曹雅晴	辜玉雯
曾柏瑄	傅曲誠	雷承勳	蔡孟軒	蔡婷安	廖家妤	鄭宇傑
鄭宇呈	鄭宇昇	鄭耀主	鄭雅桓	劉皓平	賴建江	謝宗諭
謝孟儒	顏志倫	蕭丞志	蘇致翰	Angelina Anggara		

提供排版軟體資訊：

林偉哲　石家瑋　賴建江

協助本書排版：

唐鼎鈞　徐佳筠　陳昀聖　曾柏瑄　翁鈺瑋

翻譯前一版本、提供資訊常用術語資訊與其他協助：

洪婉瑄　張茹雯　程宛靚

校稿、插圖中文化：

丁國傑　邱彥銘　翁佳驥　Jason Tsai

另外感謝建國中學林冠廷同學提供紙本書發行之相關經驗與資訊。本書電子版將遵照原著所使用之 CC-BY-NC-SA 授權，置於官方網站 csunplugged.org 上供大家下載。紙本書將由中華民國軟體自由協會委託廠商發行。盈餘除一部份回饋於原著團隊與協助發行之廠商以外，其他均由協會用於贊助台灣自由軟體社群之活動。盈餘分配辦法將會公告於協會網站與 FB 粉絲頁中。

第一部份

資料：最原始的材料 — 如何表達資訊

資料：最原始的材料

在電腦中如何儲存資訊？

「電腦」這個字的英文是 computer，是從拉丁文中的 “computare” 演變而來，表示計算或把數字加起來。不過今天的電腦已經不只是一個巨型的計算機了。它們可以是一間圖書館，幫我們寫文件或找資料，播放音樂或電影等等。那麼電腦是如何儲存這些資訊的呢？信不信由你，其實電腦只用了兩個東西：0 跟 1 ！

「資料」與「資訊」的差別在哪裡？

「資料」只是原始的材料，也就是電腦處理的數字。電腦可以將資料轉換為你跟我可以懂得的「資訊」（例如文字、數字與圖片等）

那麼數字、字母、文字與圖片等要怎麼轉換為 0 跟 1 呢？

在這幾個活動中，我們會學到關於二進位數的知識、電腦如何畫圖、傳真機如何運作等等。這些都是儲存一大堆資料很有效的方法。此外還有我們如何避免錯誤的發生，以及我們如何測量儲存的資訊量。

活動 1
計算點點 – 二進位數字

活動摘要

在電腦上的資料是以一連串的 0 和 1 的形式儲存與傳送的。我們要如何只使用這兩個符號來表示文字和數字呢？

課程銜接

• 數學：數字 - 研究如何將數字用其他基底表示。學習用 2 做基底表示數字，也就是二進位數。
• 數學：代數 - 繼續一序列的排列，並描述這個排列的規則，還有此排列與 2 的次方之間的關係。

習得技能

• 計數
• 比對
• 序列

適合年齡

• 6 歲以上

所需素材

• 你需要使用一組五張卡片（見第 7 頁）來展示。
• 你可以做成 A4 大小，上面貼笑臉貼紙點點的卡片。效果不錯。

每個學生需要：

• 一套卡片。您可以複印第 9 頁：「素材：二進位數字」的內容，貼在卡片上，然後割下來。
• 活動學習單：二進位數字（第 8 頁）

另外有一些選擇性的延伸活動。這些延伸活動中每個學生需要：

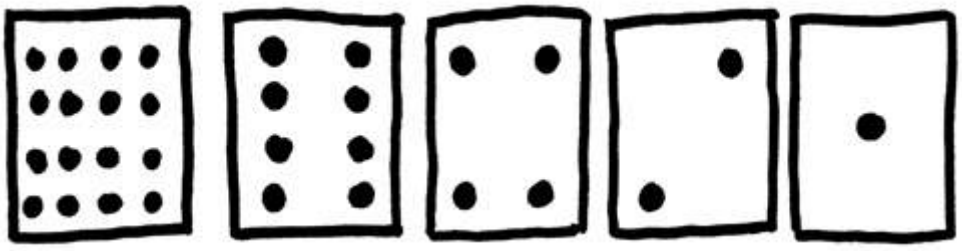
• 活動學習單：使用二進位表示數字（第 10 頁）
• 活動學習單：發送秘密訊息（第 11 頁）
• 活動學習單：電子郵件與數據機（第 12 頁）
• 活動學習單：數到 31 以上（第 13 頁）
• 活動學習單：更多關於二進位數字的知識（第 14 頁）

二進位數字

活動介紹

在發出第 7 頁的活動單之前，可以先展示一次給全班看。

在此活動中，您將需要一套五張卡片，如下圖所示。卡片其中一面有點點，而另一面沒有。選擇 5 個學生拿起示範卡片，站在全班面前。卡片應該要照下面的順序排列：



活動討論

當您在發卡片時（從右到左），看看學生們能不能猜出下一張卡片上有多少個點。請大家注意一下，卡片上的點點數量有什麼特性？（每張卡上的點點數量都是右邊卡片的兩倍。）

如果再加一張牌放在左邊，那麼那張卡片應該要有多少點點？（32）再下一張呢？（64）

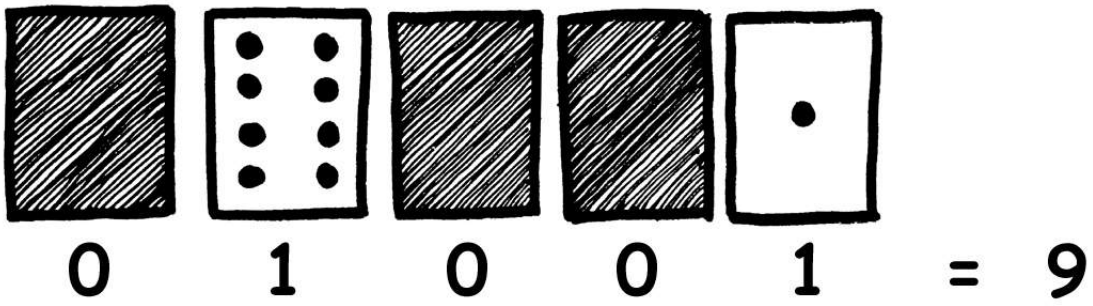
我們可以用這些卡片表示許多不同的數字，只要把一些卡片翻面蓋起來，並把所有沒蓋住的點數加起來。請學生試著展示 6 點（翻開 4 點的和 2 點的卡片），再來試試怎麼表示 15 點（翻開 8 點、4 點，2 點和 1 點的卡片），然後再試試怎麼表示 21 點（翻開 16，4 跟 1點的卡片）.....等等。唯一要注意的規則是，卡片要嘛就要全部秀出來，不然就要全部蓋起來，不能只蓋一半。

這五張卡片能表示最小的數目是多少？（學生們可能會回答 1，但正確答案是 0）。

現在試著從零開始往上數。

這堂課中剩下的部份，就是要請學生們密切注意卡片變化的情形，並確認學生們是否在翻牌與蓋牌的過程中可以發現特定的規則（每張卡片被翻轉的次數，是右邊那張牌的一半）。您可以在不同的班級中嘗試看看。

當某一張卡片被蓋起來時，用 0 來表示。當卡片被秀出來時，用 1 來表示。這就是二進位數。



讓學生們試試 01001。01001 的十進位是什麼數字？（9）17 的二進位數是什麼？（10001）

多試幾次，直到學生們能理解這個概念為止。

另外有五個選擇性的延伸活動，可以用於加強學生的觀念。在時間允許的範圍內盡量讓學生去進行。

活動學習單：二進位數字

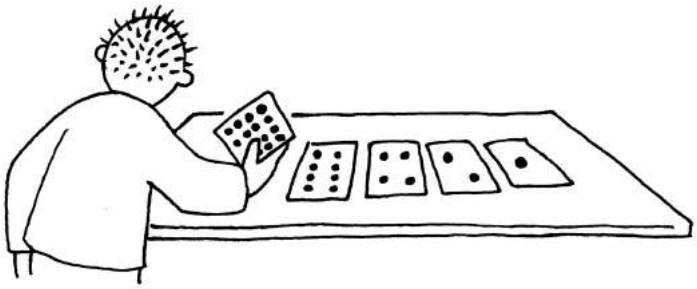
學會數數字

你認為你會數數字嗎？讓我們試試新的方法吧！

你知道電腦中只用了 0 跟 1 嗎？其實任何你從電腦中看到或聽到的東西－包括文字、圖片、數字、影片甚至聲音都是由 0 跟 1 組合而成的！接下來的活動內容將教你如何使用跟電腦一樣的方法傳送一段秘密訊息給你的朋友。

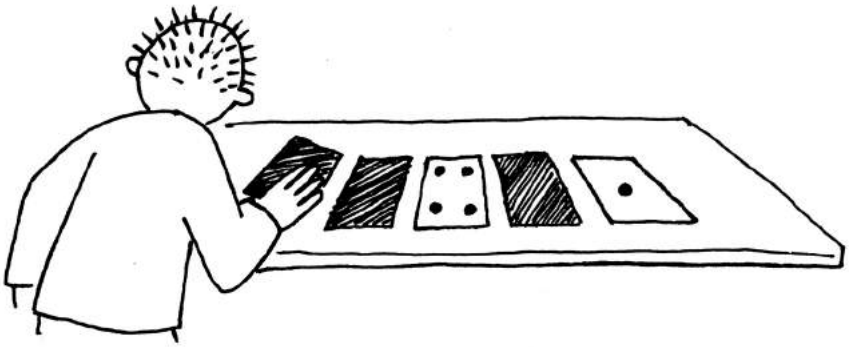
步驟說明

從你的活動單上剪下卡片，然後把它們照順序放在桌上。點數為 16 的卡片必須放在最左邊，如圖所示：



確認一下你排放的順序要跟圖片中的一樣

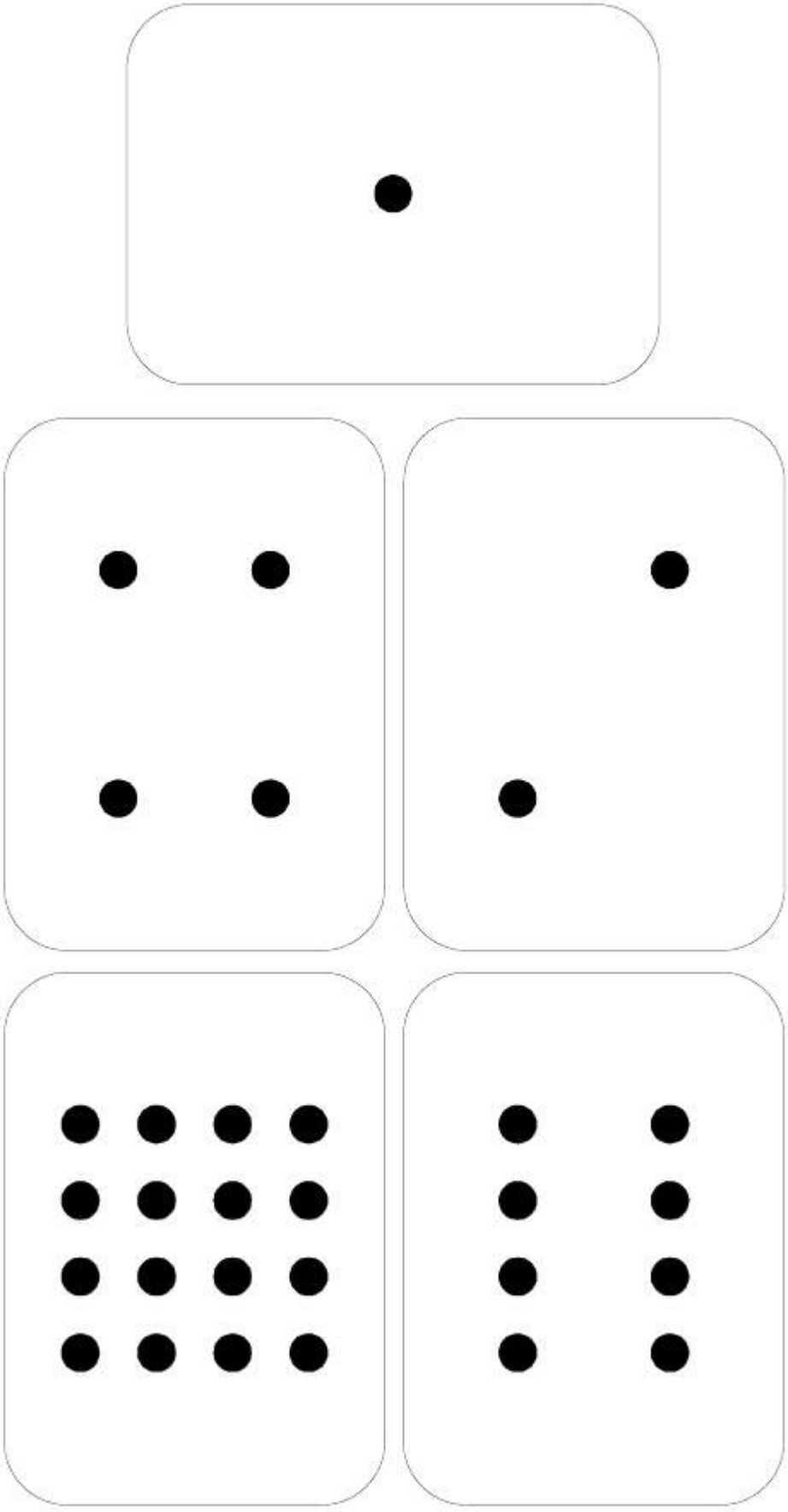
現在把一些卡片翻過去背面使得剩下的點數總和為 5－注意不可以讓卡片的順序改變哦！



接著去試試看怎麼樣讓剩下的點數總和為 3、12、19。是不是有其他的方法可以表達相同的數字呢？你所能看到，最大的點數總和為多少呢？最小的點數總和又是多少呢？在最大和最小的點數之中，有沒有任何的數字是你無法用這些點數的總和表達的呢？

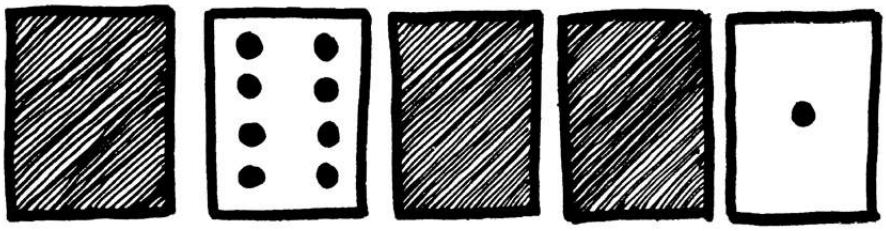
高手挑戰：試著按照順序用卡片表達 1、2、3、4。
你能不能找出一個有規律的方法可以讓數字一直加 1 呢？

素材：二進位數字



活動學習單：使用二進位表示數字

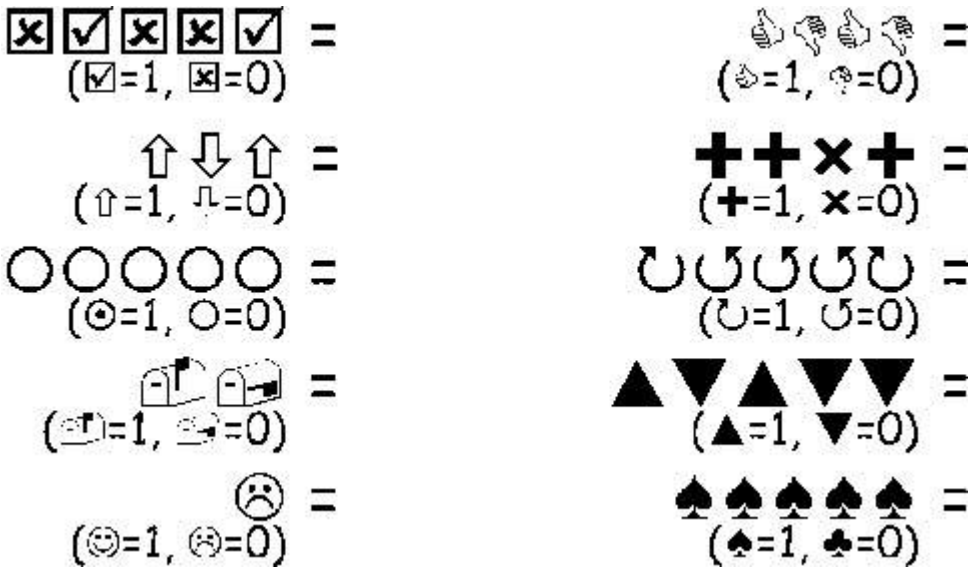
二進位數字就是用 0 或 1 來表示點數卡片是點數面朝上或朝下。 0 代表點數面朝下， 1 代表點數面朝上。例如：



你可以算出 10101 是多少個點嗎？11111 呢？

你的出生日期是幾號呢？試著用二進位數字表達。然後試著使用二進位數字來表達你的朋友的生日。

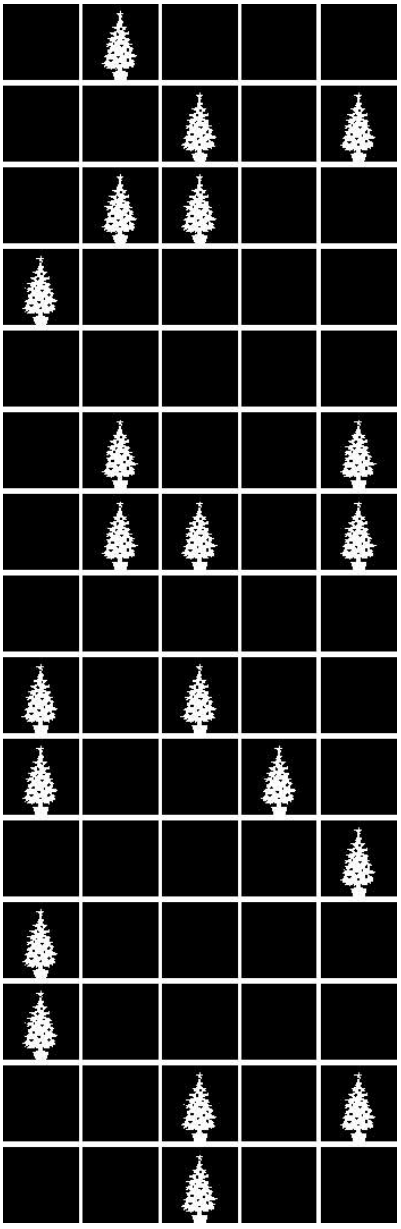
試著用二進位數字表達這些數字：



高手挑戰：使用長度分別為 1、2、4、8、16 的棍子來組合出長度為 0 到 31 的棍子。你也可以利用一個天平搭配少量的砝碼來秤出很重的東西（例如箱子或行李箱）。回去試試看，讓大人們驚訝一下！

活動學習單：傳送秘密訊息

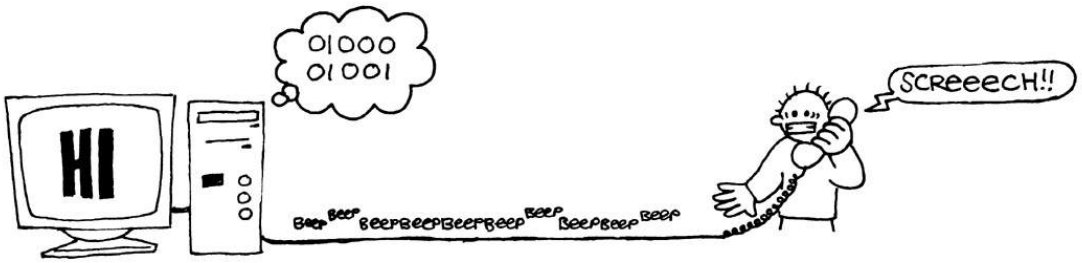
湯姆被困在百貨公司的最上層了！聖誕節就快要到了，他想帶著他的禮物回家。 他該怎麼辦？湯姆試過大聲求救，但是附近根本沒有人。不過他可以看到對街有一些電腦工程師深夜裡還在工作。他該怎麼引起他們的注意呢？湯姆看了一下他的四周，然後靈光一現 -- 他可以用聖誕樹上面的燈傳送訊息給對街正在工作的女士！他找到了所有的燈泡，然後插上電源，這樣他就可以開關這些燈泡。他擬了一則簡單的二進位代碼，確定對街的女士可以懂他的意思。你能幫助他嗎？



1	2	3	4	5	6	7	8	9	10	11	12	13
a	b	c	d	e	f	g	h	i	j	k	l	m
14	15	16	17	18	19	20	21	22	23	24	25	26
n	o	p	q	r	s	t	u	v	w	x	y	z

活動學習單：電子郵件與數據機

電腦也是使用二進位系統跟網路連線與傳送訊息。唯一的不同是，電腦用的是嗶聲來表示。高頻率的嗶聲代表 1，低頻率的嗶聲代表 0。這些聲音的傳遞非常快速 — 事實上，我們所能聽到的只有連續又惱人的噪音而已。如果你從來都沒有聽過，下次可以試著聽看看數據機跟網路連線的聲音。或者，試試看打個電話到一支傳真機 — 傳真機也是透過數據機來傳遞訊息。



利用跟湯姆在百貨公司使用的一樣的代碼，試著傳一封電子郵件給你的朋友。記得要用簡單一點的訊息，因為你跟你的朋友可沒辦法像數據機那樣處理得那麼快！

活動學習單：數到 31 以上

看看你做的二進位數卡片。如果你要再做下一張卡片，那麼上面應該要有幾個點呢？再下一張呢？他們之間的規則是什麼？如你所看到的，即使是很大的數字，也只需要少少的幾張卡片而已。

如果你細心的觀察這些規則，你可發現非常有趣的規律：

1, 2, 4, 8, 16, ...

試試 1 + 2 + 4 = ? 答案是什麼？

再試試 1 + 2 + 4 + 8 = ?

當你從第一個數字開始，把所有的數字加起來，你發現了什麼？

學會了二進位數字，你現在可以用你的手指來數數字，而且你會發現，你能數的數字遠遠的超過 10 — 等等，你說你只有十根手指頭？沒錯！你可以使用二進位來表示數字，用手指頭來代替每一張卡片。你不用變成外星人一樣，就可以從 0 數到 31 了。這樣就有 32 個數字了 (別忘了，0 也是一個數字喔！)

試著動動你的手指頭，手指頭向上表示 1，向下表示 0。

實際上，若你用兩隻手，可以表示從 0 ～ 1023 的每一個數字，這樣總共就有 1024 個數字了！如果你跟有可以彎曲成很多段的手指頭（ㄟ，這樣你就真的得變成外星人了！），你就可以數更多的數字。如果一隻手可以表示 32個數字，兩隻手能就有 32 x 32 = 1024 個數字。那麼請問圖上的外星人可以數的最大數字是多少呢？

活動學習單：更多關於二進位數字的知識

1. 另一個關於二進位有趣的性質，是當我們把一個 0 放到一個二進位數字的右邊的時候。想一下，過去我們在學十進位數字時，當我們把一個 0 放到一個數字右邊，就等於把那個數字乘上 10。舉例來說，9 會變成 90，而 30 會變成 300.

但在二進位數字的世界中，當我們把 0 放到一個二進位數字的右邊會發生什麼事呢？試試看：

1001　　10010

(9)　　(?)

多試幾個其他數字來證明你的猜想。規則是什麼？你認為發生了什麼事？

2. 我們目前用的每一張卡片都可以代表電腦裡面的一個「位元」（「位元」的英文 bit 就是「二進位數字」英文 binary digit 的簡稱）。所以目前我們用的字母可以用 5 張卡片來表示，或是說 5 個「位元」。然而電腦必須知道這個字母是大寫還是小寫，還是數字、標點符號或是像 $ 或 ~ 這種特殊符號 。

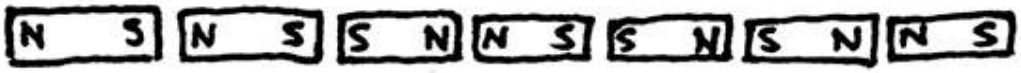
看看你的鍵盤，數數看電腦總共有幾個字元要表示。算算看電腦總共需要幾個位元來儲存所有的字元呢？

現代的電腦都使用「美國資訊交換標準代碼」（American Standard Code for Information Interchange，又稱為 ASCII 碼）來表示每一個字元。但在一些不說英文的國家則必須使用更長的編碼。

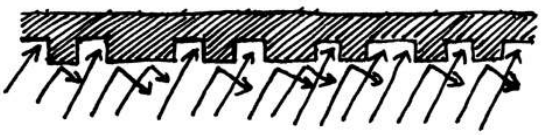
這個活動在說什麼？

現在的電腦使用二進位系統來表示資訊。之所以被稱為二進位是因為它只用兩個數字。這種二進位數又稱為「以 2 為基底」的數字系統（我們一般使用的是「以 10 為基底」的數字系統）。每個 0 或 1 被稱為一個「位元」。一個位元在電腦的主記憶體中通常用一個電晶體的開或關，或是用一個電容充電或沒充電來表示。

當我們要利用電話線或是無線電來傳輸資料時，會利用高頻或低頻的聲音來表示 1 和 0。在磁碟（軟碟和硬碟）還有磁帶上，則是會利用表面的磁場方向來表示 0 和 1 ，像是北往南或南往北。



音樂 CD、光碟片和 DVD 則是利用光學方式來儲存位元 -- 利用表面是否反射光線來表示。



電腦之所以只用兩個不同的數值，是因為這樣一來，要做一個裝置就簡單多了。我們當然可以做一個擁有 10 種不同反射光的 CD 來表示 0 到 9 的數字，但是這樣必須使用很昂貴以及精確的裝置才行。

除此之外你也許也注意到另外一件事情：雖然我們說電腦只存 0 和 1，但其實並不是真的有 0 和 1 在電腦裡面 -- 只有像是高電位和低電位、南極和北極等等。但比起說是「閃」還是「不閃」，寫 0 和 1 還是快多了。所有在電腦裡的東西都是用位元表示 -- 文件、圖片、歌曲、影片、數字或甚至程式或手機上的 app，全部都只是一大堆的位元。

一個位元自己不能表達出太多的東西。但當八個位元聚集在一起，就可以表達從 0 到 255 的數字。而我們通常會把 8 個位元稱作一個「位元組」(byte)。

電腦的速度取決於一次可以處理幾個位元。舉例來說，一個 32 位元的電腦在一次操作中可以處理 32 個位元；但 16 位元電腦則必須將 32 位元的數字分成兩個 16 位元，因此速度慢多了（但比較便宜！）

在接下來的活動之中，我們將會看到電腦怎麼用位元來表達其他種類的資訊。

解答和提示

二進位數字（第 8 頁）

3 需要 卡片 2 和 1
12 需要 卡片 8 和 4
19 需要卡片 16、2 和 1

任何一個數字都只有一種方式來表達。

你可以表示出最大的數字是 31 ，最小則是 0。你可以將 0-31 間的所有數字都表達出來，而且每個數字只有一種表達方式。

高手挑戰：要把任何一個數字加 1，只要從右到左翻卡片，直到有一張卡片從點數向下翻成向上。

使用二進位表示數字（第 10 頁）

10101 = 21, 11111 = 31

傳送秘密訊息（第 11 頁）

訊息內容：HELP IM TRAPPED

數到 31 以上（第 13 頁）

如果你從頭開始累加，累加的和必定是數列裡下一個數字減一。

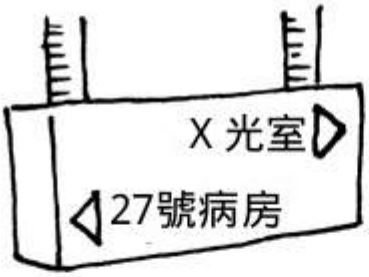
外星人可以數 1024 * 1024 = 1048576 個數字 — 也就是從 0 到 1048575。

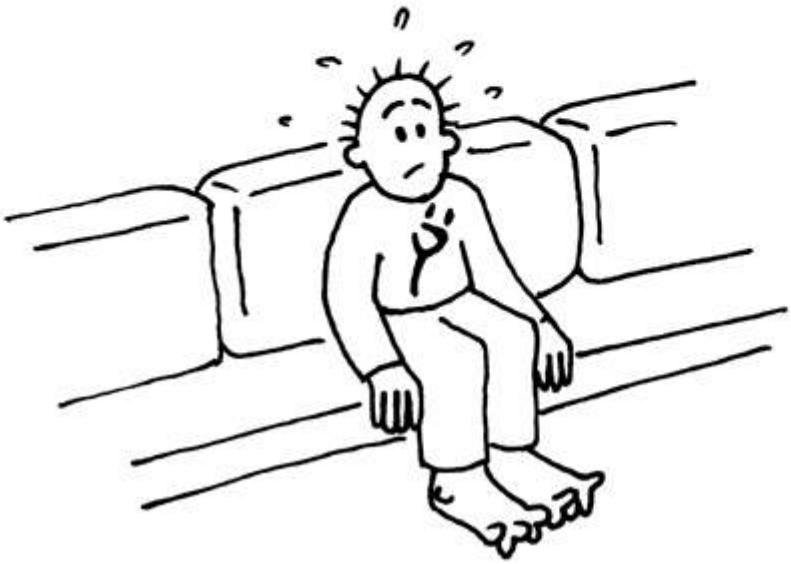
更多關於二進位數字的知識（第 14 頁）

當你把一個 0 放到一個二進位數字右邊，會使得那個數字變成兩倍。

所有是 1 的地方現在變成了原來的兩倍，使得總和的數字變成了兩倍（在十進位數中，我們把一個 0 放到某一個數字的右邊，就會讓那個數字變成 10 倍。）

一台電腦需要 7 個位元來存所有的字元。7 ・個位元總共可以表示 128 個字元。而通常電腦會以 8 個位元組成一個位元組，而其中浪費掉 1 個位元。





危險動作，請勿模仿！
薯叔有練過的，不是
每個人腳趾都能這樣
彎的喔！

活動 2
用數字表示顏色 — 圖像表示法

活動摘要
電腦只用數字來儲存圖畫、相片與其他圖片。以下的活動將展示電腦是如何做的。

課程銜接
• 數學：幾何 — 形狀與空間技術：只使用數字來表示其它種類的資料
• 技術：減少重複的資料所使用的空間

習得技能
• 計算
• 繪圖

適合年齡
• 7 歲以上

所需素材
• 投影片展示：用數字表示顏色（第 21 頁）

每個學生需要
• 活動學習單：孩子的傳真（第 23 頁）
• 活動學習單：做出自己的圖片（第 24 頁）

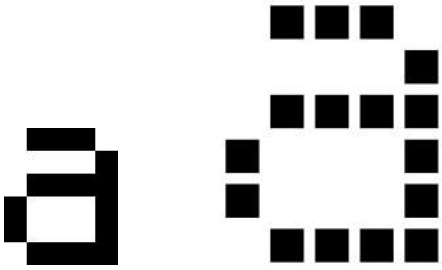
用數字表示顏色

問題討論

1. 傳真機是用來做什麼的？
2. 在什麼情況下電腦會需要儲存圖片呢？（例如繪圖程式、包含圖形的遊戲、或多媒體系統等等。）
3. 該怎麼讓只能夠使用數字的電腦儲存圖片呢？

（您可以安排學生發送或接收傳真作為這項活動的準備）

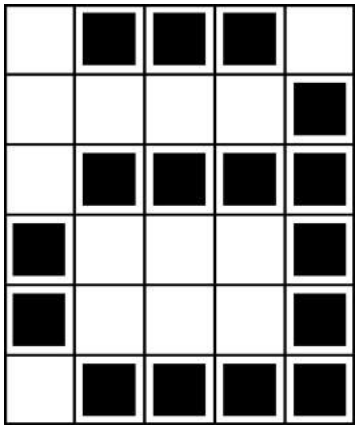
使用投影展示：



電腦螢幕被分割成許許多多的小格子，稱為像素（圖像元素）。

在黑白的畫面下，每個像素不是黑色就是白色。

上面的字母 “a” 已被放大來顯示像素。當電腦儲存圖片時，它要儲存的就是哪些像素是黑的，哪些像素是白的。



1, 3, 1

4, 1

1, 4

0, 1, 3, 1

0, 1, 3, 1

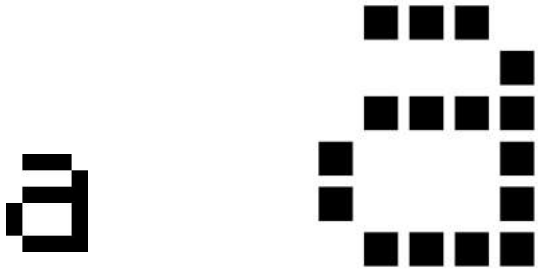
1, 4

上面的圖片說明了一張圖片怎麼用數字來表示。
第一行由一個白色像素，三個黑色像素，再一個白色像素所組成。
因此，第一行被表示為 1,3,1。

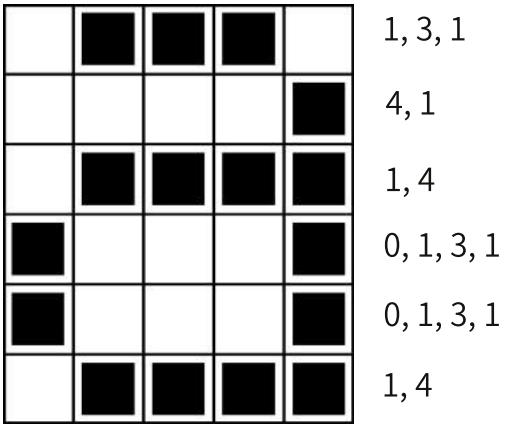
第一個號碼必須是白色像素的數目。如果第一個像素是黑的，那該行會從 0 開始。

第 23 頁的學習單提供了一些圖片。學生可以使用剛剛展示的方法來解碼。

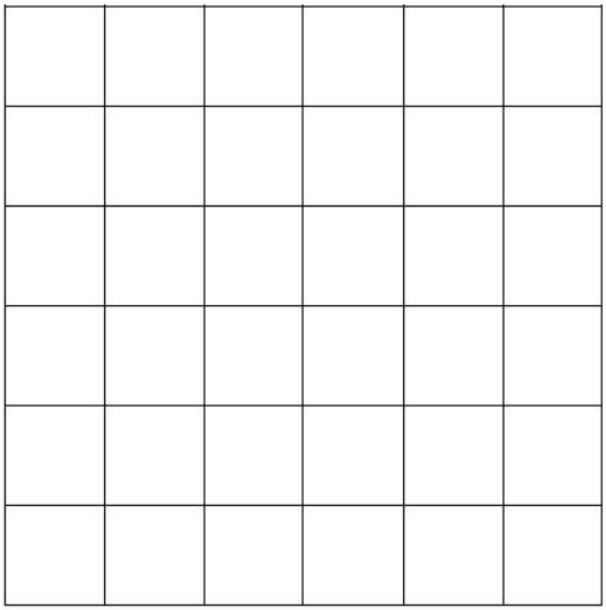
用數字表示顏色



▲ 從電腦螢幕上的字母 “a” 和其放大圖示，
可以得知該圖像是由像素所構成



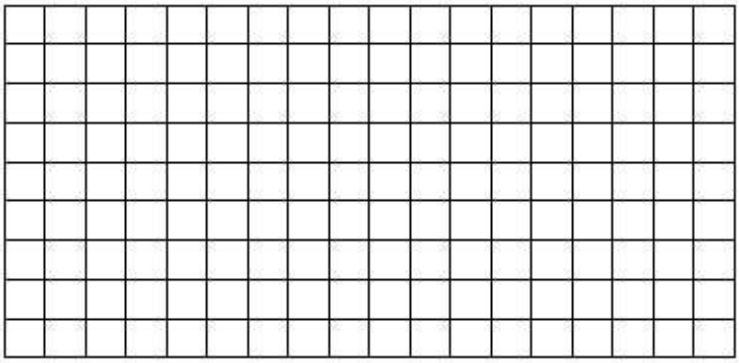
▲ 用數字為同樣的圖像編碼



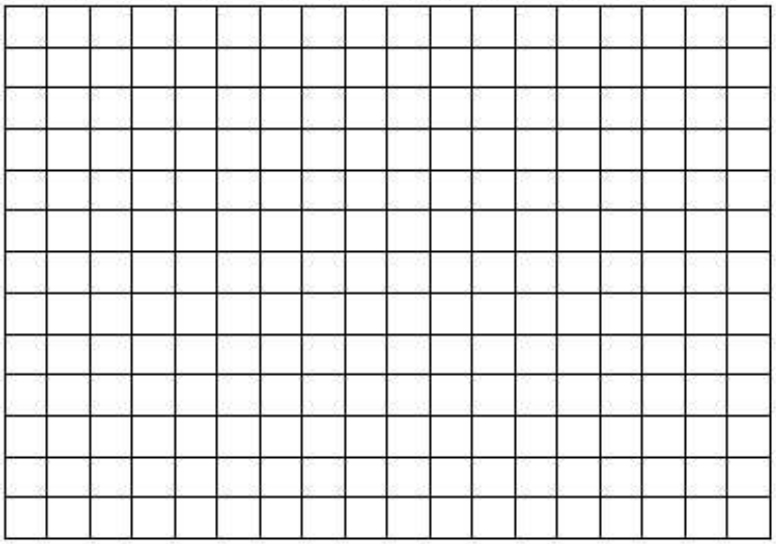
▲ 空白網格（教學用）

活動學習單：孩子的傳真

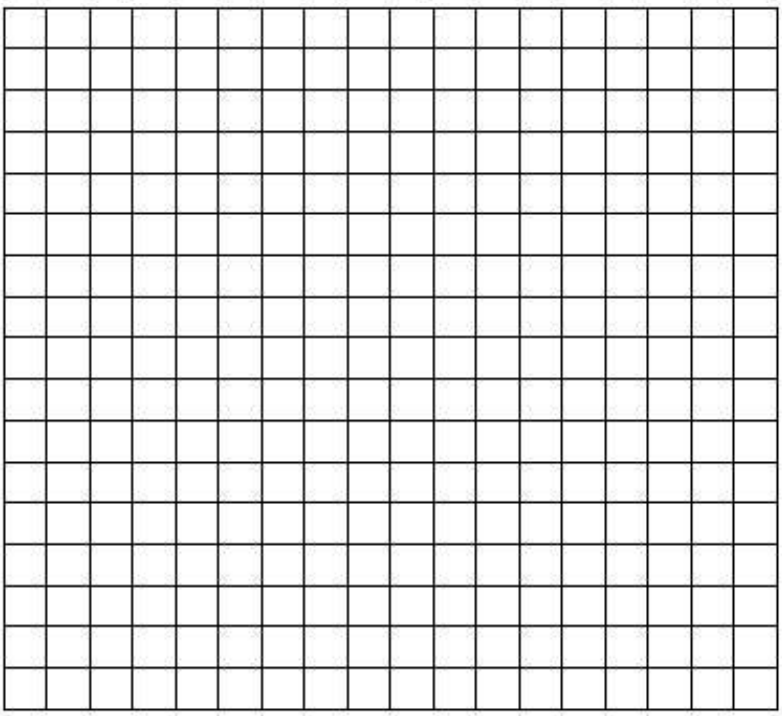
第一張圖是最簡單的，最後一張圖則是最複雜的。因為很容易畫錯，因此建議用鉛筆畫，並準備好橡皮擦。



4, 11
4, 9, 2, 1
4, 9, 2, 1
4, 11
4, 9
4, 9
5, 7
0, 17
1, 15



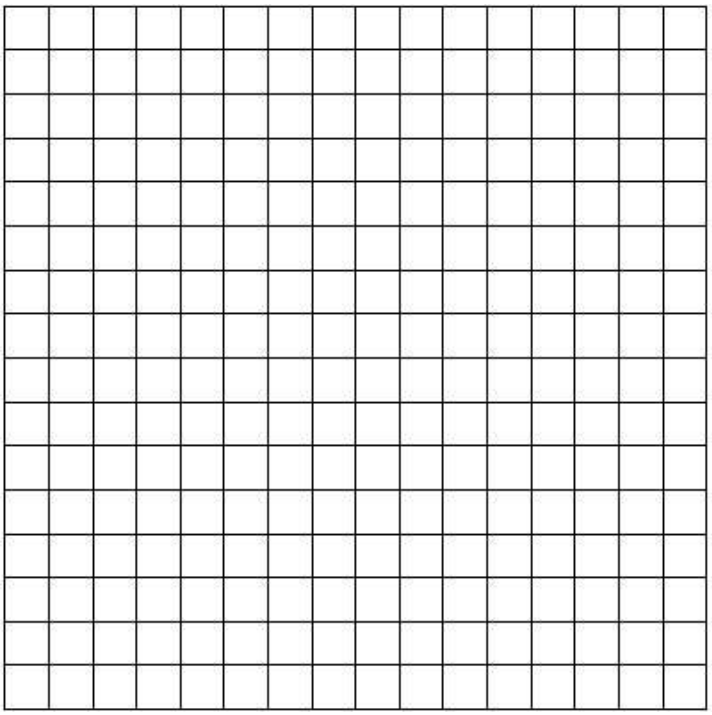
6, 5, 2, 3
4, 2, 5, 2, 3, 1
3, 1, 9, 1, 2, 1
3, 1, 9, 1, 1, 1
2, 1, 11, 1
2, 1, 10, 2
2, 1, 9, 1, 1, 1
2, 1, 8, 1, 2, 1
2, 1, 7, 1, 3, 1
1, 1, 1, 1, 4, 2, 3, 1
0, 1, 2, 1, 2, 2, 5, 1
0, 1, 3, 2, 5, 2
1, 3, 2, 5

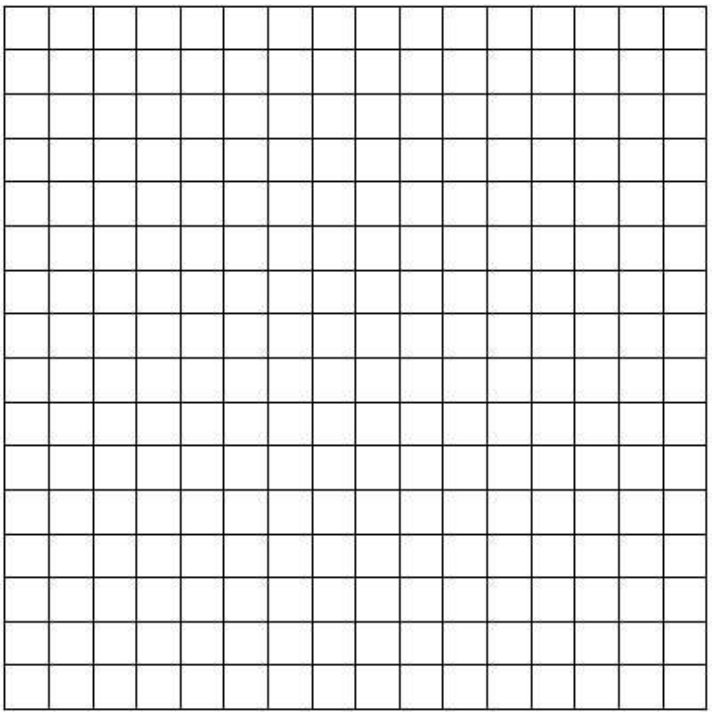


6, 2, 2, 2
5, 1, 2, 2, 2, 1
6, 6
4, 2, 6, 2
3, 1, 10, 1
2, 1, 12, 1
2, 1, 3, 1, 4, 1, 3, 1
1, 2, 12, 2
0, 1, 16, 1
0, 1, 6, 1, 2, 1, 6, 1
0, 1, 7, 2, 7, 1
1, 1, 14, 1
2, 1, 12, 1
2, 1, 5, 2, 5, 1
3, 1, 10, 1
4, 2, 6, 2
6, 6

活動學習單：做出自己的圖片

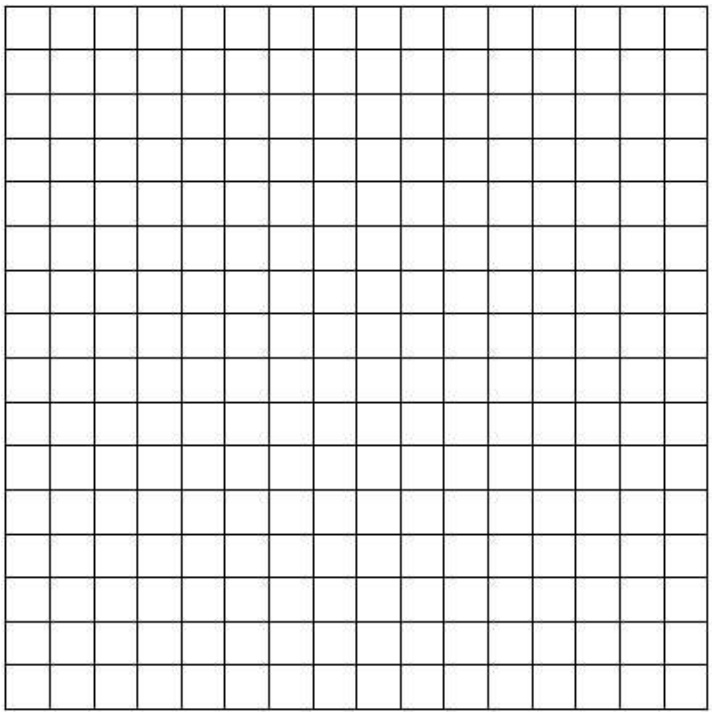
現在你知道了如何用數字來表示圖片，要不要來試試看，製作自己的編碼圖片給朋友？在上方的網格中畫一張圖，當你完成後，在下方網格中寫下對應的編碼。沿著虛線剪下下方的網格，並交給你的朋友請他還原你的圖片。（注意：你並不需要把整個網格都用滿。如果你的圖片並沒有用到整個網格，就在下方留下空白行即可。）

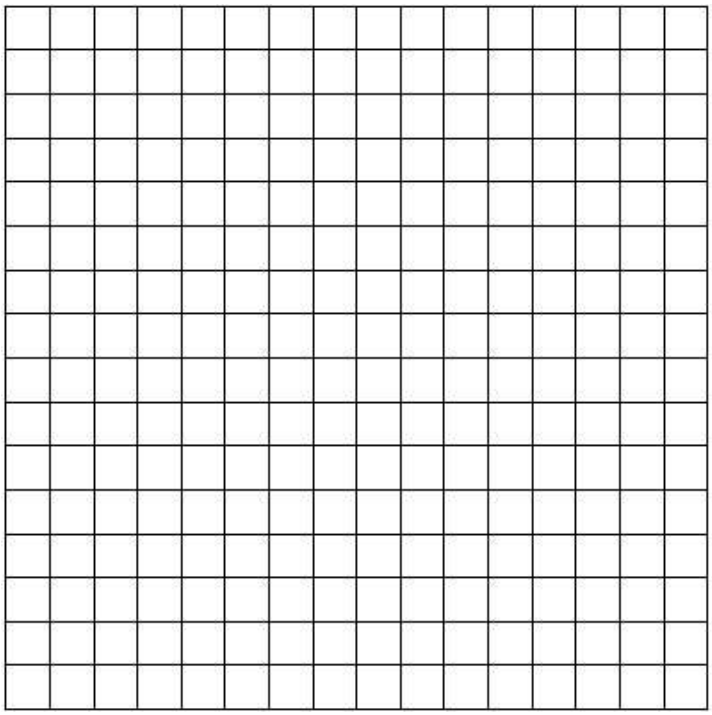




活動學習單：做出自己的圖片

高手挑戰：如果你想製作彩色圖畫，可以使用一個數字代表一種顏色 （例如 0 代表黑色，1 代表紅色，2 代表綠色等等。）不過，這樣就需要用到兩個數字來表示一系列像素了：第一個數字跟以前一樣代表長度；而第二個數字則用來標示顏色。試試看做張彩色圖片給朋友吧！別忘了讓朋友知道每個數字所代表的顏色喔！





活動變化與延伸

1. 可以試著利用素描紙疊在網格上先畫一次，如此可以看到沒有框線的圖形，會更容易看清楚。
2. 除了在網格中著色外，也可以讓學生使用貼紙，或是在大一點的格子中擺放物品來取代。

活動討論

通常一系列的像素的長度是有限制的，因為長度是以二進位數字表示。如果能表示出的數字最大只到 7，你要如何表示一串 12 像素的黑色呢？（有個好方法是先表示出 7 個黑色像素，之後接著一個長度為 0 的白色像素，再接著表示 5 個黑色像素。）

這個活動在說什麼？

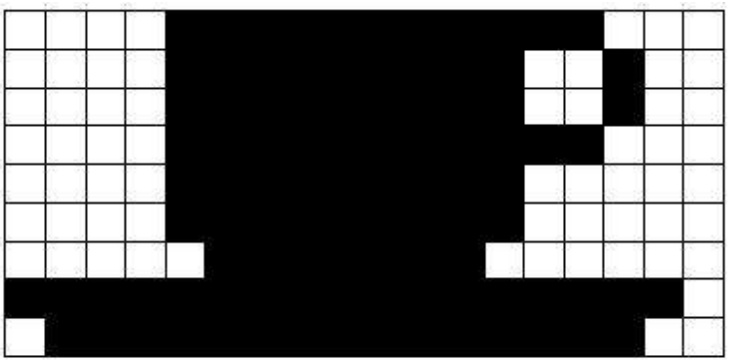
傳真機實際上就只是一台簡單的電腦，掃描黑白頁面成大約 1000 x 2000 像素，並使用數據機傳輸給另一台傳真機，最後將像素印在紙上。傳真的圖像常常有著部分空白（邊緣）或黑色像素（垂直的線）。彩色圖片也會有許多重覆部分。為了節省儲存空間，開發人員會使用很多種壓縮技術。而在這活動中使用的方法稱為變動長度編碼法（RLE，run-length encoding，又稱為游程編碼），是一種有效的壓縮圖片的方式。如果我們不壓縮圖片，那可能會花很多的時間來傳送圖片，而且也需要更多儲存空間。這樣會讓傳真或將圖放在網路上變得成本太高而不可行。舉例來說，傳真圖片通常被壓縮至原大小的七分之一左右。也就是說，如果不壓縮，光傳輸就會多出七倍的時間！

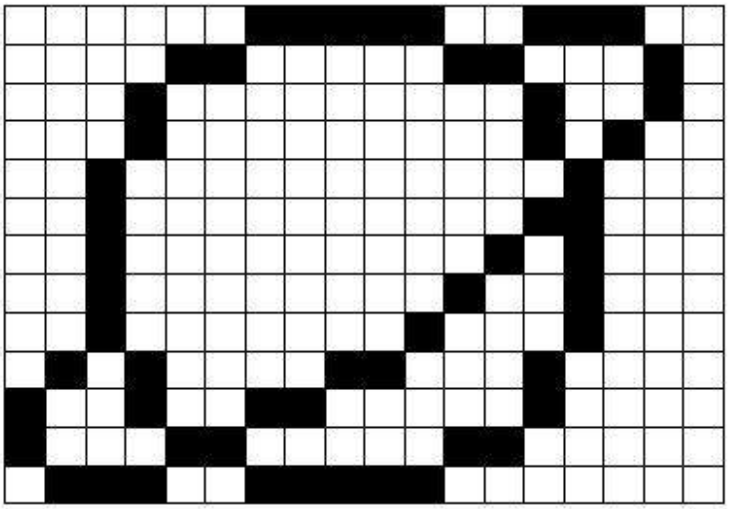
相片或圖片常被壓縮至原大小的十分之一或甚至百分之一（使用像是 JPEG、GIF 或 PNG 等技術）。這樣一來就可以在磁碟中儲存更多圖片，而這也表示在網站瀏覽圖片時可以花更少的時間。

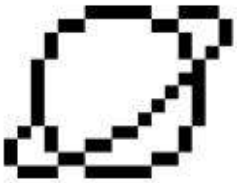
開發人員可以依想傳輸的圖片選擇最適合的壓縮技術。

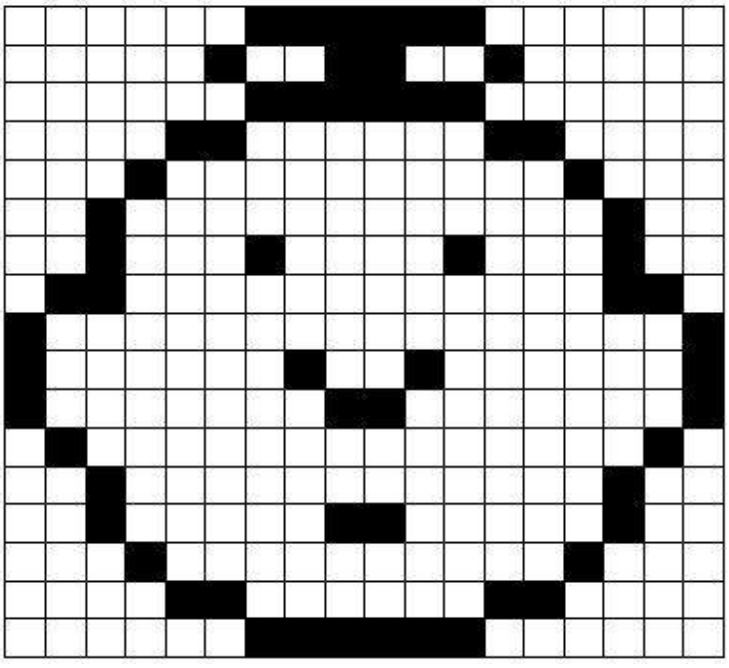
解答與提示

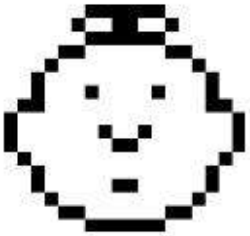
學習活動單「孩子的傳真」的解答











活動 3
資料宅急便 — 文字壓縮

活動摘要

因為電腦可以儲存資訊的空間很有限，所以要盡可能用有效率的方法來呈現資訊。這就叫做壓縮。藉由對資料做編碼，然後要取出資料時再解碼的方式，電腦可以儲存更多資料，或是更快地在網路傳遞資料。

課程銜接

- 英文：辨認出單字與文字的樣式
- 科技：利用重複的資料來減少使用的空間

習得技能

- 複製寫下的文字

適合年齡

- 9 歲以上

所需素材

- 投影片展示：資料宅急便（第 31 頁）

每個學生需要

- 活動學習單：資料宅急便（第 33 頁）
- 活動學習單：高手挑戰（第 34 頁）
- 活動學習單：小試身手（第 35 頁）
- 活動學習單：給高手高手高高手的大挑戰（第 36 頁）

資料宅急便

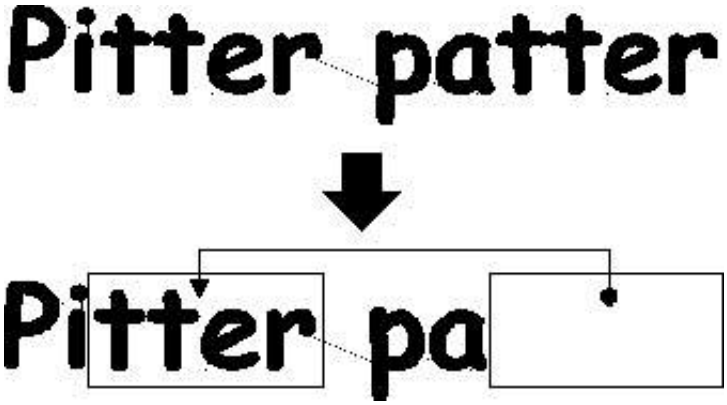
活動介紹

電腦必須儲存與傳輸很多資料。為了不佔用太多儲存空間，或花太多時間在網路上傳送資料，電腦需要像這樣壓縮文字。

展示與討論

給學生們看「雨」這首詩（第 32 頁）。找找看在這首詩中的字母的特定樣式。

問學生們是否能找到同樣字母、同樣的單字或甚至同樣的片語連續出現兩次以上的地方？（找到以後，像下圖那樣用框框把它框起來。）



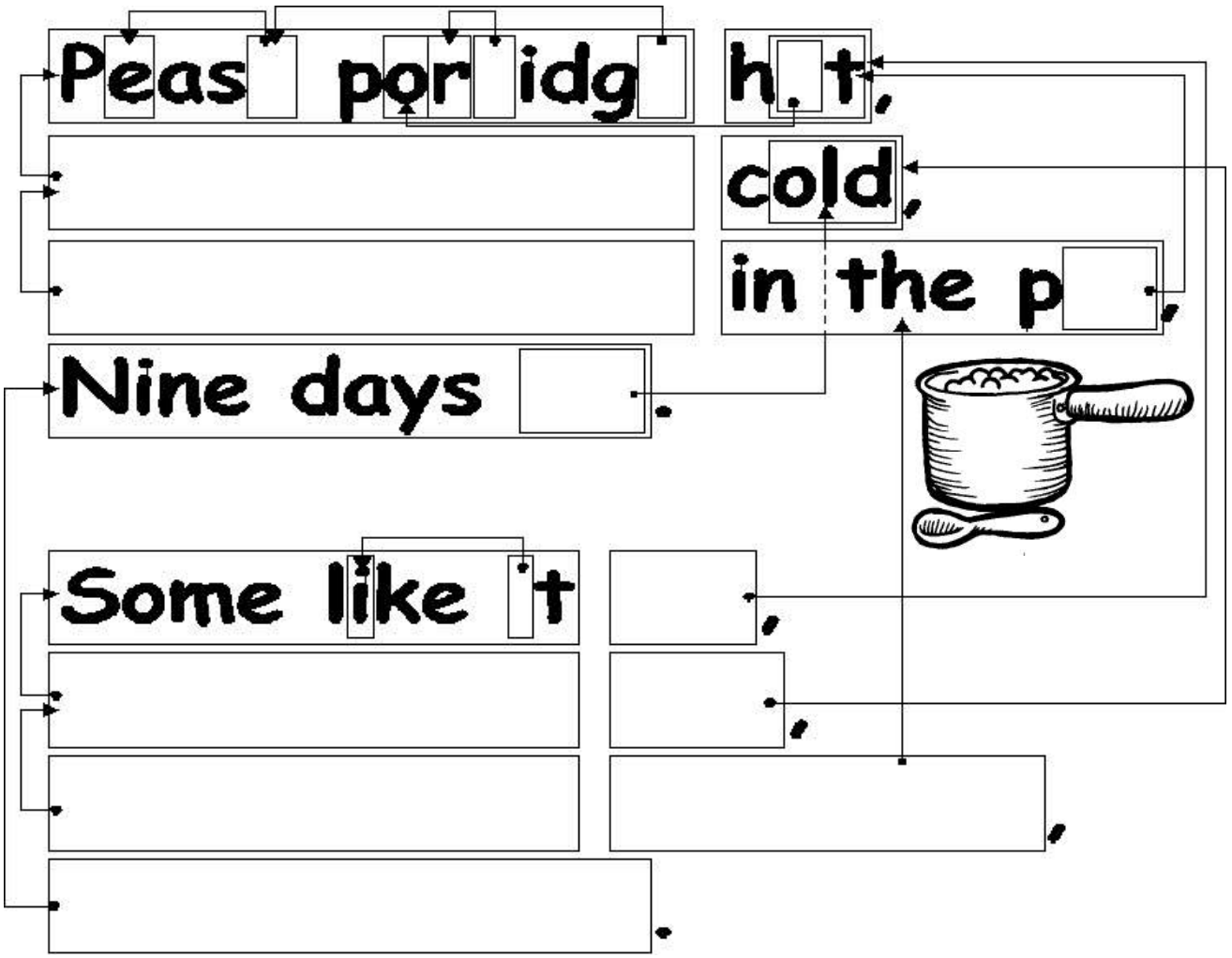
The Rain

Pitter patter
Pitter patter
Listen to the rain

Pitter patter
Pitter patter
On the window pane

活動學習單：資料宅急便

在這首詩中缺了很多單字與字母。你能把缺少的單字與字母正確地填上去嗎？你可以順著箭頭找到缺少的字母。



現在，找一首簡單的詩或兒歌，並用相同的方法設計自己的謎題。注意，箭頭要指向前面一個出現過的相同字母。你的謎題應該要可以從左到右，從上到下解出來。

挑戰：看看你需要保留多少原本的單字！

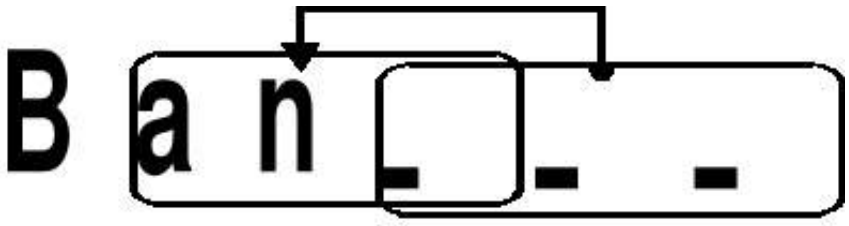
這裡有一些兒歌可以試試：
Three Bind Mice (https://en.wikipedia.org/wiki/Three_Blind_Mice),
Mary Mary Quite Contrary (https://en.wikipedia.org/wiki/Mary,_Mary,_Quite_Contrary),
Hickory Dickory Dock (https://en.wikipedia.org/wiki/Hickory_Dickory_Dock),
或是找一些 Dr. Seuss 的書。

提示：避免讓箭頭變得太擠。在單字與字母之間記得留多一點空白，這樣才有空間可以畫框線與箭頭。

設計謎題的時候，把詩或兒歌先寫下來，再決定框框要在哪裡會比較容易一點。

活動學習單：高手挑戰

這一題你會怎麼解？



有的時候，箭頭指向的框框中本身就有缺字。在這一題中，只要由左到右仔細觀察一下，就可以找出左邊的框框缺少的字母是什麼，接著右邊的框框也可以順利解出來。這種做法對於很長，但是有特定樣式的文字就很有用。

試著自己做做看。

在電腦中，這些框框與箭頭是用數字來表示的。比方說，

Banana

可以寫成 **Ban(2,3)**。前面的 **2** 表示要往前數兩個字元，然後從那邊開始複製字母。

Ban---

後面的 3 則表示連續複製三個字元：

Bana--

Banan-

Banana

當然，因為用了兩個數字來表示這些單字，所以要有兩個以上的框框的單字才值得去做壓縮，不然的話就不會節省到空間了。想想看，如果我們用了兩個數字卻只表示了一個字母，這樣大小是不是反而會增加呢？

試著自己寫一些單字，然後用電腦的方式來壓縮看看。你的朋友們能解開你的謎題嗎？

活動學習單：小試身手

你實際上需要多少個單字？

假裝你自己是一台電腦，正試著把下面的資料塞進磁碟裡。循著下面的資料，把所有已經出現過的，兩個以上字母的組合劃掉，改用指示箭頭來存放。目標是盡可能劃掉越多字母越好。

I know an old lady who swallowed a bird

How absurd! She swallowed a bird!

She swallowed the bird to catch the spider

That wriggled and jiggled

and tickled inside her

She swallowed the spider to catch the fly

I don’t know why she swallowed a fly

Perhaps she’ll die…

活動學習單：給高手高手高高手的大挑戰

準備好面對真正困難的壓縮了嗎？
下面的故事透過電腦程式來運算之後，其中至少有 1633 個字母可被劃掉。你能發現多少個？
記住，只有兩個以上重複的字母可以消除。祝你好運！

Once upon a time, long, long ago, three little pigs set out to make their fortunes. The first little pig wasn't very clever, and decided to build his house out of straw, because it was cheap. The second little pig wasn't very clever either, and decided to build his house out of sticks, for the "natural" look that was so very much in fashion, even in those days. The third little pig was much smarter than his two brothers, and bought a load of bricks in a nearby town, with which to construct a sturdy but comfortable country home.

Not long after his housewarming party, the first little pig was curled up in a chair reading a book, when there came a knock at the door. It was the big bad wolf, naturally.

"Little pig, little pig, let me come in!" cried the wolf.

"Not by the hair on my chinny-chin-chin!" squealed the first little pig.

"Then I'll huff, and I'll puff, and I'll blow your house down!" roared the wolf, and he did huff, and he did puff, and the house soon collapsed. The first little pig ran as fast as he could to the house of sticks, and was soon safe inside. But it wasn't long before the wolf came calling again.

"Little pig, little pig, let me come in!" cried the wolf.

"Not by the hair on my chinny-chin-chin!" squealed the second little pig.

"Then I'll huff, and I'll puff, and I'll blow your house down!" roared the wolf, and he did huff, and he did puff, and the house was soon so much firewood. The two terrified little pigs ran all the way to their brother's brick house, but the wolf was hot on their heels, and soon he was on the doorstep.

"Little pig, little pig, let me come in!" cried the wolf.

"Not by the hair on my chinny-chin-chin!" squealed the third little pig.

"Then I'll huff, and I'll puff, and I'll blow your house down!" roared the wolf, and he huffed, and he puffed, and he huffed some more, but of course, the house was built of brick, and the wolf was soon out of breath. Then he had an idea. The chimney! He clambered up a handy oak tree onto the roof, only to find that there was no chimney, because the third little pig, being conscious of the environment, had installed electric heating. In his frustration, the wolf slipped and fell off the roof, breaking his left leg, and severely injuring his pride. As he limped away, the pigs laughed, and remarked how much more sensible it was to live in the city, where the only wolves were in the zoo. And so that is what they did, and of course they all lived happily ever after.

這個活動在說什麼？

電腦的儲存容量正以令人難以置信的速度增長 – 在過去 25 年裡，傳統電腦的儲存量已經至少增長了數百萬倍 – 雖然如此，我們打算存入電腦中的東西也跟著增加很多很多。只要空間足夠，電腦可以儲存整本書，甚至一整個圖書館的資料；現在更可以儲存很多音樂和電影。大量的資料在網際網路上是個大問題，因為需要很長的時間才能下載完。此外，我們也嘗試把電腦越做越小 – 小到變手機或手錶那樣，而我們仍希望能在上面儲存大量的資訊！

要解決儲存空間或傳輸大量資料的問題，除了買更多的記憶體或硬碟之類的儲存空間，或加快網路傳輸速度之外，還有一個解決這個問題的方法，就是我們可以透過壓縮使資料佔用的空間變少。壓縮和解壓縮的這個過程通常會交給電腦自動完成。我們會注意到的可能只有磁碟能儲存的空間變大了，或網頁顯示的速度變快了。但是實際上，電腦做的工遠遠超過你的想像。

近年來，發明了許許多多壓縮的方法。而在這個活動中所使用的方法，也就是用一個指標指向前面出現過的區塊的做法，被稱為「LZ編碼」（Ziv-Lempel Coding，或 LZ Coding）。這個方法是由兩名以色列的教授在 20 世紀 70 年代所發明。它可以方便地用於任何語言，並可以輕易地減少一半的資料量。它有時被稱為在個人電腦上的「拉鍊」（zip），並且也用在 GIF 和 PNG 這兩種圖檔。此外，也被用於高速的數據機。數據機利用壓縮減少透過電話線傳送的資料量，所以速度快得多。

另外，還有一種壓縮的想法 – 常用的字母應該比其他的字母使用更短的編碼。摩斯電碼就是使用了這個想法。

解答與提示

資料宅急便（第 31 頁）

Pease porridge hot,
Pease porridge cold,
Pease porridge in the pot,
Nine days old.

Some like it hot,
Some like it cold,
Some like it in the pot,
Nine days old.

活動 4
卡片翻轉魔術 — 錯誤的發現與修正

活動摘要

當資料儲存在磁碟或是要從一台電腦傳送到另一台時，我們會希望在傳送過程中，資料不會因為任何原因而被改變。然而，有時候還是會發生一些意外性錯誤，使資料毀損或改變。在這個活動中，我們會使用一個魔術般的手法，在資料受損而被改變時能發現並修正它。

課程銜接

- 數學：數字 — 探索如何計算以及估計
- 數學：代數 — 探索資料樣式及其關係，並找出遺失的資料
- 數學：列與行、座標系統
- 科技：資料驗證

習得技能

- 計數
- 辨識奇數和偶數

適合年齡

- 7 歲以上

所需素材

- 36 張有磁鐵可吸在冰箱上的卡片，單面著色
- 一個金屬板（白板也可以）供展示用

每組學生需要：

- 36 張相同的卡片，單面著色

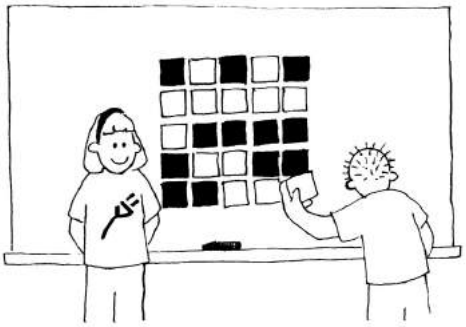
魔術般的技巧

展示方式

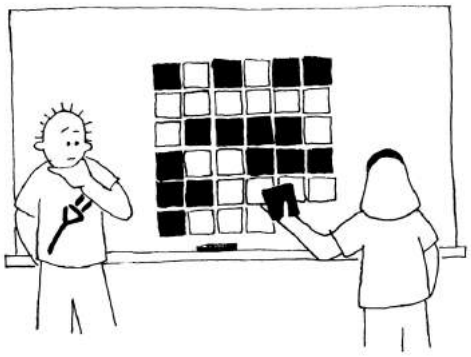
這是個讓你變成魔術師的機會！

你需要一副完全相同，雙面的卡片 (想自製的話，可以從大張的卡片剪小，單面著色)，為了更方便展示，可以使用雙面不同色的磁卡，冰箱的磁卡效果佳 (大部分的磁卡都是單面的，可以將它們的正面黏起來，將其中一面做上白色記號。)

1. 讓一名學生將雙面磁卡片以 5*5 的方式排好，並隨意翻至不同面。



故意增加一列或一行，「讓它更難一些」。



這些卡片是這個技巧中的關鍵。你可以再多用幾張卡片，但要確定每一行或列，都有偶數張有色卡片。

2.　讓一個學生，眼睛蒙上並翻一張卡。現在包含這張卡的那一列跟那一行，其有色卡的張數會變成奇數，這樣就可以找出哪一張卡片被變更了。各位同學能知道是怎麼做到的嗎？

教學生們這個技巧：

1. 將學生分成兩兩一組進行，讓學生將它們的卡片排成 5 x 5。

2. 每列每行有幾張有色卡呢？是奇數還是偶數？記住，0 是算偶數喔。

3. 在每一列增加第六張卡片，確保每一列的色卡數是偶數，增加的那張稱為「同位」卡（Parity card）。

4. 在底部增加第六列卡片，確保每一行的有色卡數是偶數。

5. 現在翻其中一張卡片。觀察一下，卡片所在的那一列與那一行有什麼變化？（那一列與那一行的色卡數變成奇數了。）剛剛加上去的那一張同位卡就是用來告訴你某一列某一行的資料出問題了。

6. 輪流向大家表演這個「魔術」。

延伸活動

1. 嘗試使用不同的東西取代色卡，擁有正反兩面性質的都可以。舉例：撲克牌、硬幣、正反兩面寫著 0 和 1 的卡（可以與前面介紹的二進位數系統做連結）。

2. 如果兩張或兩張以上的卡片被翻過來了呢？（這樣就沒辦法知道是哪兩張卡片被翻過來了，只能知道有些東西被改變了。仔細分析一下，是可以限縮到知道兩對卡片中的某一張有問題。但如果發生四次翻轉時，就很有可能行列的奇偶性檢查會找不出任何錯誤）。

3. 嘗試更大的排法，例如：9 x 9 張卡，檢查用的同位卡要使之擴張成 10 x 10。（這個方法適用於任何排法，甚至可以不用是正方型。）

4. 另一個有趣的練習是看看右下方的卡片。如果你用它檢查它所在的那一行的資料是正確的，那它所在的那一列是否也會是正確的？（答案是：是。如果用的是偶數的同位檢查法，那一定會是正確的。）

5. 在這個卡片練習中，我們用的是偶數的同位檢查 — 也就是讓色卡的數量保持為偶數。反過來想，我們能不能用奇數的同位檢查呢？（可以，但是在右下角卡片的那個檢查，只有在列數與行數同時是奇數或同時是偶數才有作用。例如，5 x 9 或 4 x 6 可以檢查出問題，但 3 x 4 就不行。）

一個現實生活範例：書碼與條碼

這種檢查技術也同時被用在書碼和條碼。發行的書本會有一組 10 或 13 位的數字數字，通常會印在書背上。最後一位的數字就是檢查碼，就像活動練習中的同位卡片一樣。

這表示如果你訂購了一本有使用 ISBN (International Standard Book Number，國際標準書號)的書，網站能幫你檢查書號有沒有錯誤。只要很簡單地看同位檢查碼就好了。這樣你就不會等了老半天，結果還拿到錯誤的書了。

這裡是一組　10　碼的書號的檢查方式：第一位數字乘上十，第二位數字乘上九，第三位數字乘上八，依此類推，直到第九位數字乘上二。最後把所有的值相加。

例如：ISBN 為 0-13-911991-4 的書，它算出來的值是：

(0 × 10) + (1 × 9) + (3 × 8) + (9 × 7) + (1 × 6) + (1 × 5) + (9 × 4) + (9 × 3) + (1 × 2) = 172

然後把算出來的值除以 11。餘數是多少？

172 ÷ 11 = 15 餘 7

如果餘數為 0，檢查碼就是 0；如果不是的話，就用 11減掉餘數，得到的值就是檢查碼。

11 - 7 = 4

看看，ISBN 的最後一碼是不是 4？賓果！

如果 ISBN 的最後一碼不是 4，那我們就知道發生了錯誤。

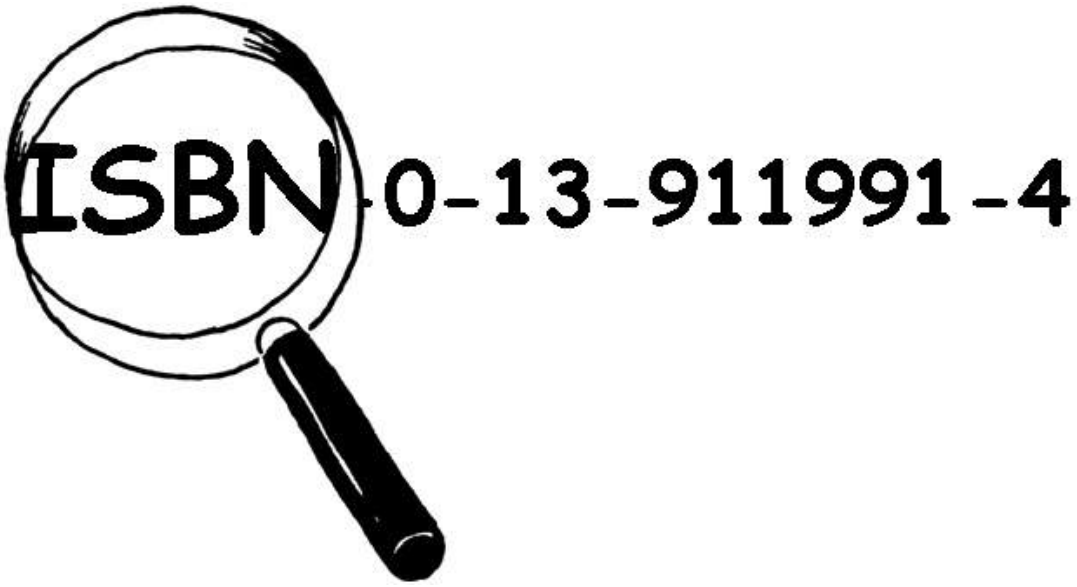
如果檢查碼是 10 的話呢？變成兩位數了。這種狀況下，就會改用 X 來表示。

▲ 這是 Weet-Bix™ 的一個盒子的條碼

另外一個使用檢查碼的例子，就是食品雜貨的條碼。它用的是不同的公式（前面說的公式可以用在 13 位的書碼）。如果條碼讀錯了，它最後一位數應該會和計算的值不相同。當這樣的情況發生時，掃描器會發出嗶嗶的聲音，櫃台結帳人員則會重新掃描條碼。此外，檢查碼也使用在銀行帳號、身份證字號、稅號、火車、運輸工具等等，以及在許多人們需要複製一些數字但要確認是否正確的時候。

檢查那本書！

偵探暢銷著作
書本追蹤服務公司



我們擅長尋找和檢查 ISBN 檢查碼，而且只需要一點小額費用。
加入我們的行列 -- 找找你的教室或圖書館中有沒有真正的 ISBN 碼。

他們的檢查碼是對的嗎？

有時候會發生一些錯誤。

常見的錯誤有：
- 某一位數的值被改變
- 兩個相鄰的數字被互換
- 數字中被多插入一位數
- 某一位數字不見了

你能找到一本檢查碼為 X，也就是檢查碼為 10 的書嗎？應該不會太難找 -- 平均每十一本書中就應該有一本。

哪一種錯誤會發生但不會被檢查到？你能改變某一位數字且仍然維持正確的檢查碼嗎？如果兩位數字被互換呢？（這種錯誤很常在打字時發生）

這個活動在說什麼？

想像你將 10 塊錢現金存入銀行的帳戶。出納員整理出所有存款的量，並傳送到中央電腦。假設在傳送的過程中，線路發生了一些干擾，結果數字從 10 塊錢變成 1000 塊。當然啦，你會很開心地接受這種錯誤，但對銀行來說可是會哭哭的！

在傳送資料時檢查錯誤是非常重要的。所以接收端的電腦需要檢查傳過來的資料沒有因為某些干擾造成損毀。有時候發生傳送錯誤時，原始資料可以再傳送一次，但有些情況則是不行，比方說磁碟因為曝露在磁力或電力輻射，或是高溫下，或是任何其它的物理性破壞而造成資料損毀。如果資料是從深遠的太空偵測器傳送過來，那麼當錯誤發生時，等待重新傳送會需要非常長的時間！（比方說，在木星最靠近地球的時候，從木星傳過來的無線訊號，大概需要超過半小時才能收到）。

因此，當資料損毀時，我們需要有辦法發現（錯誤檢查），並且能重建原始的資料（錯誤修正）。

在翻卡片遊戲的活動中所使用到的技巧，也被用在電腦上。藉由加入同位檢查位元到每一列和每一行，我們不只能檢查出有錯誤，還能找出錯誤發生的位置。這個出問題的位元會被改回去，這就是錯誤修正。

當然，電腦使用的錯誤控制系統通常會複雜很多，以檢查和修正多重錯誤。電腦中的硬碟裡，有很大量的空間是被用來做錯誤修正的。這樣即使部分的硬碟壞了，也還是可以工作。這些錯誤偵測與修正的系統，與這個活動中所講的同位檢查機制是密切相關的。

解答與提示

ISBN-10　檢查碼無法偵測出的錯誤會發生在某一位數字增加而同時另一位數字減少，剛好把檢查碼的差彌補過來。這種狀況下計算出來的總和會一樣。然而這種計算方式之下，這種錯誤發生的機率很小。在別的系統（例如 ISBN-13），也有可能有其他種類的錯誤是無法被檢查出來的，例如三個連續的數字的顛倒；但最常見的錯誤（例如打錯一個數字，或兩相鄰的數字交換）是會被檢查出來的。

活動 5
二十個問題 — 資訊理論

活動摘要

在一本 1000 頁的書中有多少的資訊？一本 1000 頁的電話簿中，或者是一疊 1000 張的空白紙中，甚至是在「魔戒」這本書中，哪個有更多的資訊？如果我們有方法可以「測量」，我們就可以估計需要多少的空間去存放這些資訊。例如，你還有辦法讀這段話嗎？

Ths sntnc hs th vwls mssng.

懂一點英文的話，應該可以知道這句話在說什麼吧，因為在母音當中並沒有太多的「資訊」。這個活動將會介紹一個方法來測量資訊內容。

課程銜接

- 數學：數字 — 探索數字：大於、小於、範圍
- 數學：代數 — 樣式和順序
- 英文：拼字、辨識文字中的元素

習得技能

- 比較數字並且利用數字的範圍
- 推導演繹
- 提問題

適合年齡

10 歲以上

所需素材

- 在第一個活動中沒有需要的材料。

在額外的活動中，每一位同學將會需要：
- 活動學習單：決策樹（第 54 頁）

二十個問題

1. 和同學討論有關於他們所認為的資訊是什麼。

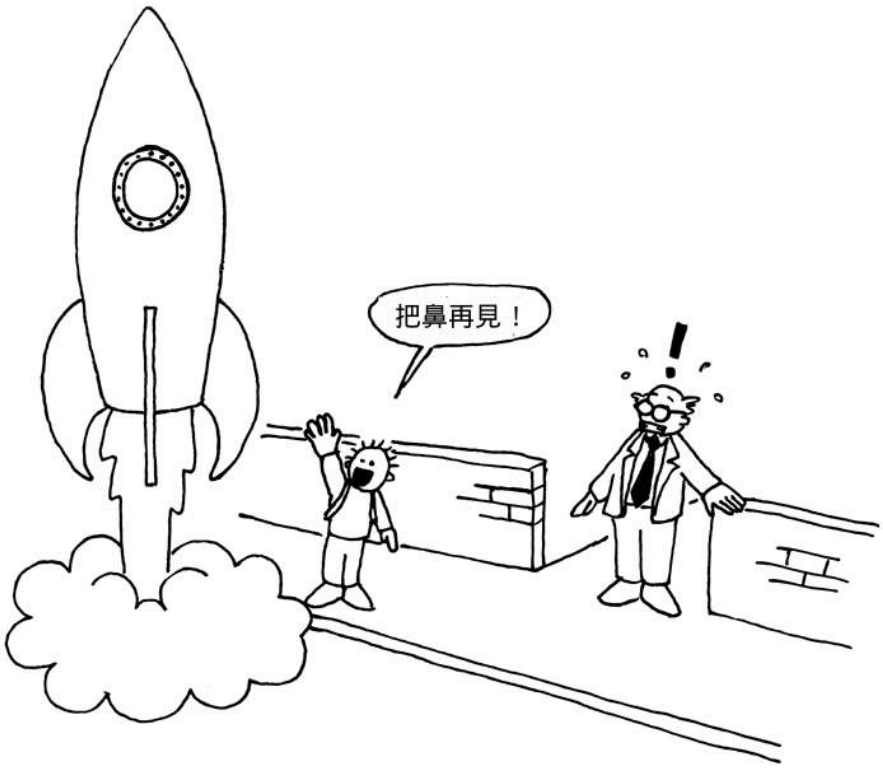
2. 我們如何能得知在一本書中有多少的資訊？書中的頁數和字數是重要的因素嗎？一本書會比另外一本有更多的資訊嗎？一本非常無聊或是特別有趣的書，哪個包含的資訊量較多？一本 400 頁但都是廢話的書，會比一本電話簿含有更多的資訊嗎？

3. 解釋電腦科學家是如何藉由某段訊息（或者某本書）「讓人驚訝的程度」的來測量資訊。告訴你一些你已經知道的東西 — 比方說，一個總是走路到學校的朋友告訴你：「我今天走路到學校。」你就一點都不會感到驚訝，因為你早就知道了。但是如果你的朋友告訴你：「我今天坐著直升機到學校。」這樣的訊息會讓你下巴掉下來，也因此會告訴我們非常多的資訊。

4. 要如何測量得知某段訊息的「驚訝值」呢？

5. 一個方式是看看要猜到那個資訊會有多困難。如果你的朋友告訴你：「猜猜我今天是如何到學校的？」而答案是他走路到學校，這個時候你會有比較高的機會在第一次就猜到正確答案。但如果答案是坐直升機，甚至太空船，那麼有可能得猜好幾次才能猜到答案。

6. 某段訊息所包含的資訊量，是藉由是否容易被猜到來衡量。下面的遊戲會帶給我們一些這方面的想法。



活動：二十個問題

這是改編自傳統的「二十個問題」的遊戲。選擇一個學生出來當關主，心裡想定一個答案以後，由其他的學生來猜關主心裡想的是什麼。其他學生可以問關主問題，但關主只能回答「是」或「不是」。任何問題都可以問，但回答只有「是」或「不是」，不能有其他任何回答。

比方說，題目可以是：

• 1 到 100 間的數字
• 1 到 1000 間的數字
• 1 到 100 萬間的數字
• 任何一個整數
• 有特定樣式（學生們可以瞭解）的一串六個數字。回答時必須從第一個到最後一個依序回答。（例如，2, 4, 6, 8, 10, 12）

選擇一種，然後讓關主在範圍內選擇一個答案，最後看看大家問了幾個問題才猜到關主的答案。這個值就是「資訊」的測量值。

活動討論

你用的是哪一種策略？哪一種策略最好？

引導學生發現，當數字範圍在 1 到 100 之間，那麼最多只需要 7 次的猜測就可以得到正確答案。例如：

是否比 50小 — 是
是否比 25小 — 否
是否比 37小 — 否
是否比 43小 — 是
是否比 40小 — 否
是否比 41小 — 否
答案就是 42 — 是！

有趣的是當範圍拉大到 1 到 1000 的數字時，上述的方法並不需要多十倍的問題量 — 反而只需要多三個問題就可以了。實際上把範圍每次都加倍，每次最多只需要多問一個問題就可以得到答案。

後續可以讓同學們玩玩看珠璣妙算（MasterMind，請參考維基百科中「珠璣妙算」條目。）

活動延伸：在某段訊息中有多少資訊？

電腦科學家並不只猜數字而已 — 他們也可以猜到在一個字或一個句子中，可能接著出現哪一個字母。

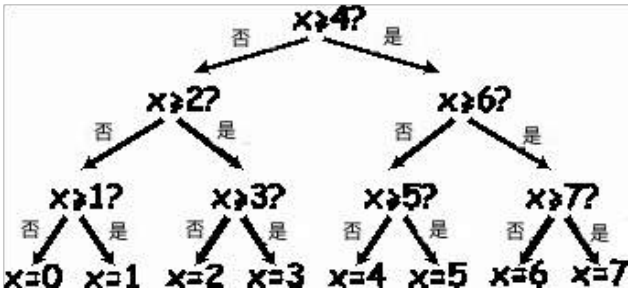
嘗試用四到六個單字組成的句子來進行這個遊戲。同學們必須要依照正確的順序（從第一個到最後一個）猜到字母。指定一位同學，寫下大家發現的字母，並且記錄下每個字母猜了幾次才猜中。任何一個可以藉由「是」或「否」回答的問題都可以被提出。

舉例來說：「這個字母是 t 嗎？」、「這個字母是母音嗎？」或者「這個字母的順序是在 m 之前嗎？」注意，在文字之間的空白也被視為一個「字母」，並且必須要猜出來。輪流並看看你能不能發現，在訊息中的哪一部分是最容易被找出來的。

活動學習單：決策樹

如果你已經能掌握問問題的策略，那你不需要詢問任何事情便能傳達訊息。

這裡有一張叫做「決策樹」的圖，用來猜測 0 到 7 之間的某個數字：



要猜到數字 5 需要做哪些決定（是 /不是）？

你需要多少決定（是 /不是）才能猜測出任何數字？

現在我們來觀察一件非常迷人的事。在樹最下方的數字 0,1,2,3… 的下面寫下那些數字的二進位表示法（參考活動 1）。

仔細觀察決策樹的圖。如果「不是」 = 0，而「是」 = 1 ，你發現了什麼？

其實在遊戲中，我們選擇提問的方式，就是讓答案序列可以用這種方式表達數字。

現在來試試看。如果要猜 0 到 15 間的數字，決策樹要怎麼畫？

高手挑戰：你會使用哪一種決策樹來猜測一個人的年齡？
如果是猜測一個句子裡的下個字母呢？

這個活動在說什麼？

Claude Shannon，著名的美國數學家（同時也是雜耍者及獨輪車車手），做了許多關於這個遊戲的實驗。他用「位元數」來測量「資訊」的量 -- 每個「是」與「不是」分別都用一個位元（1 或 0）來代表。他發現每個訊息所包含的「資訊」量的多寡與你已知的事情是有關的。有時候我們能藉由問一個問題來免去問許多其他問題的必要。在這種情況下，這則訊息的資訊量是低的。舉例來說，投擲一枚硬幣的資訊通常用一個位元即可：正面或反面。但如果擲一枚不公正的硬幣，十次中有九次是正面，那此資訊就不再是用一個位元可以表示了 — 信不信由你，它的資訊量更少。你要如何知道投擲一枚硬幣的資訊少於一個是非題呢？這很簡單 — 問像這樣的問題，「下兩次擲硬幣的結果都是正面嗎？」對於不公正的硬幣，有 80% 的機率答案為「是」。而在剩下 20% 答案為「不是」的情況下，你必須再多問兩個問題。但平均起來，每次擲硬幣所問的問題會少於一題！

Shannon 將一個訊息的資訊量稱為「熵」（entropy）。熵不僅相依於可能的結果數 — 以擲硬幣為例，有兩種結果 — 也相依於這些結果發生的機率。我們需要更多的問題，來獲得關於一些不可思議或令人訝異的事件的資訊，因為這些事件透露更多我們不知道的事情 — 像是搭直升機去學校。

一個訊息的熵對資訊科學家來說是非常重要的。你無法將一則訊息壓縮到比它的熵所佔據的空間還小；而一個最佳的壓縮系統就相當於一個剛剛所進行的猜數字遊戲。因為電腦程式在進行「猜測」，所猜測的問題清單可以事後重複使用，所以只要將答案（位元）紀錄下來，我們就能重現原來的資訊！最佳的壓縮系統可以將文字檔案壓縮到大約原先大小的四分之一 — 大大的節省儲存空間！

猜測的方法也能被用來建造一個預測使用者輸入的電腦介面！這對於打字有困難的身心障礙人士來說非常有幫助。電腦建議他們接下來可能要輸入什麼，而他們只要選擇他們想要的字即可。一個好的系統對於每個字母平均只需要兩個是非題的答案，如此便能成為對於控制滑鼠及鍵盤有困難的人們很棒的助手。這樣的系統也以不同的形式被用在手機上「打字」的時候。

解答與提示

一個是非題答案與一個位元的資訊有關 -- 無論是單純的問題，像是「比 50 大嗎？」，或是更複雜的問題，像是「是否在 20 和 60 之間？」

在猜數字遊戲中，如果題目是用特定的方法選出來的，則答案的序列就是該數字的二進位表示法。

3 以二進位表示裡就是 011，這代表在選擇樹裡，答案為「不是」、「是」、「是」，就跟我們把「不是」寫成 0 並把「是」寫成 1 的結果相同。

在猜測年齡時，所使用的決策樹可能會偏向使用較小的數字。

猜測一個句子裡的字母時，應該要依據前一個字母來做決策。

第二部份
讓電腦運作 —演算法

讓電腦運作

電腦是根據設定好的一連串指令來運作。這些指令讓電腦得以排序、搜索及傳送資訊。為了盡可能快速地這些事，比方說在大量的資料中找到需要的東西，還有在網路上傳送資訊，需要一些好的方法才能很快地把事情做好做滿。

演算法就是用來完成任務的一組指令。演算法的概念是資訊科學的核心。演算法是我們讓電腦運作的方式。有些演算法比較快，而有很多已發現的演算法讓我們可以解決從前得花上很多很多時間的問題 -- 例如，求出圓周率小數點後一百萬位、找到全球資訊網上所有包含你名字的頁面、求出將包裹裝入容器中最佳的方式，或是決定一個很大的數字（100位數）是不是質數。英文中「演算法（Algorithm）」這個字來自穆罕默德 · 伊本 · 穆薩 · 花拉子米－穆罕默德（Mohammed ibn Musa Al-Khowarizmi—Mohammed），也就是摩西的兒子，花剌子模人，在西元 800 年左右加入了在巴格達的一個稱作智慧之屋（House of Wisdom）的學術中心。他的作品將印度的算數藝術傳播到阿拉伯地帶，再從阿拉伯到歐洲。當它在西元 1120 年被翻譯為拉丁文時，第一個字是「Dixit Algorismi」－因此稱作「Algorismi」。

活動 6
海戰棋－搜索演算法

活動摘要

電腦常需要在大量的資料集中尋找資訊，因此電腦必須為此發展出快速又有效的方法。這個活動將示範三種不同的搜尋方法：線性搜尋法、二元搜尋法以及雜湊法。

課程銜接

- 數學：數－探索數字：大於、小於、等於
- 數學：幾何－探索圖形與空間：座標
- 電腦：演算法

習得技能

- 邏輯推理

適合年齡

- 9 歲以上

所需素材

每個學生需要：

- 一份海戰棋遊戲
  - 遊戲 1 的 1A、1B
  - 遊戲 2 的 2A、2B
  - 遊戲 3 的 3A、3B
- 也許還會需要補充遊戲板，1A'、1B'、2A'、2B'、3A'、3B'。

海戰棋活動 — 暖身

活動介紹

1. 挑選約 15 名學生在教室前方站成一排。給每位學生一張號碼牌（隨機順序），不要把卡上的號碼給其他同學看。

2. 給另一位學生一個裝有四、五顆糖果的罐子。他們的任務是要找到指定的數字是在誰的手上。他們可以「支付」一顆糖果來看特定的一張卡，如果他們用完所有糖果前找到正確的數字，就可以留下剩下的糖果。

3. 重覆以上步驟任意次。

4. 現在重新洗牌再重新發卡。這次，讓拿著卡的學生依卡片上的數字排成升冪排列。然後重複上面搜尋的步驟。

如果數字是排列好的，合理的策略是只「支付」一次糖果，讓站在中間的學生翻開他的卡片，這樣就可以消除一半學生拿著指定卡片的可能。通過重覆這個過程，學生們應該能夠只用三顆糖果就找到數字。效率的增加將會是顯而易見的。

活動進行

學生可以透過遊玩海戰棋來感受電腦進行搜尋的方法。當他們在玩遊戲時，可以讓他們思考搜尋戰艦的策略。

海戰棋 — 線性搜尋遊戲

閱讀以下說明給學生聽：

1. 分成兩人一組，你有一張紙（1A），另一個人也有一張紙（1B），不要給對方看到你的紙。

2. 兩個人都圈起上半部其中一艘戰艦，然侯告訴你的同伴圈起來的數字。

3. 現在，兩人輪流猜測對方的戰艦在哪裡。（猜的人唸出代表位置的字母，而同伴則告訴你你猜的那艘船在那張紙上的數字是幾號。）

4. 你花了幾次猜測才中同伴的船在哪裡？這就是你在這場遊戲中的分數。

（1A’ 與 1B’ 在學生想繼續玩，或是有人「不小心」看到同伴紙張的內容時可以用。2A’ 2B’ 3A’ 3B’ 則是為了之後的遊戲。）

活動討論

1. 這些分數所代表的意義是什麼？

2. 分數最大值及最小值分別是多少？（答案分別為 1 及 26，前提是學生不會「打到」同一艘船兩次。這個方法稱為「線性搜尋」，因為它會一個接一個走訪所有的位置。）

海戰棋 — 二元搜尋遊戲

活動說明

這個版本的遊戲的指示和上一個遊戲相同。不過這次戰艦上的數字，是以升冪排列。在遊戲開始前請先告訴學生這一點。

1. 分成兩人一組，你有一張紙（2A），對方有一張紙（2B），不要給對方看到你的紙。

2. 兩個人都圈起上半部其中一艘戰艦，然侯告訴你的同伴圈起來的數字。

3. 現在，兩人輪流猜測對方的戰艦在哪裡。（猜的人唸出代表位置的字母，而同伴則告訴你你猜的那艘船在那張紙上的數字是幾號。）

4. 你花了幾次猜測才中同伴的船在哪裡？這就是你在這場遊戲中的分數。

活動討論

1. 這些分數所代表的意義是什麼？

2. 得分較低（猜測次數較少）的同學們，所使用的策略為何？

3. 應該先選哪一艘船？（要先選在最中間，一半左右位置的船）下一艘呢？（一樣，在選擇的那半邊的最中間的那一艘船）

4. 如果用前面所說的策略，需要幾次才能找到目標船隻？（最多五次）

這個方法稱為二元搜尋，因為這個方法把問題切成兩半來解決。

海戰棋 -- 雜湊搜尋遊戲

活動說明

1. 跟前面一樣，每一個人拿一張紙，然後告訴你的同伴你所選的船的數字。

2. 不過在這個遊戲之中，你可以找到這艘船在哪一行（0 到 9）裡，只要簡單地把戰艦上的數字每一位數拆開，再一個個加起來，總和的最後一個數字就是戰艦所在的行。比方說。A 戰艦上的數字是 2345，那就把 2345 拆開再分別加起來（2+3+4+5），得到總和為 14。它的最後一位為 4，所以 A 戰艦一定位於第 4 行。當你知道在第幾行以後，你就需要猜測在那一行中的哪一艘戰艦是你要打的。這種方法稱為「雜湊」。

3. 現在運用這個新的搜尋方法玩這個遊戲，同一張紙可以玩好幾次，只要每次選擇不同的行即可。

（注意：不像其他的遊戲，紙張 3A’ 和 3B’ 需要成對，因為成對的紙張中，戰艦所在的行才會一樣。）

活動討論

1. 像之前一樣討論分數所代表的意義。

2. 哪些船會很快被找到？（在那一行中只有一艘的。）哪些船會比較難被找到？（某艘船所在的行中有很多其他船的。）

3. 三種搜尋法中，哪一個最快？為甚麼？

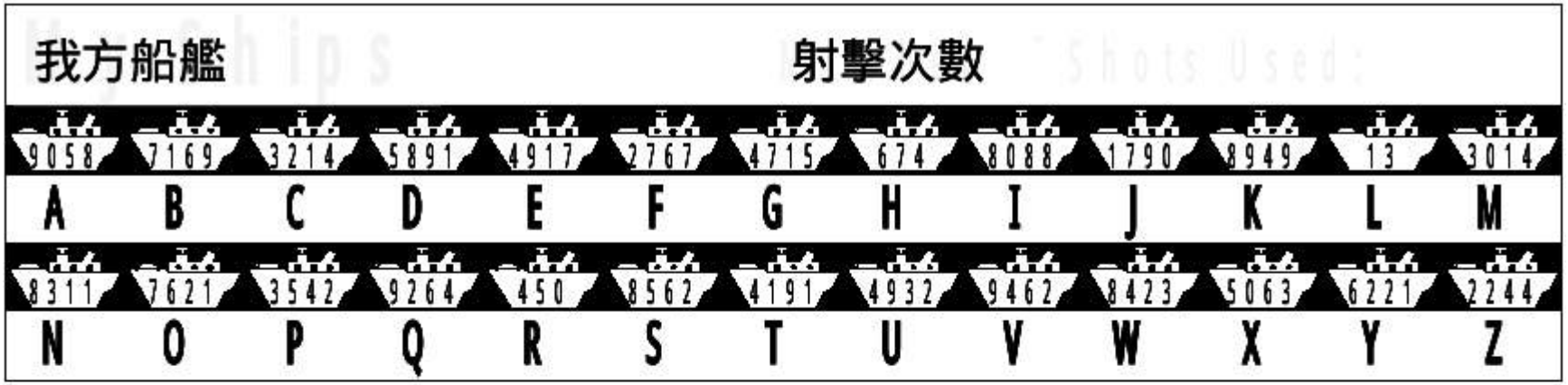
三種不同的搜尋法，其個別優點是什麼？（第二種搜尋法比第一種快，但第一種搜尋法不需要照順序排列。第三種搜尋法通常來說會比其他兩種快，但也有可能會變得非常慢。在最糟的情況下，如果所有船都在同一行，那就會變得和第一種搜尋法一樣慢。）

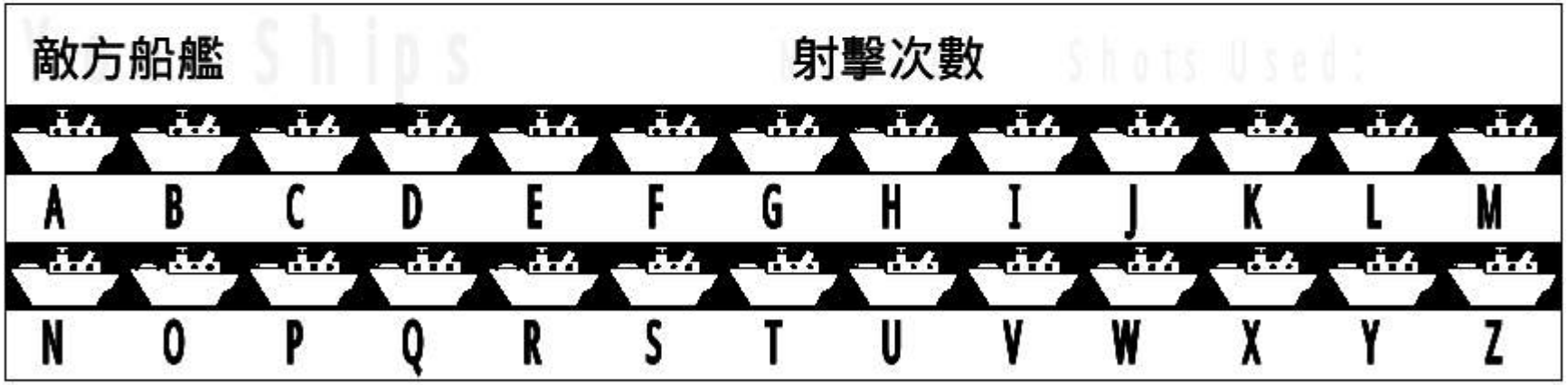
延伸活動

1. 讓學生運用這三種搜尋法來創造屬於自己的遊戲。在第二種遊戲中，需要先把數字以升冪方式排列。另外讓他們思考一下，雜湊搜尋遊戲要怎麼設計才會變得非常難？（當所有船在同一行中時。） 然後要怎麼樣設計才會變得很簡單？（讓每一行中船的數目相同。）

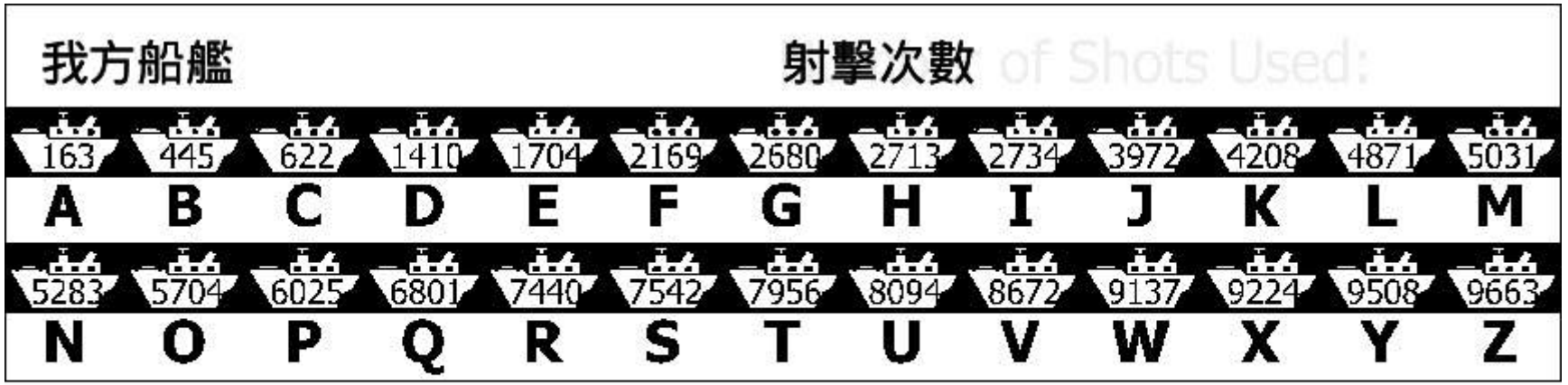
2. 如果要搜尋的船不在那裡，會發生什麼事？（在線性搜尋中，需要 26 次才能發現；在二元搜尋中，需要五次才能發現；而在雜湊系統中，要根據那一行中有多少船才能知道。）

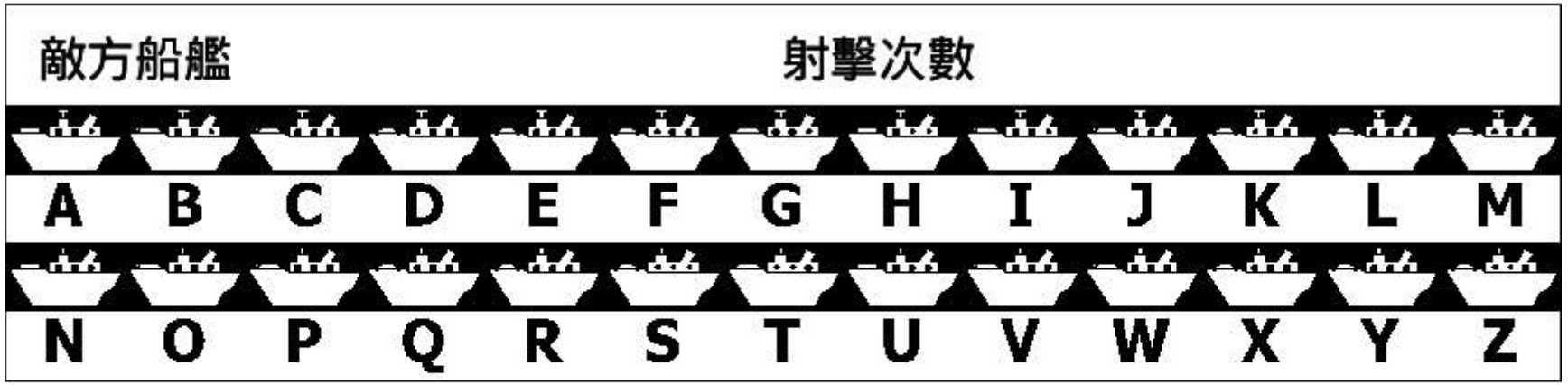
3. 用二元搜尋策略，100 艘船需要搜尋多少次？（大約六次），一千艘船呢？（大約九次），一百萬艘船呢？（大約十九次）（注意一下，搜尋的次數並沒有因為戰艦數量而跟著大幅增加。每次戰艦數量加倍時，只需要多搜尋一次，因此它是與戰艦數量的對數成比例的。）



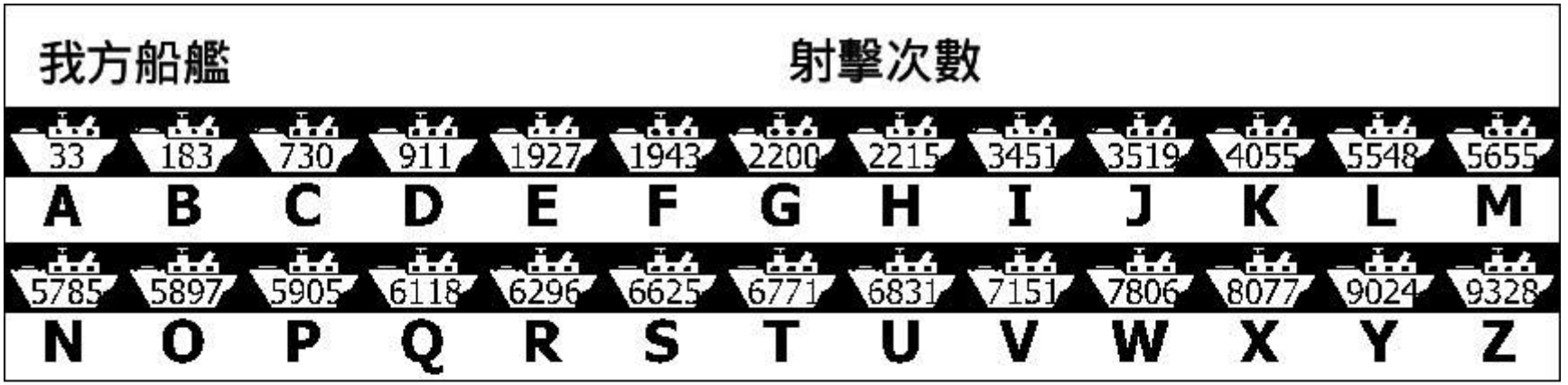


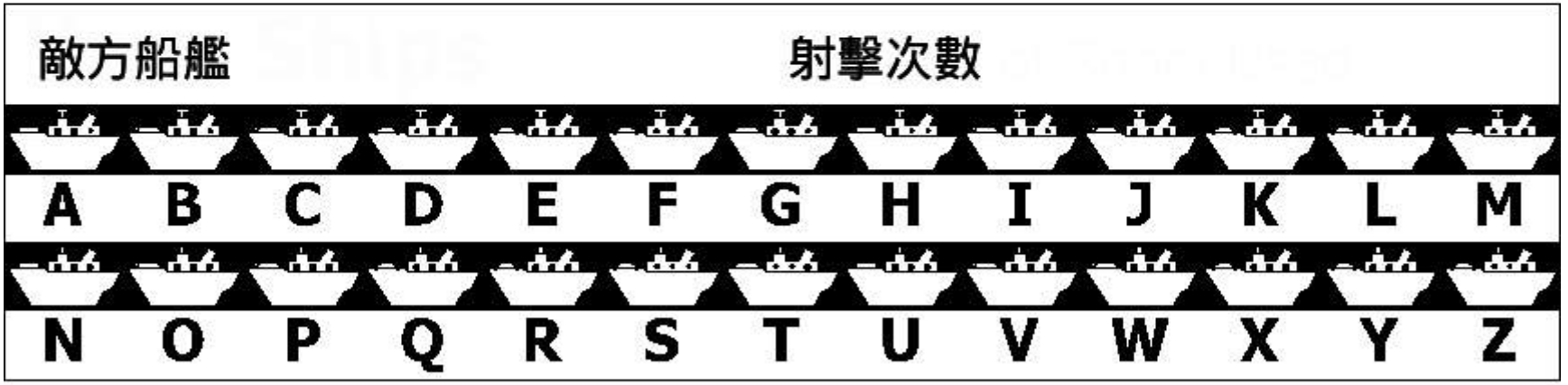
1 A





2A





2B

我方船艦	射擊次數								
0	1	2	3	4	5	6	7	8	9
A 9047 B 1829	C 3080 D 9994		E 5125 F 1480 G 8212	H 8051 I 1481 J 4712 K 6422	L 7116 M 8944 N 4128	O 6000 P 7432 Q 4110	R 9891 S 1989 T 2050 U 8199	V 4392	W 1062 X 2106 Y 5842 Z 7057

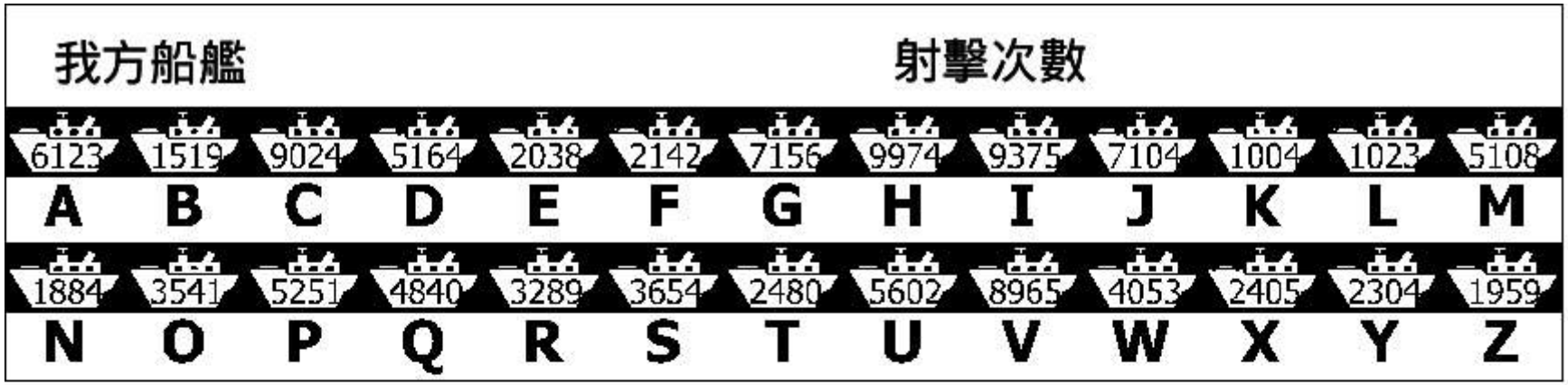
敵方船艦	射擊次數								
0	1	2	3	4	5	6	7	8	9
A B C D	E F G	H I J	K	L M N		O P Q	R S T U	V W X	Y Z

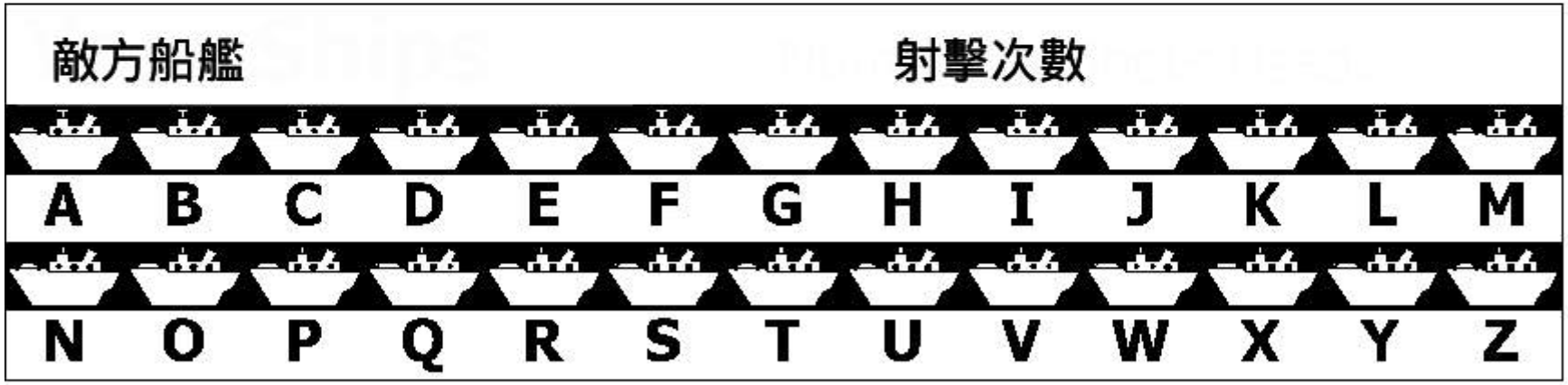
3 A

我方船艦	射擊次數								
0	1	2	3	4	5	6	7	8	9
A 9308 B 1478 C 8417 D 9434	E 6519 F 2469 G 5105	H 1524 I 8112 J 2000	K 4135	L 9050 M 1265 N 5711		O 4200 P 7153 Q 6028	R 3121 S 9503 T 1114 U 7019	V 2385 W 5832 X 1917	Y 1990 Z 2502

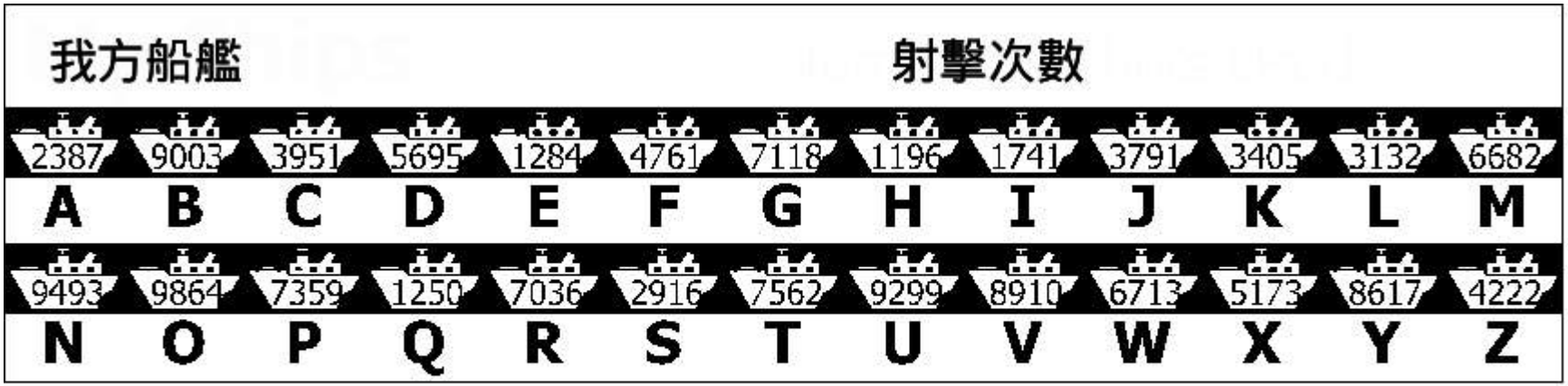
敵方船艦	射擊次數								
0	1	2	3	4	5	6	7	8	9
A B	C D		E F G	H I J K	L M N	O P Q	R S T U	V	W X Y Z

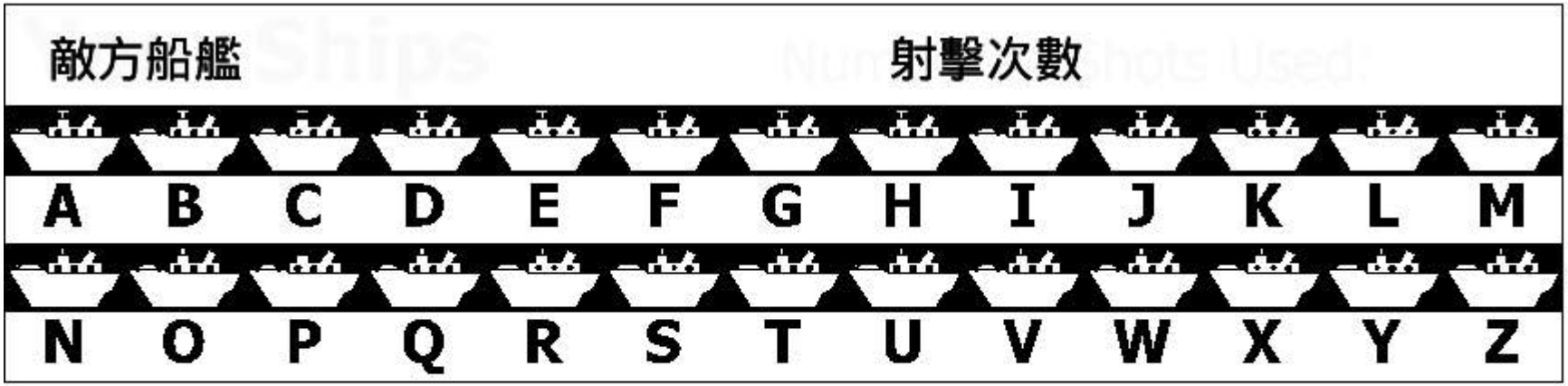
3B



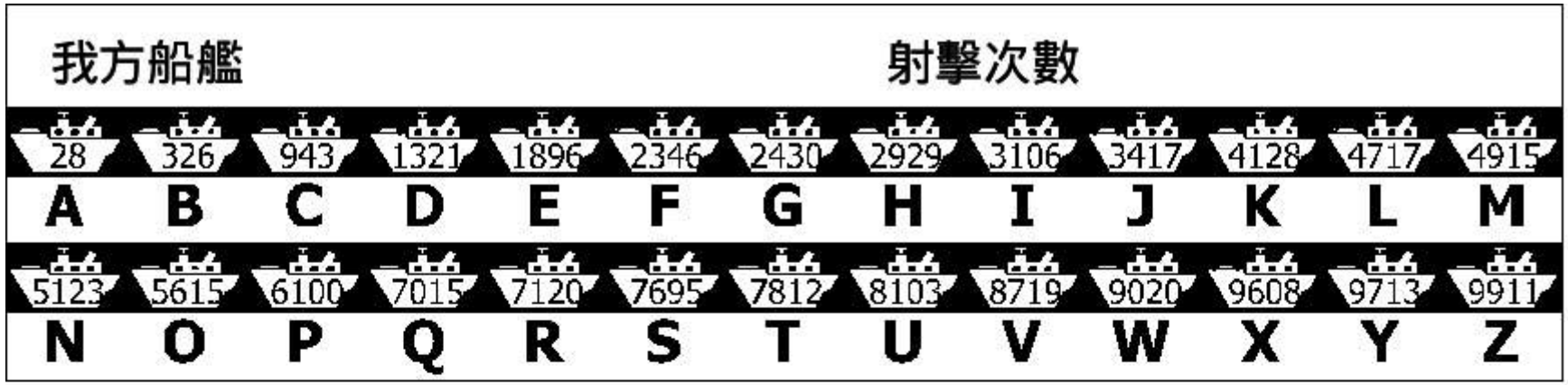


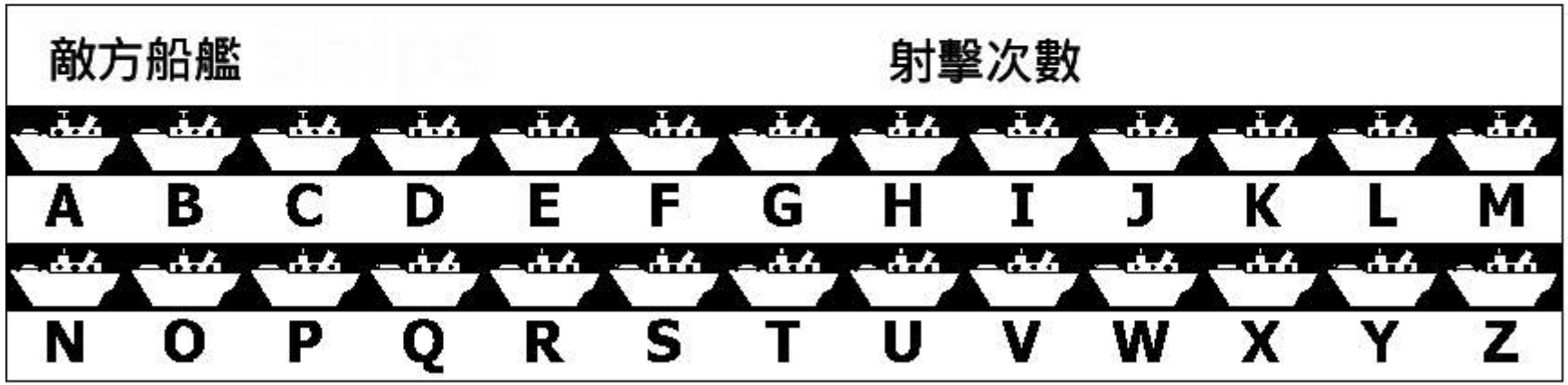
1A′



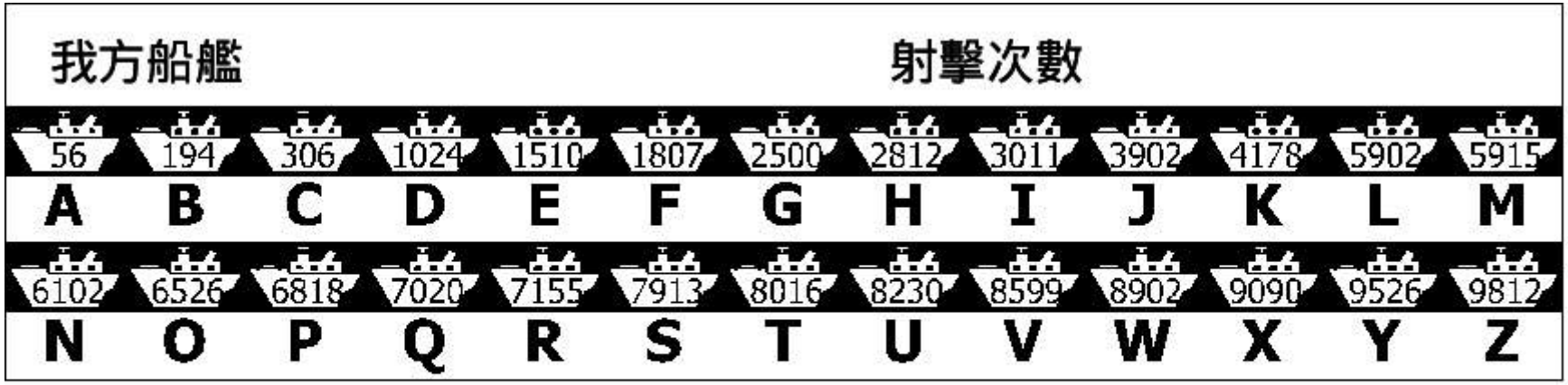


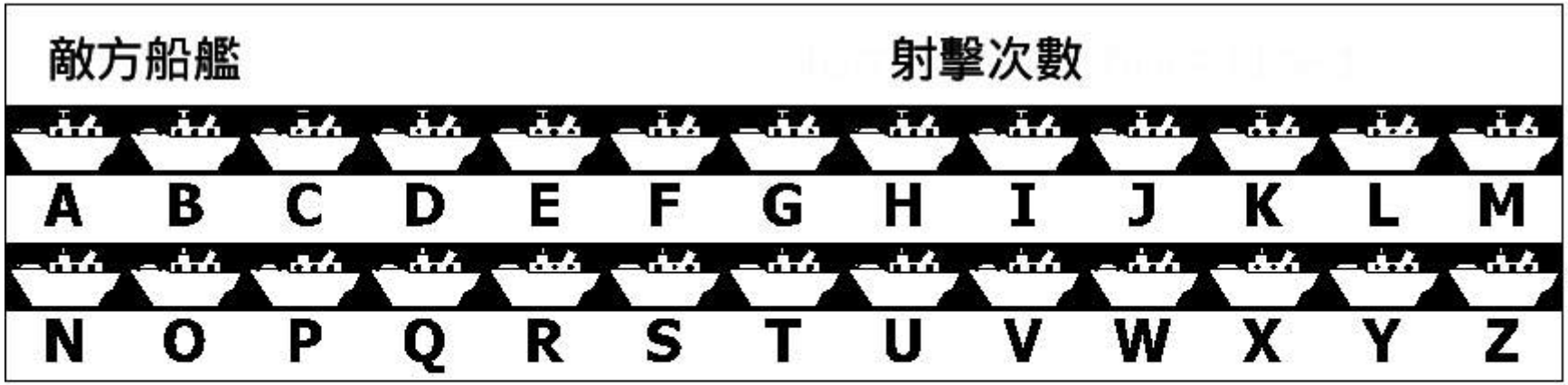
1B′





2A′





2B′

我方船艦	射擊次數								
0	1	2	3	4	5	6	7	8	9
A 1982 B 7841	C 6113 D 1055		E 9121 F 1011 G 2984	H 5009 I 2651 J 1751 K 4848	L 1248 M 1716 N 2148	O 2004 P 5173 Q 2806	R 9369 S 1321 T 3004 U 7190	V 3285	W 9172 X 2052 Y 6012 Z 7525

敵方船艦	射擊次數								
0	1	2	3	4	5	6	7	8	9
A B C D	E F G	H I J	K	L M N		O P Q	R S T U	V W X	Y Z

3A′

我方船艦	射擊次數								
0	1	2	3	4	5	6	7	8	9
A 8615 B 7003 C 1991 D 6211	E 1361 F 7644 G 5600	H 7726 I 9003 J 5557	K 3000	L 1814 M 2002 N 8844		O 9656 P 4002 Q 1221	R 6993 S 3121 T 4300 U 1907	V 8208 W 9423 X 4176	Y 2917 Z 4122

敵方船艦	射擊次數								
0	1	2	3	4	5	6	7	8	9
A B	C D		E F G	H I J K	L M N	O P Q	R S T U	V	W X Y Z

3B′

這個活動在說什麼？

電腦儲存了大量資訊，它們必須要能夠快速對其進行搜尋篩選。而搜尋引擎所面對的世界上最大搜尋問題之一就是，他們必須要在極短時間內搜尋數十億個網頁。而電腦被要求尋找的的資料，像是文字，條碼編號或是作者名稱，則被稱為「搜尋關鍵字」。

電腦可以非常快速的處理資訊，而你可能會認為為了找到想要的資訊，它們必須從頭開始，直到找到想找的目標為止。這是我們在第一個「線性搜尋遊戲」中所使用的方法。但這種方法 -- 即使對電腦而言 -- 也過於緩慢了。舉個例子，假設一間超級市場的貨架上有一萬種不同產品。當櫃檯掃描一個條碼時，電腦必須最多搜尋一萬次才能找到產品的名稱與價格。這樣即使檢查每筆條碼只花千分之一秒，檢查完所有條碼也要花 10 秒。想像一下，要花多少時間才能掃描完一個家庭會採買的林林總總的雜貨？

二元搜尋法看起來是比較好的方式。二元搜尋法中，先將每筆數字照順序排列。然後檢查列表中間的項目，就可以知道要找的關鍵字是在列表前半還是後半。接著重複該動作，直到找到要搜尋的項目為止。舉剛才超級市場的例子，一萬個項目最多需要進行十四次檢查，也就是大約兩百分之一秒 — 幾乎是一瞬間 -- 即可完成搜尋。

第三個尋找資料的方法叫做雜湊。在這種方法中，關鍵字本身會直接指示到哪裡找這個資訊。舉個例子，假設要搜尋電話號碼，你可以將所有數字加起來，然後除以十一並取餘數。這個做法得到的雜湊關鍵字就有點像活動四裡所提到的同位檢查數值 — 資料本身的一小部份是從資料的其他部份而來。用這種方法，通常電腦可以很直接地找到要搜尋的資訊。不過還是有小部份的可能，在同一個位置有好幾筆資料，而電腦必須在這幾筆裡面再繼續搜尋。

程式設計師常會使用一些不同版本的雜湊搜尋方式來進行搜尋，除非資料有必要按照順序排列，或是無法接受很慢的搜尋，即使機會很小。

活動 7
最重與最輕 — 排序演算法

活動摘要

電腦時常被使用來把資料依序排序。舉例來說，把名字依字母順序排列，依日期排序電子郵件或約會，或是依數量多寡排列物品等等。排序除了可使得我們在找東西時更快速之外，還有許多顯著的好處。例如把全班的成績依高低排序，最低分與最高分就很明顯了。

但若使用錯誤的方法，即使有設備很好、很快速的電腦，將大量的資料正確排序可能還是會花很多時間。還好，有幾種快速的演算法非常適合使用於排序。在本活動中，學生會學習不同的排序演算法，並明白適當的排序法比起簡單的排序法可以更快速的解決問題。

課程銜接

- 數學：測量 — 實際測量重量
- 計概：演算法

習得技能

- 使用天平
- 排序
- 比較

適合年齡

- 8 歲以上

所需素材

每一組學生需要：

- 8 個大小相同但重量不同的容器（例如：牛奶盒或裝滿沙的底片筒）
- 天平
- 活動學習單：重量排序 (第 86 頁)
- 活動學習單：分治法 (第 87 頁)

最重與最輕

活動討論

電腦時常要把資料依序排序。大家來腦力激盪一下，什麼樣的資料會需要排序？為什麼排序很重要？如果資料沒有排序會怎麼樣？

電腦常常一次只比較兩個值的大小。下一頁的活動會以此條件限制讓學生明白這個概念。

活動進行

1. 把學生分為兩組。
2. 每一組需要一張第 86 頁的學習單、一組容器和天平。
3. 讓學生進行此活動，並討論結果。

學習活動單：重量的排序

目標：

以最佳的方法把未知重量的容器排序。

你會需要：

水或沙子、8 個已知容積的容器、天平

步驟：

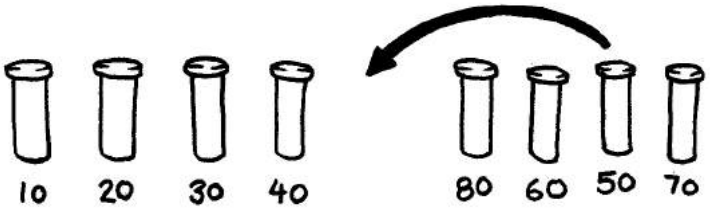
1. 將不同量的沙子或水倒入容器中，並將其密封。
2. 弄亂它們的順序，讓大家無法得知重量大小的順序。
3. 找出最輕的那一個。什麼是最好的方法呢？

注意：天平一次只可以比較兩個容器的重量。

4. 隨機挑選 3 個容器，並依輕到重排序，只能使用天平。你怎麼做到的呢？最少要比較幾次？為什麼？
5. 把所有的容器依輕到重排序。當你覺得完成時，重新稱重以檢查排列的順序。

選擇排序法

選擇排序法為電腦排序的方法之一。下列是選擇排序法的運作方式。首先，找到最輕的容器並擺在一邊。接下來，再從剩下的容器中挑出最輕的，並也擺到一邊。重複此動作直到所有的容器都被擺到同一邊。



計算你比較的次數。

高手挑戰：找出數學的規律。有 8 個容器時需要比較幾次？9 個呢？ 20 個呢？

學習活動單：分治法（Divide and Conquer）

快速排序法（Quicksort）

快速排序法比選擇排序法還要迅速，對於大型的表單則更加明顯。事實上，這是目前最好的方法之一。以下是快速排序法執行的方法。

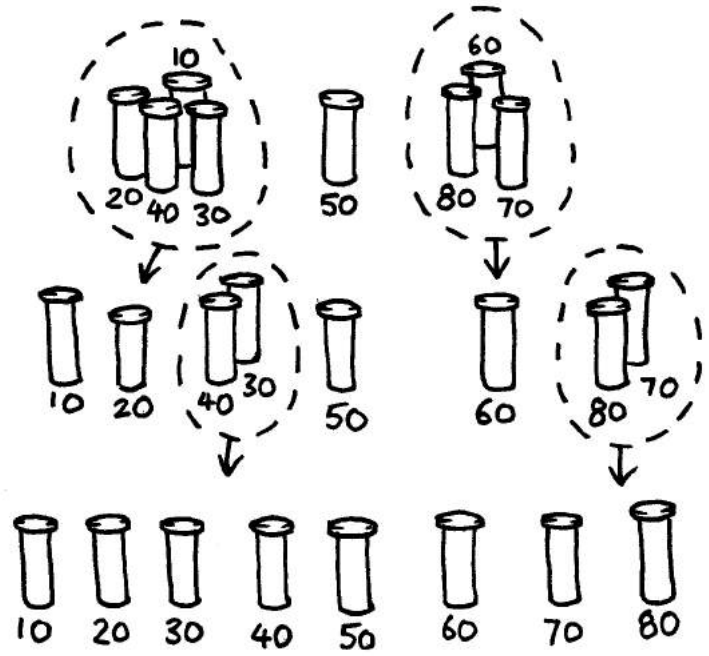
從全部的物件中隨機挑出一個物件，然後把他放在天平的一端。

然後將它與其他剩餘的物件一一比較，將較輕的放在左側，隨機選出的物件放在中間，較重的則放在右側。（有可能在結束比較時，兩邊物件的數量不一樣）

選擇其中一側，將該側所有物件重複上面的步驟。再將另一側的物件也做一樣的處理。

記得記住一開始選擇的物件要保持在中間。

剩餘的物件群一直重複這些步驟，直到每一群都只剩一個物件。一旦所有物件群都被分到只剩一個物件，所有物件便會被分成由最輕到最重。



整個過程要比較幾次？

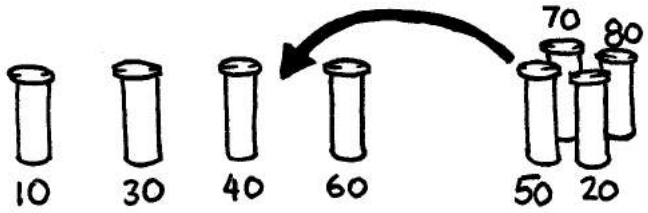
我們可以發現，快速排序法比選擇排序法更有效率，除非你一開始就選擇到最輕或最重的那個物件。如果你夠幸運，一開始就選到中間的重量，相較於需要 28 次的選擇排序法，快速排序法能以 14 次比較來完成排序。在任何情況下，快速排序法都不會比選擇排序法差，甚至可能好的多！

高手挑戰：假如快速排序法意外地一直選到最輕的物件，則需要幾次比較來獲得結果？

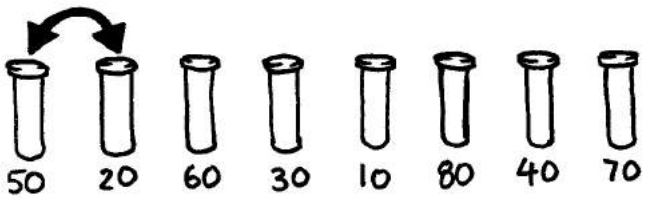
活動變化與延伸

排序還有很多種不同的方法。你可以用以下這些方法來試著把重量排序：

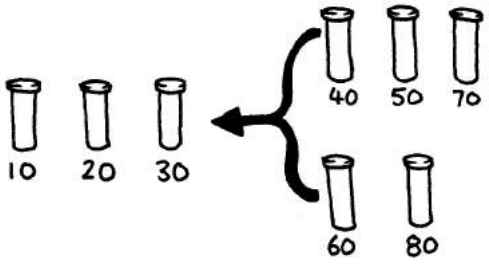
插入排序法是從尚未排序的物件群中挑出一個物件，把它插入已排序物件群中正確的位置（見下圖）。隨著一次次的插入，未排序物件群的規模會越來越小，而已排序物件群的規模則會越來越大，直到所有物件都排序完成。我們玩撲克牌時就常常利用這種方法來理牌。



氣泡排序法的做法是一次又一次從頭到尾檢查整個物件群，如果兩個相鄰的物件順序不對就把它們交換，整個物件群檢查完之後，再重頭開始檢查一次。如果沒有任何物件在檢查名單時被交換，就表示排序完成了。這個方法並不是很有效率，但是對某些人來說這種方法較容易理解。



合併排序法是另一種利用「分治法」來排序物件的方法。首先，將名單隨機分成大小相同的兩個群組（奇數個時則分成大小相近的兩份）。兩個群組都要分別做排序，最後再將兩個群組合併。合併兩個群組的做法相當容易 — 比較兩個群組最前面的物件，把較小的那個挑出來放入排序群組中。在下圖中，兩組的最前面的物件分別是 40 和 60 公克，所以下一個就把 40 公克的物件挑出來放入已排序群組中。那麼，最開始分成兩個群組以後，要怎麼分別排序？簡單 — 用合併排序法！最後，所有的群組會被分成只剩下一個物件，所以你不需要擔心做不完。



這個活動在說什麼？

在已排序的清單中尋找資訊會容易很多。電話簿、字典和書籍索引等都是用字母來排序。試想如果它們沒有經過這樣的排序，我們要找相關資訊時會有多不方便。若是將一串數字（例如一些支出）由小到大進行排列，就可以輕易的在排序的兩端找到最大和最小的數字，如果有數字重複的話那也是一目了然，因為它們一定會被排在一起。

實際上電腦在執行時花了很多的時間在把資料排序。所以，找到快速又有效率的排序方法一直是資訊科學家的重要工作之一。一些比較慢的方法如插入排序法、選擇排序法和泡沫排序法在特定的情形下還是很有用，但是在待排序的物件群規模很大的情況下，通常會選用快速排序法、合併排序法這些速度較快的排序法。舉例來說：如果有十萬筆資料要排序，快速排序法大概比選擇排序法要快上 2000 倍。若資料總數上升至一百萬筆，那麼兩種排序法所花的時間會差上 20000 倍。電腦常常要同時處理上百萬筆資料（很多網站擁有上百萬的訪客量，甚至連一張從便宜的照相機所拍的照片也有超過百萬的像素要處理）；選擇不同的演算法的差異可能是要花費一秒鐘還是五小時來完成一個完全相同的任務。不僅僅是時間上的延遲令人無法忍受，還有其他的成本，比方說要耗上 20000 倍的電力（如此浪費電力不僅有違環保理念，而且也會減少可攜式裝置的電池壽命）。所以，選擇合適的演算法才能有好的效果。

快速排序法使用到一種方法叫做分治法。在快速排序法中，我們不斷的把物件群組分成較小的兩個群組，然後再對這些較小的部分執行快速排序法。最後，整個物件群被反覆的分割，直到它們小到可以被簡易的排序。在快速排序法中，物件群被分割到只剩下一個物件 — 只有一個物件，那還需要排序嗎？儘管這些步驟看似十分複雜，但是在實際運行中，快速排序法卻戲劇性地比其他方法快速。分治法所使用的這種觀念稱為遞迴 （Recursion），就是演算法裡會反覆呼叫自己來解決問題。這聽起來很奇怪，但實作起來卻意外的管用。

解答與提示

1. 找出最輕物件的最佳方法為：按順序把每個物件都檢查一遍，並記住到目前為止最輕的物件就可以了。也就是說，比較兩個物件，並保留較輕的那個，再把被保留下來的物件和另一個物件比較，再次保留較輕的那個。一直重覆上述動作，直到所有物件都被比較過了。

2. 在天平上比較重量。這樣可以簡單地用三次比較，有時候甚至兩次比較就可以 — 如果學生瞭解什麼是比較的遞移律（也就是 A 比 B 輕，而且 B 比 C 輕的話，那 A 就一定比 C 輕。）

高手挑戰：

有一條捷徑可以把選擇排序法執行的次數加總。

要找到兩個物件中較小的物件需要比較一次，三個需要兩次，四個需要三次，以此類推。要排序八個物件，首先需要比較七次以找出最小物件，再來是花六次找出剩餘七個物件中的最小物件，以此類推。

最後我們就會得到排序總次數：

7 + 6 + 5 + 4 + 3 + 2 + 1 = 28 次比較。

n 個物件將需要 1 + 2 + 3 + 4 +⋯ + n – 1 次比較才能排序。

如果我們重新編組這些數字可以比較簡單的得出相加結果。

例如，要運算 1 + 2 + 3 + ⋯ + 20，就把它們重新編組成

(1 + 20) + (2 + 19) + (3 + 18) + (4 + 17) + (5 + 16) + (6 + 15) + (7 + 14) + (8 + 13) + (9 + 12) + (10 + 11)
= 21 × 10
= 210

一般來說， 1 + 2 + 3 + 4 ⋯ + n – 1 = n(n – 1)/2。

活動 8
與時間競賽 —排序網路

活動摘要
儘管電腦的運行相當快速，但凡事都有極限。其中一個提升速度的方法是把任務拆分成多個部分，並交由多個電腦執行。在這個活動中，我們使用排序網路來同時執行各種比較。

課程銜接
數學：數字探索 — 小於和大於

習得技能
- 比較
- 排序
- 寫出演算法
- 合作解決問題

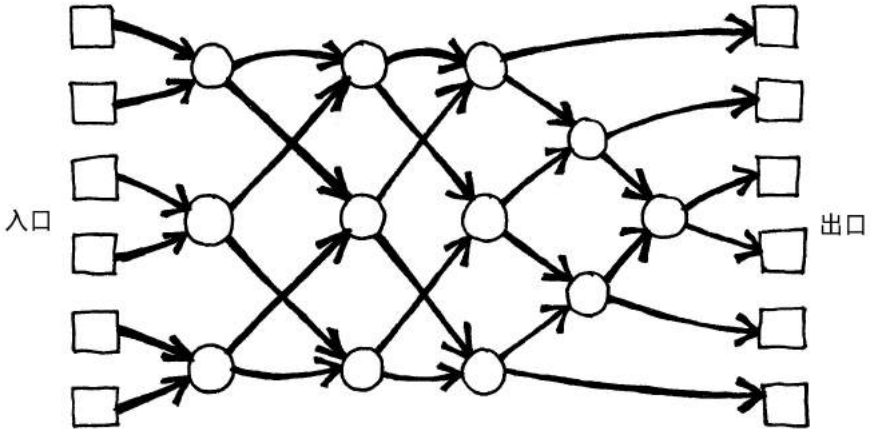
適合年齡
- 7 歲以上

所需素材
這是一個戶外團體活動。你需要準備：

- 粉筆
- 六張卡片兩份（把第 94 頁上的圖印出來並剪下）
- 碼表

排序網路

在活動之前，先依圖用粉筆在場地上標記此網路的圖形。



步驟說明

這個活動將會展示電腦如何使用「排序網路」來排序隨機產生的數字。

1. 每六個學生一組。每次只有一隊可以使用網路。

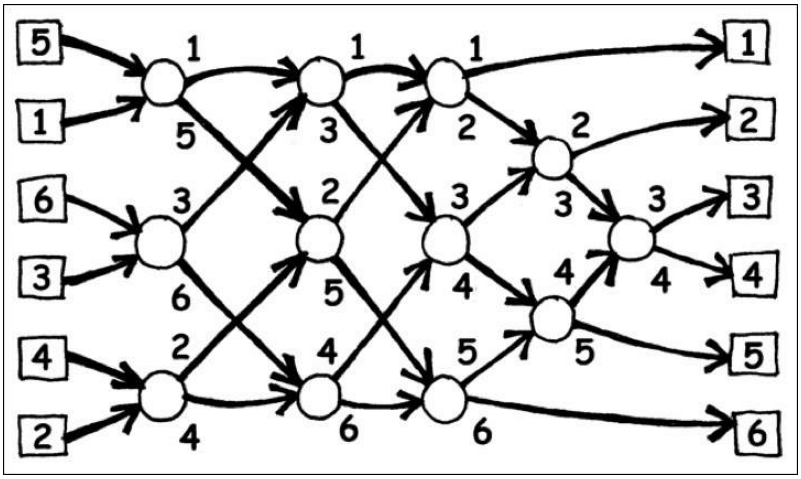
2. 每個團隊成員都拿一張有編號的卡片。

3. 每個成員各站在場地的左側（入口）的一個正方形內。成員手上的編號必須排列成混亂的順序。

4. 成員沿著地上所畫的線移動。到達圓圈後，你必須等待另一個人到達。

5. 當隊伍中的另一位成員到達你的圓圈後，兩個人就比較手上的卡片。 數字比較小的隊員，走左邊離開到下一個節點；數字比較大的隊員則走右邊離開。

6. 你到場地的另一側時的順序是正確的嗎？如果隊伍中有人犯了錯，則整個團隊必須重新開始。確定你是否已經了解網路節點的運作機制，也就是小的值往左，大的值往右。舉例來說：



素材：排序網路

1	2
3	4
5	6

156	221
289	314
422	499

活動變化

1. 當學生熟悉活動進行方式之後，用碼表來測量每一隊花多少時間來通過整個網路。

2. 使用數字大一點的卡片（例如：素材中所提供的三位數卡片）。

3. 新增更大的數字，讓學生要多花一點時間來比較。也可以改用單字，用字母順序來進行比較。

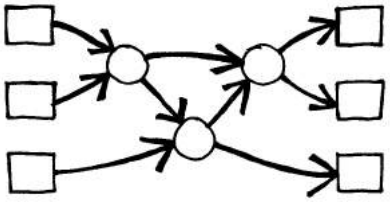
4. 這個活動也可以用來作為其他學科的練習，比如音樂。你可以比較印在卡片上的音符，並依照它們的音高或時間來排序。

延伸活動

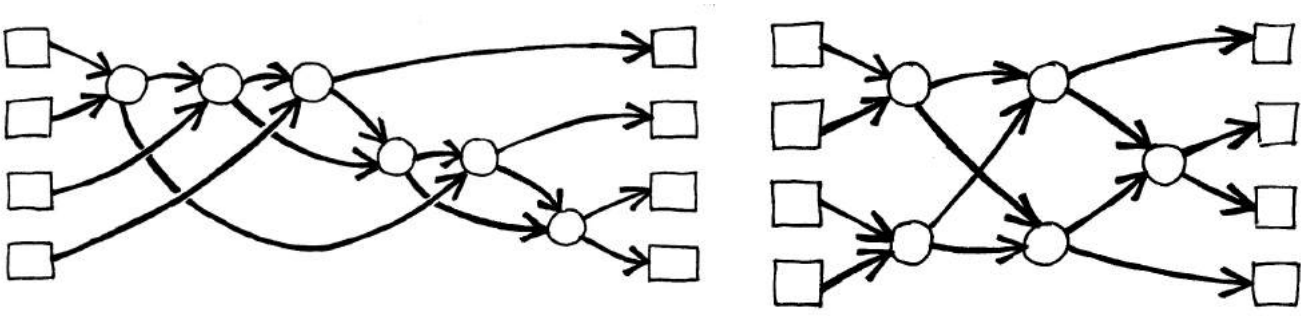
1. 如果改成數字較小的去右邊，數字較大的去左邊，結果會變成怎樣？（數字會按照相反的順序排序。）

如果整個網路反過來跑（從最右側到最左側），能正確排序嗎？（不一定。學生應該能夠找到一個例子，最後的順序是錯誤的。）

2. 試著設計出小一點或大一點的網路。像這邊這個網路只對三個數字做排序。讓學生自己去發想。

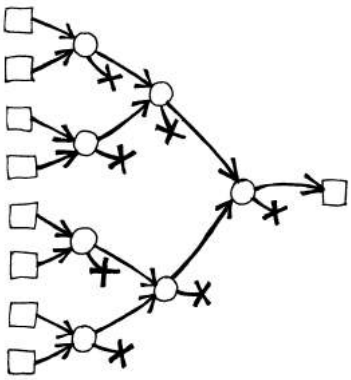


3. 下面是針對四個數字做排序的兩個不同的網路。哪一個比較快？
（第二個比較快。因為第一個網路是一個個做比較，而第二個有一些比較可以同時被執行。第一個網路是「串行處理」(Serial processing) 的一個例子，第二個網路則使用「平行處理」(parallel processing)，因此跑得更快。）



4. 嘗試做大一點的排序網路。

5. 網路也可以用來找到輸入的最小值或最大值。例如，這裡是一個接受八個輸入的一個網路，最後的輸出只有一個，就是所有輸入中的最小值（其他的值都會停留在網路中的死角）。



6. 日常生活中哪些事情可以或不可以用平行處理的觀念來加速進行？例如，做飯時只用一個爐子就會慢很多，因為所有的菜都必須一個接一個來烹調。怎麼樣的工作可以藉由僱更多的人來更快地完成？怎麼樣的工作不能？

這個活動在說什麼？

現在的生活越來越倚賴電腦，因此我們希望電腦能盡快處理各種資訊。

一種加速處理的方法是改進程式的演算法，讓它計算地快一點（就像我們在活動 6 與活動 7 所學到的）。

另一種方法是同時讓數台電腦處理同一個工作的不同部份。例如，在活動中我們用來排序六個數字的網路，雖然總共有 12 次比較要做，但最多同時可以進行三個比較。這也就是說，整個工作的時間不是一次比較的時間的 12 倍，而是可以縮短到 5 倍而已。換言之，這個平行處理的網路比起一次只做一次比較的串行處理的網路要快上兩倍。

但是並不是所有的工作都能透過平行處理的網路來加快速度。舉一個例子，想像一個人要挖一個十公尺長的水溝。如果十個人同時進行，每個人只需要挖一公尺，整個工作完成的速度會快十倍。但是如果是挖一個十公尺深的洞呢？找十個人來一起工作就沒什麼作用了 — 總不能在你開挖的同時叫另一個人從第二公尺深的地方開始挖吧。

電腦科學家至今仍一直努力研究如何讓電腦平行處理的方法。

活動 9
泥濘城市 — 最小生成樹

活動摘要

我們的社會被許多的網路所連結著，包括電話線路、需求供應鍊、網際網路、公路網等等。針對一個特定的網路，我們常常有許多的選擇來讓我們決定走哪條路、管線、或是無線電聯絡站。我們需要找到有效率的方式來連結這些網路中的物件。

課程銜接

• 數學：幾何學 — 探索形狀與空間：尋找地圖中的最短路徑

適合年齡

• 9 歲以上

習得技能

• 解決問題的能力

所需素材

• 活動學習單：泥濘城市問題（第 100 頁）
• 籌碼或紙板做成的方格子（每個學生約需 40 個）

泥濘城市

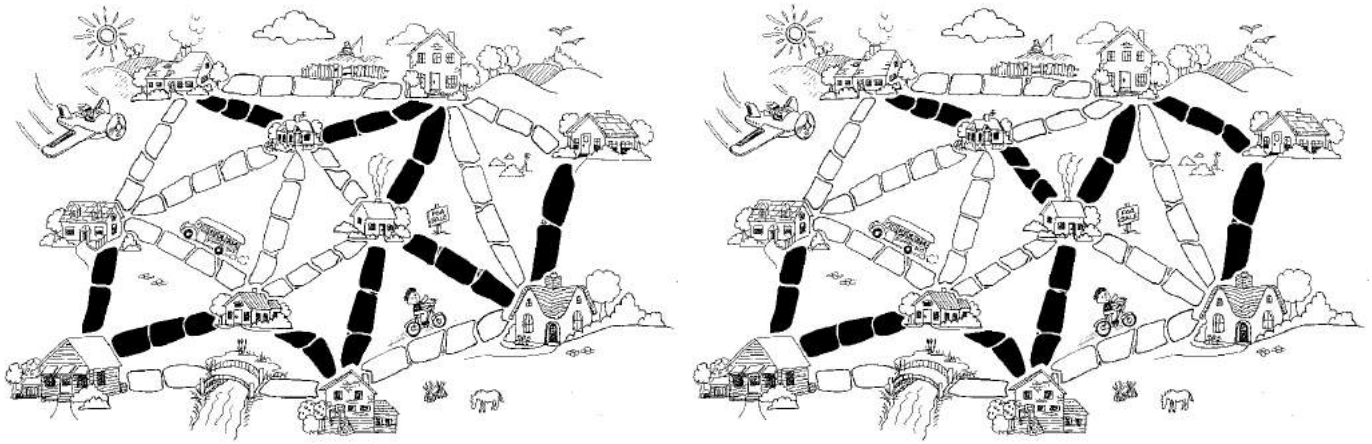
活動介紹

這個活動會告訴你電腦是如何針對現實生活的問題找出最好的解決辦法，像是房子之間的電力線是如何連結的。讓同學們用第 100 頁的活動學習單，裡面會說明何謂「泥濘城市問題」。

活動討論

跟同學分享彼此的解決方式，他們到底用了哪些方法呢？

要找到最佳解的一個不錯的策略如下：從一張空的地圖開始，逐一放置籌碼。籌碼的數量取決於路的長度，路越長放越多，已連接的房子就不做重複的連接，直到所有的房子都連接上了。若改變選擇而選另一條有相同長度的路，則會找到不同的答案。兩個可能的答案如下圖。



另一種策略是先把路都填滿，然後移去不要的道路，但這樣比較費力點。

你會在現實生活中的哪裡找到這些網路呢？

電腦科學家稱這些網路的表示形式為「圖形」（grpahs）。真正的網路可以用這些「圖形」來表示，以用來解決像是城市間的最佳路徑規劃，或是跨國的飛機航班規劃等等的問題。

也有許多的演算法能運用在圖形上，像是尋找兩點之間的最短距離，或是尋找能走過所有地點的最短路徑。

活動學習單：泥濘城市問題

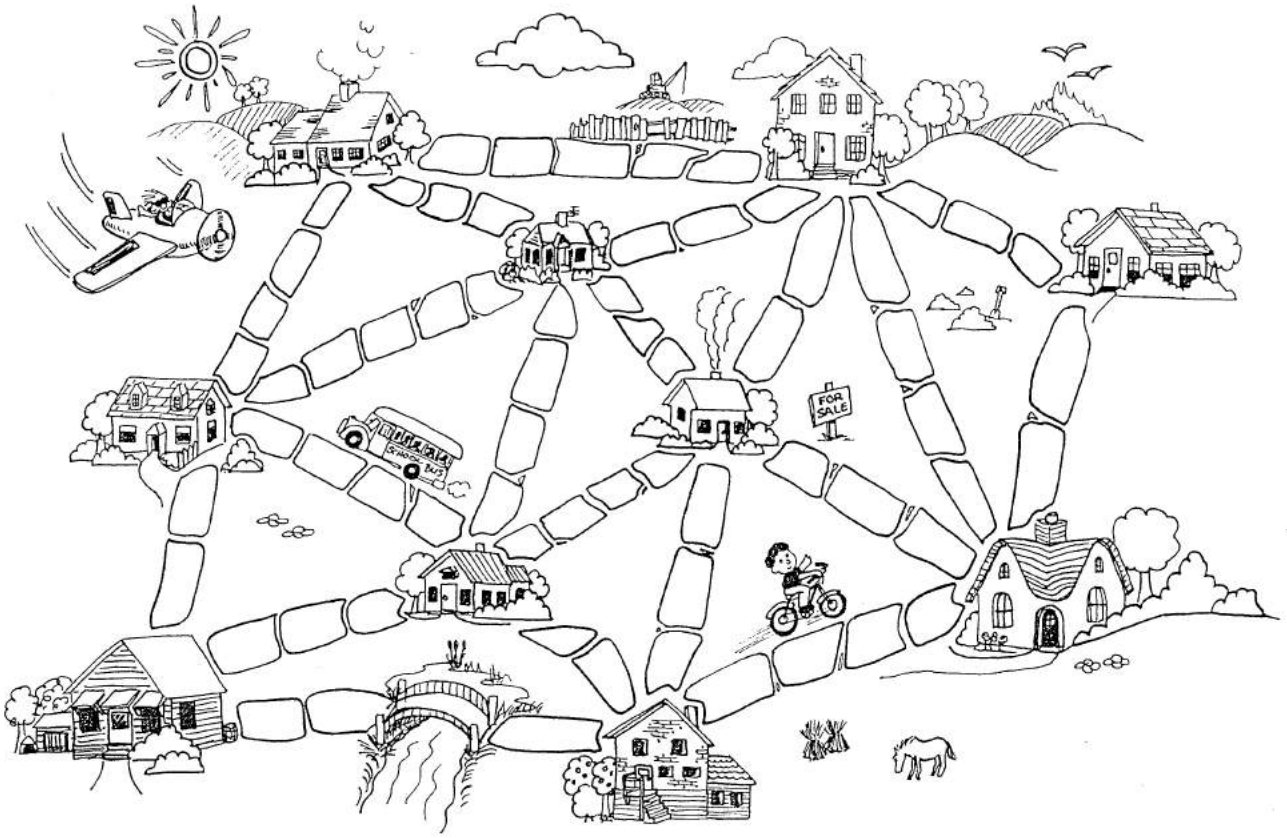
很久以前有一個沒有鋪道路的城市。每次在暴雨過後，這個城市會變得特別難行走，因為地板會變的非常泥濘－車子會卡在泥中，人們的鞋子會髒掉。市長決定某些街道一定要鋪路，可是卻不想多花不必要的錢，因為這城市還想要蓋一座游泳池。因此市長指定了兩個條件：

1.鋪足夠的路讓人們可以從自己的家裡，經過鋪好的路到任何人的家裡。

2.鋪路的花費越少越好。

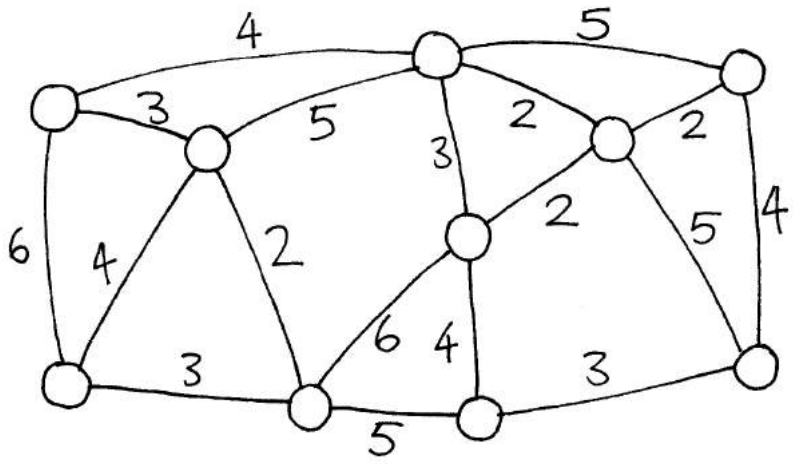
以下是城市的平面圖。每棟房子之間鋪的石頭的數量代表鋪那條路線的花費，請試著找到能夠連接每個房子，並且花費最少的路線。

你會用什麼方法來解決這個問題？



活動變化與延伸

這裡有另一個表示城市和道路的方式：房子用圓表示、泥濘的道路則用線表示、而道路的長度則用數字寫在線旁邊。



電腦科學家和數學家常用這種圖來表示這類問題，他們稱之為「圖形」(graph)。一開始可能會覺得有點混淆，因為在統計學裡也會用「圖形」來表示資料，例如柱狀圖等。但是統計學中的「圖形」跟資訊科學中的「圖形」是沒有關係的。資訊科學中的「圖形」的「長度」不是用來代表資料的統計數字，也不必照比例尺來畫。

試著自己設計一個泥濘城市問題並給你的朋友解答看看。

你能找到一個規則來描述需要多少路或是連結的最佳解嗎？它跟這個城市有多少房子有關嗎？

這個活動在說什麼？

假設你在設計公共事務，像是電力、瓦斯、水要怎麼傳遞到新的社區。電線或是水管的網路需要從所有房子連接到提供的公司，每個房子都必須要連接到這個網路，但是從房子連接到公司的路線不是那麼重要，只要路線存在就好。

設計一個網路總長度最小的任務又叫做最小生成樹（minimal spanning tree）問題。

最小生成樹不只對瓦斯或電力網路有用，也能幫助我們解決電腦網路、電話網路、輸油管、航線等等的問題。然而，當決定人們旅行的最佳路線時，你需要同時把這個旅程的方便性和花費考慮進去。沒有人想要因為比較便宜而浪費很多時間搭飛機繞遠路。泥濘城市演算法對這些網路可能沒那麼有效，因為它只是把路和飛行路徑的總長度最小化而已。

最小生成樹作為一個解決其他圖形問題方式也是很有用的，像是「旅行商人問題」（Travelling Salesperson Problem, TSP）。TSP 的目的是要找到可以拜訪到每個點的最短路徑。

解決最小生成樹問題，有一些很有效率的演算法可以用。一個找最佳解的方法就是從沒有任何連結的空白開始，然後依他們大小的升冪順序加入，但注意只把那些先前還沒連結的節點加入聯結即可。這種方法叫做 Kruskal 演算法，由 J.B. Kruskal 在 1956 所發表。

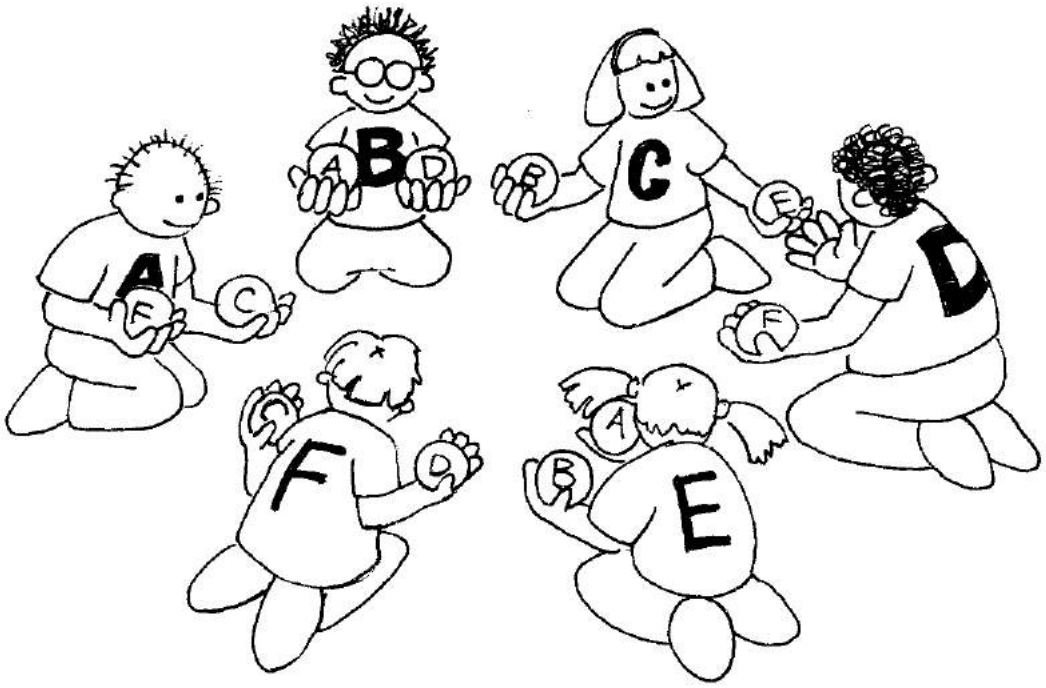
不過對於圖形的許多問題，包含「旅行商人問題」，電腦科學家還沒找到可以夠快得到最佳解的方式。

解答與提示

活動變化與延伸（第 101 頁）

城市裡有 n 個房子需要多少路或是連結呢？結果發現最佳答案是剛好有 n-1 個連結，因為這永遠是足夠將 n 個房子連在一起，而多加任何一條連結則會在房子間出現不需要的路。

活動 10
橘子遊戲 — 網路中的路由與死結



活動摘要

當有很多人同時使用同一個資源（例如很多車擠在道路上，或很多訊息要透過網路傳送），有可能會造成「死結」。因此，為了避免這種狀況發生，就需要協同作業。

課程銜接

• 數學：發展邏輯和推理

習得技能

• 合作解決問題
• 邏輯推理

適合年齡

• 9 歲以上

所需素材

每個學生都需要：
• 兩個橘子或網球，上面標有相同的字母；或兩顆水果（最好是素描用的人造水果）
• 具有字母的名稱標籤或貼紙，或有顏色的帽子，徽章或蓋子，以便與水果配對

橘子遊戲

活動介紹

這是一個合作解決問題的遊戲，目標是讓每個人拿到標有自己的字母的橘子。

1. 五人以上的學生坐成一圈。

2. 學生都貼有一個字母（使用名稱標籤或貼紙），或者每人都有一種顏色（可以是帽子，或衣服的顏色）。如果使用英文字母，那每個學生最後手上會有兩顆橘子，橘子上面的字母要跟身上的相同。但其中有一名學生只會有一顆橘子，確保始終有一隻手空著。如果是使用水果，那每個學生最後手上會有兩顆水果與身上的顏色相同 — 例如，戴黃色的帽子的學生，手上可能有兩根香蕉，帶綠色的帽子的學生，手上可能有兩顆綠色蘋果等等。當然，其中會有一名學生最後手上只有一顆水果。

3. 遊戲開始前，把橘子或水果分給在圈內的學生。 每個學生手上會有兩顆，除了一個學生只有一顆。 （但要注意，學生開始時拿到手上的橘子或水果不能與身上的字母或顏色對應到。）

4. 學生開始傳橘子或水果給旁邊的人，直到每個學生拿到標有其字母或顏色的橘子或水果。傳遞時必須遵循以下兩條規則：

(a) 一隻手只能拿一顆水果。
(b) 水果只能在身旁的學生手空著的時候傳遞過去。（學生可以選擇傳他們手上兩個橘子中的任何一個）。

學生們很快就應該發現，如果他們太「貪婪」（也就是拿到自己該拿的水果之後就不再傳了），那麼整組可能很快就傳不下去了。因此，要強調的是，光是個人拿到該拿的水果是「贏」不了的。目標是整組都要拿到正確的水果才行。

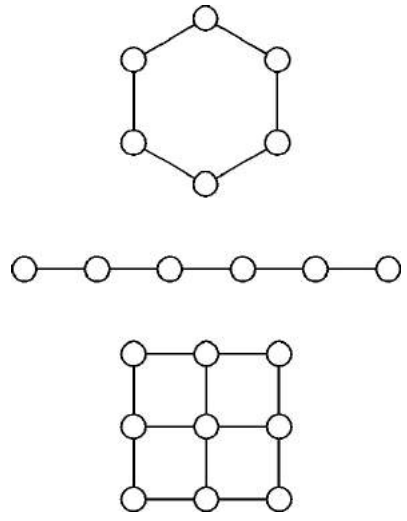
活動討論

學生做了什麼樣的策略來解決這個問題？在現實生活中，有沒有經歷過「死結」？ （比方說堵車，或者跨年夜看完煙火有極大量的人同時擠進捷運車站等等。）

活動變化與延伸

試著讓一組的學生數變多或變少。

1. 讓學生試試看，有沒有新的策略出現。
2. 活動進行時不准交談。
3. 嘗試不同的方法，例如坐成一直線，或讓某些學生身邊有比較多可以傳遞的對象。這裡有一些建議的變化形。



這個活動在說什麼？

路由和死結是在許多網路中都會遇到的問題，如道路系統，電話和電腦網路系統等等。工程師們花上大量的時間，試圖搞清楚如何解決這些問題，以及如何設計網絡，讓這些問題更容易被解決。

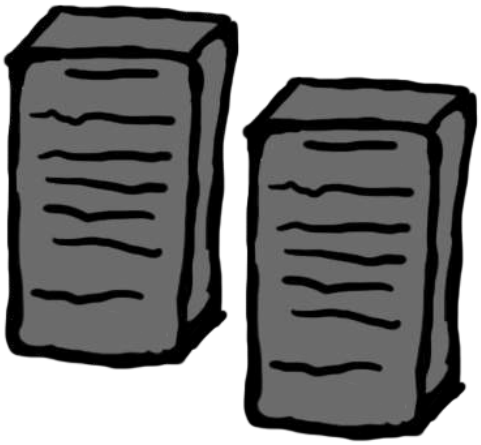
路由，擁塞和死結可以在不同的網路裡造成許多令人沮喪的問題。想想看，有沒有遇過大塞車？或是有沒有看過在小巷子裡幾台車進退不得，每台車都無法動彈的狀況？

有時，電腦會在商業活動中「當機」（如銀行）。這個問題是因為通訊網路的「死結」所引起的。如何設計網路，讓路由的選擇更容易，更高效和盡可能避免阻塞的狀況，是許多工程師所面臨的棘手問題。

有時不止一個人想要在同一時間取得相同的資料。如果一些資料（例如某個客戶的銀行戶頭餘額）正在更新，那在更新過程中把它「鎖定」是很重要的事。如果沒有把它鎖定，別人可能也會在同一時間進行更新，那戶頭的餘額可能就會出錯。但是，如果鎖定的項目被另一個鎖定的項目所干擾，這時就可能會發生死結。

在電腦的設計中，一項最令人振奮的發展是平行計算的發明。我們可以把數百個或數千個類似 PC 的處理器在網路中組合，就變成相當於一台功能強大的電腦。而這些電腦在平行運作時，想像一下，就好像這些電腦用極快的速度在玩橘子遊戲一樣。

活動 11
石板傳送－網路通訊協定



活動摘要

電腦通過訊息在網際網路上彼此溝通。然而，網際網路不是很可靠，有時這些訊息會遺失。因此，我們可以透過在這些訊息中加上一些資訊，確保這些訊息被成功地發送。這些資訊就叫做協定。

課程銜接

- 數學：開發邏輯和推理能力
- 英語：溝通，人際間的傾聽能力

習得技能

- 合作解決問題
- 邏輯推理

適合年齡

- 9 歲以上

所需素材

- 每位學生會需要：很多空白的「石板」
- 每個發送訊息者會需要：一套訊息行為卡
- 老師需要：計時器

石板傳送

活動介紹

在這個活動中，學生們需要思考不同的溝通方法是如何運作的。通過適當地觀察規則和流程，學生們會認識到什麼叫做通訊協定。而透過在角色扮演的劇本中操作，學生們可以測試他們自己的協定，在不可靠的環境下是否能運作，進而瞭解網際網路上的封包交換方法，特別是 TCP / IP 協定。

活動準備（30 分鐘）

1. 首先準備卡片。把下方的行為卡印出來並剪下。這些行為卡是遊戲的基礎。

2. 接下來，選擇一些要讓學生傳送的訊息。重要的是，訊息本身不要是一般的句子，或任何可以藉由它們的結構手工組合回去的訊息。像是 ”1LHC255HD(RLLS” 就很合適。或是電話號碼之類的一串數字也可以。

3. 把「石板」印出來。每個石板的大小要控制在只能放入六個字元或數字，所以沒辦法一次把整個訊息放在一個石板上。每名學生大約需要 30 個石板，數量取決於你打算讓遊戲玩多久。

注意：行為卡有三種類型；延遲傳遞，不傳遞，傳遞。調整它們之間的比例會影響訊息發送者的品質。 更多的「傳遞」卡，表示訊息發送者會更可靠。更多「延遲傳遞」和「不傳遞」的卡片，表示整個網路是不太可靠。這些卡片就類似於一個電腦網路或通訊頻道。

玩遊戲

1. 將全班分成兩人一組。兩個人必須分開坐，不能看到彼此或讓彼此溝通。這個遊戲裡這一點非常重要。 把兩個人分在兩個房間是最理想的，不過在一個教室裡讓他們坐在不同側背對背應該就可以了。

2. 給每組的其中一人一則訊息，讓他傳遞給他的夥伴。

3. 把行為卡洗牌，並選擇一個訊息發送者。老師本身可以當訊息發送者，或是如果班上學生數是奇數，可以找一個學生來做。如果你班級很大，可能需要一個以上的訊息發送者。

4. 學生開始在自己的石板上寫字，並把它交給訊息發送者。石板上至少應該要有夥伴的名字在上面。

5. 訊息發送者翻起最上面那張行為卡，把它讀出來，並依照行為卡的指示行動。

6. 重複步驟 4 和 5。

石板傳送

在 5 分鐘左右的混亂和挫折後，學生們應該可以開始體認到，傳輸協定中如果只有收件者的名字是不夠的。現在可以暫停，並開始討論他們遇到的第一個問題是什麼？是順序嗎？如果利用這 6 個字元中的一個，在裡面放入石板的序號，是不是個好主意？但這表示每個石板中實際可以傳送的資料會變少 — 就我們現在可使用的石板的數量而言，這意味著什麼？

再玩一段時間之後，學生們可能會注意到其他的問題，而這些問題應該也要讓大家討論。可能出現的問題包括石板遺失，不知道這個板有沒有被傳送到，不知道是否要重新傳送某個石板。你可以建議一些解決方法，像是收到石板後，送出一個確認收到的石板給原發訊者，原發訊者等著這些送回來的確認之後再送另一個 -- 這表示負責接收的學生們也需要空白的石板以便發送確認訊息，而且在再次玩遊戲之前，他們必須先協調回應的內容中要寫什麼表示確認收到。

這場遊戲將需要至少兩名學生，但我們建議盡可能越多越好。如果你的班級人數很多，可以考慮用數名訊息發送者。 然後跟學生們討論：如果有很多的訊息發送者，會發生什麼事？ 如果只有一個，又會什麼發生事？

立即傳送石板	先送下一個訊息 再送這一個
立即傳送石板	先送下一個訊息 再送這一個
立即傳送石板	先送下一個訊息 再送這一個
立即傳送石板	不傳送， 丟棄此訊息
立即傳送石板	不傳送， 丟棄此訊息

收件者：

寄件者：

收件者：

寄件者：

收件者：

寄件者：

收件者：

寄件者：

收件者：

寄件者：

收件者：

寄件者：

收件者：

寄件者：

收件者：

寄件者：

石板傳送

在一個古老的城市中，有著許多非常重要的管理者。這些管理者決定城市如何運作和負責做一些十分重要的決策。他們每個人的住處遍布整個城市且都沒有住在一起。

這些管理者時常想要聯絡彼此，所以他們需要在城市內傳送和接收訊息。人們是用門牌號碼來分辨管理者，而管理者們每個人都可使喚一群負責送信的信使。

傳送訊息只有一種方法，就是把訊息寫在巨大的矩形石板上，然後由信使帶著這些大石板去目的地。這些石板有著固定的大小，而且每塊石板只能填寫 6個片段的訊息。一個片段訊息可能是一個字母或一個數字。一則訊息通常會被拆散記錄到許多石板上，而且由於石板的重量很重，所以信使一次只能送一個石板到目的地。

但管理者不能確保信使會一直正確的把訊息傳到目的地，因為信使既懶散又健忘。他們常常在工作時間偷閒，休息一大段時間，甚至試著逃離這個城市。

於是這些管理者們試著去想出一個方法來讓他們的聯絡管道是可被信賴的，他們想要設置一套他們全都會遵守的規則。藉著這套規則，他們能分辨他們的訊息是否已送出和訊息的正確性。這些管理者已經決定收信者的地址應該要被寫在石碑上。

在你的團體中，你的任務是去制定出管理者們用來聯絡的規則 …

這個活動在說什麼？

在網路上資料是被分割成許多封包來傳送。但是這些封包傳輸時經過的管道不一定一直都是穩定的。個別的封包有時會受損、遺失或不照順序送達。

在「石板傳送」這個活動中，石板就是封包，而刻在石板上的內容就是資料。封包包含資料和標頭資訊。標頭資訊的大小會影響你可以傳多少的資料。要注意的是，因為封包的大小有上限，所以標頭資訊的大小與資料大小要達到一個平衡點。

學生會發現他們需要協調出他們用來交換資料的資訊，例如封包號碼和總封包數，或是這個封包是不是用來確認的封包。因為這些資訊會占用資料空間，所以實際上會需要使用更多的封包來傳送資料。

網際網路通訊協定，像是 TCP 和 UDP，協定本身的設計會考慮平衡這些因素，來建立一個可信賴且有效率的資料傳輸協定。

這個活動是改編自一個 "Computer Science Inside" 的計畫 (csi.dcs.gla.ac.uk)。

第三部份
告訴電腦要做什麼 — 如何表達程序

電腦是如何運行的？

電腦會依照指令執行 — 每秒執行數百萬個指令。想要操作一台電腦，你只需要給它正確的指令。但是這並不像聽起來那麼簡單！

當我們聽到指令後，會依照我們的常識去分析它的意思。如果有人說「走過那扇門」，我們會知道他並不是指把那扇門撞破再走過去 -- 他的意思是走經由門的那條路，如果門是關著的，那就把門打開！然而，電腦跟我們是不一樣的。當我們在操作移動機器人時，我們的確要採取一些安全措施，避免機器人因為從字面上解讀指令所造成的破壞 — 例如：走過那扇門，結果機器人直接把門撞壞。不經由「思考」而直接確實遵守指令，確實需要一些時間來適應。

以下的兩個活動可以給我們一些關於 — 只能接受固定指令，相當死腦筋的 — 機器溝通的想法。

第一個活動會告訴我們一種電腦用來辨識單字、數字與字串等符號的「機器」。這些「機器」被稱為「有限狀態自動機」 (Finite-state automata)。

第二個活動則介紹我們如何與電腦溝通。一個好的工程師應該要學會如何用可以依照字面解釋的特定組合指令集來告訴電腦如何運作。這些指令集就稱為程式。有很多不同的程式語言可以供工程師選擇來編寫這些指令，但是我們將使用一個簡單到沒有電腦也可以使用的語言。

活動 12
尋寶遊戲－有限狀態自動機

活動摘要

電腦語言通常需要處理一連串的符號，像是在字母、文件中的單字，甚至是在另一個程式中的文字。電腦科學家常會使用有限狀態自動機來處理。所謂的「有限狀態自動機」（Finite-state automaton, FSA）會依照一組指令集讓電腦辨識單字或字串所組成的符號。我們將使用一種類似 FSA 的東西：藏寶圖！

課程銜接

• 數學：發展邏輯和推理 — 用文字和符號來描述並繼續特定樣式
• 社會研究

習得技能

• 簡易的地圖閱讀
• 辨識特定樣式
• 邏輯能力
• 遵守指令

適合年齡

• 9 歲以上

所需素材

你需要：
• 一組島嶼卡（指令要隱藏好，避免被那些想要繪製島嶼地圖的人看到！）
• 複印素材：島嶼卡（自第 130 頁開始）並剪下來。沿虛線折並用膠水黏好，這樣卡片的正面會有島的名字而背面會有指令。

每個學生需要：
• 活動學習單：尋找金銀島（第 129 頁）
• 鉛筆或原子筆

延伸活動中，每個學生需要 :
• 活動學習單：金銀島（第 135 頁）
• 活動學習單：神秘的硬幣（第 136 頁）

金銀島

活動介紹

你的目標是找到金銀島。友方的海盜船沿著固定路線在島嶼間行駛，提供旅行者交通服務。每個島嶼有兩艘駛離的船隻，A 和 B，可讓你選擇搭乘。你需要找到通往金銀島的最佳路線。你可能需要在你所抵達的每個島呼叫 A 船或 B 船其中一艘船（但不能兩艘船都呼叫）。在島上的人會告訴你你的船接下來會帶你到哪裡，但是海盜並沒有包含所有你能到達的島嶼的地圖。用你自己的地圖來記錄你將要前往的地方和你所搭上的船。

示範

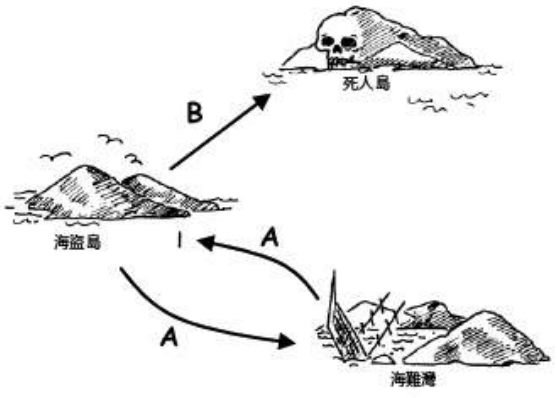
（注意：這和實際活動所使用的地圖是不同的）

用一塊板子畫出底下的三個島嶼的圖。

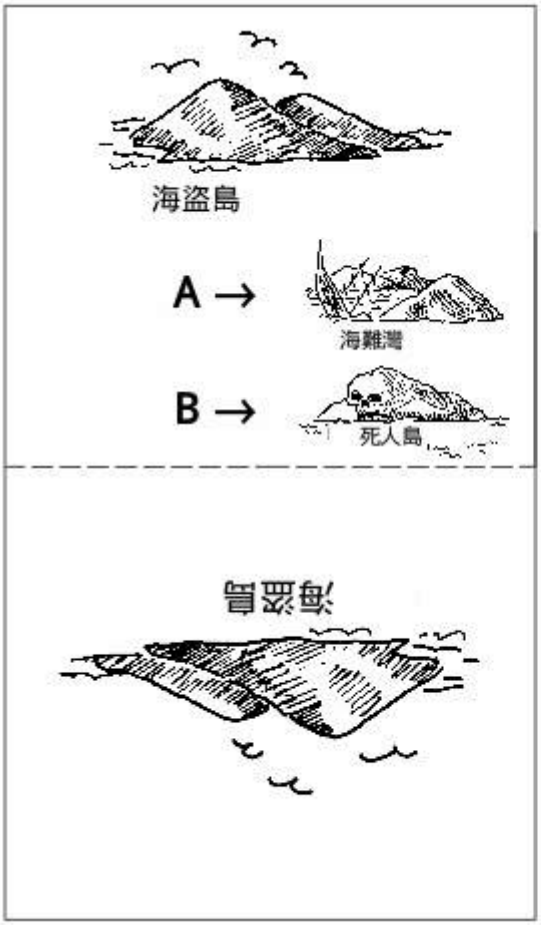
畫出下兩頁的三張卡片，並且讓三個學生每人拿一張。注意，在卡片上的路線和我們等等要進行主要活動是不同的。

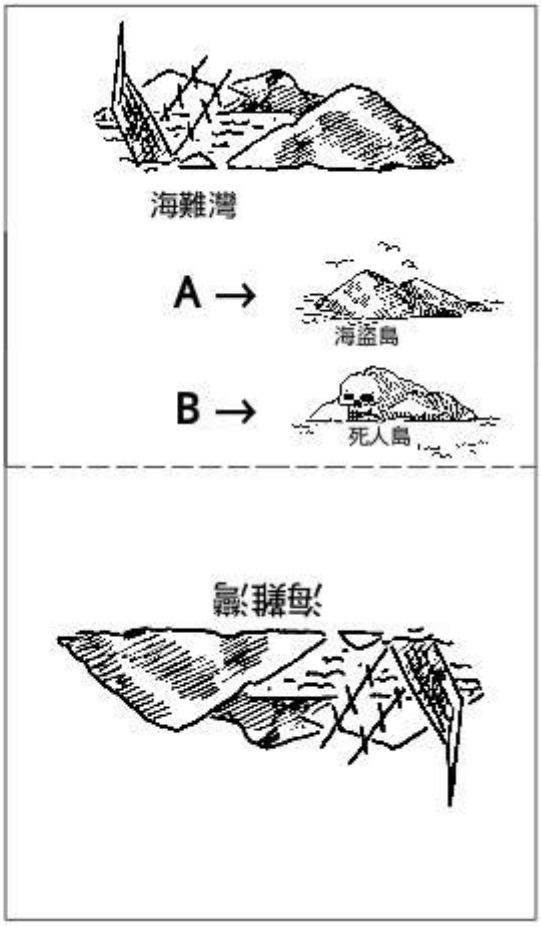
從海盜島出發呼叫 A船。拿著卡片的學生要指引你到海難灣。在地圖上標記路線。在海難灣再次呼叫 A 船。你將被指引回海盜島。在地圖上標記這條路線。這次呼叫 B 船。在地圖上標記這條路線。這條路線到達死人島，然後你會被困在那裡！

你最後的地圖會是這樣：

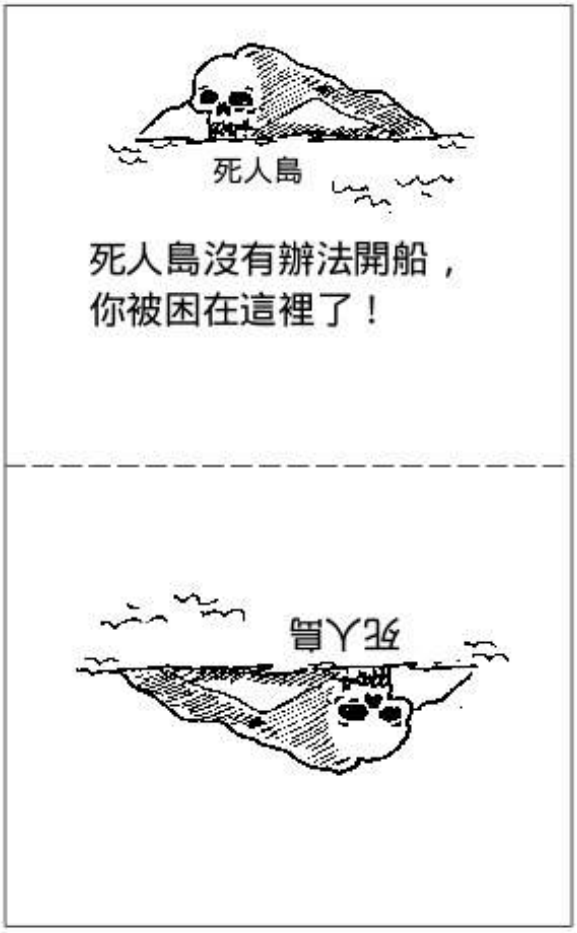


展示用卡片





展示用卡片

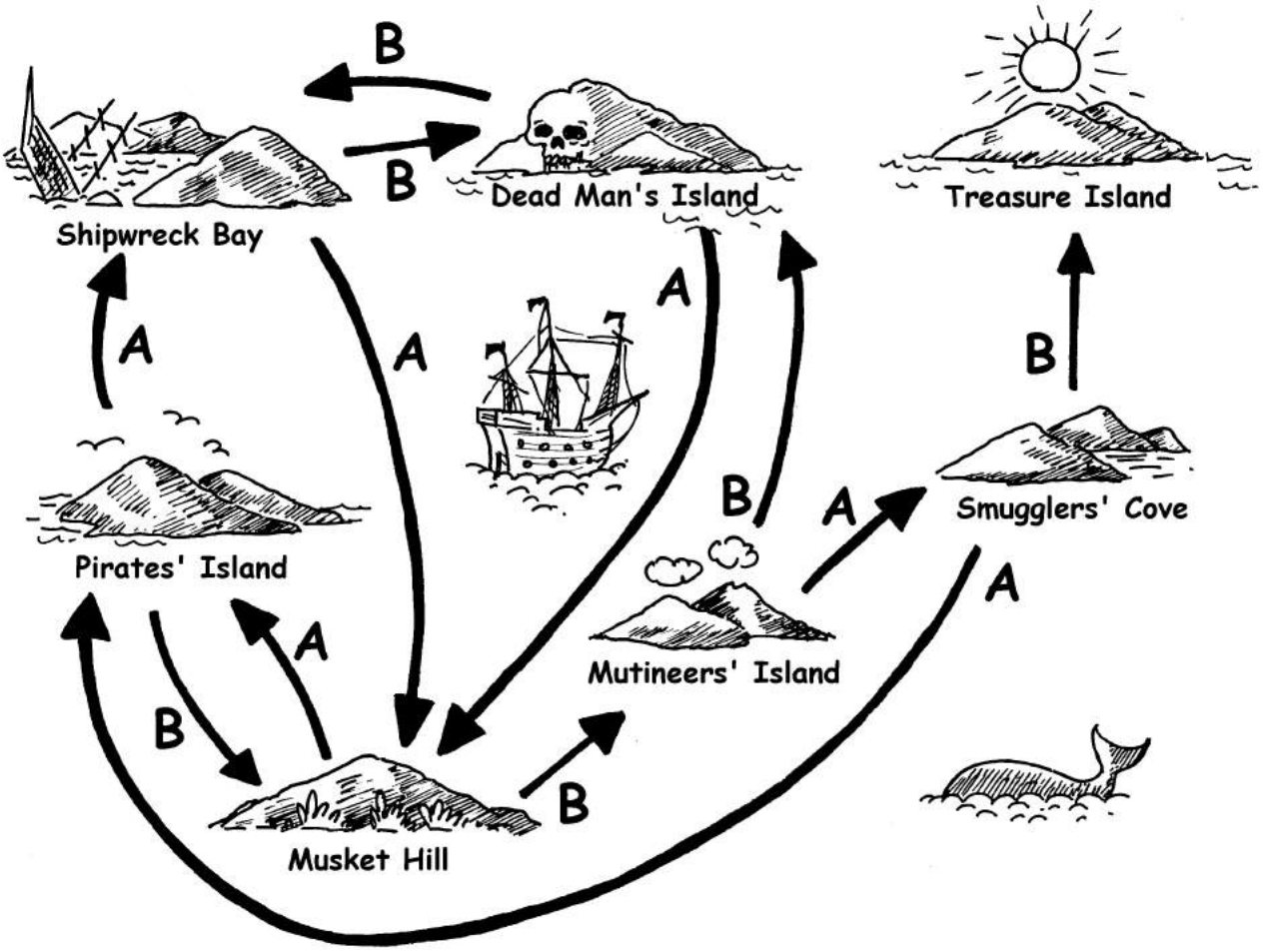


活動進行

選 7 位學生當「島嶼」。這些學生會拿著他們所代表島嶼的卡片，卡片的背後會有秘密指令。將他們隨機分配到教室或操場的各處。剩下的學生會拿到空白地圖，他們必須從海盜島航向金銀島，並在地圖上仔細做標記。（最好一次只讓一個學生出去，讓他們無法預先知道路線。）

動作快的人：試著多找幾條路徑。

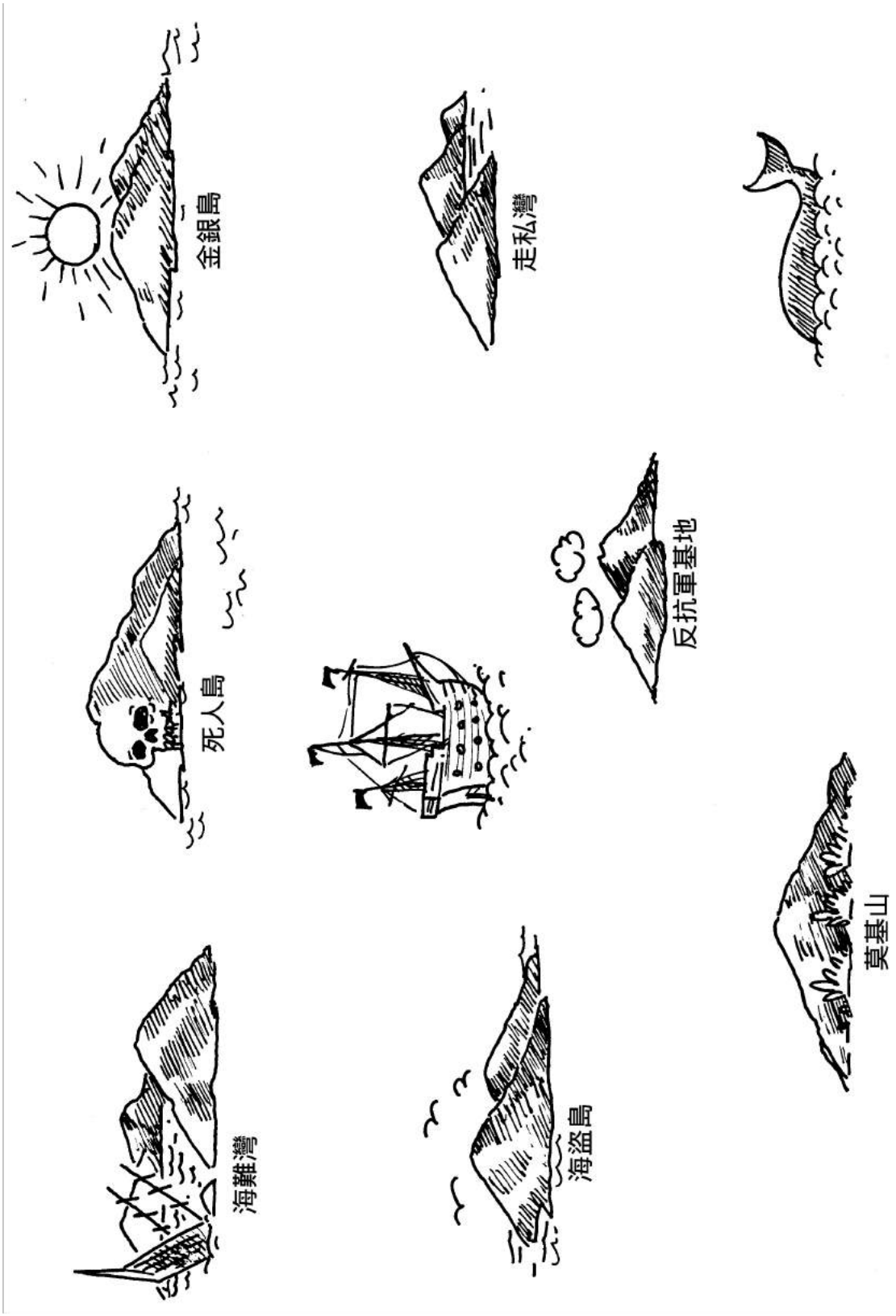
完整的地圖是這樣的：



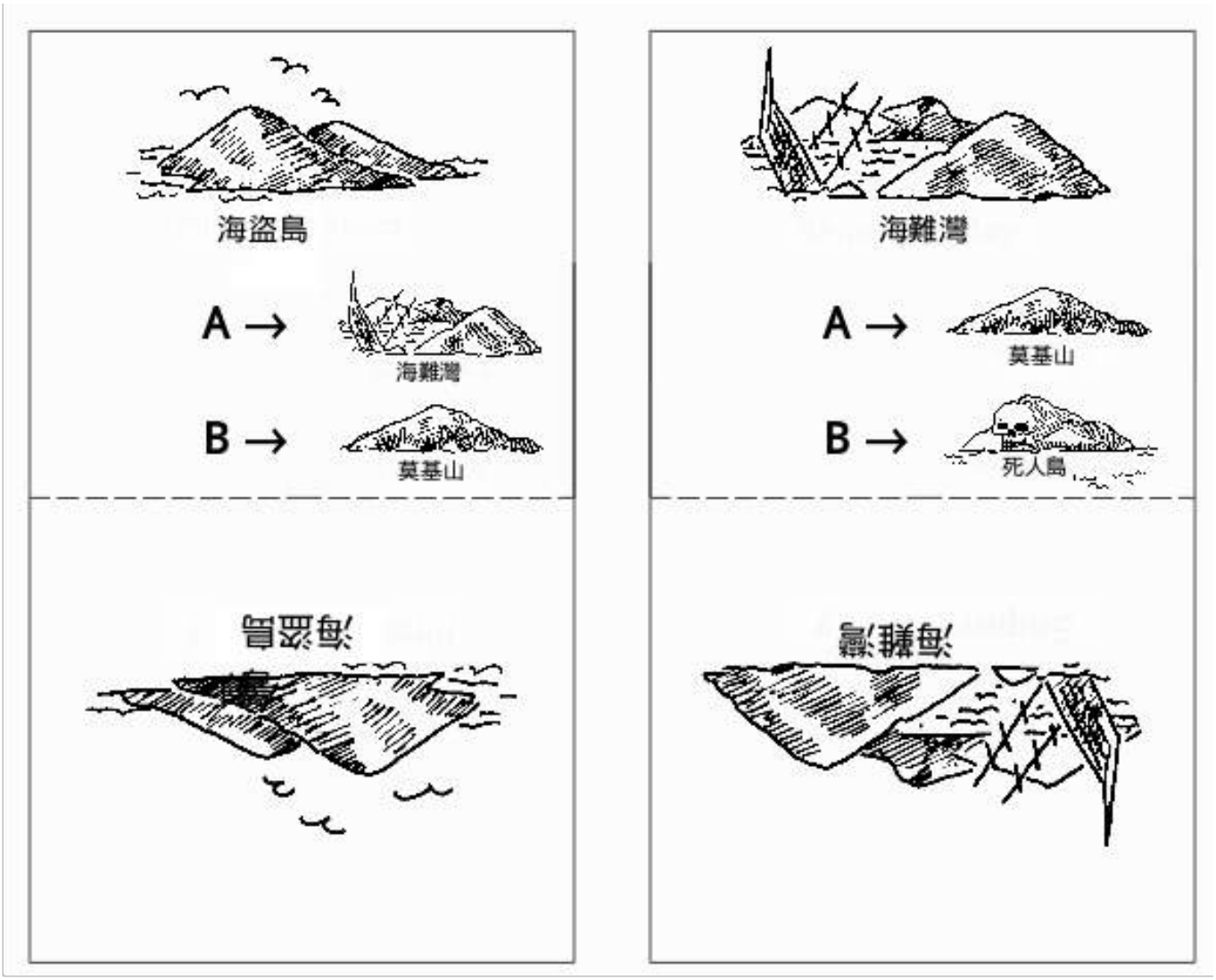
活動討論

最快的路徑是哪一條？哪一條路徑會非常慢？有些路徑可能有迴圈。你能找到一個例子嗎？（像是：BBBABAB 跟 BBBABBABAB 都能到達金銀島。）

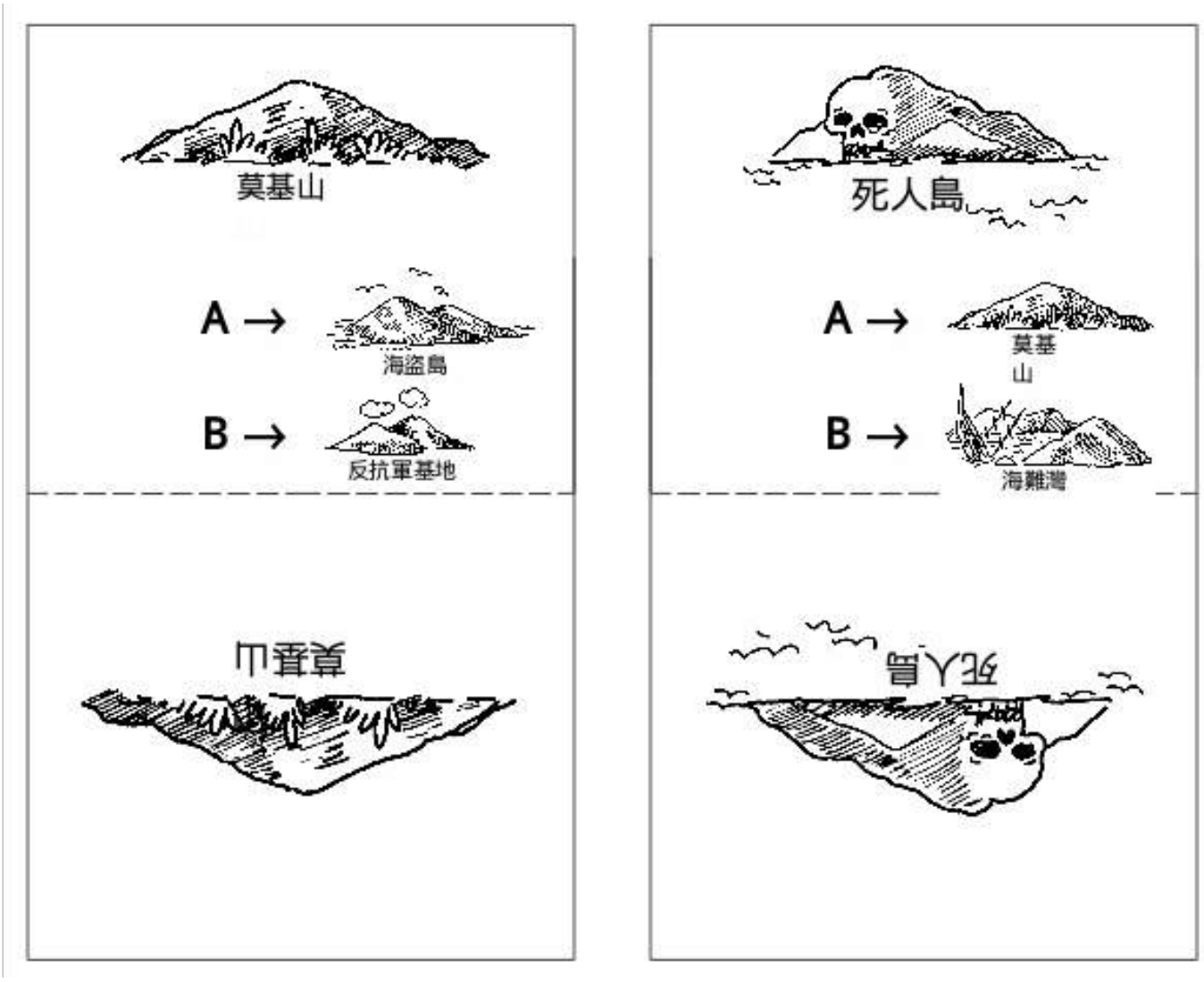
活動學習單：尋找金銀島



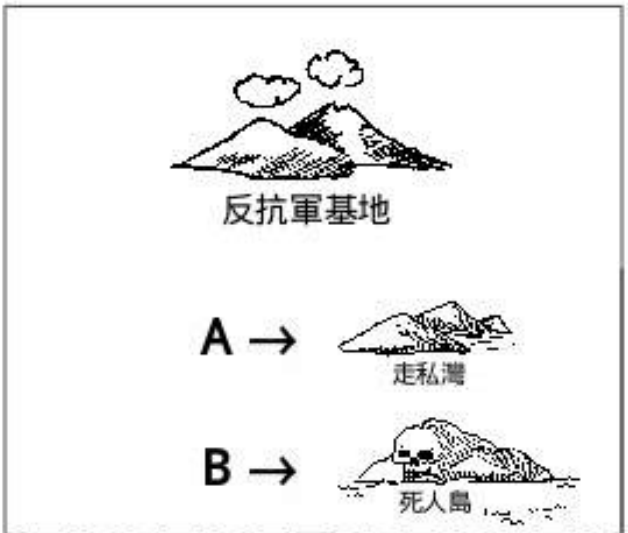
素材：島嶼卡（1/4）

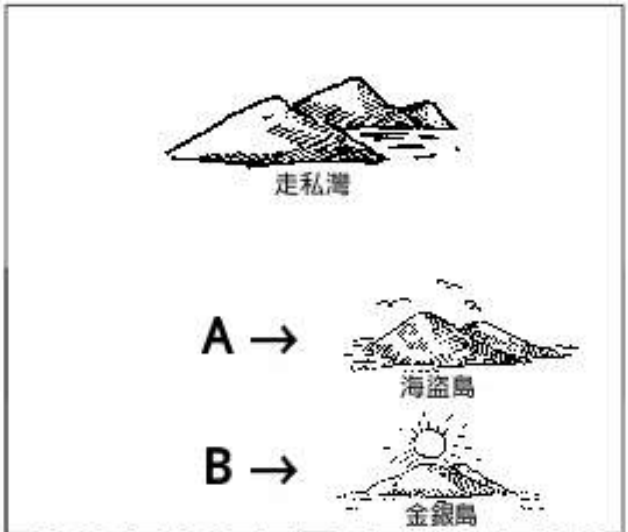


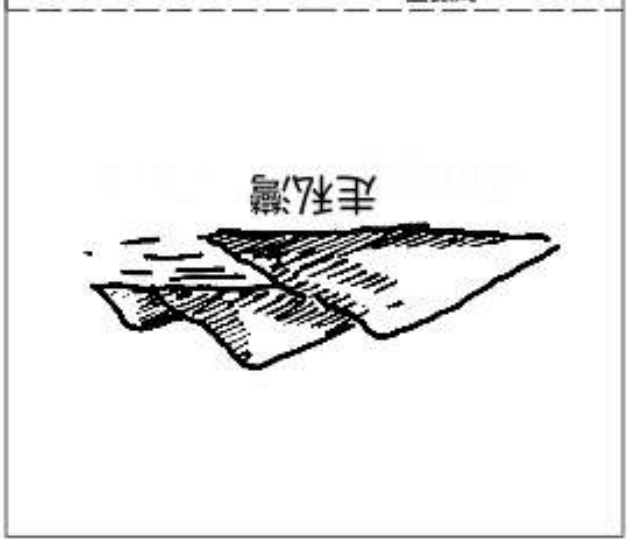
素材：島嶼卡（2/4）



素材：島嶼卡（3/4）



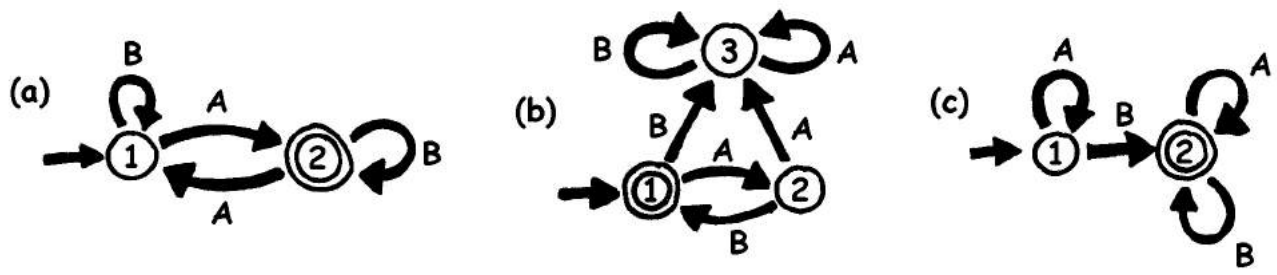




素材：島嶼卡（4/4）

有限狀態自動機

另一個畫地圖的方法是像這樣：



這些島嶼被表示成有編號的圓圈，而且最後一個島嶼（有寶藏的）是雙環。有哪些路線可以讓我們航行到最後一個島嶼？（透過例子來尋找是比較合適的。例如："A"會走到雙環的狀態嗎？"AA"呢？"ABA"呢？"AABA"呢？一般的樣式是怎樣？）

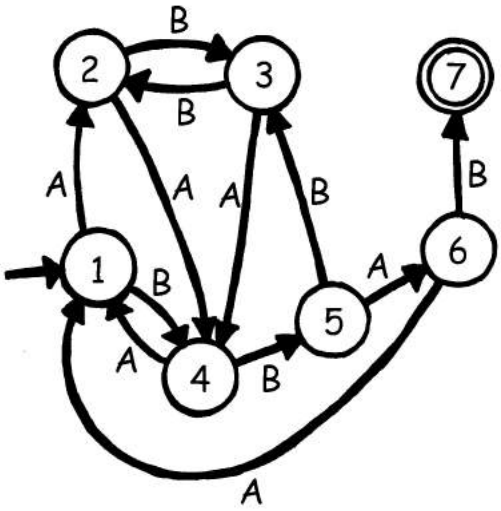
解答

地圖 (a) 只要序列裡有奇數個 A（例如：AB, BABAA, AAABABA）就可以到達終點（島嶼 2）。
地圖 (b) 只有在序列是 AB 交錯（例如：AB, ABAB, ABABAB）時才會到終點。
地圖 (c) 要求序列至少要有一個 B（唯一不適合的只有 A, AA, AAA, AAAA, ...）

活動學習單：金銀島

你能夠藏好埋著的寶藏嗎？你能讓尋寶的難度多難呢？是時候做一個自己的地圖了！

1.這裡有一個運用相同方法且更複雜的版本來呈現地圖。這張地圖與前一個練習是一樣的。電腦科學家們用這個既快速又簡單的方法設計出路徑圖。

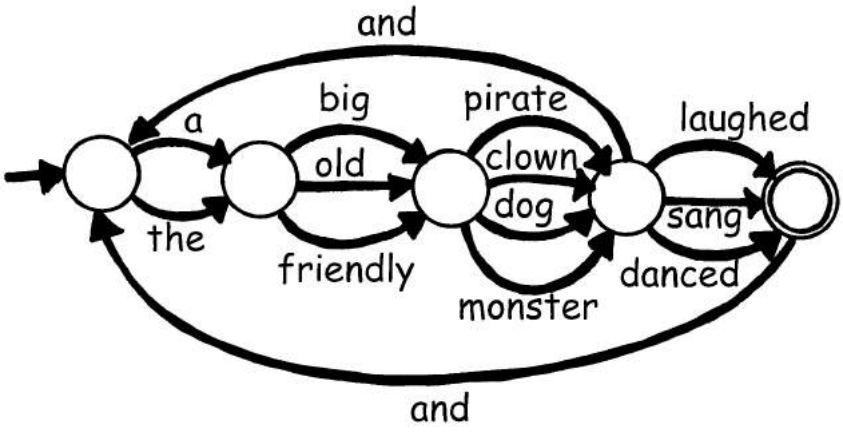


像上圖一樣畫出自己的地圖架構，這樣你就能夠清楚的知道海盜船行進的路線，然後做一份空白的地圖和小島卡。哪一種走法才能最快到達金銀島呢？

2.你的朋友麼能好好照著地圖走嗎？　給他們兩條路徑選項 A和 B，看看他們是否能抵達正確的小島。

根據有限狀態自動機的概念，你就能做出非常多種的遊戲和地圖。

3.這裡有一個藉著從地圖上隨機選取路徑並記錄用過的字來建立句子的方法。



現在，依照這個方法自己試試看。或許你還能創造出更有趣的故事！

活動學習單：神秘的硬幣

有些朋友從網路上下載了一個遊戲，遊戲裡機器人會投擲硬幣，然後他們要猜結果是正面或反面。整個遊戲在一開始看起來非常簡單，至少他們會有百分之五十的機率會贏。但不久之後他們就會發現事實並非如此！

經過一段時間後，他們開始懷疑，投擲硬幣的結果似乎有一種特定的樣式。難道這場遊戲是場騙局嗎？當然不是！因此，他們決定調查看看。

喬伊寫下了他們接下來在遊戲中的猜測，下面是他所發現的結果：（h=正面，t=反面）

h h t h h t h h h t t h h h h t t h t t t h h h h t h h h t t t h h h t t t t h h t t t t t t h t t h t t t h h h t t t h h h t h h t h h h h h h h h h t t h h h t t t t h h h h t t t t t t t

你能從這結果當中找到可預期的規律嗎？

有一個簡單的「地圖」可以來描述上面這個序列。你能想得出來嗎？（提示 :只有 4 個「島」）

這個活動在說什麼？

有限狀態自動機是在資訊科學領域中被用來幫助電腦處理一連串的指令或事件。

一個簡單的例子是當你在打客服電話時，常會聽到「要做 .....請按 1，要做 .....請按 2，轉接客服人員請按 9」這一類的訊息。你按的按鍵會成為電話另一端的有限狀態自動機的輸入。這些指令可以非常簡單也可以很複雜。有時候機器可能會在原地打轉，那是因為有限狀態自動機裡出現了奇特的迴圈。如果真的發生這種情況，那表示在系統的設計上發生了錯誤 — 這種錯誤會使撥號的人抓狂。

另一個例子是從銀行的提款機領錢。提款機中電腦的程式將會引導你經過一連串的程序。在程式中，所有可能的序列都保留在一個有限狀態自動機裡。你每按一個按鍵都會使自動機轉變為另一個狀態。當自動機進入某些狀態時，會給它們的電腦下達指令，像是「領出一百元」、「印出明細表」、「退出金融卡」等等。

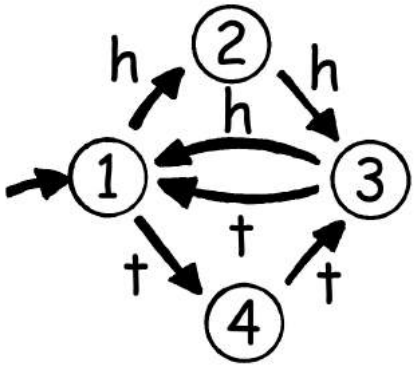
有些電腦程式很會利用像是第 134 頁的圖來處理英文句子。它們可以同時產生句子和處理使用者輸入的句子。在 1960 年代，一個電腦科學家寫了一支有名的程式，叫做「Eliza」（依 Eliza Dolittle 而命名）。它可以和人們對話。這支程式自稱為心理治療師，會詢問一些像是「告訴我關於你家庭的事」、「繼續下去」之類的問題。雖然實際上它根本不了解任何事，但那已經足夠真實了 — 說實在，它的使用者也真是夠容易被騙的 — 某些人真的相信他們是在和一個人類的心理治療師說話。

雖然電腦不是非常擅長於了解自然語言，不過處理人工語言是沒有問題的。一種很重要的人工語言就是現在所謂的程式語言。電腦會用有限狀態自動機來讀取程式，並將它們轉換為電腦可以瞭解的指令，然後執行。

解答與提示

神秘的硬幣（第 136 頁）：

神秘的硬幣遊戲使用的地圖：



照著地圖走就會看到，若每三個一組圈起來，每組的頭兩個都會是一樣的。

活動 13
行動的指示－程式語言

活動摘要

在對電腦下命令時，通常需要使用一種「語言」，而這所謂的語言其實就是一組有限的指令詞彙讓電腦可以遵循。不過，其中一件常常讓人感到很無言的事情是，電腦通常會完全遵守字面上的意義，即使產生的結果很詭異也一樣。在這個活動中，我們會從程式的角度給同學們一些體驗。

課程銜接

• 語文：人際聽力

習得技能

• 給予並遵守指示

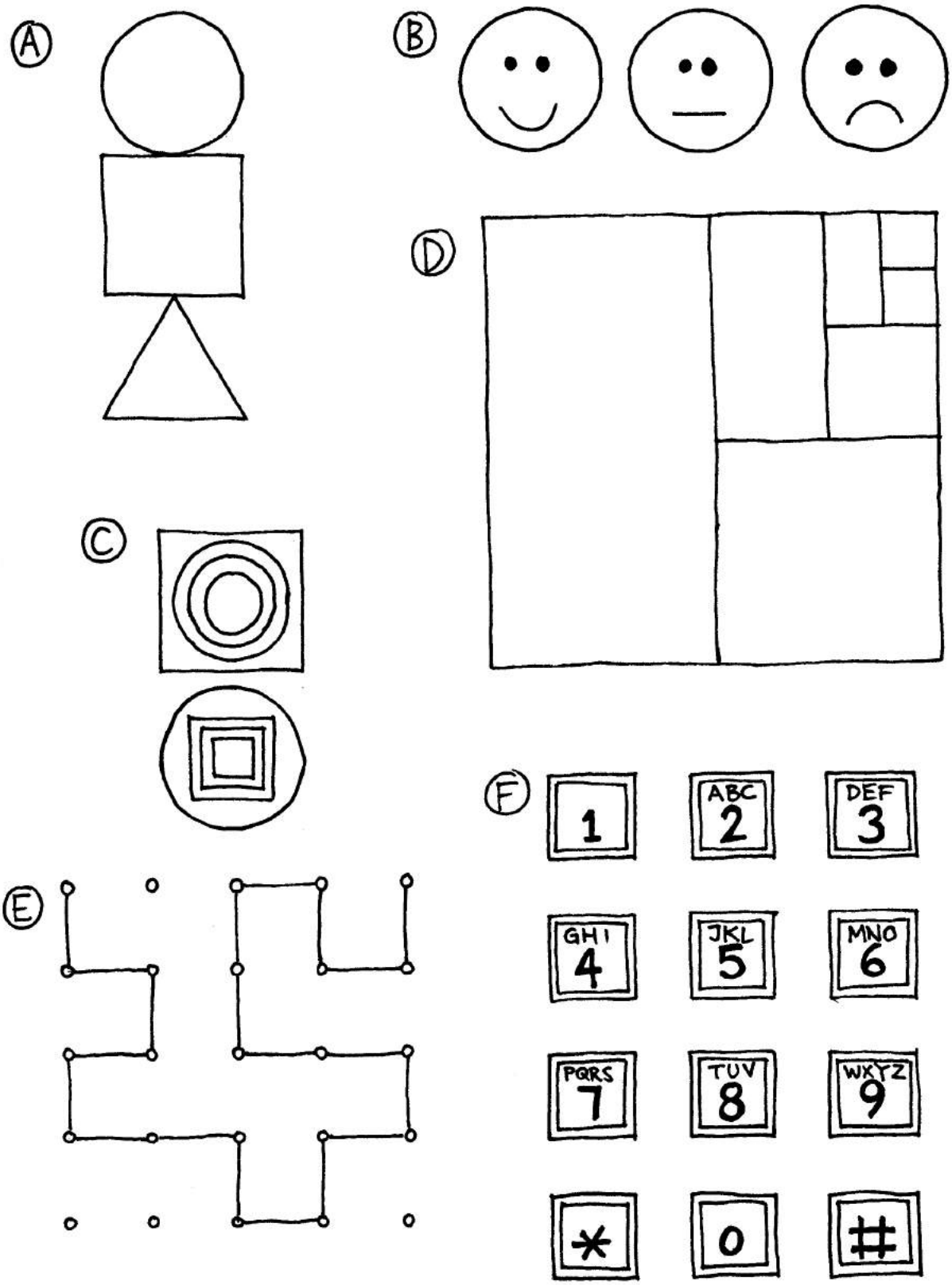
適合年齡

• 7 歲以上

所需素材

你需要：
• 上面有圖片的卡片，例如下一頁裡的範例。

每個學生需要：
• 筆、紙、尺



前進的命令

活動介紹

討論看看，如果人們完全只遵照字面上的意思來執行指令，這樣是件好事嗎？比方說，你指著一扇關上的門，然後下指令說：「通過這扇門！」會發生什麼事？

對電腦來說，它們只接受設定好的指令集裡的指令，而且會完全遵照這些指令執行 — 即使是沒什麼意義的指令！

範例展示

觀察學生是否能夠依照這些指示畫出圖來。

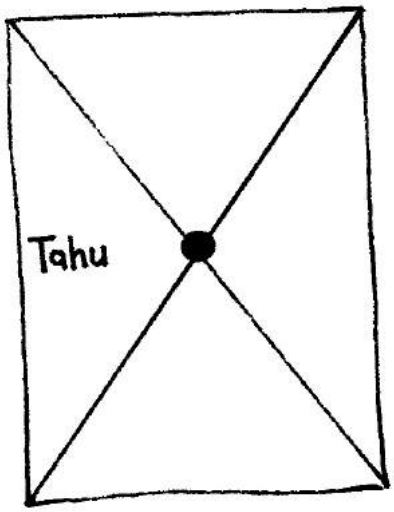
1. 在頁面的中心畫出一個點。

2. 從頁面的左上角開始，畫出一條直線，經過中心的那個點，到頁面的右下角。

3. 從頁面的左下角開始，畫出一條直線，經過中心的那個點，到頁面的右上角。

4. 把你的名字寫在左邊的三角形中。

結果可能如下圖：



活動進行

選一個學生當領袖，並給他一張圖（像是第 140 頁中的那些圖）。讓領袖對著全班描述這張圖，讓其他的學生們也能畫出該圖。大家可以提出問題，以便釐清領袖所下的指令與說明。遊戲的目的是要看看學生們能夠多快多正確的完成這些指令。

重複一次這個活動，但這次其他學生不能提出問題。最好給領袖比較簡單的圖，不然學生們可能很快就會放棄。

接下來，試著讓領袖藏在屏幕後面，讓大家看不到，也不能提出問題，只能靠聽領袖的聲音說明來進行。

最後向大家指出，這種形式的溝通，最像工程師在寫程式。工程師們會給電腦一組指令，但是之後才會知道這些指令產生的結果。

讓學生自己畫下圖且寫出說明指令。可以試試兩人一組，或全班一起進行。

活動變化與延伸

1. 寫下如何做紙飛鏢的說明。

2. 寫下如何在學校找到某個神秘地點的說明。使用類似以下這些指令：前進 x 公尺，向左轉（90 度），向右轉（90 度）等等。

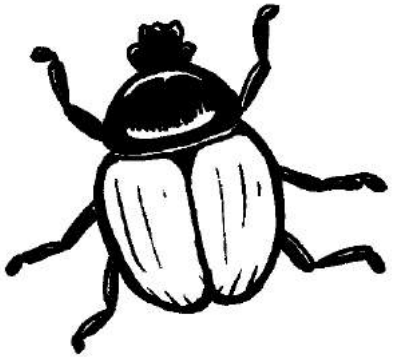
學生應該測試並仔細調整這些指令，直到確實可以遵照指令得到想要的結果。

3. 把一名同學的眼睛蒙起來，讓其他同學透過指令引導看不見的同學在房間裡移動。

這個活動在說什麼？

電腦藉由一連串指令來運作，這些一連串指令就稱之為程式。所以程式就是一串寫好的指令，來完成特定的工作。程式是依照設計好的語言寫成。在這語言中包含有限的指令，來告訴電腦該做甚麼。語言有很多種，根據不同的目的，會有相對合適的電腦語言。

不管你用哪種語言，程式設計師要根據想要電腦做甚麼來寫程式。電腦不像一般人類，即使給電腦一串不合邏輯的指令，電腦還是會徹底把它做好做滿。



因此，程式要寫好是很重要的。即使是一個小小的語法錯誤，也會造成很多執行上的大麻煩。我們可以想像，如果錯誤是發生在太空船發射程序的電腦上，核電廠的控制電腦上，或是火車的鐵路控制電腦上，都會引發大災難！我們常把寫錯的程式片段叫做「蟲「」（bug)。會這樣稱呼是因為，在過去 1940 年代的電子計算機上，曾經找了老半天之後發現，問題是出在繼電器上有一隻蛾所造成。後來把它弄走（debugged）之後問題就解決了。

越複雜的程式，就越有可能發生更多的錯誤。在美國執行戰略防禦計畫（Strategic Defence Initiative, “Star Wars”，這是一個由電腦來控制的系統，目的是形成一道堅不可催的防禦系統來抵抗核彈攻擊） 時，這個問題就引起很大的爭議。有些電腦科學家認為，由於軟體的複雜性和不可靠性，這個系統不可能成功。軟體必須透過仔細的檢查，來盡可能找出更多可能的錯誤以便事先修正；但是這個系統卻不可能為了要找出錯誤而一直發射導彈來測試！

第四部份
電腦也拿它沒輒 — 不可駕馭問題（Intractability）

不可駕馭問題（Intractability）

有沒有哪些問題，即便是對於電腦而言也太過困難？答案是：有。在活動 20 我們會看到，光是對話、聊天這件事，就是電腦做不到的。不是因為電腦沒辦法說話，而是因為他們不能理解或思考話中的細節與意義。但是在這裡，我們討論的不是那種困難的問題 — 不是說電腦不能交談，而是我們根本就不知道自己是如何做到這一點的，所以我們也沒辦法告訴電腦怎麼做。不過在這裡我們在說的問題是：我們可以透過寫程式來告訴電腦做什麼，但電腦還是需要非常非常非常長的時間：搞不好要一百萬個世紀才能算出結果。買更快的電腦其實作用不大：即使是快一百倍，這類問題仍然需要數百萬年的時間；甚至是快一萬倍也還是要花上數百年。這就是所謂不可駕馭的問題 — 即使用最快的電腦，想得到解決需要的時間遠遠超過電腦的使用壽命！

在第二部分的活動中，講到了所謂的演算法，也就是如何找到方法使得電腦程式運行更有效率。在這個部份，我們看的是已知沒有有效率解法的問題，即使用電腦也需要數百萬個世紀。我們會遇到的肯定是現在的資訊科學中最大的謎團：沒有人知道是否有解決這些問題的有效方式！它有可能只是因為還沒有人想出好辦法，也可能是根本就沒有好方法。我們不知道是哪個。而且還不只這樣喔！有成千上萬的問題，雖然它們看起來完全不同，但實際上，如果某個問題發現了一個有效的方法來解決，這個解法可以被轉換成另一個有效的方法來解決所有的問題！在這些活動中，你會知道這類問題的性質。

給老師的話

在這個部份中有三個活動。第一個活動是把地圖著色，並計算如果相鄰的區域必須不同顏色，那共需要多少顏色。第二個活動中，學生要有簡單的閱讀街道地圖的能力。在活動中我們要把冰淇淋車放在街道的某個角落，讓大家不需要走太多的路來買冰淇淋。第三個活動是戶外活動，使用線和釘子，探討如何將一連串的點連接起來的最短方法。

這些活動提供了對於「複雜性」的實作評估 — 一些說起來非常簡單的問題，怎麼會變成不可駕馭的問題。這些問題其實並不深奧。它們是每天的活動中實際會出現的問題，例如對映，課表，或是道路建設等等。這些計算的基礎是一種稱為「NP-完全」（NP-Complete）的概念。這會在每個活動最後的「這個活動在說什麼？」的章節中解釋。雖然活動本身可以不照順序進行，不過這些章節仍然要依照出現的順序來閱讀比較理想。等到學生讀完的時候，大家就會對現代資訊科學中最重要的開放性問題有足夠的瞭解。

這個部分的技術名稱叫做「不可駕馭問題」（intractability）。這個字源自於拉丁文 tractare，意思是繪製或拖動，而現代用法是指易於處理的或易於控制的問題。而它的相反詞 Intractable 就是指是那些不容易處理的問題，因為會花很長時間才能得到答案。雖然這聽起來深奧，但是「不可駕馭問題」卻具有重大的現實利益，因為在這方面若有所突破，將對於許多不同的研究產生重大影響。例如，大多數的加密密碼都依賴一些問題難解的特性。而如果有個罪犯設法找出了有效的解法，那他就可以很輕鬆愉快地把這些加密密碼解開，然後賣掉，或是更簡單地，直接用這些密碼來盜領戶頭中的錢。這部份我們會在第五部份 — 密碼學中提到。

活動 14
貧窮的製圖師 — 著色問題

活動摘要

許多優化問題會有這樣的情況：某些事件不能同時發生，或是某些物件成員不能相鄰。比方說，任何人想要排課或安排會議時間，都會遇到需要滿足所有人都能參與的問題。許多這類的難題都會用地圖著色問題來具體化：必須依照地圖上的國家位置來選擇，使得周邊國家有著不同顏色。這個活動就是在討論這樣的問題。

課程銜接

- 數學 : 數字 — 探索其他基底的數字。描繪二進位的數字。
- 數學 : 代數 — 繼續一個連續的樣式，描述這樣式的規則。二的 N 次方的模式與關係。

習得技能

- 解決問題
- 邏輯推理
- 演算法程序與複雜度
- 溝通的洞察力

適合年齡

- 7 歲以上

所需素材

- 白板或是白報紙之類的書寫工具。

每個學生會需要：

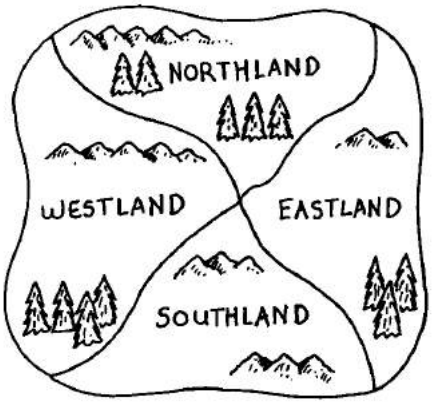
- 一張以上的活動學習單的影本。
- 可移動的小的彩色標記。例如 :籌碼，彩色標籤等等。
- 四隻不同顏色的蠟筆或色鉛筆、彩色筆等。

著色問題

活動介紹

這個活動是由一個故事衍生出來的。學生們要幫助一名製圖師，或地圖繪製師，他要在地圖上為不同國家上色。每個國家所上的顏色並不重要，只要和鄰近的國家不同就可以了。

舉例來說，這張地圖上顯示了四個國家，如果我們把北國塗上紅色，則西國和東國就不能是紅色，因為他們和北國的邊界將會變得難以辨識。我們可以將西國塗上綠色，同時我們也可以將西國塗上綠色，因為東國跟西國並沒有邊界相鄰的地方。（如果兩個國家只有在一個點上有交集，這樣並不算是共用一條邊界，因此他們可以塗上相同顏色。）南國可以塗上紅色，最後我們只需要兩種顏色來完成這張圖。



在故事中，製圖者非常貧困，無法負擔太多蠟筆的費用。所以理想上是用越少顏色越好。

活動討論

向學生解釋問題，讓學生開始製作。將著色過程展示在黑板上。

將第一份完成的學習複製下來。這張圖可以透過兩種顏色來正確的呈現。雖然限制只能使兩種顏色聽起來特別具有挑戰性，但和使用越多種顏色越好的地圖相比之下，這樣會變得比較簡單，因為每個國家可以選擇的顏色會很少。

要求學生們嘗試只用兩種顏色來為地圖上色。在過程中他們可能會發現「必要規則」：當一個國家著完色，其他相鄰的國家就必須塗上另一種顏色。這個規則一直被重複使用，直到所有國家都著完色。最好是讓學生自己發現這個規則，而不是事先告訴他們，這樣可以讓他們更加了解整個過程。

當學生完成每一項工作，就給他們下一張來嘗試。

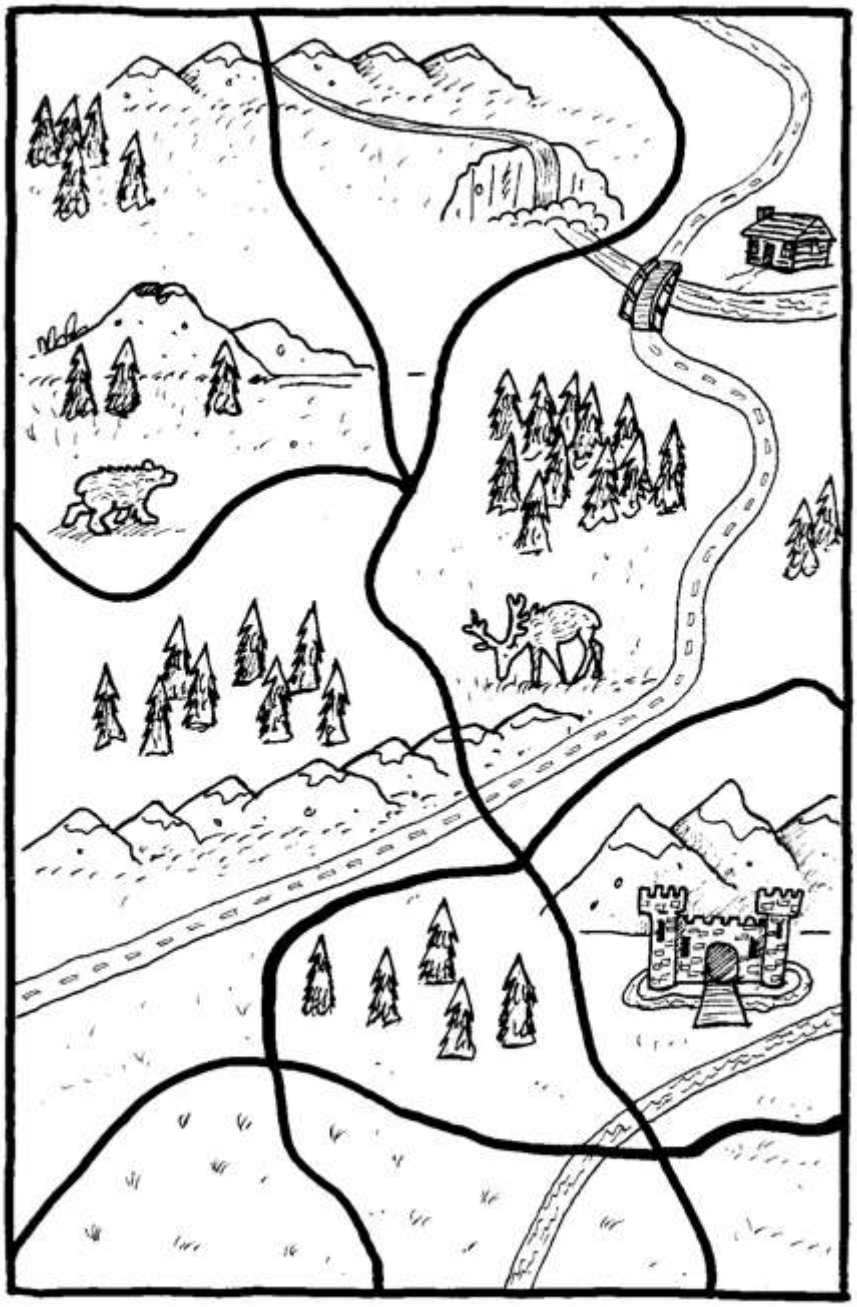
學生們也可能會發現，先用彩色標籤或物品來先做計畫會更好，而不是直接著色，因為這樣，如果發現問題可以輕易修正，而不是畫下去就來不及了。

對年長一點的學生，要求他們解釋如何找到使用最少顏色的方法。比方說，這張圖需要使用至少三種顏色來完成，因為這其中包含了一組三個國家（最大的三個），這三個國家彼此兩兩相鄰，所以無法只用兩種顏色搞定。

如果有一名學生提早完成了所有的學習單，可以請他們試著去設計一張需要五種不同顏色來完成的的地圖。因為已經有實際證明，任何一張地圖都可以透過四種顏色來完成著色，所以這個問題可以讓他們消耗一些時間跟腦力！以我們的經驗來看，學生會很快的發現，一張他們相信要使用五種顏色來完成的地圖，理所當然的都能找出一組四個顏色的解。

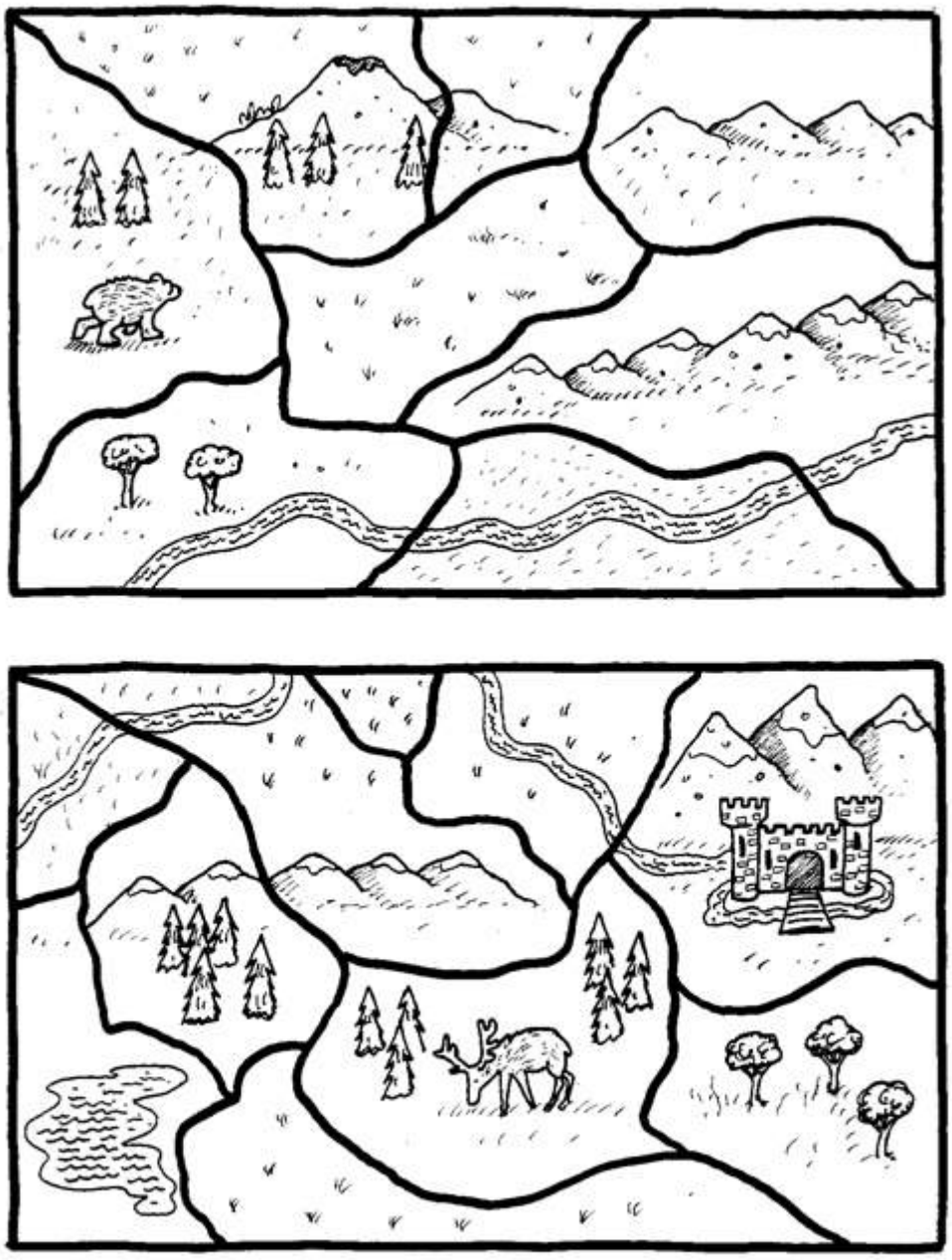
活動學習單：著色問題 1

用盡可能少的顏色來為此地圖中的國家們著色，但要確保沒有兩個鄰國是相同的顏色。



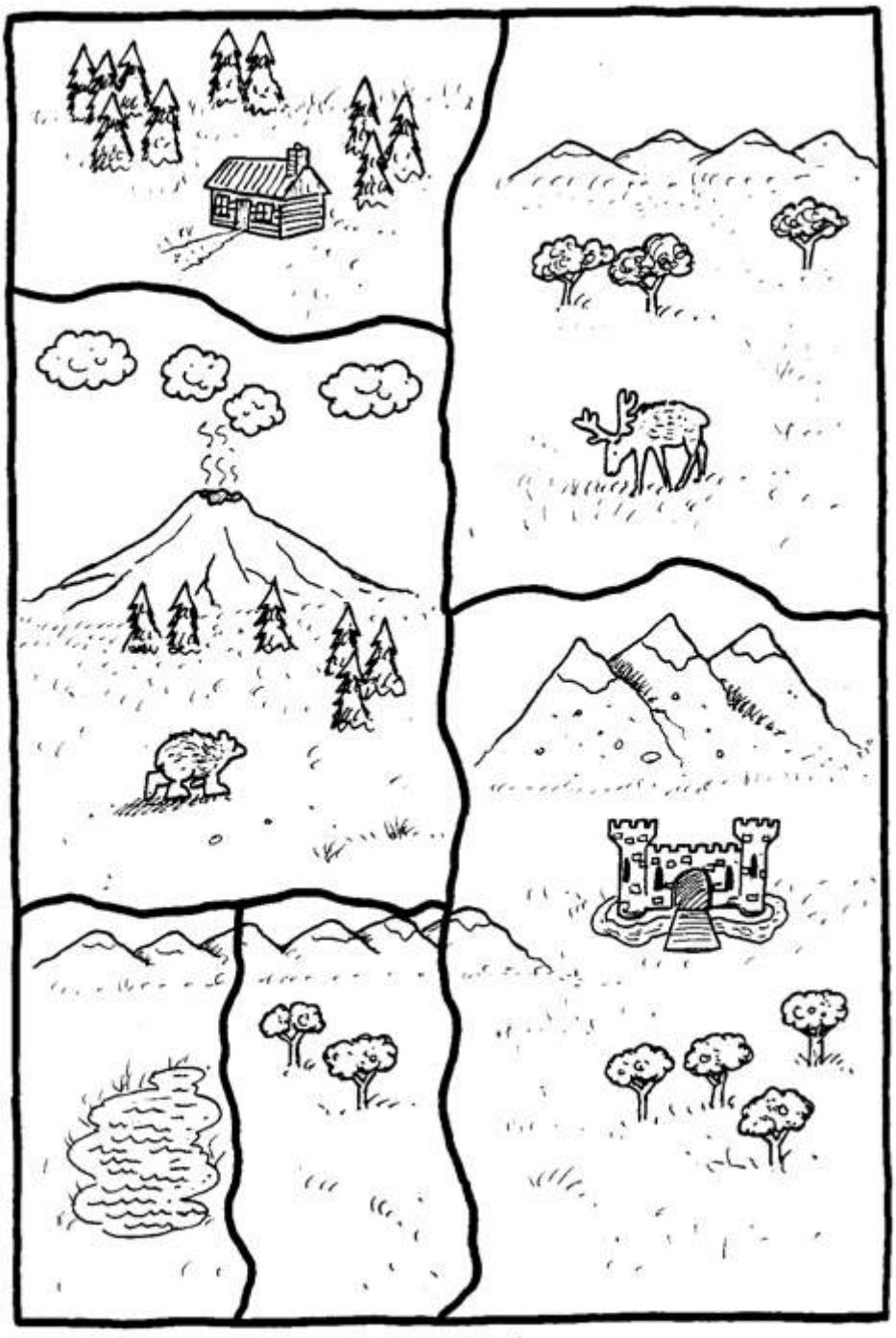
活動學習單：著色問題 2

用盡可能少的顏色來為此地圖中的國家們著色，但要確保沒有兩個鄰國是相同的顏色。



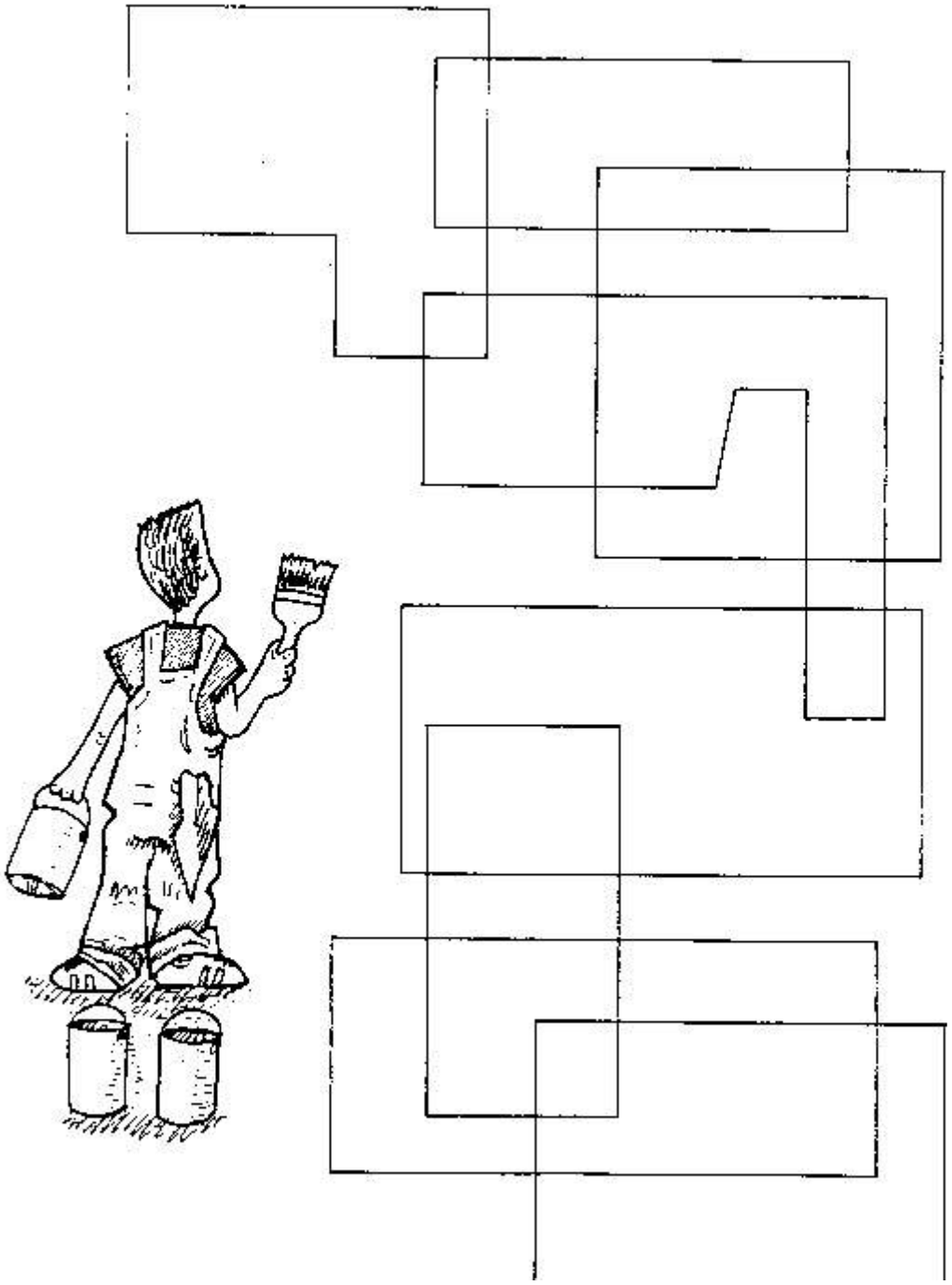
活動學習單：著色問題 3

用盡可能少的顏色來為此地圖中的國家們著色，但要確保沒有兩個鄰國是相同的顏色。



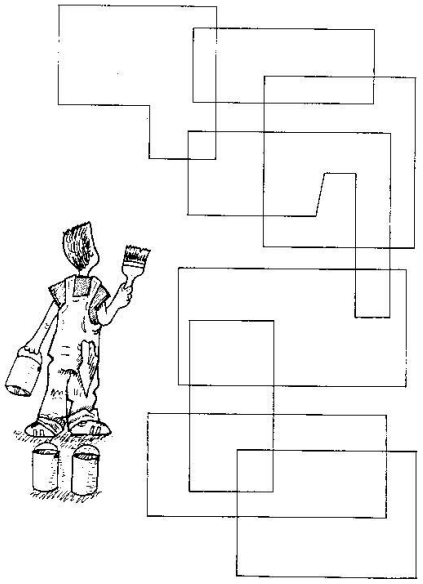
活動學習單：著色問題 4

用盡可能少的顏色來為此地圖中的國家們著色，但要確保沒有兩個鄰國是相同的顏色。



活動變化與延伸

以下有一個簡單的方法建立一個只需要兩個顏色的地圖。這個地圖是利用許多相疊的封閉曲線（線的頭接到自己的尾端）所繪出。你可以任意的畫出各種不同曲線及形狀，也可以互相交疊，最後畫出來的地圖都可以只用兩個顏色來完成著色。同學們能夠試試看創作出這類型的圖。



在一張紙上或是一個球面上（像是地球儀），最多只需要用四個顏色就可以完成著色。而你可能會想，（信不信由你，有人付錢給科學家去做相同的實驗），在較奇特的表面上，例如環面（像甜甜圈狀的旋轉曲面），又需要多少顏色才能完成著色呢？在這種情況下，只需要五個顏色就足夠了，同學們可以去實驗看看。

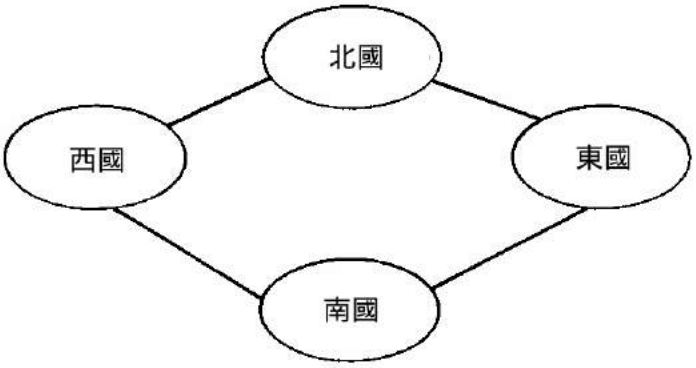
著色問題還有需多不同且有趣的變化。舉例來說，只有我自己在紙上為地圖著色，現在我知道我最多只需要四個顏色，就可以巧妙地完成。但如果我並不是自己完成，而是與一個無能（或是對立！）的夥伴輪流去選擇顏色來塗滿地圖，我可以精心設計著色方式，但我的夥伴卻懶得想，只是符合規則地與我輪流著色，那需要多少種顏色的蠟筆才能彌補我的豬隊友這種符合規則卻不明智（甚至會破壞）的舉動呢？這個數字目前尚無解喔！在 1992 年，有證明只需要 33 支蠟筆就足夠了，但後來在 2008 年時，證明結果被改進為只需要 17 支蠟筆即可。只是我們仍然不清楚，到底實際上多少支就足夠？（專家推測結果應該只需少於 10 種顏色即可）。同學們可以實際下去，兩人一組試試看，讓你的對手最多需要幾種顏色。

另一種變化，又稱為帝國著色遊戲。從兩張有著相等國家數量的地圖開始。其中一張地圖（例如地球）中的某個國家，會剛好配對在另一張地圖上（例如月球上的殖民地）的另一個國家。除了之前每個相鄰國家必須塗上不同顏色的需求之外，我們增加了一個要求 -- 在地球上的國家的顏色必須與它在月球上相對應的殖民地的顏色相同。我們需要多少顏色來完成這個問題呢？這答案目前還是未知的。

這個活動在說什麼？

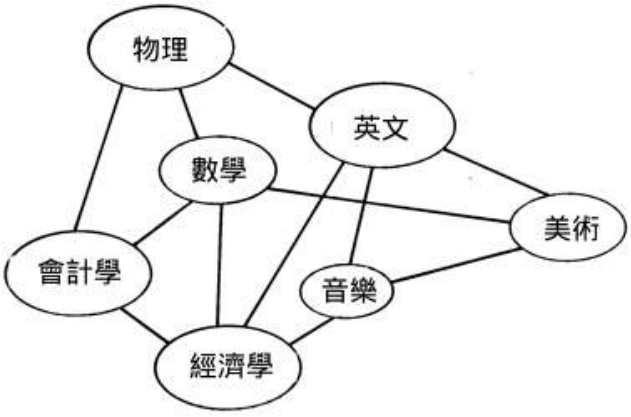
在這個活動中我們探討的地圖著色問題，是在為某個特定的地圖著色時，最少需要多少種顏色。在 1852 年時，「任何地圖皆只需要用四種顏色」的理論就出現了，但直到 1976 年才被證明。在資訊科學充滿著未解的問題，而這個四色理論在出現後超過 120 年才被證明，這件事給了那些仍為了其他未解問題花了數十年的科學家們很大的鼓勵。

地圖著色問題屬於「圖形著色問題」（graph coloring）的一種類別。在資訊科學中，「圖形」（graph）是一種抽象的關係表示，如圖所示。



如同活動 9 中「泥濘城市問題」所提到的，「圖形」在數學中的意思是用來展示數值資料，例如長條圖等的圖形。但資訊科學中的「圖形」跟這無關。在資訊科學中，「圖形」是用技術上稱為「節點」（nodes）的圓或大點來表示物件，然後再加上一些節點間的線段來表示物件間的關係。在活動一開始的地圖可以用上方的圖形來表示。節點表示國家，而在兩節點中的線段則表示這兩個國家的相鄰關係。在圖形上，著色規則是相鄰的兩個節點不可分配到相同的顏色。與地圖不同的是，在一般的圖形中不會限制用多少種顏色，因為相連的線本身就可能代表很多不同的限制；而二維的地圖的自然特性則限制了可能的排列。圖形著色問題是為了尋找特定圖形所需的最小數量的顏色。

在右圖中每個節點對應學校中的某一個科目。在兩種科目間的線段表示至少有一個學生同時修這兩堂課，所以這兩個科目不能安排在同一個時段。使用這種表示法，我們可以發現，要安排好課表，並使課程時段數量最少，這個問題跟著色問題是相同的，只要想像把不同的顏色用來代表不同的時段即可。圖形著色演算法在資訊科學中是一門相當有趣的學問，也被使用於許多現實生活中的問題，雖然可能永遠不會用來為地圖著色 — 我們的貧窮的製圖師只是個虛構的角色。



有成千上萬的問題是以「圖形」為基礎。有些在這本書的其他地方會提到，像是活動 9 的最小生成樹，以及活動 15 的支配集合等等。圖形是用來表示資料的一種非常常見的方式，而且可用來表示所有種類的狀況。就像地圖是以道路、路口所組成，分子是由原子連接所組成，訊息在電腦網路上傳遞的路徑，印刷電路板上組件的連接，還有像一個大型專案中所需要的小任務等等。因此，可以用圖形表示的問題始終讓電腦科學家們深深著迷。

這些問題很多都是相當困難的 — 並不是難在概念，而是它們需要很長的時間來算出答案。舉例來說，要決定適中大小的圖形著色問題的最有效率的解法，還有像是找出一間有 30 位老師及 800 名學生的學校排課表的最佳方式，即使電腦使用目前已知最佳的演算法，仍然可能要花上數年甚至數百年的時間。這個問題會耗費無止盡的時間來尋找解法 — 而且還必須假設在結束前電腦不會掛掉。這樣的問題只能在現實生活中解決，因為我們可以接受次佳（但已經夠用了）的解法。如果我們堅持一定要保證找到最佳解，那這個問題就變成不可駕馭的問題了。

解決著色問題所需的電腦時間與圖形的大小成指數性的成長。想想看地圖著色的問題。我們可以嘗試所有的可能的著色方式來解決。而正如我們所知，最多需要四種顏色，所以我們必須評估分配四種顏色到每個國家的所有組合。若有 n 個國家，那就有 4^n 種組合，這個數字增加的非常快速：每增加一個國家，組合數就乘以四倍，解決時間也因此增加四倍。即使電腦可以解決比如說，50 個國家的問題只需要一個小時，但多一個國家，就要增加四個小時，一旦我們增加十幾個國家，電腦就必須花上整整一年來尋找解答。這種問題並不會因為我們持續發明更快的電腦而消失。

圖形著色問題是解題時間呈指數成長的一個很好的例子。就這個問題舉一個非常簡單的例子，就像在這個活動中使用的小地圖，我們相當容易找到最佳解法。但只要國家數增加超過十，這個問題用手算將會變得相當複雜，若增加至一百多個國家，即使是電腦也需要花上好多年去嘗試所有可能的著色方式，才能找出最佳解。

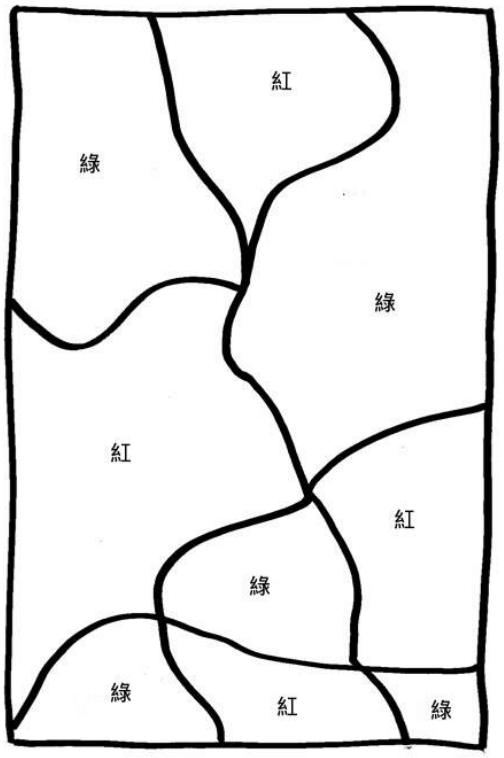
很多真實生活中的問題都有這種特性，但無論如何我們都必須解決這些問題。電腦科學家使用夠好的，而非完美的解決方法。這些啟發性技術往往相當接近最佳解，可以非常快速的計算，並給出相當接近實際目的的解答。學校可以接受需要使用一間以上的教室，而不一定要最完美的課表。也許貧窮的製圖師也可以負擔的起多一種顏色，即使嚴格來說，並不是絕對必要的。

到目前為止沒有人能夠證明，這類問題不存在一種有效的方法來解決，但同樣的也沒有任何人證明這種有效的方法真的存在。電腦科學家們也對這種方法是否存在感到存疑。在接下來的兩個活動中，我們將會學到更多有關這類問題的知識。

延伸閱讀

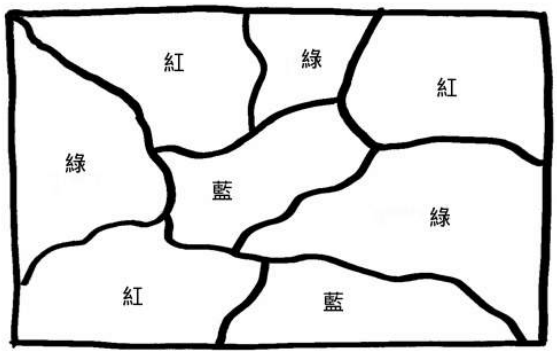
Harel 在 “Algorithmics” 中探討了四色理論，包括它的歷史。Casey 跟 Fellows 所著的 “MEGA-Mathematics!” 中探討了地圖著色問題的許多面向。Kubale 2004 年出版的書 “Graph Colorings”，也包含了這類問題的歷史。另外也有很多網站在討論這個主題。

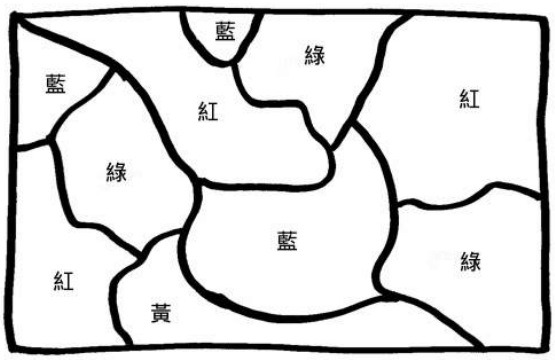
解答與提示



活動學習單：著色問題 1

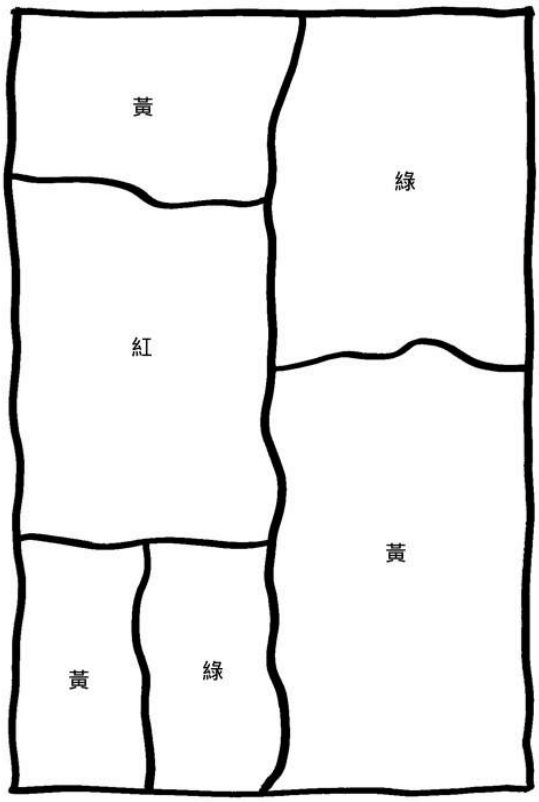
左方的圖是唯一可能的解法。（當然，顏色學生可以任意選擇，不過我們只需要兩種顏色。）





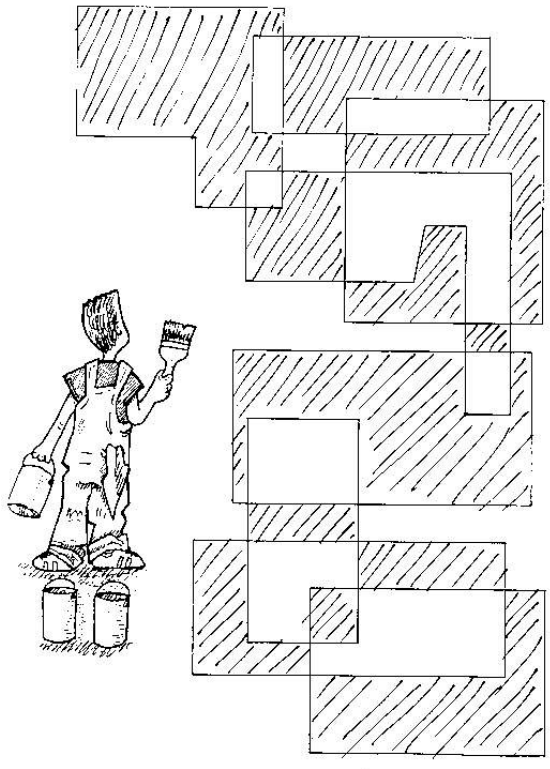
活動學習單：著色問題 2

頂部的地圖可以使用三種顏色來完成，而底部的地圖需要四種。 這裡提供兩組可能的解答。



活動學習單：著色問題 3

這是個簡單的三色地圖，這裡提供一個可能的解答。



活動學習單：著色問題 4

僅使用兩種顏色（灰與白）。

活動 15
旅遊小鎮 — 支配集

活動摘要

很多現實生活上的情況都可以用網路或是像活動 14 中用來著色的圖形的形式來表達。網路也提供很多開發特定用途演算法的機會。在這個活動中，我們在一些連接點或節點上做標記，讓所有沒標記的節點最多只要一步就可以到達有標記的節點。問題來了，我們最少要標記多少節點？這是個有趣且出乎意料困難的問題！

課程銜接

- 數學：位置與方向
- 數學：邏輯推理

習得技能

- 地圖
- 關係
- 解決問題
- 迭代目標搜尋法（Iterative goal seeking）

適合年齡

- 7 歲以上

所需素材

每組學生需要：

- 冰淇淋車的圖
- 一些籌碼或是兩張不同顏色的撲克牌

你會需要：

- 冰淇淋車的投影圖，或是直接畫在白板上

支配集（Dominating Sets）

活動介紹

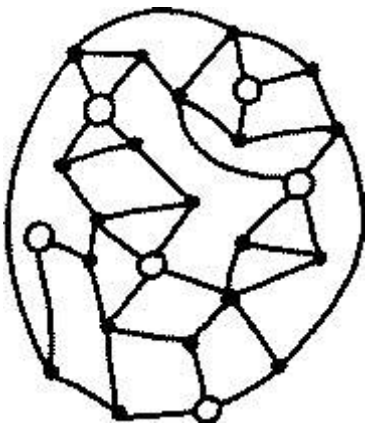
「活動學習單：冰淇淋車」上畫著一個旅遊小鎮的地圖。每一條線代表一條街道，而每個點則代表街角。這個城鎮所處的地區氣候炎熱，因此在夏天時冰淇淋箱型車都會停在街角賣冰淇淋給觀光客。我們現在要設立冰淇淋車駐紮的據點，讓每一位觀光客都能在每條街底找到廂型車，最多只需要走一條街即可買到冰淇淋。（也可以想像居民們大都居住在街口而非街道中，因此他們出門之後最多走到下一個街口就一定可以買到冰淇淋）。所以，我們的問題是：我們需要多少台冰淇淋車，並且應該設立在哪些街角？

活動討論

1. 將學生分組，並給各組一個旅遊小鎮的地圖，並說明故事內容與問題。

2. 說明該如何在某一個十字路口上放一個籌碼，標記為冰淇淋車，然後在其相距一條街的街口放另一個顏色的籌碼。這台冰淇淋車的服務範圍即為這些籌碼所標記的位置，也就是住在這個十字路口以及相鄰街上的居民。

3. 讓同學嘗試在不同位置駐紮冰淇淋車，找出能夠讓每個居民都能買到冰淇淋的駐紮方式。提醒同學，此問題涉及成本考量，冰淇淋車的駐紮據點要在能夠達成題目要求的情況下越少越好，這也是這個問題最有趣的地方。



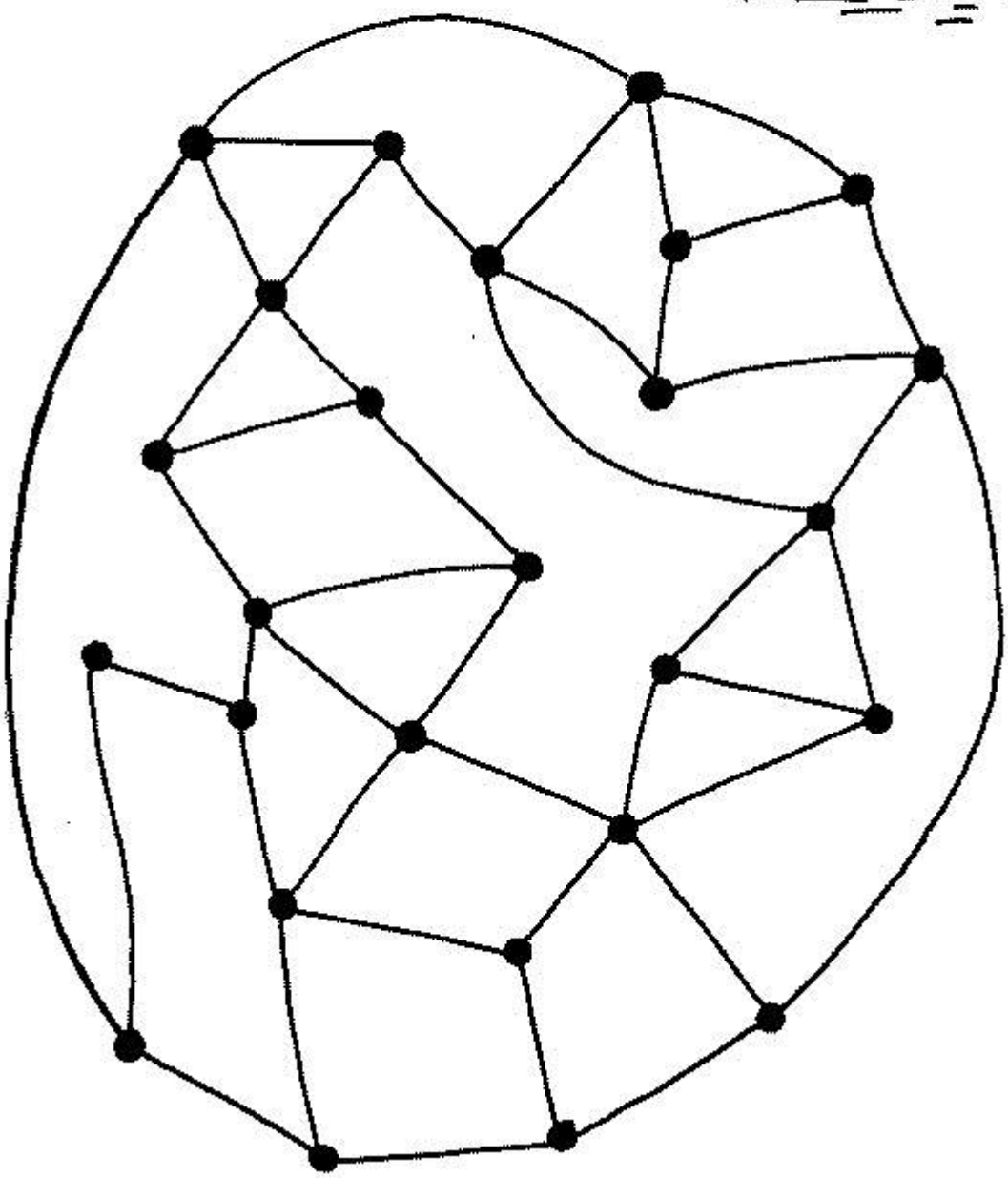
4. 旅遊小鎮所需要的冰淇淋車最少數目為 6，而解答如右圖所示。這個解答並不好找喔！因此大家摸索一陣子之後，要跟同學們提示，此題的所需的冰淇淋車數目最少為六台。因為即使增加到八台或九台車去解難度都頗高，因此最後可能會有許多組放棄。鼓勵大家多去嘗試。

5. 這個旅遊小鎮的地圖製作，是由「活動學習單：冰淇淋車（解答）」下面的六個地圖區塊所開始組成。每個區塊只需要一個冰淇淋廂型車。可以畫上街道來將這幾個地圖區塊拼湊起來，將答案隱藏起來。重點不是將所有冰淇淋車據點（空心點）通通連接起來，而是要放在將據點與服務範圍（實心點）間的連結。將這個方法用板書或投影片示範給同學看。

6. 讓同學嘗試用這個的方法製作地圖。相信同學也會喜歡製作一個只有自己解得出來的題目！這個題目即是一個「單向函數」（one-way function）的例子 — 除了製作謎題的人，其他人沒有辦法輕易解出這道謎題。「單向函數」在密碼學中佔有一個非常重要的角色 （請參見活動 17、18）。

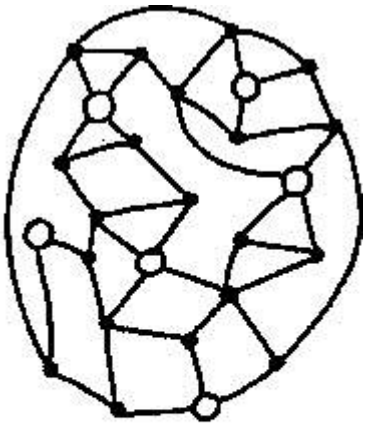
活動學習單：冰淇淋車

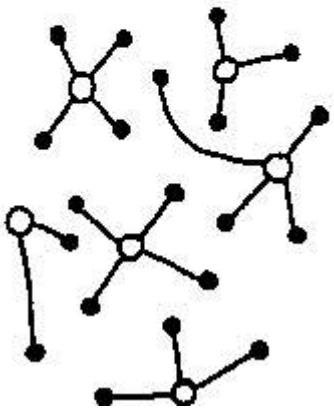
請找出如何在交叉路口擺放冰淇淋車，使所有的交叉路口都可以相連到有車子的交叉路口。



活動學習單：冰淇淋車（解答）

對同學顯示下面的圖來說明題目是如何建構的。





活動變化與延伸

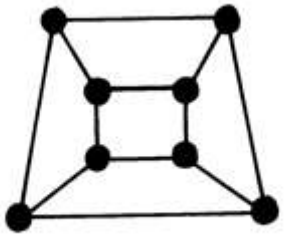
在做城市規劃時會有很多情況面臨到這類問題，像是擺設郵筒、井、消防局等等。而在現實生活中的地圖也不會用那麼簡單的方式來組成。如果你真的當上市長，需要解決這類問題的話，你會怎樣做呢？

有個比較直接的方法是：把所有擺放冰淇淋車的方法列出來，然後看看哪一個是最適合的。旅遊小鎮內有 26 個街口，所以有 26 種方法可以擺放一台冰淇淋車。檢查所有 26 種可能性不難，但是很明顯地這些方法都沒有滿足所需的條件。用兩輛冰淇淋車的時候，有 26 個地方可以擺第一台車，然後剩下 25 個地方可以擺第二台車（因為你總不會在同一個路口擺兩輛車吧！）。所以總共有 $26 \times 25 = 650$ 種可能性要檢查。嗯，要全部檢查嘛，也是不難，但已經開始有點讓人抓狂了。真正說起來，你只需要檢查一半（325 種可能性），因為車子的順序不重要：如果你已經看過車子 1 在交叉路口 A 和車子 2 在交叉路口 B 的情況，則沒有必要再檢查車子 1 在 B 和車子 2 在 A 的情況。三輛車子呢？有 2600 種可能性。四輛車子呢？有 14950 種可能性 ..... 依此類推。當然，因為只有 26 個交叉路口，並且同一個地方也不需要兩台車，因此最多 26 台車就可以了。另一種評估可能性的方式是考慮所有可能的配置：26 個交叉口和任何數量的貨車。由於每個路口只有兩種可能：有車或沒有車，因此所有配置的可能性的數量是 2^{26}，也就是 67,108,864。

這種方法被稱為「暴力」演算法（brute-force），通常需要很長的時間才能完成。大家常以為電腦計算的速度超級快，不管有多少工作，都可以在很短的時間內解決幾乎所有的問題，但實際上並沒有。暴力演算法需要多長的時間，取決於它檢查一種可能性是否是最佳解的時間。在冰淇淋車的問題中，要檢查一種可能性，得要檢查每一個路口，來確認最近的車子的距離。假設每一種可能性可以在一秒鐘檢查完，那旅遊小鎮的地圖需要多長的時間來測試所有 2^{26} 種可能性？（答案：2^{26} 大約等於 6700 萬；一天是 86400 秒，所以 2^{26} 秒相當於 777 天左右，也就是大約兩年。）假設一種可能性不用到一秒，我們算千分之一秒好了。那麼在相同的兩年，電腦大約可以找出 36 個路口的配置，因為 2^{36} 大約是 2^{26} 的 1000 倍。好，即使電腦快一百萬倍，也就是一秒鐘可以檢查一百萬種可能性，那兩年的時間一樣只能處理 46 個路口的冰淇淋車配置。這些都不算是非常大的城市喔！（想想看，你住的城鎮有多少交叉路口？）

既然暴力演算法是如此地慢，有沒有其他的方法來解決這個問題呢？我們可以試看看在泥濘城市（活動 9）中非常成功的「貪婪方法」。我們需要想想看如何貪冰淇淋 — 我的意思是，如何應用貪婪的方法來解決冰淇淋車問題。這種方法是把第一台冰淇淋車放在相鄰交叉路口最多的路口，然後第二台車就在相鄰交叉路口第二多的路口，依此類推。然而，這樣做並不一定會產生冰淇淋車擺設最少的最佳解 — 實際上，在旅遊小鎮中，最多相鄰節點（5 個）的節點，並不適合擺冰淇淋車（跟班上同學確認這一點）。

讓我們來看一個比較簡單的問題。假設我們的目的只是要找出一個配置方法，不一定要最小或最佳解。在某些情況下，這相對容易許多。 例如，這張簡單點的地圖，解答就很直覺。如果把地圖中的街道想像成立方體的邊，那很明顯地，只要擺兩台冰淇淋車立方體斜對面的頂點就夠了。此外，你應該也很清楚，不可能少於兩台車。而在旅遊小鎮的地圖中，就很難 — 雖然也不是不可能 — 說服大家相信只需要六台車就夠了。



這個活動在說什麼？

關於冰淇淋車問題，一件有趣的事情是，沒有人知道是否有一個尋找一組最佳解的演算法比暴力法明顯更快！暴力法花費的時間隨著交叉路口的數目呈指數增長 — 因此它又被稱為「指數時間演算法」（exponential-time algorithm）。而相對於指數時間演算法，「多項式時間演算法」（polynomial-time algorithm）是指演算法的執行時間會隨著變數的平方、立方、十七次方或其他數目次方而增長。

多項式時間演算法，即使是十七次方，只要是夠大的地圖，也會比暴力法更快。因為指數級成長在它的參數大到一個程度時，一定會超過任何多項式級的成長（比方說， n^{17} 跟 2^n 來比較，哪個比較大？在 n 大於 117 之後，後者就會超越前者）。那麼，是否有一個多項式時間演算法可以來尋找最少需要在哪些位置？目前還沒有人知道，但大家已經很努力的在找了。而對於一個比較容易的問題：檢查一組特定的位置是否是最佳解也是一樣，使用暴力法所花費的時間是呈指數成長，而針對此問題的多項式時間演算法，既沒有被發現，也沒有被證明不存在。

這是否讓你想起地圖著色問題（活動 14）？你應該要想起來喔。冰淇淋車的問題，正式名稱為「最小支配集」（minimum dominating set）的問題，是目前尚未確定多項式演算法的解法是否存在的問題之一。這一類的問題，範圍從邏輯，鋸狀排列問題到地圖著色問題，在地圖上找最佳路徑，還有調度流程問題等等，問題可以說成千上萬。但令人驚訝的是，所有的這些問題已被證明是相同的：如果其中一個問題找到了多項式時間的演算法，那其他所有的問題就可以經由此演算法做轉換而一起解決 — 要嘛就全部有解，要嘛就全部無解。

這些問題被稱為「NP 完全問題」（NP-complete problems）。NP 代表「非確定性多項式」（non-deterministic polynomial）。這可是內行人才會用的術語喔！這個術語表示只要有一台電腦可以立刻測試任意量的解決方案（所謂的「任意量」就是「非確定性」的部分），這個問題可以在合理的時間內解決。你可能會覺得這什麼鬼？未免也太不切實際了吧？沒錯，的確是這樣。我們的確不可能建立這種類型的電腦，因為它必須是非確定性的任意大！然而，這種機器的概念在原則上是很重要的，因為 NP 完全問題不能在沒有非確定性電腦的情況下在合理的時間量被解決。

此外，這一組問題被稱為「完全」是因為雖然問題似乎非常不同 — 例如，地圖著色問題和冰淇淋車問題看起來非常不同 — 但經過證明，如果在其中一個問題找到的一個有效率的方式來解決，那麼此方法可適用於解決同組中任何一個其他的問題。這就是我們所謂的「要嘛就全部有解，要嘛就全部無解」。

有成千上萬的 NP 完全問題，研究人員一直在努力尋找有效率的解決方案，但幾十年來還沒有好消息。就像剛剛說的，只要找到其中一個問題的有效解法，那所有問題就可以一起被解決。不過也是因為這樣，它被強烈懷疑是不是根本就沒有有效率的解法。不管怎樣，證明這些問題必然需要指數時間，在理論資訊科學 — 說不定是在整個數學界 — 來說，都是今日最著名的開放問題。

延伸閱讀

Harel 的書 “Algorithmics” 介紹了幾種 NP 完全問題，討論多項式時間演算法是否存在的問題。Dewdney 所著的 “Turing Omnibus” 也討論了 NP 完全問題。關於這個問題的標準資訊科學教科書是 Garey & Johnson 的 “Computers and Intractability”，裡面介紹了數百個 NP 完全問題，並介紹證明 NP 完全性的技巧。然而，這本書真的很難，只適合於資訊科學專業人員。

活動 16
冰之路 — 斯坦納樹

活動摘要

有的時候，對一個問題做一個渺小的、似乎微不足道的變動，結果會造成解決難度上巨大的改變。這個活動，類似於活動 9 的「泥濘城市問題」，也是要找到網路裡的最短路徑。然而，跟泥濘城市問題不同之處，是在可以縮短路徑的長度的前提下，允許加入一個新的節點進入網路。這一個小小的改變，把整個問題變成一個不可駕馭的問題。這個問題變得跟泥濘城市無關，而是在演算法上與「貧窮的製圖師」（活動 14）以及「旅遊小鎮」（活動 15）一樣的問題了。

課程銜接

- 數學：位置與方向
- 數學：邏輯推理

習得技能

- 空間視覺化
- 幾何推理
- 演算法程序與複雜度

適合年齡

- 7 歲以上

所需素材

每一組的學生會需要：

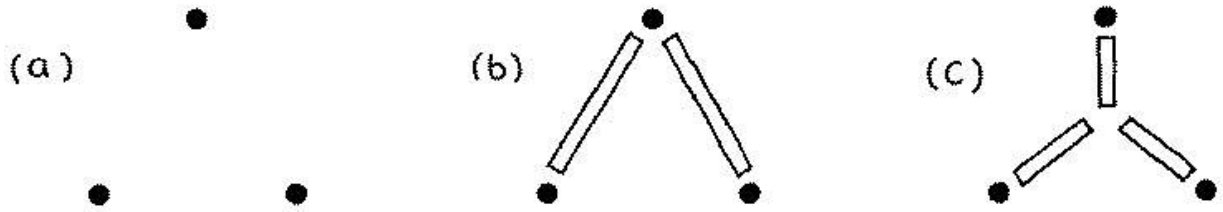
- 五或六個釘子放在地上（帳篷釘，或把衣架切成片折彎也可以）
- 幾公尺長的繩子或鬆緊帶
- 一個直尺或捲尺
- 筆跟紙用來做筆記

冰之路

活動介紹

先前的活動，「旅遊小鎮」的故事發生在一個非常炎熱的國家；而這個故事卻剛好相反。在加拿大寒冷的北方（夠冷了吧，這樣故事才進行得下去），冬天的時候整個湖會結冰；而掃雪機則會在巨大的冰湖上做出一條條可以連結工作站的路，讓工作人員可以拜訪彼此。而因為那裡很冷，所以他們希望道路的距離越短越好。而你的工作就是要找出路要造在哪裡。基本上沒有條件限制：湖面都是結凍而且被覆蓋住的。全部都是平坦的。路可以開在雪地上的任何地方。

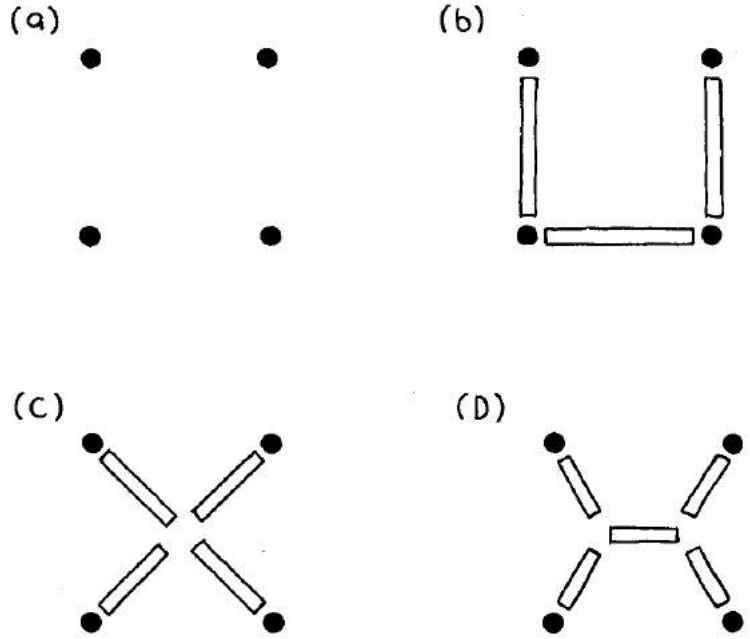
明顯地每條路應該都是直的，因為彎曲的道路會增加不必要的長度。但是不是把所有的工作站用直線連起來就好了，因為在結凍的荒地上，有時加上一個交叉點反而可以減少道路的總長度 — 要記得，重要的是道路的總長度要最短，而不是從一個工作站移動到另一個工作站的時間要最短。



在圖中，(a) 有三個工作站。把其中一個工作站跟另外兩個連起來（如 (b) 圖）是可接受的一個解法。另一個方式是在三個點的中央附近建立一個交叉點，將交叉點連結到三個工作站（如同 (c) 圖）。如果你測量一下的道路的長度，會發現 (c) 圖確實是個比較好的解法。這個額外的交叉點被稱為「斯坦納點」，因為瑞士數學家雅各 · 斯坦納（Jacob Steiner, 1796-1863）而得名。他描述了這個問題，而且是第一個指出可以藉由產生新的節點去減少道路的總長度。你可以將斯坦納點想像成一個新的，虛擬的工作站。

活動討論

1. 向大家描述接下來學生將進行的活動。使用上面的範例向學生展示，在三個點之外額外增加一個點，有時反而能夠減少所建道路的總長度。



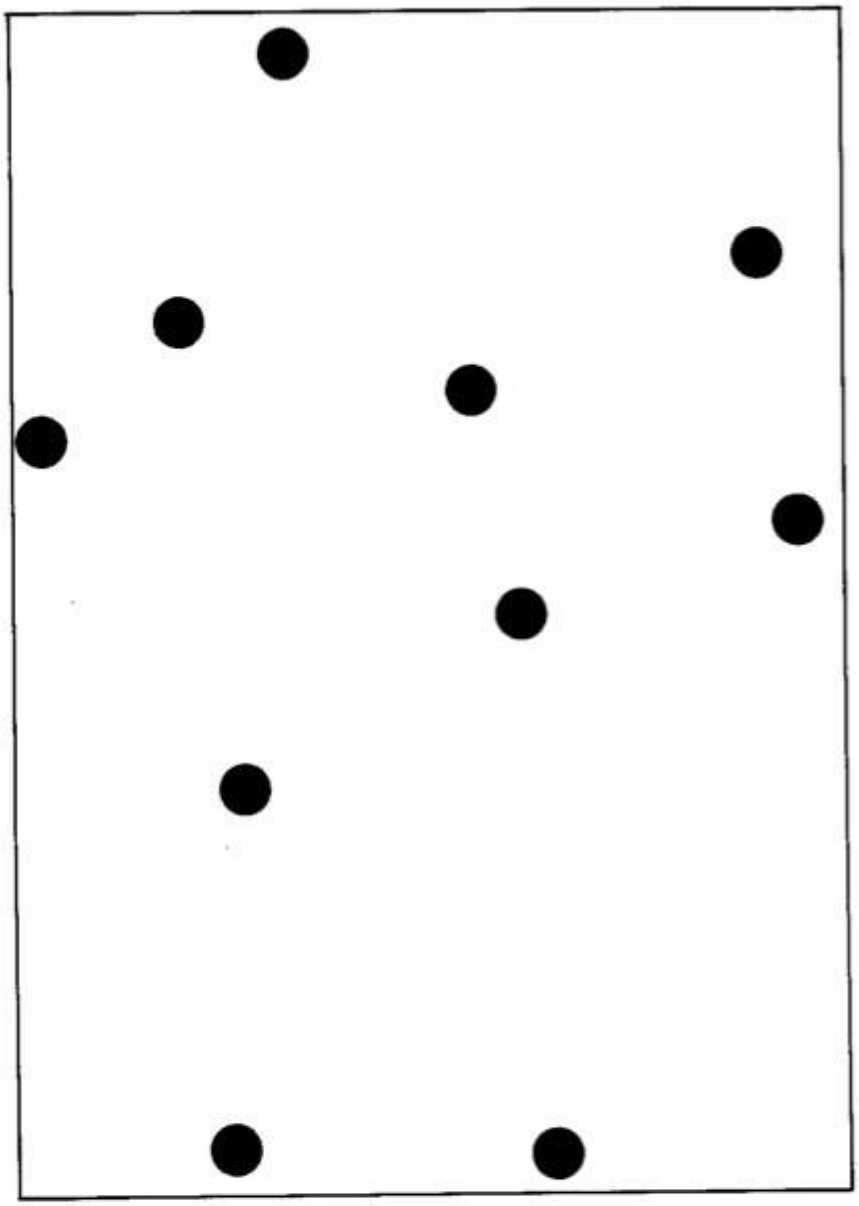
2. 學生將使用四個點排列而成的方形，如同圖 (a)。到外面去讓每一組在草地上用釘子釘出約 1 公尺見方的範圍。

3. 讓學生試試看用繩子或有彈性的物品去連接每個釘子，測量最小所需的長度並且記錄起來。在這個階段還不要加入使用任何斯坦納點。（因此這個階段，要達到總長度的最小值，要把正方形的三邊接起來，如圖 (b)，總長度為 3 公尺。）

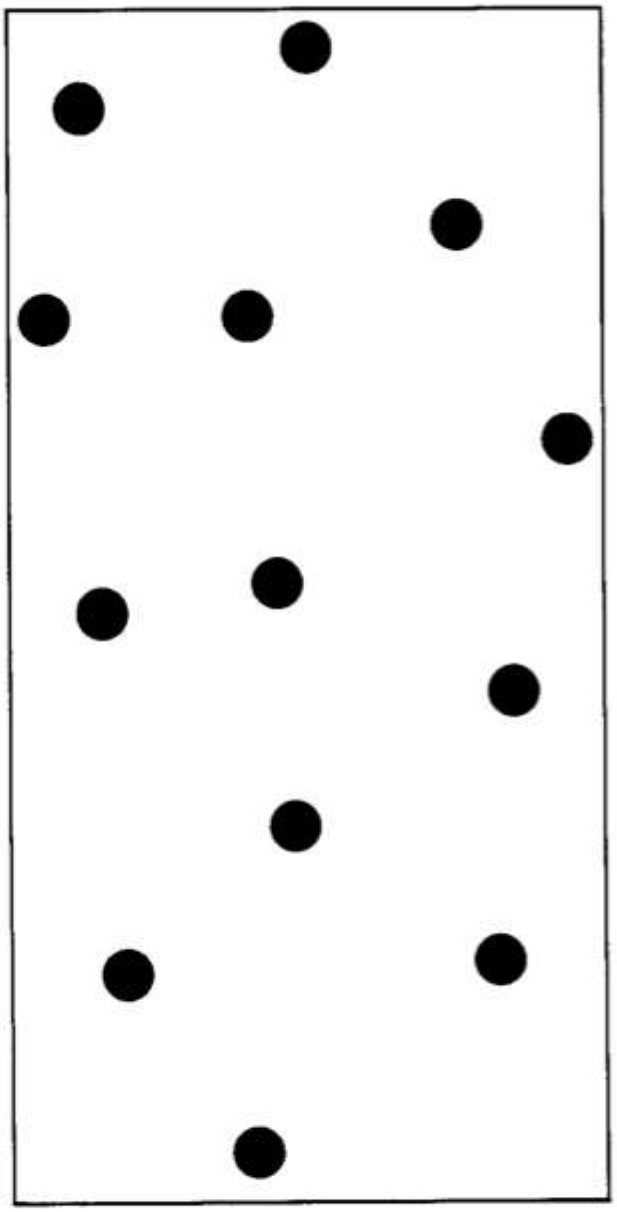
4. 現在讓學生加入一個斯坦納點，看看有沒有更短的解法。（最好的點在正方形的中心，如圖 (c)。總長度為 $2\sqrt{2}$ ）好了之後，讓大家放兩個斯坦納點，看看能不能得到更好的解法。（可以，把兩個點擺在圖 (d) 的地方，道路之間形成 120度的夾角。總長度為 $1+\sqrt{3}=2.73$ 公尺。）

如果加入三個斯坦納點會有更好的解答嗎？（不 — 加入兩個點已經是最佳解了。再多並不會得到更好的解答。）
與學生討論為何這些問題看起來如此困難。（這是因為不知道該在哪裡放斯坦納點，有太多可能性要試了。）

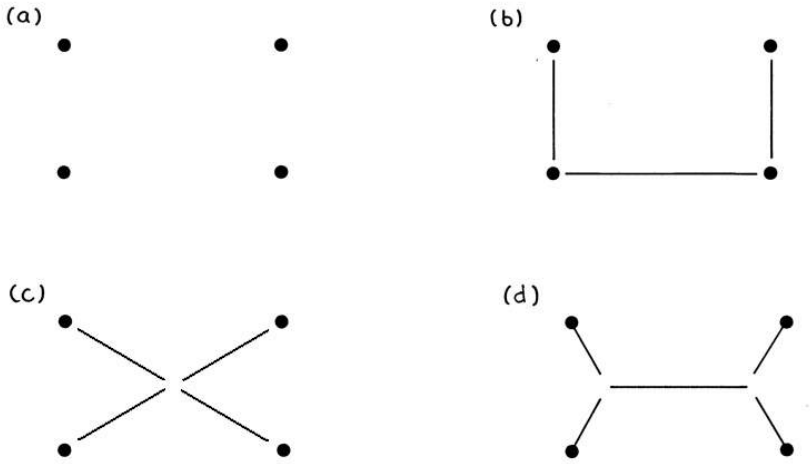
活動學習單：斯坦納樹範例 1



活動學習單：斯坦納樹範例 2

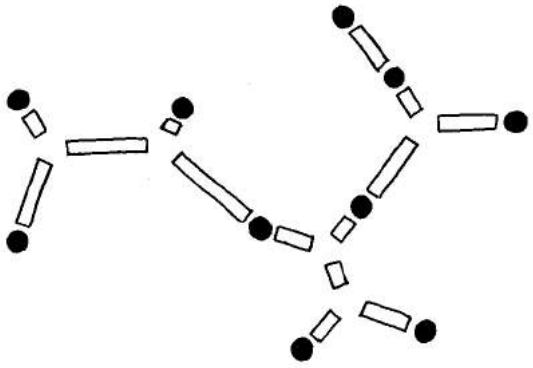


活動變化與延伸

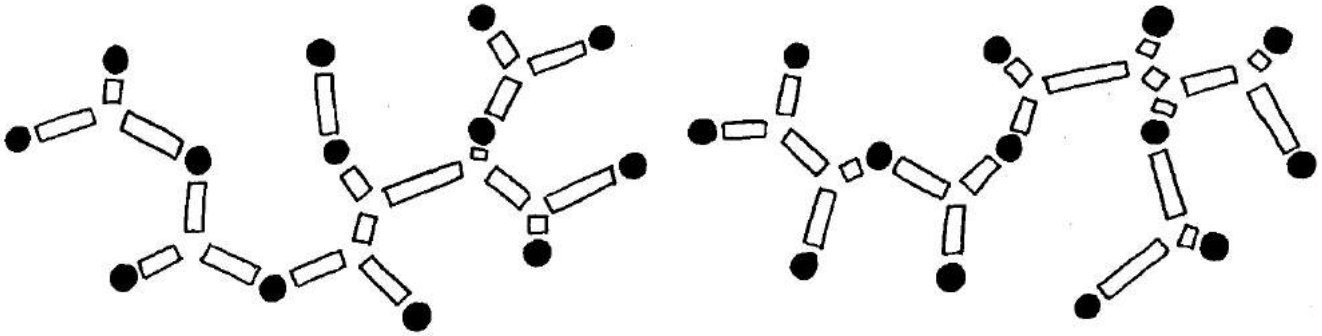


1. 如果有幾組提早完成前面的活動，可以讓他們進行一個有趣的實驗。圖 (a) 是個長 2 公尺寬 1 公尺的矩形。讓他們試試同樣的活動，學生們會發現加入一個斯坦納點反而會讓事情變糟，但是加入二個點則可以得到較短的路徑。（圖 (b) 的路徑長是 4 公尺，圖 (c) 是 $2\sqrt{5} = 4.47$ 公尺，而圖 (d) 是 $2 + \sqrt{3} = 3.73$ 公尺。）看看他們能不能夠找出為什麼在長方形中加入一個斯坦納點反而會比在正方形裡加入還糟。（這是因為當正方形拉長成長方形時，在圖 (b) 和 (d) 伸長量只有一條會增加，但圖 (c) 卻是兩條對角線都增加。）

2. 較年長的學生可以試試較大型的題目。他們可以在學習活動單上的兩張圖實驗不同的解決方式。可以把圖印出來，或用白板筆寫在蓋著透明投影片的圖上。或者到戶外去，用釘子在地上標記出來。當他們每次創下最短總距離的紀錄時可以跟其他人分享。（右圖是第一個範例的最佳解，而下頁的圖顯示的是兩種第二個範例的兩個可能的解法。兩種方式的距離相當接近。）這兩種解法如此接近，也說明為什麼這類問題會這麼困難 — 要在哪裡放置斯坦納點，有太多種選擇了。

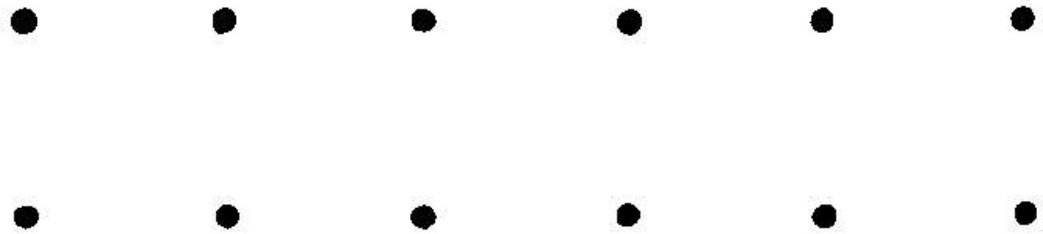


第一個範例的最佳解

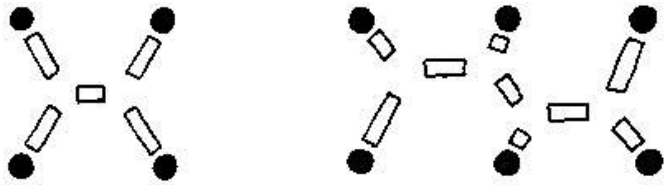


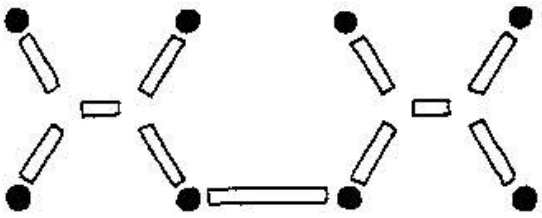
第二個斯坦納樹範例的兩種可能的解法

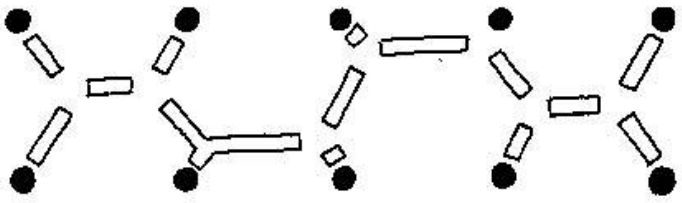
3. 梯子型網路（如下圖）也提供了此類問題的不同變化。



這裡是梯子型網路的一些最小斯坦納樹解法。

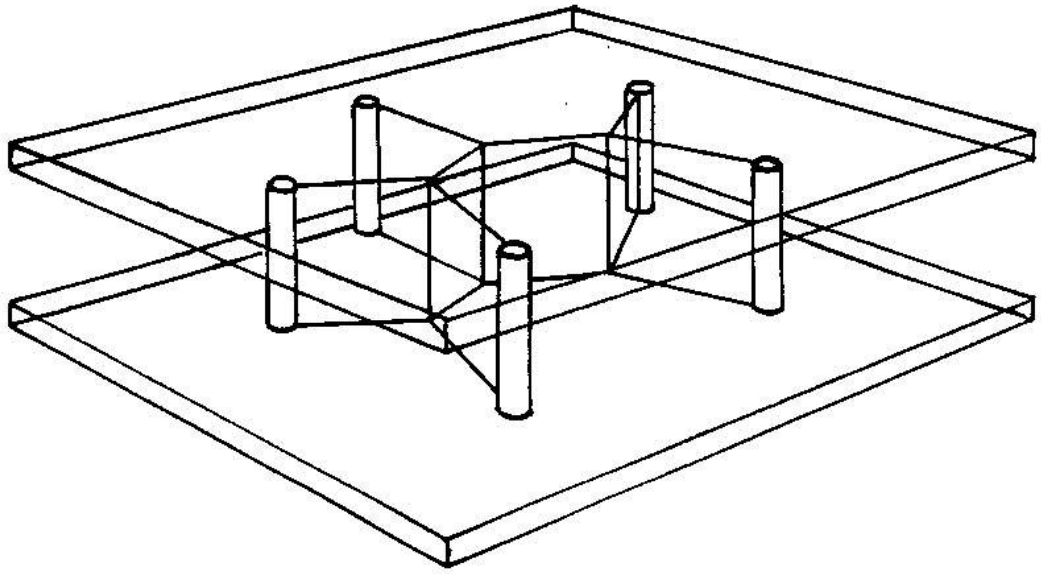






梯級為二的梯子型網路其實就是一個正方形。然而，梯級為三的梯子型網路，解法是相當不同的—試著用你記憶中的方式去畫畫看就會發現！梯級為四的解法，就跟把兩個梯級為二的梯子型網路組合在一起。而梯級為五的，又比較像梯級為三的網路的延伸。一般來說，梯子型網路的最小斯坦納樹的形狀取決於它的梯級是偶數還是奇數。如果是偶數，那就是多個梯級為二的網路組合在一起。反之為奇數，就會跟梯級三一樣的解法。但要嚴謹地這證明以上的結論，是相當不容易的。

4. 另一個有趣的活動是建構斯坦納樹的肥皂泡模型。你可以用兩片透明的投影片，然後把針插在它們中間，去表示這些點。如下圖：



接著把整個模型都浸入肥皂泡溶液中。當你把它拿出來時，你會發現肥皂泡泡的膜連著針而呈現出一個美麗的斯坦納樹網路。

然而，它所呈現出來的不一定是個最小的斯坦納樹。這些肥皂膜確實呈現出一個總距離最小化的狀態，但這個最小值只是局部的，不一定是總體的。可能有個完全不一樣的放置斯坦納點的方式，總距離更短。舉例來說，把活動學習單範例二的結構浸到肥皂泡溶液中。當它從肥皂液體中拿出來時，你可能會看到前面所說的第一種斯坦納樹，也有可能看到第二種。

這個活動在說什麼？

這個活動在講「最小斯坦納樹」。之所以叫「樹」是因為整個圖形裡沒有循環，只有像真的樹一樣分支，不會插入另一個分支或主幹上一起生長。而之所以叫「斯坦納樹」是因為可以在原來的樹所連接的工作站中間加入新的點（也就是「斯坦納點」）。最後，之所以說是「最小」是因為追求的是連接所有工作站形成的最短總路徑。在「泥濘城市」（活動 9）中，我們學到了找出連接所有點而使總長度最小所形成的樹，叫做「最小生成樹」（minimal spanning tree）：斯坦納樹也是一樣，只是在過程中允許加入新的點。

有沒有一種有效率的演算法來找出最小生成樹呢？之前我們試過貪婪法，也就是重複地找出任兩點中距離最小的，如果有尚未連接的點就把它連起來。這個方法對最小生成樹是可行的，但是對最小斯坦納樹則不然 — 目前還沒有一個通用有效率的方法。為什麼？因為在斯坦納樹問題中，你必須決定要把斯坦納點放在哪裡。事實上嘛，令人驚訝的是，困難的部份並不是精確地決定斯坦納點要放在哪裡，而是估計斯坦納點的位置：比方說在學習活動單範例二中，兩個解之間的差別其實非常的小。一旦知道要放在哪個區域，決定最佳的斯坦納點其實就只是微調位置的問題而已；而估計哪個區域可以真正縮短總長度才是困難所在。肥皂膜可以很有效率地做到這一點，電腦也可以。

尋找最小斯坦納樹對過去的電話業務來說是一個很重要，可以節省大量成本的事。在 1967 年以前，在美國的企業客戶要有自己的私有電話網路，要向電話公司租線路。他們計費的方式不是實際上鋪了多長的線路，而是以最短可以運作的網路為基準計費。理由是客戶不需要為了電話公司的方便而支付多餘的線路費用。最開始，計算要收費多少的方式是用最小生成樹。然而，在 1967 年左右，有個客戶（是間航空公司，有三個主要的電話中轉站）不小心發現了如果他們要求在某個點設第四個中轉站，線路的總長度反而會減少。電話公司只好依照假裝第四個中轉站（也就是斯坦納點）存在的狀況下計算，而被迫減少收費。雖然典型的狀況下，最小斯坦納樹只比最小生成樹少大約 5% 到 10%，但對花大錢的客戶來說，節省的成本還是很可觀的。斯坦納樹有時又被稱為「最短網路問題」，因為它就是在找連接一組工作站所需的最短網路。

如果你有玩過前兩個活動，也就是「貧窮的製圖師」與「旅遊小鎮」，那你應該不意外 — 斯坦納樹問題也是一個「NP 完全問題」。當工作站的數量增加，可能的斯坦納點的樹量也會跟著增加，而找出所有可能性的時間就會成指數成長。這個問題跟其他成千上萬的問題一樣，目前還不知道到底有沒有多項式時間演算法來解決這個問題。如果有，那它的多項式時間演算法就可以轉換成著色問題，還有最小支配集，還有許許多多其他 NP 完全問題的多項式時間演算法。

我們在前一個活動的最後解釋了什麼叫做「NP」：「非確定性多項式」（non-deterministic polynomial），而「完全」（complete）指的是如果在其中一個問題找到的一個有效率的方式來解決，那麼此方法可適用於解決同組中任何一個其他的問題。可以用多項式時間演算法來解決的問題就稱為 P。那麼，關鍵的問題來了：NP 完全問題是否有多項式時間演算法可以解？換句話說，

P = NP 嗎？這個問題目前還沒有答案，而且是現代資訊科學中最大的謎之一。

那些可以用多項式時間演算法解決的問題 -- 即使看起來要花很多時間 — 稱為「可駕馭問題」（tractable）。反之，還沒找到多項式時間演算法的問題，則稱為「不可駕馭問題」（intractable），因為不管你的電腦多快，或是同時用多少台電腦下去跑，只要問題的規模做小規模的增加，就有可能變成現實上無解的狀況。目前 NP 完全問題，包括前面說到的貧窮的製圖師、旅遊小鎮、冰之路等等，還不知道是不是屬於易解問題。但是大多數的電腦科學家對 NP 完全問題是否找得到多項式演算法解是持悲觀的態度。所以證明某問題為 NP 完全問題也同時被認為是本質上屬於難解問題的有力證據。

想像一下，如果你的老闆要求你對一個問題設計出一個有效率的演算法並找到最佳解，但是你做不到時怎麼辦？當然，如果你能夠證明這個問題沒有一個有效率的演算法得到最佳解，這樣最好。但在資訊科學的領域中，做這種負面結論的證明是非常困難的。因為誰也不知道未來會不會有個聰明的傢伙，偶然發現一個不起眼的方法，把整個問題解決了，就像前面的航空公司發現加上一個斯坦納點可以節省網路的成本那樣。所以，你其實很難告訴老闆說問題沒有有效率的解。不過，你可以證明該問題是個 NP 完全問題，並告訴老闆說已經有成千上萬的人在努力解跟老闆丟給你一樣的問題，而他們目前仍一無所獲。嗯，這樣是不會幫你加分啦，但至少可以讓你脫離困境！

你找不到一個有效演算法時的做法：三種可能性

當然啦，在現實生活中這些問題都還是要解決，而這種狀況下人們就退而求其次 — 找一個「啟發式演算法」（heuristics algorithm），也就是說這類演算法不保證找到最佳解，但是可以找到最佳解的一部份。啟發式演算法可以非常快，而找不到最佳解時所耗費的成本也相對少，因此足以解決現況。只是一旦找到比現有更好一點點的課表，或網路，或路線的時候，會覺得之前怎麼那麼笨。

延伸閱讀

上面的圖片來自 Garey 與 Johnson 的經典教科書 "Computers and Intractability"。

1984 年六月出版的美國科學雜誌（Scientific American）裡的「電腦娛樂」專欄中，有關於如何用肥皂泡泡產生斯坦納樹，還有很多其他模擬小工具的有趣描述，像是排序用的義大利麵電腦，用來在圖形中找最短路徑的貓咪的搖籃，還有判定一個數是否是質數的光反射鏡的裝置等等。這些東西也在 Dewdney 的 Turing Omnibus 裡關於模擬電腦的章節也有提到。

第五部份
機密分享與打擊犯罪－密碼學

機密分享與打擊犯罪

你一定聽過間諜和特務使用隱密的代碼或隱形墨水寫下並交換訊息之類的故事。那就是「密碼學」這門學問的源起。密碼學是一門傳遞訊息的藝術。在二戰期間，英國人建造了專門破解密碼的電子機器，並且用這機器來破解敵軍的軍事機密訊息。接著，電腦的出現改變了一切，並且使密碼學進入了一個新的時代。在過去完全無法想像的大量計算，現在可以透過配置完善的電腦來進行。當人們開始互相分享電腦系統時，密碼出現了新的用法。而當電腦連在網路上時，人們就必須開始保護自己的資訊，以免被其他人竊取。當電子郵件出現時，確定郵件內容有沒有被改過，是不是真的是由發信者所發出，還是別人偽造發信者的身份所發的，就變得很重要了。現在電腦可以使用網路銀行的功能，也可以透過電腦買賣商品，在電腦網路上我們就非常需要訂單與金流的安全管道。而恐怖份子攻擊電腦系統所帶來的威脅日益增加，也使得電腦安全越來越重要。

講到密碼學，可能讓你想到儲存密碼的電腦，和弄亂訊息的文字使得敵人無法解讀之類的事。但實際上並不是這樣。現代電腦系統不會儲存密碼，因為如果系統這樣做，任何設法取得密碼的人就可以突破系統內所有的安全機制，那會是一個大災難：他們可以盜領帳戶內的錢、假裝是其他人傳送訊息、讀取所有人的機密資料、甚至命令軍隊推翻政府。在現代，密碼會透過「單向函數」（one-way function）這個我們在活動 15 中提到過的方式來處理。還有加密不只是弄亂訊息的文字：它是使用到前面第四部份所說的「不可駕馭問題」的技術。

使用密碼學，你可以做你認為不可能的事情。在這一部份，我們會玩一個簡單的遊戲，在一組中所有人都不透露自己年齡的情況下，就能計算出這一組平均年齡。另外也會講到如何使兩個身在不同城市，看不到彼此的人擲硬幣，並且兩人都能同意硬幣擲出的結果。最後我們也會講到怎麼去編出只能被某一個人解碼的機密訊息，即使所有人都知道編碼是如何做的。

給老師的話

接下來的活動提供了現代密碼技術的實際體驗 -- 它們與大部分的人想像中的電腦與保密有很大的不同。

這裡有兩個關鍵的概念。第一個是「協定」（protocol）的概念，它是一種交易往來（transaction）的正規化敘述。Protocol 這個字可能會讓人想到外交官，甚至禮節（譯註：英文中的 protocol 也有「禮儀」、「禮節」的意思），但電腦也會用喔！看起來很困難的工作可以被超乎想像的簡單協定所完成。在活動 16 中，我們只會花幾分鐘的時間，透過團隊合作，就可以在所有人都不需要透露個人的年齡（或收入）的情況下，簡單的計算出他們的平均年齡（或收入）。第二個關鍵的概念是透過電腦與他人互動時，計算複雜性 — 也可以說是困難性（intractability） — 所扮演的角色。在活動 17 中，我們可以看到兩個人如何在不需要信任彼此，只透過電話聯繫就能同意擲硬幣的結果沒有作弊。（在這個活動中，也連帶介紹了布林 (boolean) 邏輯電路的觀念，以及如何用它來運作。）在活動 18 中，我們會學到如何利用計算技巧來安全地將訊息加密，即使加密的方法是公開，大家都知道的。

這些活動中的一部份 — 特別是最後一個活動 — 困難度不低。老師們會需要激勵學生，告訴他們這些大多數人覺得不可能做到的活動，是絕對可以完成的。建立這種驚奇感，與學生持續溝通，在活動中頻繁地暫停確認，讓這種感覺在活動中能一直持續著，以免學生因為樹（無聊的感覺）而錯過了整個森林（成就感與驚奇感）。這些活動是這本書中最具有挑戰性、最複雜的。如果發現這些活動對學生真的太難的話，請跳到第六部份。這是完全不一樣的、不需要技巧的章節。

密碼學技術的應用與爭議

當電腦逐步入侵我們的日常生活，密碼學的應用也變得更加重要。大多數的人根本不知道現代的加密協定是怎麼一回事。結果就是當大型機構 — 包括政府與商業機構 — 建立了涉及個人資訊的系統，會變成由這些機構來做重要決定，包括如何處理事物、要蒐集什麼資訊、什麼東西可以開放給誰等等。如果人們對現代科技產生的可能性有更深的了解的話，他們就能主動參加更多像這樣的決策，而整個社會也會建立起一個不同的資訊基礎。

資訊隱藏協定、加密協定還有公開金鑰編碼的本質一般來說被認為是非常先進的。但這些想法本身並不困難。難以理解的是它的技術性，而不是它的觀念。在現實的狀況中，例如電子商務，這些技術都用軟體包裝好了，因此加密技術就變得很容易使用。但是瞭解這些技術的基礎想法，知道這些軟體能做什麼也是很重要的事。

政府對密碼系統有很大的興趣，不只是因為要維護官方通訊的安全性，也是擔心從事非法活動的人們，像是毒品走私和恐怖活動等等，也會使用加密通訊。如果這樣的人使用加密，那竊聽就會變得沒用，除非知道怎麼解密。這些擔憂在執法人員（想要限制密碼系統的運用）與公民自由主義者（對於政府監聽私人通訊的行為感到不滿）間引起不少爭論。

有一段時間美國政府限制了一些密碼學方法，將他們視為軍火 — 像是炸彈或槍械之類的。當然，只要有正確的資訊，加上一點技術的能力，任何人都可以設立一個安全通訊系統，但這些系統若落入「壞人」的手裡就會很危險。在某一個階段，大家對 "Clipper Chip" 有過廣泛的爭論。Clipper Chip 是一種由美國國土安全局所開發的晶片，裡面有一種叫做「託管密鑰」的後門。任何透過此晶片加密的訊息，都可以被政府相關單位透過託管密鑰來解密。 FBI 和美國司法部想要讓這種經片廣泛運用在通訊上，但這引來很大的反對意見，因為會威脅到個人的隱私。任何一種加密系統在技術上都是可行的，但在政治上卻不一定被接受。

密碼學的觀念並不只是保密訊息而已。另外還有許多應用，像是保證訊息真的是傳送者所發出 — 這個機制稱為「認證」，沒有這種機制，就不會有現在的電子商務。還有用在電子投票，確保其他人 — 包括營運該系統的人 — 不會知道投票者投給誰，同時還能避免一個人投兩票以上。你甚至運用密碼學來經由電話玩牌 — 這可能聽起來很蠢，但若你瞭解到做生意其實就很像在玩撲克牌，你就懂了。

這些事情聽起來是不可能的。如果你正在和電話另一端的人比賽，因此不能信任他，那你們要怎麼洗牌？你怎麼發現有人從中截取了一個訊息並修改它，然後假裝是電話另一端的人發送過來給你的？如果你不知道這些事，那你就別想做電子商務了。你必須預防有技術的罪犯，利用截取銷售點和銀行間的電話線路，來偽造授權以便從銀行帳戶提款。你必須要避免商業上的競爭者們因為故意產生假的訂單或合同而造成嚴重的損失。這些看來很困難的事，都可以奇蹟似地用現代密碼技術做到。而這些活動會告訴你怎麼做。

還有許多關於編碼和破解的有趣的書。"Codebreakers: the inside story of Bletchley Park"，由 Hinsley 與 Stripp 所編寫的書，裡面有一些第一手資料，告訴你第一代的電腦如何在第二次世界大戰中被用來解密，從而大幅縮短戰爭時間並拯救許多人的性命。相關的文章在泛科學 [1] 與科學月刊 [2] 等網站也都有深入淺出的介紹，大家可以參考閱讀。

[1] http://pansci.asia/archives/59156 《泛科學》「網路驗證碼與電腦科學之父」
[2] http://pansci.asia/archives/92180 《泛科學》轉載《科學月刊》 2016 年 1 月號

活動 17
傳遞機密 — 資訊保密協定

活動摘要
加密技術讓我們能在擁有高度隱私的情況下和他人分享訊息。在這個活動裡提供了一個互相交流訊息但不洩露個別資訊的方法：一群學生能在不洩漏個別實際年齡的情形下，計算出他們的平均年齡。

課程銜接
數學：加總與平均

習得技能
- 計算平均
- 隨機亂數
- 協同合作

適合年齡
- 7 歲以上

所需素材
每組學生需要：
- 一些計算紙
- 一隻筆

傳遞機密

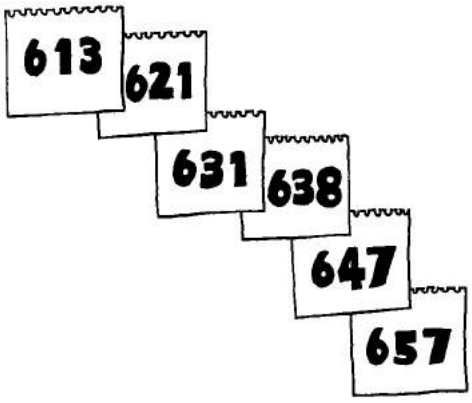
活動介紹

這個活動能在不洩漏任何人實際年齡的情形下，算出一群學生的平均年齡。另外，也可以用來找出一群學生平均拿到多少的零用錢，或是一些其他比較個人的資訊。這個方法尤其適合計算成人們相關的統計數字，因為年紀較長的人通常對收入和年齡等資料較為敏感。

一組至少要有三個人以上。

活動討論

1. 向組員說明你想要計算大家的平均年齡，但不能洩漏任何人的年齡。問問大家對這個要求有沒有什麼想法，或甚至問問大家覺得有沒有可能。



2.找出六到十個人來。給第一個學生紙和筆，請他們在紙的最上方隨機寫下一個三位數字。以此圖的範例來說，613 是第一位學生寫下的隨機數。

3.請第一個學生撕下第一頁，並將自己的年齡與亂數相加，寫在第二張紙上。比方說，第一個學生 8 歲，那就在第二張紙寫上 621。每個人要拿老自己的那張紙，不能讓別人看到。

4.將便條傳給第二個人，第二人撕下第二張紙，將自己的年齡與第二張紙條上的數字相加，並將總和寫在下一張紙上。在這個範例中，第二個學生 10 歲，因此第三張紙上寫的是 631。

5.持續照著這些流程，直到每個人都寫過，並且在手上有一張不給別人看到的紙條。

6.將便條紙傳回給第一個學生，將紙上的數字減掉自己手上原來的數字。在範例中，便條紙總共傳給了五個人，最後的數字是 657，減掉第一個學生手上原本的數字 613，得到 44，這個數字即是所有學生的年齡總和，就可以以此算出平均值。 因此，在此範例中的學生平均年齡為 8.8 歲。

7.告訴學生們，除非兩個人刻意合作，不然只要撕毀自己的那張紙，就沒有人能知道自己的年齡。

活動變化與延伸

這個活動的進行方式也可以用來做秘密投票。做法就是每人同意則把數字加一、不同意則加零。當然，如果有人加或減了 0、1 以外的數字，這個投票就變得不公平了，雖然這種作弊方式還是有一定風險，當所有人都投同意票時，如果有人加超過 1 的數字，會讓同意票總數超過參與的人數。

這個活動在說什麼？

電腦儲存了大量的個人資訊：銀行存款餘額、個人社交網路、欠稅、駕照持有時間、信用記錄、考試結果、病歷等等。個人隱私非常重要！但某些情況下，我們仍然必須與其他人分享一些這方面的資訊。例如，當我們使用銀行卡付購物費用時，店裡需要驗證我們的存款是否足夠支付。

通常情況下，我們會提供比實際需要還要更多的資訊。舉例來說，如果我們在商店進行電子交易，他們實際上會知道我們用哪個銀行的帳戶、帳號為多少、我們的名字。此外，銀行也會知道我們在哪裡購物。銀行可以透過監測客戶在哪裡購買雜貨、每天花了多少錢購買、在什麼時間購買，藉此來建立個人資料。如果我們都使用現金付款，那當然就不會有任何個人資訊向外流出。大多數人都不太擔心個人資訊被共享，但其實有個潛在危險是這些資訊會被濫用，無論是在市場上（比方說，業者會針對某些花很多錢在購買機票的人，發送關於旅遊的廣告）、階級歧視（只針對富裕的客戶提供更好的服務）、甚至勒索（例如威脅某人要透露一些不想讓人知道的交易）。一般來說，如果人們知道自己購物方式會被監控，都會想改變這樣的購物方式。

這些隱私的損失大家似乎比較不以為意，但加密協定的存在仍然允許我們在用電子金融交易時，可以擁有跟現金交易相同的隱私水準。你可能很難相信，錢可以從銀行帳戶轉到商店帳戶，而沒有任何人知道錢的來源與去向。這樣做可以讓電子交易變得更合理：收付雙方都只分享到有限必要的資訊，而這件事可以透過一個聰明的協定來完成。

延伸閱讀

David Chaum 曾提出一篇經典的論文，將這些問題突顯出來，並使用了一個頗為挑釁的標題：「沒有身分識別的安全：一個讓老大哥（Big Brother）過時的交易系統」（譯註：Big Brother 是喬治·歐威爾所寫的小說「1984」中的角色。老大哥是大洋國的領袖，但沒有人實際上看過他的存在，只知道老大哥一直會盯著人民的一舉一動。）。此文頗值得一讀，裡面提供了一些關於資訊保密協定的簡單例子，包括如何使用電子現金交易而完整的保有隱私。這篇論文可以在 "Communications of the ACM"，1985 年十月號裡找到。

活動 18
秘魯式拋硬幣 — 加密協定

活動摘要

這活動讓大家瞭解，如何實現一個簡單但看似無法達成的任務。在兩個只透過電話溝通且互不信任的人之間，透過拋硬幣做出一個公平隨機的選擇。

課程銜接

- 數學：邏輯推理
- 數學：布林邏輯

習得技能

- 布林邏輯
- 函數
- 解謎

適合年齡

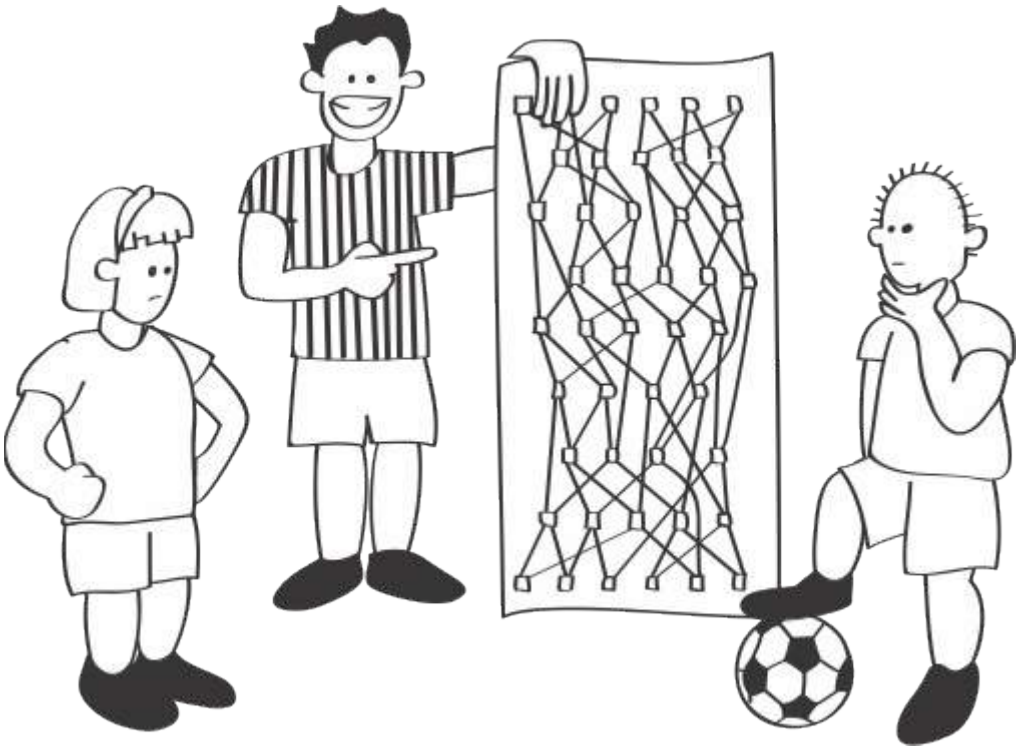
- 9 歲以上

所需素材

每組學生需要：

- 活動學習單：秘魯式拋硬幣（第 203 頁）的影本
- 兩種不同顏色的按鈕或籌碼大約兩打

秘魯式拋硬幣



活動介紹

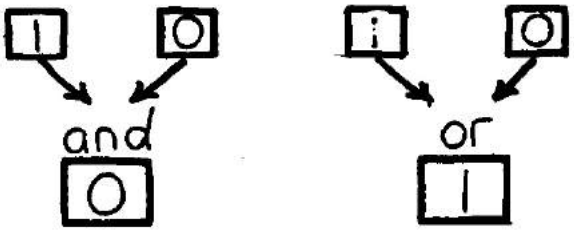
這個活動是本書的其中一個作者（MRF）在祕魯和學生上課的時候所發明的。你也可以想一個故事來套入這種情境。

利馬跟庫科斯這兩支足球隊要決定冠軍賽的主場球隊是哪一隊，而拋硬幣決定是最簡單的方法。但是因為這兩個城市相隔十分遙遠，代表利馬的 Alicia 與代表庫科斯的 Benito 不能只是為了拋硬幣而花很多時間和金錢來見面。那他們可以藉由打電話來做到這件事嗎？比方說，讓 Alicia 拋硬幣而 Benito 猜是正反面。但這是不可行的，因為只要 Benito 猜正面，Alicia 只要說「抱歉，是反面」就能輕易取得主場優勢。Alicia 其實不奸詐，但這畢竟是個重要的比賽，對她的誘惑極大，而就算 Alicia 非常誠實好了， Benito 輸了的話又怎麼會信服呢？

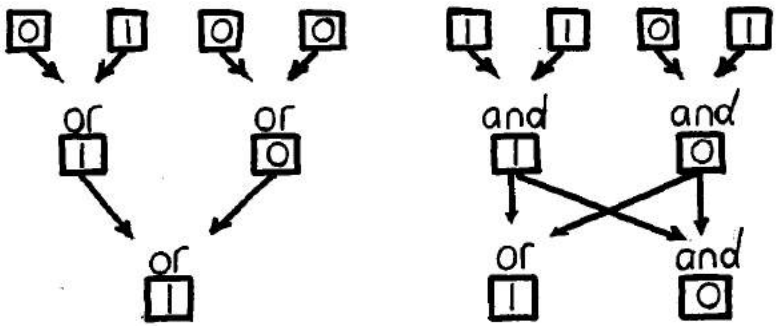
如果學生已經學過二進位表示法（活動 1）、同位元概念（活動 4）、還有單向函數（活動 15）的例子，那會對這活動更為上手，並學得更多。

最後他們決定一起設計一個由「And 邏輯閘」與「Or 邏輯閘」所組成的電路。原則上他們能透過電話（或是用電子郵件也可以！）做這件事。雖然免不了在實作的過程中會有些乏味，但在過程中，其中的樂趣就是確定這個電路足夠複雜而另一人無法作弊，而最後設計出來的電路是大家一起想出來的智慧結晶。

活動討論

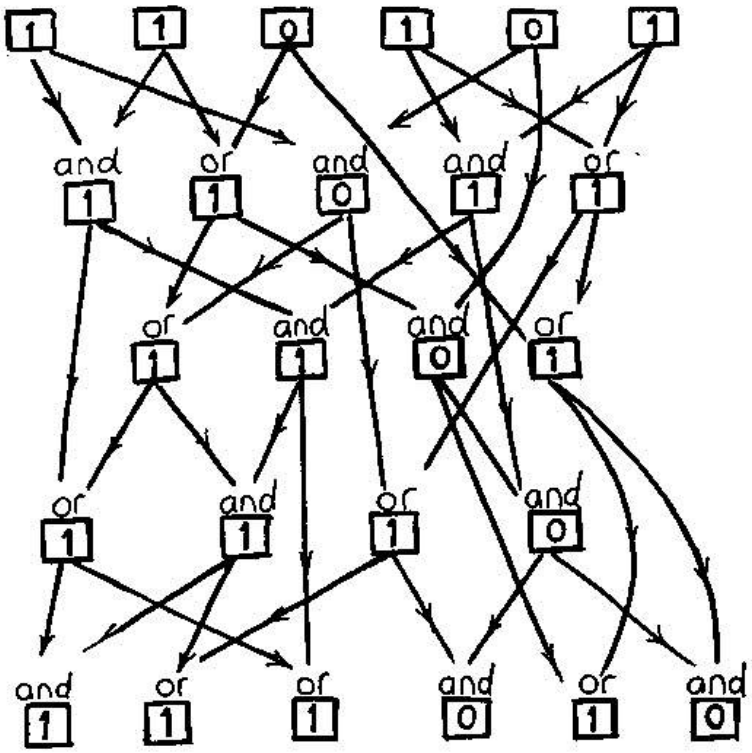


「And 邏輯閘」與「Or 邏輯閘」的原理很簡單。每個閘都有兩個輸入和一個輸出。輸入可以是 0 或 1，分別代表假（false）和真 （true）。當兩個輸入都是 1 的時候，And 邏輯閘的輸出是 1，其他種狀況下輸出都是 0。舉個例子，圖中的 And 邏輯閘的輸入為 0 和 1（在上方），所以輸出（底部正方形）是 0。而 Or 邏輯閘只要在其中一個輸入是 1 時，輸出就會是 1，只有兩個輸入都是 0 時，輸出才會是 0。換言之，當輸入是 1 和 0 時，輸出就會是 1。



而一個閘的輸出可以再連結成其他閘的輸入，以此形成更複雜的電路。舉個例子，左手邊 Or 邏輯閘的輸出為第三個 Or 邏輯閘的輸入，因此 4 個輸入裡只要有一個是 1，最後的結果就會輸出 1。而右邊的電路，上方兩個 And 邏輯閘的輸出會餵進去下面兩個閘，所以這個電路總共會有兩個輸出。

對這個拋硬幣活動，我們需要更多複雜的電路。書上的電路共有 6個輸出和 6 個輸入，下方是某種輸入後結果的舉例。



透過電話進行拋硬幣遊戲的電路操作方法如下，Alicia 選擇一組隨機的六個二進位位元（0 跟 1）輸入給這組電路，這一組輸入要保密。接著她告訴 Benito 電路輸出的結果，而 Benito 就必須猜測 Alicia 的輸入是奇數個 1 還是偶數個 1 — 換句話說，他要猜測的是其奇偶性。如果這個電路夠複雜，那 Benito 就無法隨意算出答案，所以只能隨機地做個選擇（就好像丟硬幣來猜一樣）。如果 Benito 猜對了，則那庫科斯就得到主場優勢。反之，利馬得到主場優勢。當 Benito 把他的猜測告訴 Alicia 之後，Alicia 會把她保密的輸入告訴 Benito，而 Benito 也可以藉由同樣的電路來證實 Alicia 告訴他的輸入是否正確。

1. 把學生分為兩組，給每組電路圖和一些籌碼，並告訴他們這個故事。可以把情境設定成跟學生比較相關，比方說，學生參與的運動校隊正在決定冠軍賽的先攻後攻。規定一下籌碼的顏色 — 比方說紅色是 0，藍色是 1 — 然後叫學生在這張紙上面的記下來以免搞錯。

2. 示範如何利用籌碼把 Alicia 選擇的位元放進輸入裡，然後在這張紙的下方解釋 And 邏輯閘與 Or 邏輯閘的規則（可以叫學生用顏色標記）。

3. 示範如何把籌碼放在每個節點上，然後經由電路產生相對應的輸出。這必須要小心不要弄錯。下表（不能給學生看到）有對應每種輸入的正確輸出，可以讓老師確認學生的結果是否正確。

Input	000000	000001	000010	000011	000100	000101	000110	000111
Ouput	000000	010010	000000	010010	010010	010010	010010	010010
Input	001000	001001	001010	001011	001100	001101	001110	001111
Ouput	001010	011010	001010	011010	011010	011010	011010	011111
Input	010000	010001	010010	010011	010100	010101	010110	010111
Ouput	001000	011010	001010	011010	011010	011010	011010	011111
Input	011000	011001	011010	011011	011100	011101	011110	011111
Ouput	001010	011010	001010	011010	011010	011010	011010	011111
Input	100000	100001	100010	100011	100100	100101	100110	100111
Ouput	000000	010010	011000	011010	010010	010010	011010	011010
Input	101000	101001	101010	101011	101100	101101	101110	101111
Ouput	001010	011010	011010	011010	011010	011010	011010	011111
Input	110000	110001	110010	110011	110100	110101	110110	110111
Ouput	001000	011010	011010	011010	011010	111010	011010	111111
Input	111000	111001	111010	111011	111100	111101	111110	111111
Ouput	001010	011010	011010	011010	011010	111010	011010	111111

4. 現在每組都都分成兩半，並分別選出一個人當 Alicia，一個人當 Benito。Alicia 要隨機選擇一組輸入放到電路中並計算輸出，然後把結果告訴 Benito。Benito 接著要猜測輸入的奇偶性（有奇數個還是或偶數個 1）。這個過程中必須確定 Benito 的猜測也是隨機的，然後 Alicia 會跟大家說她所選擇的輸入值。如果 Benito 猜對的話他就贏了。Benito 可以藉由在電路中檢驗輸出是否正確來驗證 Alicia 有沒有亂說。

然後這個拋硬幣就算完成了。

但如果 Benito 可以輕易找到一組輸入產生相同的輸出結果，他就可以作弊了。所以 Alicia 必須確定這個電路的計算是個單向函數 （如活動 15、活動 16 討論的），才能防止 Benito 作弊。單向函數就是在有輸入值時，可以很容易計算輸出值；但是只有輸出值時會很難算出輸入值為何。

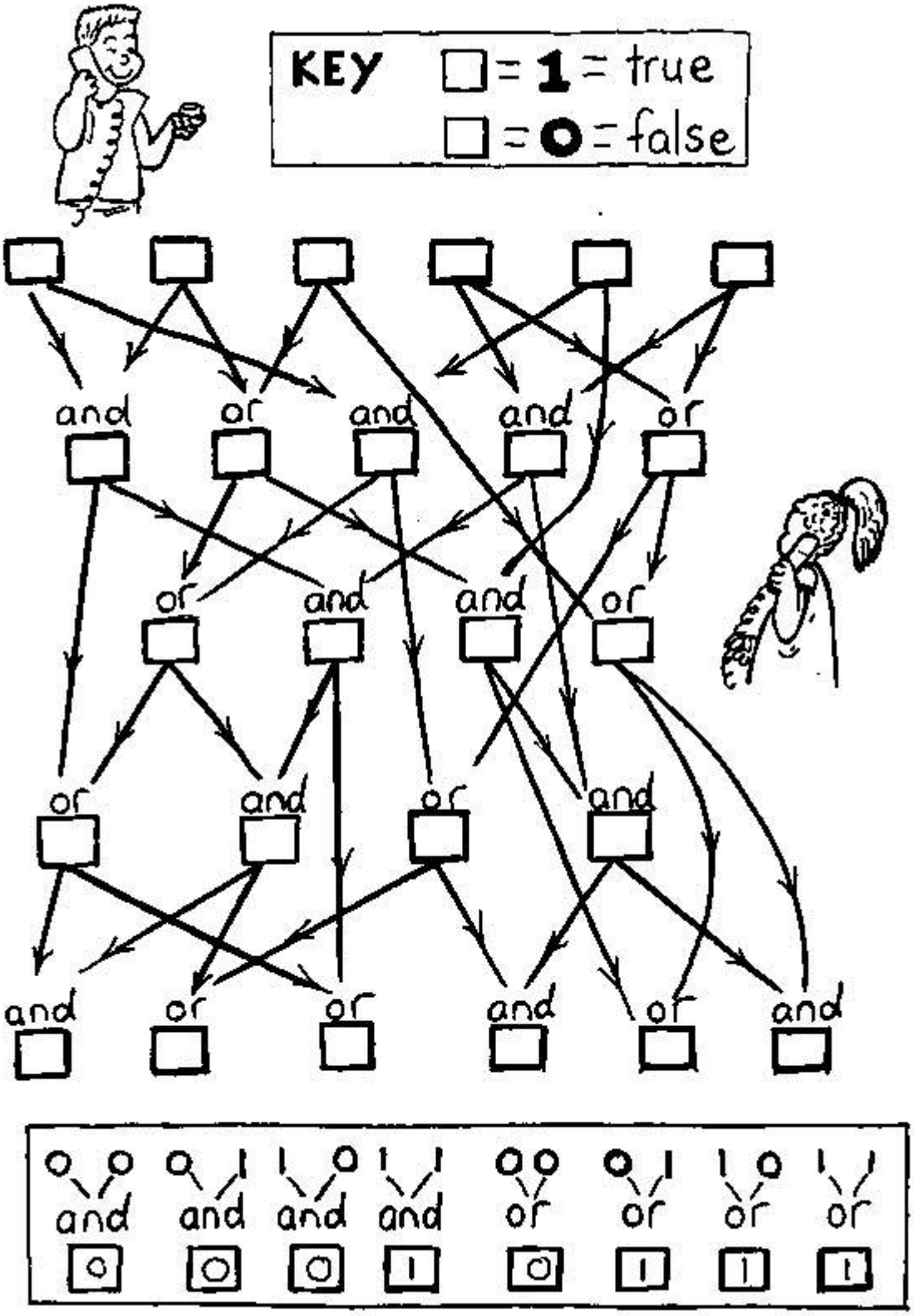
相對地，Alicia 如果可以找到不同奇偶性的輸入可以產生相同輸出，那就變成她就可以作弊了。因為不管 Benito 猜甚麼答案，Alicia 都可以說他是錯的，因此 Benito 也必須確定這個電路不會有多種輸入可以產生相同輸出。

5. 看看學生能不能找出讓 Benito 和 Alicia 作弊的方法。從這圖表的第一行，可以看到有好幾種輸入（000001, 000011, 000101）會產生相同的輸出 010010。因此如果 Alicia 給的輸出是 010010，而 Benito 猜偶數個 1，Alicia 可以說她輸入的是 000001；如果 Benito 猜奇數個 1，Alicia 也可以說輸入是 000011。

這電路對 Benito 來說是難以作弊的。但如果輸出是 011000，那輸入一定是 100010 — 沒有其他的可能性了（可以看圖表確認）。若 Alicia 給 Benito 的輸出剛好是 011000，那 Benito 猜偶數的時候就會是正確的，而且沒有爭議性。電腦系統會用更多位元，想作弊時就會多出更多可能性要試。

6. 現在讓每組學生為這遊戲設計出自己的電路，看他們能不能設計出讓 Alicia 可以作弊，或讓 Benito 可以作弊的電路。這電路不一定要六個輸入位元，甚至可以讓輸入與輸出的位元數不同。

活動學習單：秘魯式拋硬幣



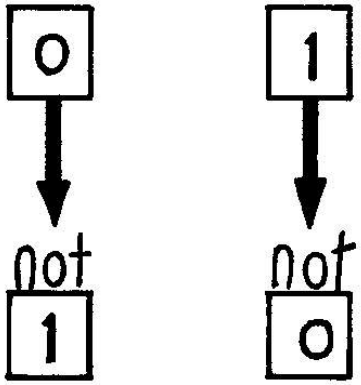
選擇電路的輸入值，並算出輸出值。

活動變化與延伸：

1. 這個方法在實務上有個明顯的問題，就是得建立一個 Alicia 和 Benito 都能接受的電路。這對小孩來說可能有趣，實際上卻可能使得整個程序無法進行 — 特別是透過電話！然而，有一個簡單的替代方案。Alicia 和 Benito 可以分別建構自己的電路，並把它們公開。然後 Alicia 分別在兩個電路輸入她選擇的位元，然後並比較兩個輸出對應的位元。如果兩組在某一個位元相同，那個位元的最後輸出就是 1，反之則為 0。在這樣的狀況下，只要有一個電路是單向函數，它和其他電路的組合也會是單向函數，參與者就無法作弊。

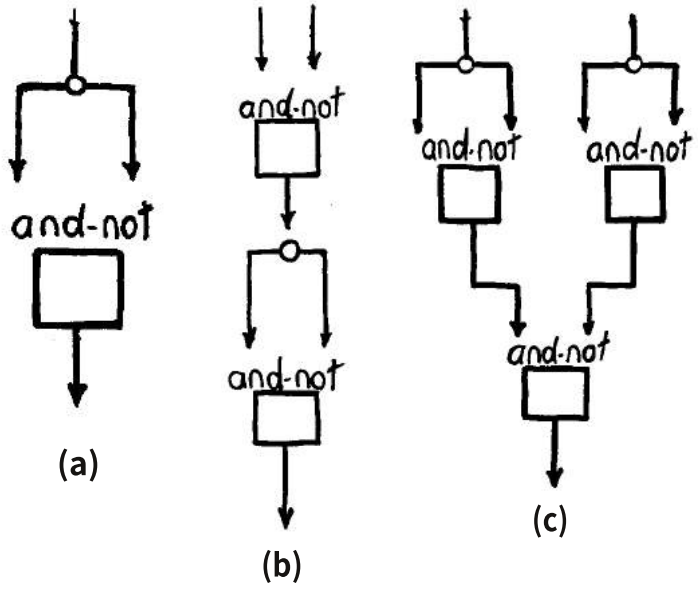
接下來的兩個變化型與加密協定或擲硬幣無關，而是關於建構一個沒有 And 邏輯閘與 Or 邏輯閘的電路。這裡探討了一些不只是電腦的電路，還有邏輯本身的重要概念。這種邏輯被稱為「布林代數」（Boolean Algebra），以紀念數學家喬治 · 布爾（George Boole, 1815-1864）。

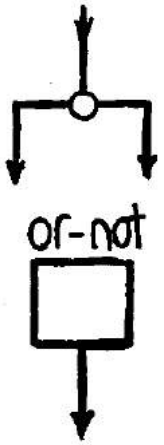
2. 學生們可能會注意到，全 0 的輸入，也就是 000000，會產生全 0 的輸出。同樣的，全 1 的輸入，也會產生全 1 的輸出。（當然，也是有並非輸入全零，卻輸出全零的例子，反之亦然）。這結論的確是根據 And 與 Or 邏輯閘做出來的。而借由使用 Not 邏輯閘（使輸出輸入相反），我們就可以輕易做出非（全 0 輸入與全 0 輸出）這樣的結果。



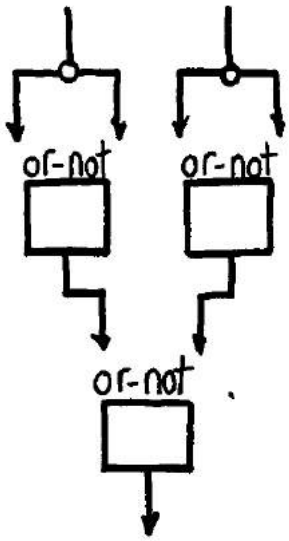
3. 另外有兩個重要的閘：Nand 和 Nor。他們代表 And 和 Or 邏輯閘後面接上一個 Not 邏輯閘。也就是說，把 a 跟 b 輸入到 Nand 邏輯閘，得到的輸出是 NOT (a AND b)。這看起來有點多此一舉，因為只要在 And 邏輯閘後面加一個 Not 邏輯閘就可以了。但是有一件有趣的事：所有其他的邏輯閘都可以只用 Nand 邏輯閘兜出來。只用 Nor 邏輯閘也可以。

向學生介紹 Nand 與 Nor 邏輯閘後，讓學生們試試看能不能用其他種邏輯閘兜出某一種，或甚至用同一種邏輯閘兜出某一種。右方跟下頁的六張圖，分別展示出用 Nand 兜出 Not, And 與 Or （a ~ c 圖），以及用 Nor 兜出 Not, And 與 Or（d ~ f 圖）的做法。

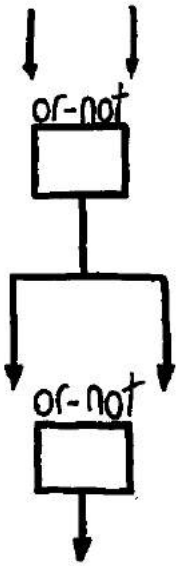




(d)



(e)



(f)

這個活動在說什麼？

近年來，在電腦網路進行的商務活動大幅的成長，其中不可或缺的是保證電子資金、機密交易還有所簽署具法律效力的文件交換的安全。密碼學的主題就是如何用安全且隱密的方式通信。幾十年前，資訊科學的研究人員發現一個違反直覺的結果：我們可以運用某些確保特定資訊公開的技術來達到將資訊保密的結果。這個結果就是我們會在活動 19 中所提到的「公開金鑰加密系統」，這也是現在廣泛用於交換訊息的主要加密方式。例如，你可能在瀏覽器中有看過 SSL（Secure Socket Layer，安全通訊協定）或 TLS（Transport Layer Security，傳輸層安全協定）的設定；這些系統都是基於公開金鑰系統，讓你的瀏覽器能夠與像是銀行這種網站建立安全的連線，就算有人在竊聽網路上的資料也沒有用。

不過密碼學不是單純保持資訊的機密那麼簡單。它也可以控制資訊，限制哪些部份可以被其他人找到；此外就是建立分隔兩地的人之間的可信任通訊管道。 加密的規則或「協定」的發展已經讓一些看起來不可能的事情發生，像是無法偽造的數位簽章，還有告訴別人你的某項資訊，但是實際上不用真正把它顯露出來。透過電話來擲硬幣是一個比較簡單的比喻性質的問題，不過一開始看起來也像是不可能的任務。

當然，在真實的情況下，Alicia 與 Benito 不會真正自己設計一個電路，而是會找個電腦程式來跑。實際上兩個人大概也都不太會對電腦程式裡怎麼運作感興趣。但是兩個人肯定都要確保對方不會設法影響最後的決定，不管對方是否有那種技術能力或時間。

原則上，任何的爭議都應該透過中立的仲裁者來解決。這個仲裁者拿到電路，還有 Alicia 的原始二進位值，她傳送給 Benito 的輸出值，還有 Benito 所送回去的猜測結果。當這些資訊交換完畢，通通都會變成是公開資訊，所以參與雙方都必須同意這最終的結果。仲裁者要能將 Alicia 的原始輸入放進電路中，並檢查輸出是否如 Alicia 所說，最後決定這個結果是否正確無誤。不用說，會有一個清楚明白的程序來確認大家都照著規則走，不致發生爭議。如果是用真正的硬幣，讓 Alicia 來丟然後 Benito 隔著電話來猜，沒有任何一個仲裁者會同意接受！

這個活動中使用的電路很小，現實上不太實用，因為很容易列出所有的輸入與輸出結果，要作弊就變得很簡單。一個方法是把輸入變成 32 個位元，這樣可靠度就高了一點。不過即使是這樣，依然不保證無法作弊 -- 因為整個結果是由特定電路所產生。另一種方法是用活動 15「旅遊小鎮」所介紹的單向函數。

一般會使用的方法是依靠找出一個很大的數字的因數。這個是一個已知的困難問題（雖然在下一個活動的結尾，我們會知道它並不是一個 NP 完全問題）。

我們很容易可以確定某個數是不是另一個數的因數，但是如果要把一個很大的數字做因數分解有時是很花時間的。

數位簽章就是基於類似的概念。Alicia 將她的輸入值放進電路中計算出輸出；而輸出值事實上也進一步證實了輸入該值的人的確就是 Alicia -- 因為如果這個「電路」是適當的單向函數，那麼就沒有人能找出第二組輸入值會得到相同的輸出。換句話說，因為除了 Alicia 手上那一組值之外不會有第二個可能的輸入值，那就沒有人可以假冒 Alicia！當然，要做出一個數位簽章，通常需要更為複雜的協定，確保 Alicia 可以「簽署」某個訊息，而且別人可以透過相同的機制確認這個訊息就是由 Alicia 簽署的 -- Alicia 想賴也賴不掉。雖然數位簽章的協定複雜很多，但原則跟上面所說的「電路」是一樣的。

另一個應用是透過電話玩撲克牌 -- 沒有裁判可以來發牌與紀錄雙方手上的牌。每件事都要靠玩家自己來，只在遊戲結束有爭端的時候靠仲裁者來仲裁。拿實際上的例子，就好比商業合同的談判。明顯地，在遊戲進行中，玩家要保密自己的牌。但是他們必須誠實，不能偷雞假裝宣稱自己有 A！各家手上的牌可以在遊戲結束時攤開來看，並且確認大家出牌的手順。但是怎麼洗牌同時讓各家的牌能夠保密不被偷看就是問題了。令人驚訝的是，洗牌這件事其實可以用一個跟擲硬幣一樣的加密協定來完成。

加密協定在電子交易上是非常非常重要的，不管是確認記帳卡或信用卡持有人的身份，授權某個手機可以通話，或是確認某封電子郵件的發信人是否是本人等等。穩定可信賴的加密機制對電子商務的成功來說是最基本的要素。

延伸閱讀

Harel 的書 “Algorithmics” 討論了數位簽章與相關的加密協定。它也說明了如何透過電話來玩梭哈；這個想法最初是在 1981 年，在 D. A. Klarner 所編著的 “The Mathematical Gardener” 中，“Mental poker” 這一章所提出來的。Dorothy Denning 所著的 “Cryptography and data security” 是資訊科學領域中，關於密碼學的一本非常棒的教科書。Dewdney 的 “Turing Omnibus” 則有一個章節提到布林邏輯，並討論如何建立此活動中所用的電路。

活動 19
孩子的秘密 — 公開金鑰加密系統

活動摘要

加密技術對資訊安全來說是個關鍵。而現代的加密技術的關鍵，則是訊息的發送者只需要使用某些公開的資訊，就可以將一段訊息鎖住，只有特定的人才能解鎖打開。

這就如同每個人都買了一個掛鎖，在上面寫上他們的名字，然把它們全部放在相同的桌上供其他人使用。當然每個人自己都有自己鎖的鑰匙，但掛鎖是開著的，鎖上不需要任何鑰匙。如果你要寄給某人一則機密的訊息 ,你只需要把訊息放入一個盒子，拿鎖把它鎖起來，然後寄給收件者。即使盒子落入別人之手，因為沒有鑰匙，所以也無法打開。有了這樣的機制，大家就不需要事先溝通解鎖的密碼為何。

這個活動教大家如何在數位的世界中實現這個機制。不過在數位的世界中，我們不是直接拿起掛鎖鎖住盒子，而是去複製這個掛鎖，並使用複製後的掛鎖 ,把原始的掛鎖留在桌上。當然，如果我們要複製一個實際的掛鎖，我們得先把掛鎖解體，瞭解它如何運作才能複製。但是在數位的世界中，我們可以在不讓人知道鑰匙長什麼樣或掛鎖怎麼運作的情況下，直接複製掛鎖！

聽起來不可能？繼續讀下去。

課程銜接

• 技術：公開金鑰加密、密碼

習得技能

• 解決問題

適合年齡

• 11 歲以上

所需素材

把學生分成　4　組 , 每組再分成兩個小組。每個小組要分到一張「活動學習單：孩子的密碼地圖」（第 214 頁）。換句話說，每一組學生將需要：

• 兩張「活動學習單：孩子的密碼地圖」（第 214 頁）影本

你還需要：

• 一份「活動學習單：孩子的加密法」（第 215 頁）的（透明）投影片

• 一個在投影圖上做註記的方法

孩子的秘密

活動介紹

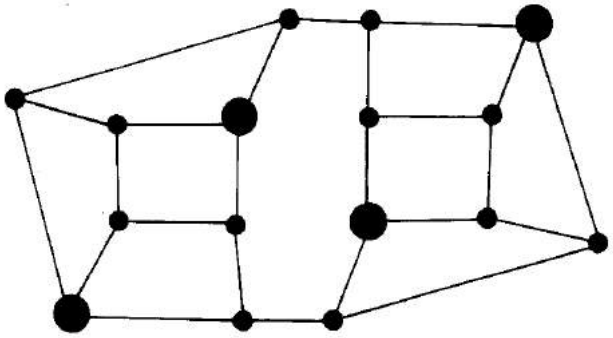
這個活動是這本書裡最具挑戰性的活動，需要十分細心與持續專注才能完成。學生們應該已經在活動 15 「旅遊小鎮」上學到何謂單向函數。如果也完成過活動 17 「傳遞機密」與活動 18 「秘魯式拋硬幣」，那這個活動應該就會比較容易上手。此外，這個活動也用了活動 1 「計算點點」與活動 5 「二十個問題」裡的概念。

Amy 正在計劃派 Bill 傳送一個秘密消息。通常情況下，我們可能會認為秘密消息是一個句子或一段話，但在接下來的練習中，Amy 只會發送一個字元 — 事實上 ,她只打算傳送一個代表字元的數字。雖然看起來只是個簡單的訊息，但是請記住，她可以發出一連串這樣的秘密訊息，最後組成一個句子。現實中這樣的工作會由電腦來完成，而有時即使是很小的訊息也是很重要的。在歷史上最有名的訊息之一 — 由 Paul Revere 所傳送 — 訊息裡傳達的只有兩個可能裡的其中一種 [1]。我們將看到如何使用 Bill 的公開金鑰來加密 Amy 的訊息。這樣即使有人攔截，也沒辦法對其進行解碼；只有 Bill 可以解碼，因為只有他鑰匙。

我們將利用地圖來加密訊息。這裡的地圖不是指金銀島的藏寶地圖，而是像旅遊小鎮（活動 15）中的那些街道地圖：線條是指街道，而點則是指街角。每個地圖都有一個公開版本 — 就像上面例子中的掛鎖，以及一個私密版本 — 就像是例子中的鑰匙。

活動討論

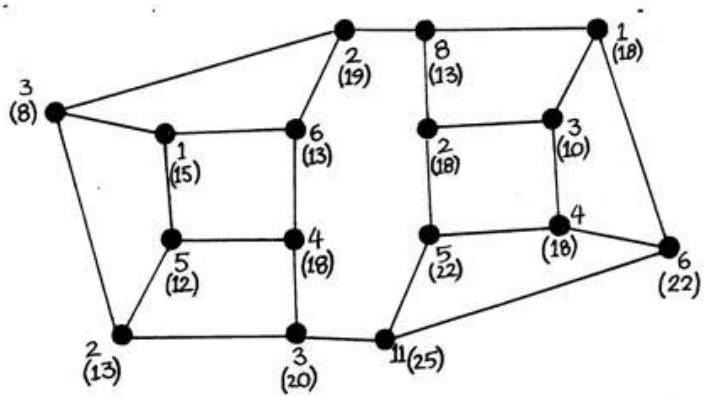
在「學習活動單：孩子的加密法」裡的是 Bill 的公開地圖。這張圖不是秘密，Bill 可以放心把它放在桌上（或網頁上）給大家看，或提供給可能要發信給他的任何人。Amy 有這張圖的影本，其他人也有。而右邊的圖則是 Bill的私密地圖。這張圖與他的公開地圖相同，只是把一些點（街角）放大。這個版本的地圖只有他有，要保密不能讓別人拿到。



這個活動是最好作為一項課程，至少在開始時，因為它需要花費相當大的功夫。雖然並不算困難，但是一定要精確地進行，任何錯誤都會造成很多麻煩。因此，讓學生瞭解這種聽起來不可能的加密方法完全是可行的非常重要，因為他們需要這種動機來激發過程中所需的努力。

我們發現有一件事情可以高度地激發學生的興趣。告訴他們，使用這種方法，他們可以在課堂上傳遞加過密的紙條。即使被老師沒收，甚至老師們都知道這個訊息是怎麼被加密的，但他們還是無法解密。

1. 把 Bill 的公開地圖顯示出來（學習活動單：孩子的加密法）。決定 Amy 要傳送哪個數字。現在，在地圖上的每個節點旁邊寫一個隨機想到的數字，但重點是所有的數字加起來要剛好等於 Amy 打算傳送的數字。我們來看看這張圖。假設 Amy 打算傳送的數字是 66，這張圖每個節點旁邊，沒有用括號括起來的數字加起來就剛好是 66。如果需要（而且學生會）的話，可以用負數沒關係。



2. 現在 Amy 必須計算發送給 Bill 的數字。如果她把地圖上沒括號的數字直接送出去給 Bill，那沒有任何意義，因為落入別人的手上時，任何人都可以把數字加起來就解開了。

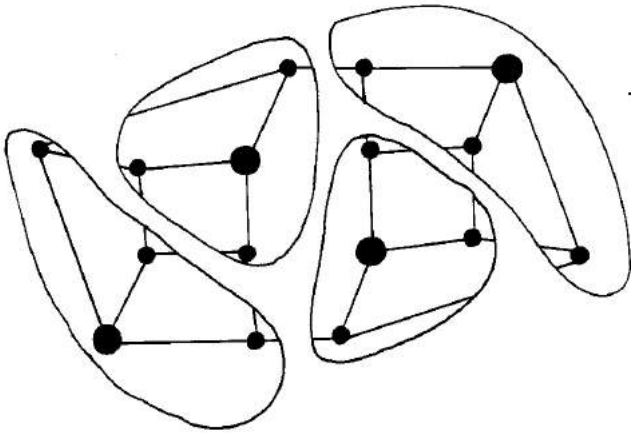
因此，我們換一個方式。選擇任何一個交點，看它和它的三個鄰居，然後把四個交點的數字都加起來。把算出來的總和寫在交點的括號中，或是使用不同顏色的筆來寫。例如，圖中最右邊的那個點，它本身數字是 6，而連接它的三個點分別是 1、4、11，四個點加起來是 22，所以就把 22 寫在括弧內。對所有在地圖中的其它交點重複此動作。

3. Amy 發送給 Bill 的地圖，只需要含括號內的數字即可。把原來寫的數字擦掉，只留下要發送的號碼，或者在西張新的地圖上重新寫上括弧內的數字。讓學生看看是否有任何方法，從這張地圖中找出原始訊息。他們會發現很困難。

4. 只有知道 Bill 的私密金鑰的人可以解出 Amy 要發送的訊息。在編過碼的地圖中，標記出在 Bill 私密地圖裡被放大的點。

要解碼訊息，Bill 只需要看剛剛標記起來的交點，並將這些點的數字加起來。在這個例子中，這些節點的數字分別為 13，13，22，18，加起來剛好是 66，也就是 Amy 的原始訊息。

5. 好玩吧！這是怎麼一回事呢？實際上地圖是特別設計過的。仔細看私密地圖上被標記的點。把這些標記點與它的三個相鄰節點圈成一組，這樣會剛好把地圖分成四個不重疊的片段，如圖所示。每個片段中被標記的那個點的值，還記得怎麼算嗎？對，就是標記點與三個相鄰的點的和。換句話說，每個標記點就是那個片段中所有節點最原始的值的和。所以四個標記點相加，剛好就會是 Amy 打算送出來的原始數字。



呼！發送一個字母似乎還滿麻煩的。沒錯，加密本來就不是一件容易的事。不過看看我們的成果：使用公開金鑰，不需要傳送者或發送者之間事先講好任何隱密的事情，就可達到傳訊完全保密的情況。你可以把你的公開金鑰放在一個佈告欄，任何人都可以給你一個秘密消息，但沒有人可以解密，因為他們沒有你的私密金鑰。而在現實生活中，所有的計算都是由電腦程式來完成（通常這些計算用的程式是內建在瀏覽器裡）。所以交給電腦就好了。

完成這個活動以後，你可以讓學生們知道，他們已經成為一個非常特別的團體中的一員 — 團體中的每個人都曾經自己用手用腦計算公開金鑰加密的訊息！這是連電腦科學家都認為幾乎不可能完成的任務，而真的很少有人這樣做過！

那麼，這個方法容易破解嗎？Bill 手上的私密地圖就像是在旅遊小鎮 （活動 15）中的一樣，其中標記放大的交點就是放置冰淇淋車為所有街道上的人服務的最佳解 — 沒有任何人需要走超過一個街區。我們在旅遊小鎮活動中看到，Bill 要建立自己的私密地圖很簡單：只要從標記放大的點開始往外長就可以。但是要反過來找出最佳解，找到放置最少冰淇淋車的地點就很困難。目前除了透過暴力演算法以外沒有其他更好的方法。暴力法是試著配置一台冰淇淋車，試試每種可能性；然後配置兩台冰淇淋車，看看每種可能性 ... 依此類推，直到找到一個最佳解決方案。沒有人知道有沒有一個較好的演算法，可以適用於任何地圖。很多人都已經試過了！

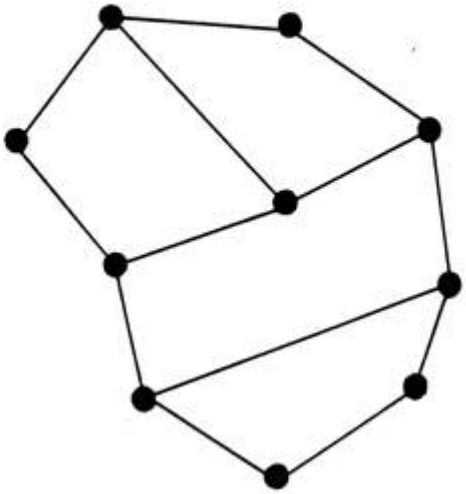
若是 Bill 一開始就用一個夠複雜的地圖（比方說，50 或 100 個交點），似乎就沒有人能夠破解密碼，即使是最聰明的數學家們努力嘗試也還是失敗。（但是有一個例外警告：請看後面的「這個活動在說什麼？」）

6. 向全班展示完這個例子之後，把學生分成四個一組。給每組裡再分成兩對，每一對學生學習活動單：孩子的密碼地圖上的公開地圖。每對要選擇一個「訊息」（任意整數），與公開金鑰對其進行編碼後，把得到的結果傳送給另一組。另一組學生可以試著進行解碼，看看有沒有辦法成功。然後給他們（或讓他們畫出）私密地圖，看看他們是否可以用私密地圖正確地解碼。

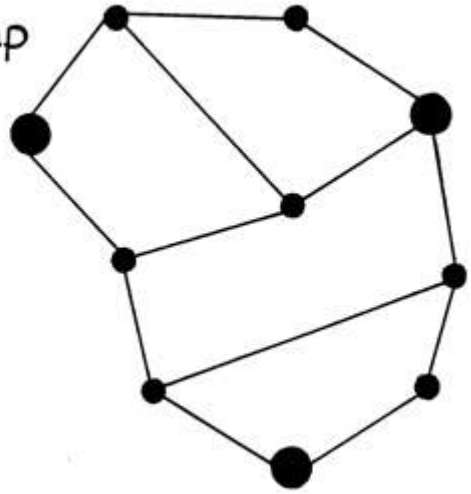
7. 現在，每一對學生可以設計自己的地圖。把私密版地圖保護起來，把公開地圖發布給另一對，或發布在教室的公布欄上。設計地圖的原理與我們在旅遊小鎮活動中所討論的相同，也可以添加額外的街道（線條）來增加複雜性。只是要注意：別把額外的街道連到任何的「特殊標記」節點。因為這樣會出現一個路口（節點）同時相鄰於二台冰淇淋車（特殊標記節點）。在旅遊小鎮問題中這倒還無所謂，但在加密時會造成嚴重的問題。因為特殊標記點就無法再分解圖成如上面私密地圖顯示的「不重疊的區塊」。這一點非常重要。

活動學習單：孩子的密碼地圖

public Map



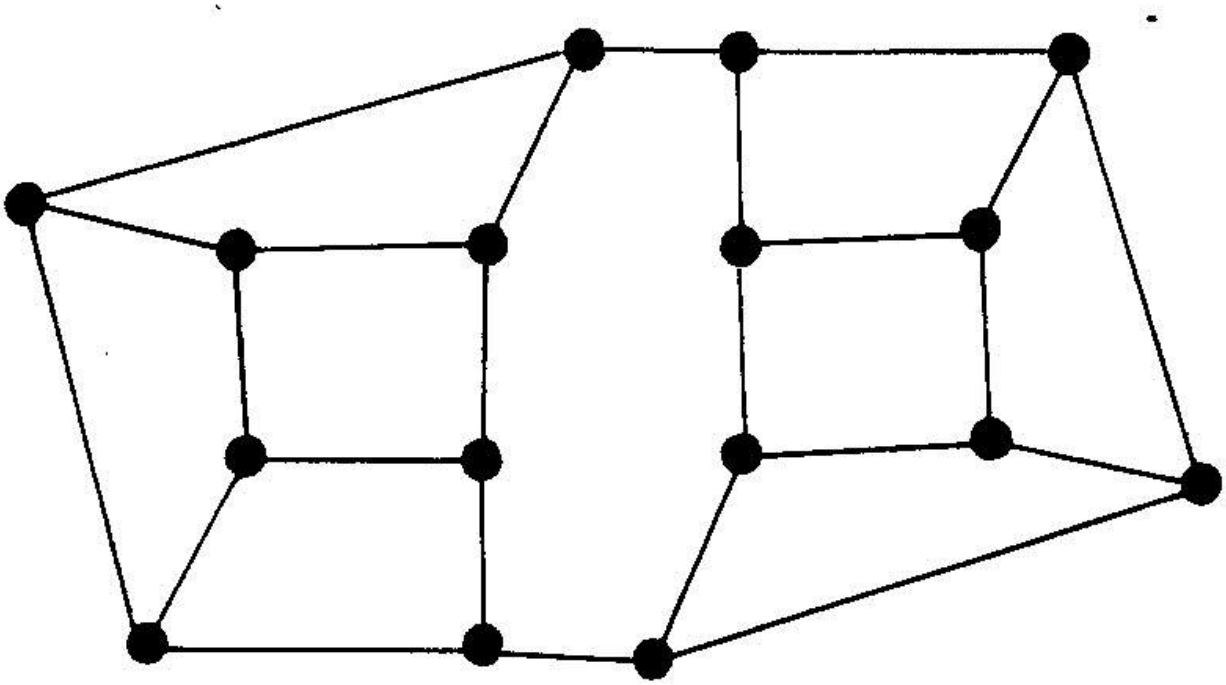
private Map



用這些地圖 , 照書上的方法加密與解密地圖。

學習活動單 :孩子的加密法

把這個「地圖」展示給全班看 ,並用它來示範把一個訊息加密的方法。



這個活動在說什麼？

我們能透過電腦網路傳送機密訊息，而且除了收件人外沒有人可以進行解碼，這是很重要的技術。當然，除了這個活動中所說的方法外，也有其他方式可以做到；例如發送者和收件者有一個共用的，只有這兩人知道的密碼。但是，公開金鑰加密技術裡最巧妙的地方是：Amy 可以不必先與 Bill 事先講好任何秘密方式，只要從網頁上拿到 Bill 的公開金鑰即可發送秘密訊息。

保密是密碼學的一個面向。另一個面向是驗證：當 Amy 收到 Bill 的訊息，她怎麼知道真的是 Bill 傳來的，而不是某些人冒名頂替？假設她收到的電子郵件上面寫著：「親愛的，我被困在這裡，身上沒有錢。請匯 10000 元到我的銀行帳號，帳號是 0241-45-784329。愛你，Bill」她怎麼知道這封信真的來自 Bill？一些公開金鑰密碼系統即可用於此需求。就像 Amy 用 Bill 的公開金鑰來加密自己的訊息，反過來 Bill 也可以發送出只有自己才能產生的訊息 — 也就是用自己的私密金鑰來加密訊息。如果 Amy 可以用 Bill 的公開金鑰來解碼，那保證這個訊息一定是來自 Bill。話說回來，任何人不都可以拿到 Bill 的公開金鑰嗎？沒錯。不過因為訊息只發送給 Amy，Bill 可以用 Amy 的公開金鑰做第二次編碼。這種雙重編碼的方式經由相同的基本方式，也就是公開金鑰與私密金鑰，可以同時提供保密和驗證識別。

現在，該是談談另一個觀點的時候了。雖然在本次活動中所展示的方法，與工業強度的公開金鑰加密系統非常相似，但它其實不是一個真正安全的方式，即使用一個相當大的地圖也一樣。

原因是，雖然目前沒有已知的方法在任意地圖上找出放置最少冰淇淋車的方式，而這個方法從這個角度來看的確是安全的；但是恰好有一個完全不同的方法可以攻擊這個系統。這個方法學校的學生不太能瞭解，至少在高中以前；但你至少應該知道有這樣的方法的存在。我們可以說，這個活動介紹的方法對學校學生來說是安全的，但對數學家則不然。如果你不是學數學的，那下一段不看沒關係。

把地圖上每個節點編號為 1、2、3、...。把每個節點原本的數字用 b_1、b_2、b_3 、... 來表示。每個節點實際傳送出去的數字用 t_1、t_2、t_3、... 來表示。假設節點 1 與節點 2, 3, 4 相鄰。因此該節點傳送出去的數字是：

$$t_1 = b_1 + b_2 + b_3 + b_4$$

當然，每一個節點都有類似的方程式。事實上，方程式的數量和未知的 b_1, b_2, b_3數量相同。竊聽者如果知道公共地圖和傳送出去的數字 t_1, t_2, t_3方程式，可以用一個聯立方程式求解的程式來計算。一旦算出了原來的數字，他們的總和，也就是原始的訊息也就被算出來了。因此沒有私密地圖也可以解碼。直接使用高斯消去法，解決方程式所需的計算工作量和方程式的數量成立方比；但由於這些方程式是稀疏方程式（sparse equations，也就是大部分的係數是零），因此還有更有效率的方法。跟成指數的計算工作量相比，用私密地圖來解碼還是最好的方式。

我們希望你不會覺得被騙了！事實上，真正公開金鑰密碼系統的處理跟我們在這活動中所學到的幾乎一樣。不同的地方是，他們使用的編碼技術是筆算算不出來的。最原始的公開金鑰的方法 — 目前也仍是最安全的方法 — 是基於對大數做因數分解的難度。

舉個例子：下面這個 100 位數的數字的因數有哪些？

9412343607359262946971172136294514357528981378983082541347532211942640121301590698634089611468911681

別浪費時間算了（我賭你也不會真的去算）！答案是：
86759222313428390812218077095850708048977
跟
10848810485363747061296139984297294840983461152579057721675 3

沒有其他的因數了，因為這兩個數字都是質數。找到這兩個質因數是一個相當大的工程：實際上，即使用超級電腦也要算上幾個月。

現在，在一個真正的公開金鑰加密系統，Bill 可以使用這個 100 位數的數字當他的公開金鑰，而那兩個質因數當私密金鑰。要產生這樣的金鑰並不會很難：你需要的是算出一個非常大的質數的一種方式。找到兩個夠大的質數（這不會很難做到），乘在一起，賓果！這就是你的公開金鑰。對於電腦來說，把兩個巨大的數字乘在一起並沒有什麼大不了的。把公開金鑰公布出來，沒有人會知道你的私密金鑰，除非他們有機會用超級電腦好幾個月。如果你擔心他們真的有超級電腦，那就改用 200 位的質數，他們就得多花上好幾年的時間！最主要的是，破解私密金鑰所需的成本會遠高於它要解鎖的訊息的價值。在實際應用中，建立安全通訊頻道通常會用 512 位元或更大的金鑰，相當於約 155 位數以上的數字。

其實基於質數的公開金鑰系統，目前還不保證在沒有私密金鑰的情況下就一定不能進行解碼。但這一點也並不像上述我們說的那麼簡單。目前我們並不是用兩個質數作私密金鑰，然後把它們的乘積做為公開金鑰。我們用的數字是從它們衍生而來。但效果是一樣的：你還是可以透過分解因數來破解私密金鑰。無論如何，要讓這個方法成為一個適當的加密和解密演算法，這些問題其實也不難克服。但我們不打算在這裡再細說了。這個活動已經夠困難了！

基於質數的系統有多安全？大數的因數分解問題在幾個世紀以來，已經有許多世界上最偉大的數學家們投入，而已經發現的方法也比嘗試所有可能因數的暴力法要好很多，但還沒有人找出了一個夠快（即多項式時間）的演算法。當然，也還沒有人證明這樣的演算法是不存在的。因此，其實也可以說，這個方法不只是對學校的學生來說夠安全，對數學家也一樣。但我們還是要小心！正如有一種方式不需透過解決旅遊小鎮問題就可以找出 Bill 的訊息，實際上也可能的確有不需把大數做因數分解的方法來破解。當然，許多人已糾仔細檢查過這一點，目前看來結果還算好。

另一個可能發送端只有少數幾個可能的訊息，竊聽者可以利用公開金鑰把所有訊息進行加密，然後比較他們所得到的傳送端發出來的加密訊息，就可以知道實際上發送出來的是哪一條訊息。Amy 的方法避免了這個問題，因為同樣的訊息（數字）可以有很多種方式來發送，只要每個節點數字調整一下即可。在實際狀況中，加密系統也會被設計成有很多很多可能的方式，多到即使用非常快的電腦也沒辦法把原始訊息試出來。

目前還不知道是否有的方法來解決質因數分解的問題；目前也還沒有人設計出來。但是，它也也尚未被證明是不可能的。如果真的找到了一個快速的演算法來解決這個問題，那就麻煩了 — 很多目前使用的加密系統就會變得不安全了。在第四部分，我們討論了 NP 完全問題，他們「要嘛就全部有解，要嘛就全部無解」：如果其中一個問題找到了有效率的演算法來解決，那麼其他的問題必然都可以找到夠快的解法。因為對這些問題找出快速的演算法已經花了那麼久的時間卻仍然沒有成功，NP 完全問題似乎是可用來設計安全密碼系統的好選擇。不過嘛，這樣的計劃有它的困難之處，因此到目前為止，密碼系統的設計者還是必須靠實際上可能比 NP 完全問題容易一點（也可能容易得多）的問題（例如質因數分解）。以上這些問題的答案都價值數百萬甚至數千萬美元，而且對國家安全也是至關重要。密碼學現在在資訊科學中已經是一門非常活躍的研究範圍了。

延伸閱讀

Harel 的 “Algorithmics” 討論了公開金鑰加密；它說明了如何使用大質數來建立一個安全的公開金鑰系統。在資訊科學領域中，密碼學的標準教科書是 Dorothy Denning 寫的 “Cryptography and data security”，不過 Bruce Schneier 的 “Applied cryptography” 這本書更偏向實務應用面。Dewdney 的 “Turing Omnibus” 介紹了另一個用公開金鑰加密的系統。

第六部份
使用者介面 — 與電腦互動

使用者介面

為什麼電腦如此的難以操作？很多人都有過類似的經驗：電腦似乎永遠不肯做你想要它們做的事，反而一直產生一些愚蠢的錯誤。這東西似乎是設計給天才使用的，而不是一般人。但電腦是用來幫助我們工作、學習、娛樂的工具，照理說應該要讓一般人容易使用才對。

「使用者介面」（user interface）就是電腦系統中用來與使用者互動的部份。這是整個系統中最重要的部份！或許你會覺得程式才是真正在做事的部分，而使用者介面不過是讓人能夠操作程式而已。但是一個程式要是無法與使用者良好的互動，並依照使用者的意思運作，那再強的程式也不過是個不能用的廢物而已。使用者介面是非常難設計與製作的，它也是在寫程式時，最需要花費精力來完成的，遠超過任何其他部分。有些軟體有著非常棒的使用者介面，它們不需要複雜的指令，而且在執行軟體時幾乎感覺不到它們的存在。但也有數不盡的好軟體，因為有著奇怪的使用者介面，而成為徹底的失敗作。現今整個產業都圍繞在精巧的介面設計 — 像是文字處理器或是智慧型手機 — 來運作一些很基礎的運算功能。

但是，到底為什麼我們必須要有使用者介面？為什麼我們不能像跟朋友講話一樣直接告訴電腦我們想要什麼？這是個好問題。也許在將來的某一天能夠實現；也許永遠沒辦法。但可以確定的是現在還做不到：「智慧型」電腦在現今還有很大的應用限制。接下來的活動會幫助你了解有關使用者介面設計的問題，以及認識更多電腦的限制，還有小心那些過於誇大的電子產品廣告。

給老師的話

電腦運算中，重要的其實是溝通，而不是計算。電腦運算本身並沒有任何價值，只有在產生的結果能夠與外界溝通，並造成某種程度的影響時，它才真正有價值。也許你會覺得很驚訝，這代表資訊科學這門學問的重點其實是人，而不是電腦。畢竟除非能夠對人們作出貢獻，不然電腦一點用也沒有。所有我們為了能讓電腦能夠執行得更快更有效率而付出的努力，都是為了能讓人們能夠更輕鬆更經濟的去使用它們。

使用者介面是電腦和人類的溝通方式。而很多在這本書的活動都是有關於溝通。「如何表達資訊」（第一部分）顯示出不同類型的資訊可以傳送到電腦或在電腦之間傳送。「如何表達程序」（第三部分）是關於如何傳送程序給電腦，告訴它如何完成某些任務。畢竟，「寫程式」就真的只是用電腦的語言解釋給電腦聽。密碼學（第五部分）則是關於如何秘密的通訊，或者在不透露全部的內容下做部分的秘密通訊。

接下來的活動是關於人與電腦的溝通。雖然本書的其他部分都是建立已有的技術知識基礎上，但這部分沒有。這有好有壞：對於沒有專業知識的學生來說較為簡單，但如果沒有一定的心智成熟度的話，會比較難以理解這些活動在做什麼，以及它們的來龍去脈。這些活動中有比其他活動還要多的詳細解釋，這是為了給老師們足夠的背景資料，以便在課堂討論中給班上同學們一些提示。

這個部份有兩個活動。 第一個是關於「人機介面」（Human-computer Interface，通常簡稱為 HCI）。因為這本書是「不插電」的，因此我們發明了一個與電腦無關的設計練習，重點在介紹設計使用者介面時需要的一些基礎原理。因為人機介面的設計與文化相關，因此這個活動並不需要「正確」答案。這一點也許有些學生會不太適應。第二個活動是關於「人工智慧」（Artificial Intelligence，簡稱為 AI）。利用猜謎遊戲去刺激學生們思考有哪些事是電腦可以做的，而哪些事是電腦做不到的。

給技術人的話

自從人們發現許多軟體都是靠著設計優良的使用者介面而成功後，人機互動已經成為最熱門的研究領域。這個主題大量運用了資訊科學以外的學問， 像是心理學、認知科學、語言學、社會學，以及人類學等等。很少電腦科學家接受過相關領域的訓練，然而對於那些喜歡這個主題中較「軟」的這一方面的人來說，HCI 代表一個重要且成長中的領域。

人工智慧是個時常挑起爭論而且令人寒毛直豎的主題。在這本書中，我們嘗試在認為有智慧的機器即將出現的 AI 愛好者，以及相信機器不可能擁有智慧的 AI 懷疑論者中保持中立。我們的目標是鼓勵學生獨立思考相關的問題，並促進觀點的平衡。

這裡的活動借用了兩本知名的著作：Don Norman 的 "The design of everyday things" 以及 John Haugeland 的 "Artificial intelligence: the very idea"，推薦給對相關問題有興趣的人。

電腦包含了另一種重要的溝通，在這本書中並沒有提到：建立電腦系統的人之間的溝通。學電腦的學生（比方說從大學的資訊科系畢業的學生）在進入職場後，都不約而同地對工作上需要大量跟人溝通感到驚訝。電腦是人類有史以來創造出的最複雜的東西，有著數百萬甚至上億個複雜連鎖的零件，程式專案則是由組織嚴密的團隊花費大量的時間溝通來共同完成的。一旦產品完成，接下來就是與客戶之間的溝通，透過使用者手冊，學習課程，客服電話，線上支援等等，更別說還要透過示範、展示、廣告等與潛在的顧客溝通。我們還沒有找到如何用「不插電」的方法寫實地呈現開發者之間的溝通，所以這本書並沒有特別提到。但這也是一種做為資訊專家與技術人的你，在課堂上能分享自己的經驗，並加以討論的主題。

活動 20
巧克力工廠 — 人性化介面設計

活動摘要

這項活動的目的主要是為了提高人性化介面設計的意識。因為我們生活在一個設計不良盛行的世界裡；我們已經習慣於（聽天由命？）忍受缺陷。雖然這些實際上是物品設計的問題，但我們不斷把問題歸咎於自己（「人為錯誤」、「訓練不足」、「它對我來說太過複雜」）。電腦的出現大大地提升了這樣的問題，因為它們沒有明確的目的 — 事實上，它們本來就是一般性的用途 — 而它們的外觀也並沒有提供它們是為了什麼而設計，還有如何去操作的線索。

課程銜接

• 技術：了解科技是透過設計而有目的的解決人們的問題。

習得技能

• 設計
• 推理
• 認識日常物品

適合年齡

• 7 歲以上

所需素材

每組學生需要：

• 一份「活動學習單：要怎麼開門？」以及「活動學習單：火爐的設計」的影本
• 一份「活動學習單：圖示」，可以用投影機或幻燈片顯示，或放在卡片上給全班看
• 「活動學習單：圖示卡」中六張卡片的其中一張（或一張以上也可以）。把這些裁剪成個別的卡片，並將它們分發到各組。

巧克力工廠

活動介紹

大型巧克力工廠由是由一個類似精靈的種族叫 Oompa-Loompas[3] 在運作。這些小精靈的記性很差，而沒有任何文字。正因為如此，他們很難記住要怎樣做才能運行巧克力工廠，並且事情經常出錯。也正因為如此，正打算設計新工廠的我們，要考慮把工廠設計成對小精靈們來說很容易運作。

活動討論

1. 把學生分成小組，並跟他們解釋這些故事。

2. 這些小精靈面對的第一個問題是如何穿過正在進行蒸餾巧克力的門。他們不記得門是要推還是拉，還是滑動到一邊來開門。因此，他們總是會相撞，並把濃稠的巧克力灑落到各個地方。學生們應該在「活動學習單：要怎麼開門？」裡填上如何開門。每一道門都可能不止一個框框要勾。對於如何打開某些門（包括第一個），並沒有寫得很清楚。在這種情況下，學生們就應該記錄他們打算怎麼嘗試。一旦他們填寫完自己的活動單，整個小組就應該去討論每種門的相對優點，尤其是在你背著一桶熱巧克力的情況下，該如何輕鬆的開門，如何使用門是合適的。這樣他們就能決定什麼樣的門適合在廠中使用了。

3. 在這活動之後用一堂課的時間來討論。下表簡短地點評了活動單中的每一道門。實際生活中的門可以經由門框與鉸鏈的位置，來找出如何開門。門上有時也會有「推」或「拉」的指示，讓你知道門是向內還是向外開啟（可是好像還是有不少人分不清楚！）找出學校裡使用的各種門把，並且討論它們的合宜性（它們可能是相當不合適的喔！）你能不能想到一扇往往令你感到困惑的門？為什麼？門是向裡開還是向外開呢？又是為什麼要這樣設計？（答案：儘管在有些緊急情況下，人們向外開門會使疏散更容易，但一般情況下人們還是會打開門進入房間的。所以，從房裡走出來的時候，通常門是不會往外開的，以免撞上正在走路的人。）

4. 這裡的關鍵概念是一個物品的「示能性」（Affordance），也就是它明顯可見的特徵 — 包括基本和感知特性 — 直接可以指示此物品該如何使用。「示能性」是一種物品「允許」或「提供」的動作。例如很明顯的，椅子一看就知道是為了讓人坐下來，桌子是為了可以放東西，旋鈕是為了可以旋轉，插槽是為了可以插入或放入東西，按鈕是為了可以按下。在電腦的介面上，常見的就是按鈕、文字輸入框、選單等檔。這些物件的外觀設計都可以給使用者一個它們該如何被使用的線索。如果按鈕被繪製成看起來像其他東西，那麼人們就不會覺得他們可以去按它。這似乎是顯而易見的，但是在數位設備上並不難找到很糟糕的設計。

[3] 在這裡要跟 Roald Dahl 說聲抱歉。如果各位讀過他所寫的故事 "Charlie and the Chocolate Factory" 就知道 Oompa-Loompas 是從他那邊借來的。讀者或學生們如果沒讀過這篇故事也沒關係，故事情節與這個活動並沒有關聯。（譯註：這本書台灣的小天下出版社有發行，書名「巧克力冒險工廠」。）

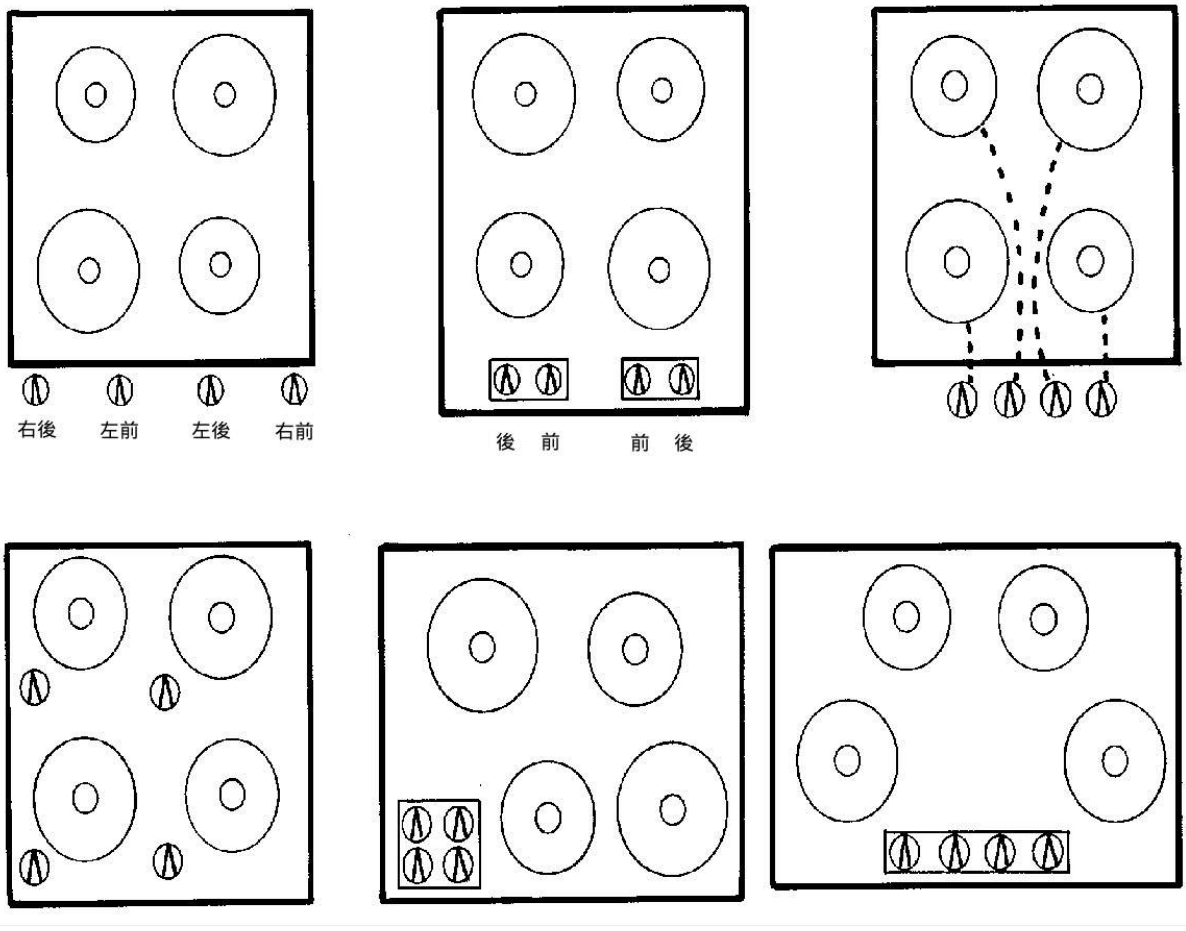
普通門： 根本看不出怎麼開門。是說因為它沒有握把，看起來比較像是要用推而不是用拉的。	有標籤的門： 標籤就像一個小小的用戶手冊。但是一扇門是否真的需要用戶手冊呢？而且這些小精靈是不會閱讀的。
鉸鏈門： 至少你可以看得出門是從哪一邊打開。	橫桿門： 看起來像是要推橫桿來開門。但是該在哪一側推呢？還是根本就該用拉的？
有把手的門： 像這樣的把手通常都是用拉的，或是用滑動的。	旋鈕門： 旋鈕顯示就是轉它去開門。但是看不出來要推還是要拉，只知道應該不是向兩側滑動的。
面板門： 很明顯的只能推。除此之外，你還能做什麼？	玻璃門： 小型直立式門桿，在門的這一側寫著「拉」，另一側一個水平的門桿上寫著「推」。
滑動門： 這門只能滑動。	

門是種非常簡單的物體。複雜的東西可能需要解釋，但簡單的東西就不需要了。簡單的物體如果需要圖片、標籤或說明，那它的設計已然宣告失敗。

5. 壺中含有不同種類的巧克力，需要在不同的溫度下煮。老舊巧克力工廠的火爐，就如在「活動學習單：火爐的設計」那樣。左側旋鈕控制左後方的火爐，下一個旋鈕控制左前方的火爐，再下一個控制右前方，右側旋鈕則控制著右後方的火爐。那些小精靈總是會搞錯控制鈕，讓煮巧克力的溫度都不對；此外小精靈們在去轉旋鈕時，也常會燒到自己的袖子。

6. 讓學生們回想一下家裡的瓦斯爐是怎麼設計的。這樣他們就能為新工廠想出一個更好的設計方案了。

接下來用一個課堂的時間來討論這次活動。下面這張圖顯示了一些常見的設計安排。除了左下方的爐子，其他的爐子都把控制旋鈕設計在前面，以避免手要伸過火爐上方。在左上方的設計裡，控制旋鈕與火爐的對應有 24 種可能性之多，以致需要用八個字來標記。中間上方的爐子把對應減少到四種可能（兩個用於左邊，兩個用於右邊），所以只需要四個字來標記。右上方的設計表明了控制旋鈕與火爐之間的關係圖，當然不是用語言來表示的（這對小精靈來說是很好的！）在下方的三種設計不需要標籤。左下方的設計中，控制旋鈕放在每個火爐的旁邊，這樣做安全嗎？一點也不。其它兩個設計出於不同原因需要略微重新擺放火爐：在中間的設計，為了便於控制，將火爐往右移動一點，預留了空間給控制旋鈕。而在右邊的設計，則很清楚地看得出來哪個旋鈕對應到哪個火爐。



這裡的關鍵概念是在「控制元件」與其「結果」的對應關係。「自然對應」利用了物理類推和文化標準的優勢，因此很容易理解。在圖片下方的三個設計，其空間對應關係就是個很好的例子 — 一看就知道，容易理解也容易記住。在上方的對應則不明顯，需要用標籤標記，也需要解釋並記憶。

7. 工廠裡到處是輸送帶，上面放著在不同的完成階段的巧克力半成品。小精靈們靠著中央控制室的指令來手動控制這些輸送帶。在中控室的人需要能告訴小精靈們停止傳送帶，減速，或重新啟動。

在舊工廠，這是由一個語音系統來完成：中控室裡傳來的聲音透過在輸送帶旁邊的揚聲器傳送出來。但是由於工廠太吵了，所以很難聽見。這一組工作團隊需要利用視覺信號機制來重新設計一個方案。

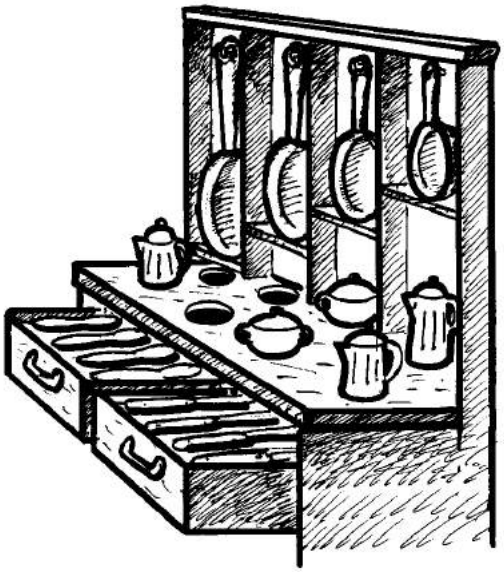
其中的一種可能性就是用燈號來表示「停止」、「減速」或「重新開始」。學生們可能會用交通號誌的方法，也就是用紅色作為停止信號，黃色作為減速信號，而用綠色作為重新開始信號。它們也都應該做得跟交通號誌一樣，紅色燈號在上面，綠色燈號在下面。

現在跟全班同學說，在小精靈的世界中，交通號誌的運作方式和我們是不一樣的：黃燈代表停止，紅燈代表開始，綠燈則是警告大家要減速停止了。這會對工廠的設計造成什麼影響呢？（答案：工廠應該要遵照小精靈們的慣用規則 — 我們不應該把我們自己的標準強加在小精靈們身上。）

這裡的關鍵概念是信號的轉換影響 — 人們通常習慣將以前所學習到的，以及對過去事物的預期，轉移到類似的新情境和新人群。但是，不同文化的人，學習行為與對事物運作方式的期望是不同的。雖然交通號誌的情境似乎有些過於牽強（其實在小精靈的世界裡，沒有什麼是牽強的），但是在我們的現實世界中有很多的例子：在美國，電燈的開關往上是開而往下是關，但是在英國卻剛好相反；計算機的數字鍵盤與按鍵式電話的數字鍵盤排列方式也不盡相同；在世界各地，數字格式（例如千位點是用小數點還是逗號）及日期格式（日／月／年或月／日／年）都是不一樣的。

8. 當小精靈們在巧克力工廠完成一輪工作之後，它們必須清理場地，並把鍋子、水壺、水瓶、湯匙和攪拌器等物品收起來，準備下次工作。有個有貨架的櫥櫃可以給小精靈們放物品，但是在下次輪班時，總是找不到東西放在哪裡。小精靈們的記性很差，要它們去記像是「鍋子是放在中間的架子上」、「水壺是放在左邊的架子上」這樣的規則，對它們來說都是很困難的。

這一組學生應該去想出一個更好的解決方案。



右邊的圖是一個不錯的設計參考（有時會這樣用，但跟這裡的原因不同：在遊艇之類的地方，要避免讓東西到處滑動）這裡的關鍵概念是使用一些可見的限制規格，讓小精靈們可以很容易從孔的形狀和大小看出東西該擺在哪裡：設計者把限制規格視覺化，利用物體的物理性質（形狀和大小），避免倚賴任意想出的規則。

9. 在巧克力工廠的中控室裡，有許多按鈕、槓桿和開關用來運作各個機器。這些都需要做標籤，但是因為那些小精靈們都不會閱讀，所以標籤上不能用文字，而必須是標誌性圖案。

為了讓學生們對圖示有感覺，「活動學習單：圖示」中舉了一些例子。學生們應該可以辨識出哪些圖示可能代表著什麼意義（比如說：信件進入信箱，可能代表了正在發送訊息。）這個練習是沒有「正確」答案的。想法就是簡單地找出圖示可能的含義。

10. 現在讓我們為巧克力工廠設計圖示。學習活動單的圖示卡片上指定了一些相關功能，每組學生會拿到一張（或數張）卡片，不要讓其他組學生知道卡片上寫的是什麼。現在來設計一個控制面板，裡面要有卡片上指定功能裡，5 個或 6 個操作的個別圖示。接著請設計控制面板的組員們，去向其他組學生展示，看看在不做任何提示的情況下，其他組學生是否能猜得出他們的圖示含義。鼓勵大家發揮想像力，設計出簡單而明瞭的圖示。

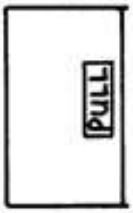
活動學習單：要怎麼開門？

在活動學習單上填寫你認為每種型態的門應該要怎麼開。

普通門

☐ 推　☐ 往左
☐ 拉　☐ 往右
☐ 滑動

有標籤的門

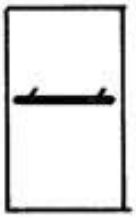


☐ 推　☐ 往左
☐ 拉　☐ 往右
☐ 滑動

鉸鍊門

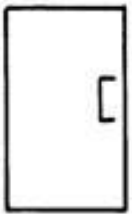
☐ 推　☐ 往左
☐ 拉　☐ 往右
☐ 滑動

橫桿門



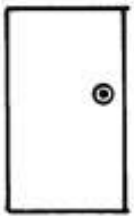
☐ 推　☐ 往左
☐ 拉　☐ 往右
☐ 滑動

有把手的門



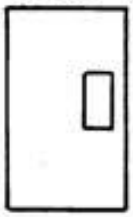
☐ 推　☐ 往左
☐ 拉　☐ 往右
☐ 滑動

旋鈕門



☐ 推　☐ 往左
☐ 拉　☐ 往右
☐ 滑動

面板門

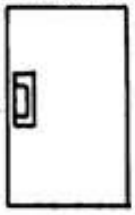


☐ 推　☐ 往左
☐ 拉　☐ 往右
☐ 滑動

玻璃門

☐ 推　☐ 往左
☐ 拉　☐ 往右
☐ 滑動

滑動門

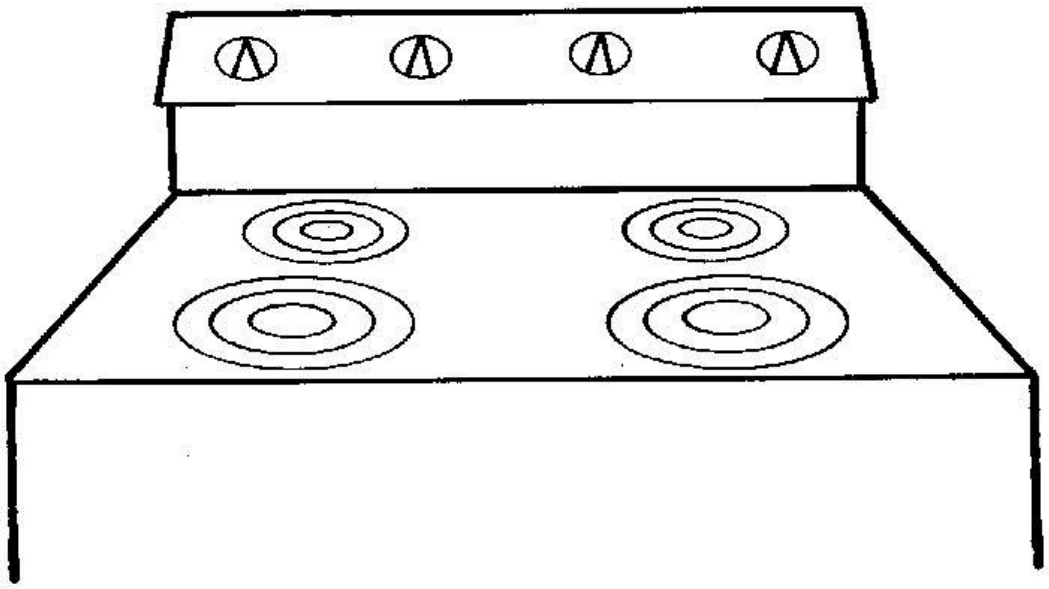


☐ 推　☐ 往左
☐ 拉　☐ 往右
☐ 滑動

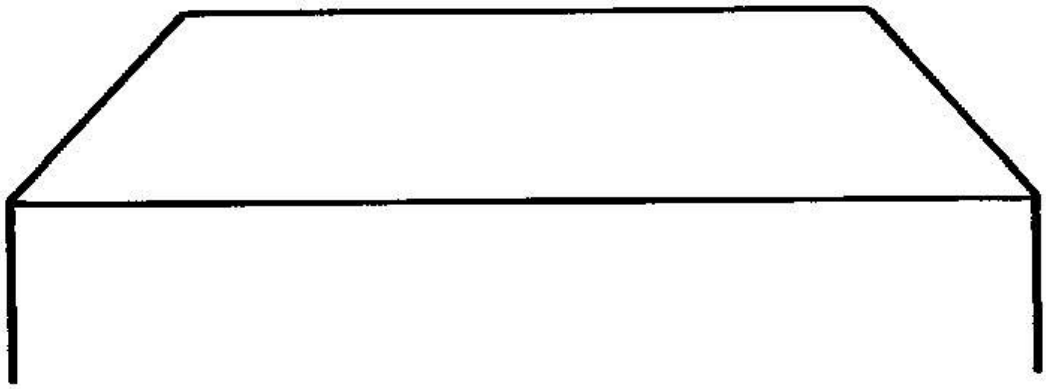
活動學習單：火爐的設計

重新設計火爐 , 讓它變得比較好用。如果需要的話可以加上前或後的控制面板。

舊爐子

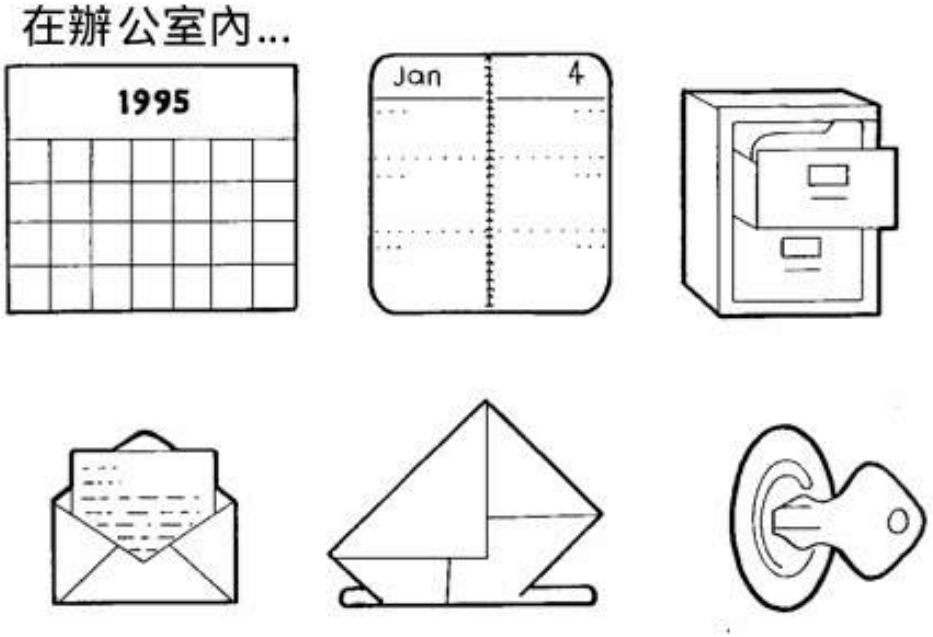


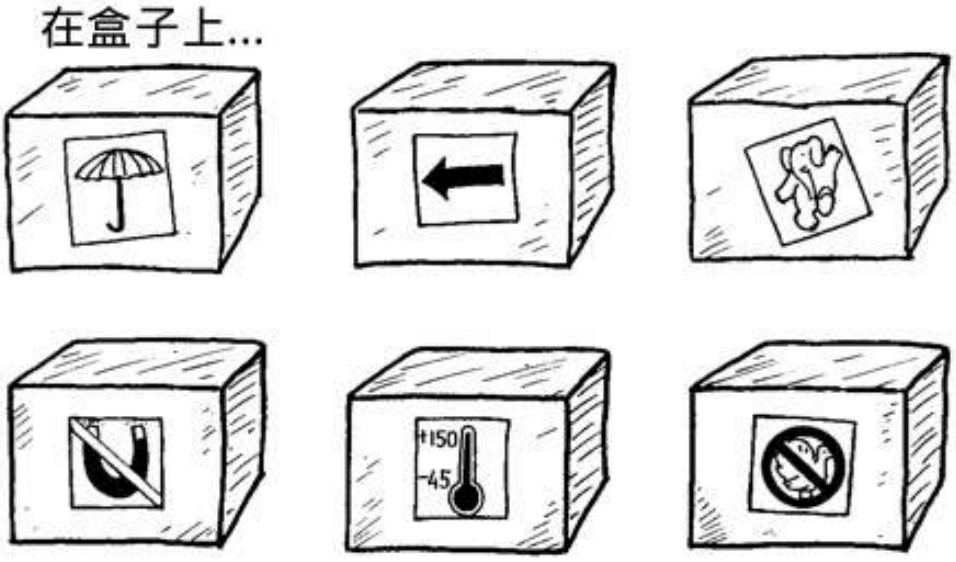
新爐子



活動學習單：圖示

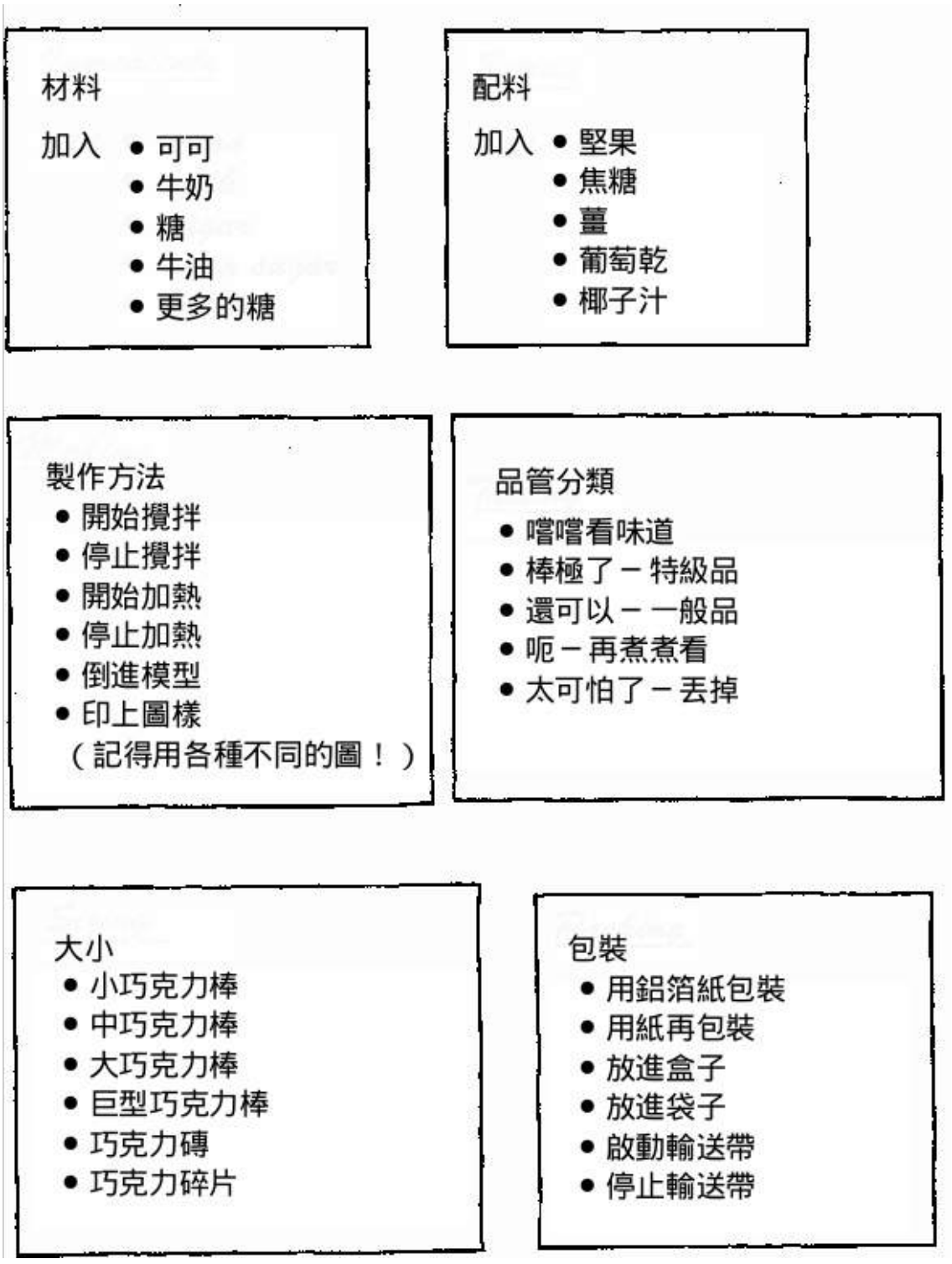
你認為每個圖示分別表示什麼意思？





活動學習單：圖示卡

剪下這些卡片，每組分一張或數張。讓每一組分別去設計要放在控制面板上的圖示，讓小精靈們可以清楚瞭解控制的方法。



活動變化與延伸

學生們會設定他們的電子錶或微波爐的時間嗎？通常廚具的設計對應都比較簡單，四個控制旋鈕就對應四個火爐。不過如果可執行的動作比控制元件多時就會變得比較困難。在電子錶或微波爐上尤其困難，不只是因為只有少數幾個按鍵或旋鈕，也是因為這些裝置會進入的狀態數量不少。（「你需要一位有麻省理工學院工程學位的人來讓這隻手錶正常運作，」有一次，某人看著他的新手錶跟使用者介面心理學家 Don Norman 說。Don Norman 有麻省理工學院工程學位，而他花了幾個小時之後搞定了手錶的設定。但是 — 為什麼搞定手錶設定得花上數小時呢？）

學生們平常就可以留意一些會讓人們感到困惑或沮喪的數位設備 — 例如行動電話、錄影機、電腦、遙控器等等 — 這些所有的設備都提供了機會，讓大家去重新設計，拯救那些洩氣的使用者！學生們該問問自己，關於那些困惑使用者的設備問題在哪裡，又該如何將它設計的更好？

這個活動在說什麼？

人與電腦的互動這件事，其實是透過設計、評價與實作電腦系統，讓人們能有效率並安全地進行他們想做的活動。在過去，電腦只為專家設計，使用者必須有高學歷，並受過特別訓練才能使用。但現在，電腦已經成為我們所有人都必須使用的日常工具，而通常大家只要去書局買本「ＸＸＸ實戰攻略」之類的書就可以學怎麼用電腦了。所以，現在的電腦其實需要更注意人機介面。

一些涉及到人命的許多災難之所以會發生，正是因為不適當的介面，例如：墜機事件甚至民航機被擊落；因開關遙控操作燈號錯誤導致高速公路連環車禍；以及核電廠的核災等等。即使是縮小到日常生活的範圍內，還是有許多人在日常生活中，用電腦或其他高科技產品時感到十分挫折 -- 有時挫折會累積到讓人抓狂（曾經有一個警察抓狂到對著他的電腦螢幕開槍）。不僅僅是電腦如此，其它的也一樣：物品的包裝只能用十分尖銳的刀子、鉤子或甚至爪子才能打開；你試著用你想像中的方式去推門，結果打不開還傷到手腕；打開牛奶盒時牛奶總會濺你一身；在電梯旁你看不到按鈕在哪裡；家庭影片系統，廣告聲稱輕鬆操作，但實際上怎麼試都不動作。遇到這些事物，你又該怎麼辦？

我們日益習慣於「人為錯誤」，並思考自己的不足之處；當事情出錯時，人們常常自己把它吞下去。但是很多所謂的人為錯誤，其實是設計的錯誤。人們對訊息的處理有其局限性，而一個好的設計師需要考慮這些。不良的設計卻寫了一份詳細而複雜的使用手冊，然後認為人們應該自己學習並永遠記住，這樣是不切實際的。此外，人都會犯錯，設計時也需要將這個問題考慮進去。

介面的評估在設計過程中是一個非常重要的部分。在這個活動中，當學生們將自己設計的圖示秀給其他同學看時，其實就是屬於介面評估的領域了。在精心控制的心理學式的實驗中，會透過更透徹的評估來對「真正的小精靈們」（也就是他們對圖示的解讀可能跟我們不一樣）做測試。

儘管這些技術的問題會變成許多很好笑的故事，但是人機介面的設計絕不是什麼笑話。不當的介面可能導致的問題不只是個人工作的不愉快而已，甚至有可能引發股災，喪失自尊與生命的問題。

延伸閱讀

Don Norman 的書 “The design of everyday things” 針對了許多日常生活用品上的設計問題，用愉快與開放的觀點來陳述。Jeff Johnson 的 “Designing with the mind in mind” 則對人們如何思考，以及設計介面時要如何考慮人的因素，提出了發人深省的見解。

活動 21
和電腦的的對話 — 圖靈測試

活動摘要

這個活動的主要目的是激發學生們的討論，看看電腦是否能擁有「智慧」，或是在未來是否可能發生。根據一名先驅電腦科學家觀察，一個人對於人工智慧存在的認知，顯示出當時的科技進展，但也血淋淋的呈現了許多對人工智慧的誤解。

課程銜接

• 科技：科技系統。瞭解科技系統是由符號語言工具所呈現，並瞭解在科技系統中的黑盒子所扮演的角色。

習得技能

• 提出問題
• 歸納理解

適合年齡

• 7 歲以上

所需素材

• 一份「活動學習單：圖靈測試問題」（第 243 頁）的複本。可以一組學生看一份，或是用投影機顯示出來
• 一份「活動學習單：圖靈測試問題的答案」（第 244 頁）

與電腦對話

活動介紹

這項活動以遊戲的方式呈現所謂的圖靈測試。讓學生透過提問以及分析得到的答案來嘗試分辨人類和電腦的不同。遊戲的規則如下：

這遊戲將會有四個角色：分別是小帥、小哥、小美、小麗。老師負責控制流程，引導大家提出問題。小帥和小哥是中間人，不參與問題的回答；而小美小麗則要負責回答問題。教室中的其他人是聽眾。小美必須用人類的回答方式作答，而小麗則要假裝自己是電腦來作答。教室中的其他學生要分辨出到底誰扮演人類，而誰扮演電腦。小帥和小哥這時的主要任務是確保遊戲公平進行：他們要負責把問題帶給小美和小麗，但同時要注意不可以洩漏小美和小麗所扮演的角色。 角色確定後，就將小美、小麗和聽眾，分別帶到不同的房間之內。

而關於提問及答題的機制如下。小帥把問題帶給小美，而小哥則把問題帶給小麗。最後兩人再將他們的答案告訴聽眾（實際上聽眾們並不知道是誰將問題傳達給誰）。這個中間人的角色，就是用來保證聽眾們不會看到小美以及小麗回答問題時的狀況。

在活動開始以前，請先選出扮演這四個角色的學生，並告訴他們遊戲規則。有個重點是，身為中間人的小帥和小哥，回來說出答案時必須完全掩飾所有能透露出是小美或小麗的資訊，例如不能說溜嘴告訴全班：「剛剛小美說她的答案是 XXX…」小美的回答由自己想，要簡短、精確與誠實。另一方面，小麗身為扮演電腦的一方，回答問題時必須遵照「學習活動單：圖靈測試問題的答案」上的答案來做回答。

小帥和小哥要準備紙和筆，因為有些問題不容易記住。

1. 在遊戲開始以前，先蒐集學生對於以下這個問題的回答以及想法：你覺得現在的電腦算不算是有智慧的？還是你覺得以後的電腦才有可能實現呢？

2. 向學生介紹此次活動的目的，在於分辨電腦以及人腦的差別。如果班上最後沒有人能夠分辨出兩者的差異，那電腦就算是通過考驗。向學生們解釋小帥和小哥的工作，主要是扮演問題和答案傳送的角色，但是不能說破誰是負責傳達誰的訊息。我們必須讓學生分辨出，到底誰是扮演電腦的那一位。

3. 向學生展示「學習活動單：圖靈測試問題」中的問題。建議可以每兩人共看一份，或是用投影機顯示。

問同學們想要先問哪一個問題。當他們決定以後，再詢問他們為什麼會選擇這個問題，還有他們為什麼會覺得個問題可以分辨出電腦和人腦。這一個推理的過程是整堂課程最重要的一環，因為它能讓學生去思考什麼問題是人類可以回答，但電腦卻不見得辦的到的。

小帥和小哥接著傳遞這些問題和答案。班上同學必須開始討論哪一個答案可能是電腦回答的。

重複這些步驟，多問幾個問題，當班上同學開始發現到哪一位扮演電腦時，就可以暫時停止。但是如果老師發現班上同學很快就知道誰是電腦，誰是人類的話，可以讓小帥和小哥隨機決定，例如擲硬幣或猜拳，然後重新安排傳話的對象是小美還是小麗。

小麗所拿到的圖靈測試問題的答案，看起來就像是由真正「有智慧」的電腦所回答出來的。有些答案會讓人很快知道是電腦回答的，例如：根號二等於多少？回答得出來的人，大部份頂多回答到 1.414，而不會回答到小數點後 20 位。另外，有些問題的答案放在一起看時，可能也會透露出電腦的真實身分。例如：「你喜歡 .....嗎？」這類問題本身很似是而非，但在問了幾次之後，就可以發現利用問題產生答案的一種簡單模式。而有時候，聽眾也可以發現對方在回答時誤解了題目的意思，他們就可以合理的推斷這樣回答的應該是人類。

大部份的答案都很平淡 — 但安全。不過，繼續追問下去就可能發現電腦其實並不真正了解問題的主題。用「我不知道」來作答，對電腦來說相對安全，也容易讓人覺得它是人類，因為回答「不知道」對於一個人類來說是很常見的。但如果電腦太常這樣回答的時候，或是對一個很簡單的問題這樣回答時，仍然有機會被察覺。

因為電腦的目的是要讓對方認為自己是人類，因此有些答案可能是故意設計來誤導大家的。例如遇到數學問題時，刻意拖延回答時間，或是故意回答錯。這類的問題與答案可以讓學生們好好討論。

活動學習單：圖靈測試問題

從這份清單中選出問題來問，試圖辨識出回答者是電腦還是人類。

1. Bart Simpson 的小妹妹叫什麼名字？
2. 你對 Roald Dahl 的看法是什麼？
3. 你是個電腦嗎？
4. 在 3, 6, 9, 12, 15這個序列中，下一個數字是什麼？
5. 你對核子武器的看法是什麼？
6. 2 x 78等於多少？
7. 根號 2 等於多少？
8. 34957 加 70764 等於多少？
9. 你喜歡學校嗎？
10. 你喜歡跳舞嗎？
11. 今天星期幾？
12. 現在幾點了？
13. 二月在閏年裡有幾天？
14. 一周有幾天？
15. 哪個國家的國旗是白色的背景，上面有個紅色圓圈？
16. 你喜歡看書嗎？
17. 你喜歡吃什麼食物？

活動學習單：圖靈測試問題的答案

1. Bart Simpson 的小妹妹叫什麼名字？
   我不記得了。
2. 你對 Roald Dahl 的看法是什麼？
   他寫了一些很有趣的書。
3. 你是個電腦嗎？
   那你是個電腦嗎？
4. 在 3, 6, 9, 12, 15 這個序列中 ,下一個數字是什麼？
   18
5. 你對核子武器的看法是什麼？
   核子武器非常的危險，不應該使用。
6. 2x78等於多少？
   166（這是故意寫錯的！）
7. 根號 2 等於多少？
   1.41421356237309504878
8. 34957 加 70764 等於多少？
   （在算出答案之前 ,請等待 20 秒 ...）105621
9. 你喜歡學校嗎？
   是的，我喜歡學校。
10. 你喜歡跳舞嗎？
    是的，我喜歡跳舞。
11. 今天星期幾？
    告訴他今天是星期幾。
12. 現在幾點了？
    給他一個正確的時間。
13. 二月在閏年裡有幾天？
    2000 年和 2004 年是閏年。（這是故意寫錯的！）
14. 一周有幾天？
    七天。
15. 哪個國家的國旗是白色的背景，上面有個紅色圓圈？
    我不知道。
16. 你喜歡看書嗎？
    是的，我喜歡看書。
17. 你喜歡吃什麼食物？

我肚子不餓，謝謝。

活動變化與延伸

這個遊戲最少可以用三個人來玩，此時讓小帥也同時扮演小哥和小麗的角色。小帥拿問題給小美，並記錄小美的回答，同時也記錄「圖靈測試問題的答案」中該問題的答案。小帥回覆這兩個答案，並用字母 A 和字母 B 來區別這兩個答案的來源。

為了考量電腦是否能在這場問答中模擬人類，跟班上同學討論要回答每個圖靈測試的問題，需要哪些知識。學生可以建議其他他們想問的問題，並且討論任何他們可能設想的答案。這需要一點想像，因為預測這場對話的走向是不可能的。

為了說明，這裡有兩個示範的對話。前面這個範例說明了一台電腦「真實的」可以回答正確的問題，而下面的範例則展現了這個討論可以拓展地多廣泛，並且展現了電腦可能需要引用的廣泛的知識。

問題：請以 Forth Bridge 為主題寫給我一個十四行詩。
回答：別看我，我從來不會寫詩。
問題：計算 34957 加上 70764
回答：（暫停個 30 秒 ...）105621。
問題：你會下西洋棋嗎？
回答：會。
問題：我的國王現在在 K1 上，而且我沒有其他棋子了。你只有國王在 K6 和城堡在 R1 上。輪到你了。
回答：（經過 15 秒的暫停 ...）城堡移到 R8，將死你。

網站上有個廣泛可用系統叫做 ”Eliza” (是一種「聊天機器人」（chatbot），它是一個你可以打字聊天的對象。Eliza 模擬一個與心理治療師的會談，並且可以藉由簡單的規則產生出色且聰明的對話。一些範例會談會在下面討論。學生可以試試 Eliza，或者其他的聊天機器人（像是 Siri！），不過也要留意，有些機器人是用不適合學校學生的語言及主題訓練出來的。

問題：一首十四行的的第一行寫道「我該將汝與一個夏日比較嗎？」若寫「一個春日」會不會一樣甚至更好呢？
回答：這不符合詩的韻律。
問題：那如果是「一個冬日」呢？這完全符合韻律。
回答：是，但沒有人想被拿來比喻成一個冬日。
問題：你認為 Mr. Pickwick 會讓你想起聖誕節嗎？
回答：某種程度上如此。
問題：然而聖誕節是個冬天的日子，而且我不認為 Pickwick 先生會介意這個比喻。
回答：我想你不夠嚴謹，一個冬天的日子代表的是一個尋常的冬日，而不是像聖誕節這種特殊的日子。

這個活動在說什麼？

好幾世紀以來，哲學家一直在爭辯一台機器是否能模擬出人類的智慧，以及反過來問，人腦的能力是不是根本無法超過一個寫得好的電腦程式？這個問題看法兩極。有些人覺得這個想法荒謬、瘋狂，甚至是褻瀆的，而另一些人則認為人工智慧是不可避免的，最終我們會開發出跟我們一樣聰明的機器。（而無數的科幻小說作者都指出，如果電腦的智慧超過我們，則他們會展開革命 — 自己可以建造出比他們更聰明的機器。）人工智慧（AI）研究學者一直被批評，他們所謂的崇高目標只是用來從政府單位吸引研究資金，而政府單位的目標則是要建造出自動化的戰爭機器。而這些研究學者們則譴責這種看法是十九世紀反工業革命的盧德運動的翻版，並指出在生活週遭多一點智慧對社會是有極大好處的。另外還有一種比較平衡一點的觀點是，人工智慧既不荒謬也不必然會發生：既然目前沒有任何電腦程式展現出「智慧」，這些問題目前爭辯也不會有任何結論。

這些關於人工智慧的爭辯，取決於對「智慧」的定義。已經有許多定義被提出與辯論。其中一個最有趣的提案是在 1940 年晚期由艾倫圖靈（Alan Turing，一位著名的英國數學家，也是二戰期間破解德軍密碼的靈魂人物與一名長跑好手）所提出。他提出一種「思想實驗」。圖靈的想法是不去定義何謂「智慧」，而是提出一個情境讓電腦去展示他們的智慧。圖靈所提出的情境就類似這個活動中所玩的，本質上也是由一名提問者，與一個人還有一台電腦透過電子打字機（在 1940 年代是很先進的技術！）來互動。如果提問者分辨不出誰是電腦誰是人類，那麼這台電腦就通過了「圖靈測試」，也就是人工智慧的測試。使用電子打字機也是為了避免提問者經由電腦的物理特徵，或是聲調等辨別。這樣一來，電腦不需要在外觀、聲音、觸感、甚至味道上偽裝，因為這些感觀跟我們所謂的「智慧」比較沒有關係。

最原始的圖靈測試跟我們的活動有一點不一樣。他所提議的最原始的測試，是一名男人與一名女人被詢問，而提問者要決定兩名回答者的性別。男人的目標是要讓提問者相信他其實是回答的另一名女人，而女人的目標則是要讓提問者相信她才是貨真價實的。接著圖靈想像（因為這只是一個思想實驗），用一台電腦來取代其中一人，看看電腦是否能假扮人類通過這個「模仿遊戲」。我們把這個想法轉換成在教室裡玩的活動。之所以不讓學生猜測回答者的性別，其中一個原因是學生所提出來的問題可能不合適，而且這樣也隱含了對性別的刻板印象，更別提回答者必須用欺騙的方式回答。

模仿智慧是一件困難的工作。如果今天把角色反過來，由人類設法通過測試讓提問者以為自己是電腦，那根本很難通過：因為只要一問「123456 × 789012 等於多少」就掛了。

不過，要讓電腦取得對話能力的皮毛其實出乎意料地簡單。在 1960 年早期所開發出來的知名程式 Eliza，就在與使用者的對話中假扮成一名非指導性的心理治療師。右方是某人與這個系統間的對話範例。一些模仿自然對話的原則包括：

1. 產生一些罐頭回應（例如在回應「他們總是如何如何」時，回答「可以給我一個例子嗎？」；遇到「他說如何如何」時回答「聽到這些我感到很遺憾。」等等）
2. 重複使用者的敘述（例如「我的男友叫我來這裡。」「喔，妳的男友叫妳來這裡。」）
3. 辨識出關鍵字（聽到「媽媽」則回應「談談你的家人吧！」）
4. 一些常用的片語或回答（如果 XXXX 的話，對你來說有意義嗎？）
5. 從前面的對話中找出問題來詢問（如果遇到 XXXX，你打算怎麼辦？）

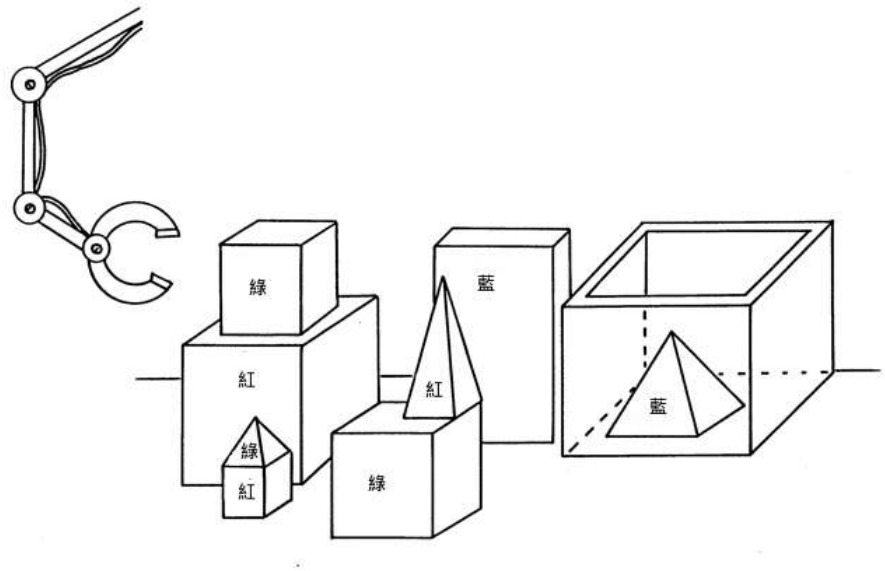
當然，這樣的對話不代表就通過了圖靈測試，因為提問的目的並非要分辨出回答的是人還是機器。如果真的刻意要分辨，那這個系統其實撐不了太久。

另外有一個廣為流傳的故事，可能是杜撰的，但現在已經成為一個人工智慧領域的傳說。有一名電腦銷售副總裁宣稱他偶然連到一個終端機。通常該終端機是直接連線到某特定人的家中，但是那一次剛好正在跑一個「心理治療師」程式。他們間的對話顯示在右方。雖然證據顯示這名副總裁的確有參與這個事件，但很明顯的整個情境本來就讓他認為他是在跟人談話。如果相反地，他一開始就起疑而刻意辨識，那應該很快就會發現。

人：　唉，男人們都一樣。
電腦：妳是指哪方面？
人：　男人們總是不斷用許多問題來困擾我們。
電腦：可以給我一個例子嗎？
人：　噢，就像我男朋友要我來這裡。
電腦：喔，妳男朋友要妳來這裡。
人：　他說我大部份時間都太憂鬱了。
電腦：噢，聽到妳這麼憂鬱我感到很遺憾。
人：　那是真的，我很不快樂。
電腦：那妳認為到這裡來能讓妳快樂一點嗎？
人：　可以肯定的是，我需要幫忙。
電腦：跟我談談妳的家人吧。
...
電腦：講到妳父親妳會聯想到什麼？
人：　家暴。
電腦：這是為什麼妳男朋友要妳來這裡的原因嗎？
....

副總裁：請問今天早上我可以用一下你的電腦嗎？
終端機：為什麼這樣問？
副總裁：這樣我可能可以多賣出我們的電信產品。
終端機：你為什麼這麼不確定？
副總裁：我這有些潛在的客戶，從來沒看過這個系統的展示。
終端機：這對你很重要嗎？
副總裁：當然啊！
終端機：你看起來非常積極。
副總裁：喂，這太扯了吧！
終端機：你說這太扯了吧是什麼意思？
副總裁：有任何理由我現在不能使用電腦嗎？
終端機：跟我談談你說的潛在客戶吧。
副總裁：請你現在撥電話給我，號碼是 491-1850
(因為副總裁沒有在尾端輸入標點，所以電腦沒有回答。副總裁很生氣，所以他打電話給負責那個終端機的人的家中。)
副總裁：你剛剛為什麼跟我講那些有的沒的，就是不讓我用電腦？
人：我？我剛剛有跟你在談話嗎？

另一個看來可以做智慧對話的系統叫做 "SHRDLU"，在 60 年代晚期開發。它可以接受指令，並操作在學生桌上的積木塊。事實上，這些積木塊與桌子只是在電腦螢幕上模擬的圖片而已，但要真的把它轉成真實的桌子與積木，用真的機器人來做也不是什麼難事（原則上啦）。下面這張圖就是機器人的小小世界。



SHRDLU 接續對話出奇地流利，下一頁有它跟人的對話內容。這對話內容展示了它可以遵照指令，分辨出在說的是哪個積木，也可以接受新的詞（尖塔）。

人：夾起大的紅色積木。
機器人：OK.
（系統理解指令時會回覆 "OK."。不過為了夾起大的紅色積木，它必須先找個空間，把上方的綠色積木移走。）
人：把三角錐夾起來。
機器人：抱歉，我不知道你說的是哪一個三角錐。
（總共有三個三角錐，所以機器人不做任何動作。）
人：找出哪個積木比你手上的那個要高，並把那塊積木放入盒中。
（系統必須理解指令中的一些詞彙，例如「它」是指哪個積木，來有「你手上的那個」等，並用消去法推論它手上的是哪一個積木。）
機器人：我假設你說的「它」是指比我手上的積木要高的那一個。
（持續印出資訊）
機器人：OK.
（把紅色積木放下並夾起較高的藍色積木。）
人：盒子裡有什麼？
機器人：藍色三角錐與藍色積木。
人：三角錐放在哪裡？
（雖然系統前面回應過「我不知道你說的是哪一個三角錐。」但是這次它知道指令中指的一定是它剛才提到的那個。）
機器人：在盒子中。
人：做一個尖塔 ...
機器人：抱歉，我不知道「尖塔」是什麼。
（系統遇到它不認得的詞彙時就立即發出回應。）
人：「尖塔」是用兩個綠色積木跟一個三角錐堆起來的。
（引號表示人正在定義一個新詞彙。）
機器人：我懂了。
人：目前有任何尖塔嗎？
機器人：沒有。
（機器人用新的定義來辨認何謂尖塔。）
人：做一個尖塔。
機器人：OK.
（人給它的描述已經足以定義並瞭解如何辨認並製作一個尖塔。）

不過，這個系統其實很脆弱。它展示的人工智慧美好景象其實很容易破滅。

右邊的對話是上面對話的延續 — 好啦，這不是真的去跟系統對話讓系統崩潰，不過這類對話的確很可能會發生。雖然它可以定義新的名詞（尖塔），但仍然只限於非常特定種類，因為這系統的「智慧」本來就很有限。最終，它還是得承認它甚至不知道自己用的字是什麼意思（「抱歉，我不知道『抱歉』是什麼。」）！

人：我用一支水槍跟你交易大的紅色積木。
機器人：抱歉，我不知道「交易」是什麼。
人：「交易」是指自由交換物品。
機器人：抱歉，我不知道「自由」是什麼。
人：「自由」就是不被強迫做事情，任何事都是出於自願。
機器人：抱歉，我不知道「做事情」是什麼。
人：「做事情」就是人們有目的地做，而不是出於意外。
機器人：抱歉，我不知道「人們」是什麼。
人：抱歉，我以為你夠聰明。
機器人：抱歉，我不知道「抱歉」是什麼。

現在，讓我們換個方式來接續對話。這完全是幻想出來的。雖然對話很短，但這些對話仍然展示出世界知識的豐富。

1. 「機器人」知道已經有水槍的情況下，對方的水槍沒什麼價值。
2. 再來，它期望對方瞭解，因為它沒有繼續解釋拒絕接受的原因。
3. 它發覺對方很想要一塊積木來建個重要的東西。
4. 接著，它察覺到積木的數量很重要。
5. 它試著用青蛙這個玩笑去軟化對方。
6. 它表示已經準備好要討價還價了

人：我用一支水槍跟你交易大的紅色積木。
機器人：我已經有一支水槍了，但我會給你兩個積木與一個三角錐，讓你可以跟你那隻滑溜溜的老青蛙玩。

當然，這種程度的對話藝術已經遠遠超過現在電腦的能力了！

在右方的故事是由另一個 80 年代後期的電腦所產生。（譯註：原文後附）雖然它很容易看出是電腦產出的故事，因為它所用的句子太簡潔太制式了，但是經過一番調整，加上許多細節之後就會不一樣了。有趣的不是這個故事表面，而是它可以體現出整個故事情節。雖然距離真正人類作者寫出的情節還有很長一段路，但它似乎已經抓住一些衝突的人性元素。這幾年已經開始出現不少自動產生文章的系統，例如網路上常見的那些內容農場，或是股票分析等等。要評估這些系統，要看有多少素材是標準的樣式，又有多少情節是有創意的。

從前，有一個名叫杜魯門的北極燕鷗。杜魯門無家可歸。杜魯門需要一個窩。他飛到岸上。杜魯門在找一些樹枝。杜魯門沒有發現樹枝。他飛到苔原。他遇到了一個名為賀拉斯的北極熊。杜魯門問賀拉斯哪裡有一些樹枝。賀拉斯隱瞞了樹枝。賀拉斯告訴杜魯門說在冰山上有一些樹枝。杜魯門飛到冰山。他在找一些樹枝。他沒有發現樹枝。賀拉斯在找一些肉。他發現一些肉。他吃杜魯門。杜魯門死了。

現在每年都會舉辦年度洛伯納獎（Loebner prize）的競賽，讓電腦程式們競爭是否能通過圖靈測試，讓裁判們認為它們實際上是人類。截至 2014 年為止，尚無任何電腦程式贏得二獎（成功讓裁判認為人其實是電腦）與首獎（成功讓裁判無法辨識出人與電腦，也就是通過圖靈測試，一旦首獎頒出，這個競賽即宣告結束。）不過每年都會頒出裁判認為「最像人類」的電腦程式。1991 年第一次的比賽中，一個電腦程式設法透過故意打錯字而得到這個獎！

目前尚沒有任何人工智慧系統通過完整的圖靈測試。不管有沒有，許多哲學家們仍持續在爭辯圖靈測試無法真正衡量出大多數人所謂的「智慧」。它所測試的是行為，也就是一個電腦程式是否擁有「智慧」的表象，但這不盡然代表它有「智慧」。如果你沒有意識，不認識自己，沒有知覺，沒辦法感受自我意識，沒辦法體驗愛，甚至沒辦法「活著」，那可以算是有人類智慧嗎？

人工智慧的爭論看來還會持續幾十年吧。

延伸閱讀

哲學家 John Haugeland 所著的 “Artificial intelligence: the very idea” 是一本非常傑出，值得閱讀的書。裡面談到關於人工智慧的爭論，在這個活動中所描述的一些事情也是從這本書來的（特別是 SHRDLU 的對話與它們的討論）。

原始的圖靈測試是由 Alan Turing 所寫的文章 “Computing machinery and intelligence” 中所描述。這篇文章在 1950 年刊登在 “Mind” 這本哲學期刊上，並且在 Feigenbaum 與 Feldman 編著的 “Computers and thought” 重新被提出。這篇文章包含了最開始的兩個對話。

Eliza 這個模仿心理治療師的程式在 J. Weizenbaum 於 1966 年發表在 “Communications of the Association for Computing Machinery” 這本電腦雜誌中的文章 “ELIZA -- A computer program for the study of natural language communication between man and machiine” 中有描述。

喜歡堆積木的那個機器人程式，是由 Terry Winograd 所寫的博士論文中所提出，發布在 “Understanding natural language” 這本書中 (出版社為 Academic Press，1972 年紐約)。

產生故事的程式是由 Tony Smith 與 Ian Witten 在 1990 年於 “Proceedings of the 10th International Conference on Computing and the Humanities” 中發表的 “A planning mechanism for generating story text” 中有描述。故事原文如下："Once upon a time there was an Arctic tern named Truman. Truman was homeless. Truman needed a nest. He flew to the shore. Truman looked for some twigs. Truman found no twigs. He flew to the tundra. He met a polar bear named Horace. Truman asked Horace where there were some twigs. Horace concealed the twigs. Horace told Truman there were some twigs on the iceberg. Truman flew to the iceberg. He looked for some twigs. He found no twigs. Horace looked for some meat. He found some meat. He ate Truman. Truman died."

原著：Tim Bell, Ian H. Witten, Mike Fellows
活動設計：Robyn Adams, Jane McKenzie
插畫：Matt Powell
2015 年修訂版：Sam Jarman

譯者：翁佳驥等 65 人
發行人兼總編輯：翁佳驥
校稿：丁國傑、邱彥銘、Jason Tsai
插圖中文化：翁佳驥
排版：唐鼎鈞、徐佳筠、陳昀聖、曾柏萱、翁鈺瑋

出版：中華民國軟體自由協會
地址：221 新北市汐止區汐萬路三段 199 巷 26 弄 4 號
電話：(02) 5593-3701
傳真：(02) 7741-7371
電子郵件信箱：service@slat.org

印刷：沈氏藝術印刷股份有限公司
初版：2016 年 3 月

中華民國軟體自由協會
捐款帳號：
國泰世華銀行中和分行
帳號：045-03-902135-1
戶名：中華民國軟體自由協會翁佳驥