



國中資訊科技這樣教

張凌倩

國立臺灣師範大學 邏輯與程式教育組

新北市立新店高中

大綱

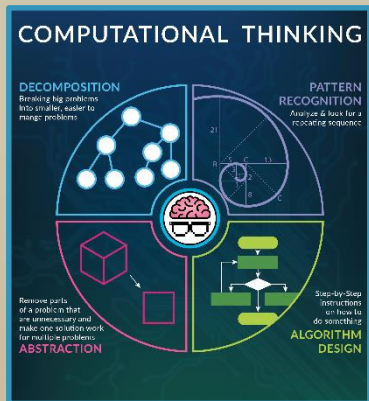


- 全球資訊教育趨勢 & 資訊科技教育的目的 (複習)
- 教學設計
- 資訊科技教學策略
 - 運算思維導向教學設計
 - 不插電的電腦科學
 - 動手實作
 - 視覺化
 - STEM/STEAM教學



全球資訊教育 趨勢

全球資訊教育趨勢



以**運算思維**為主軸



以**程式設計**為工具



以**電腦科學**為基礎



以**高階能力**為目標

這樣的資訊教育，學生了解演算法的概念嗎？





資訊科技教育 的目的

資訊科技教育的目的



- 學習電腦科學不是只有讓學生使用軟體、建置手機App等，而是可幫助學生發展凌駕於各種學科之外的創造力、創業思維
- 可幫助學生發展系統性思考、問題解決、邏輯演繹、更精確表達自我、批判性思考等高階能力

資訊科技教育的目的



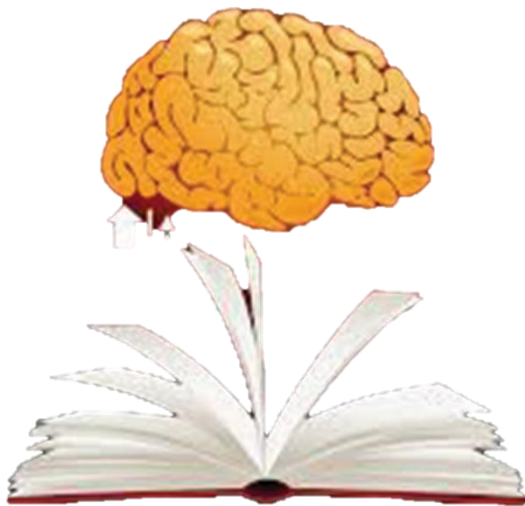
- 什麼是**批判性思考**?是批評、責難的思考嗎?
 - 客觀分析以**分辨**有用與無用的**資訊**，以進一步作合理的**判斷**或**解決問題**
 - 須能**評估****資料**、**事實**、**現象**、**研究結果**等



資訊科技教育的目的



■ 高階能力皆奠基於紮實的基礎知識



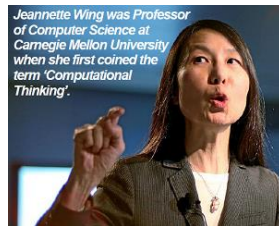


如何教 運算思維

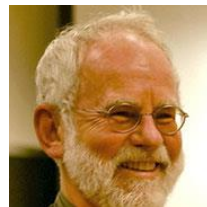
什麼是運算思維-定義



- “運算思維是利用**電腦科學**的基本概念進行問題解決、系統設計與人類行為理解的思維模式” (Wing, 2006)



- “運算思維讓我們能擁有**電腦科學**家面對問題時所持有一種的思維模式” (Grover & Pea, 2013)



資料來源：

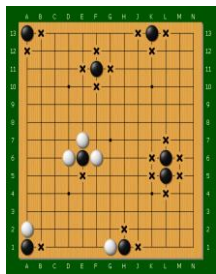
Wing, J. M. (2006). Computational thinking. Communications of the ACM, 49(3), 33-35.

Grover, S., & Pea, R. (2013). Computational Thinking in K-12: A Review of the State of the Field. Educational Researcher, 42(1), 38-43.

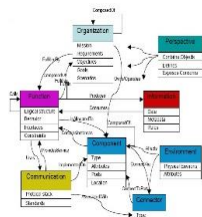
運算思維



- 運算思維在運算歷程中處處可見



問題解析

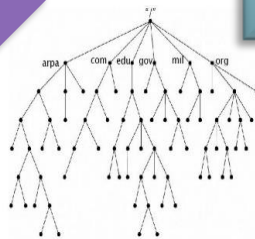


資料表示

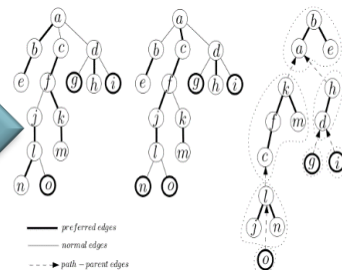
a_{ij} m colonne, i cresce
 n righe, j cresce

$$\begin{pmatrix} a_{11} & a_{12} & \dots & a_{1m} \\ a_{21} & a_{22} & \dots & a_{2m} \\ \vdots & \vdots & \ddots & \vdots \\ a_{n1} & a_{n2} & \dots & a_{nm} \end{pmatrix}$$

matrice $n \times m$



演算法思維



有一定的順序嗎？

你是電腦科學家，有固定的解題順序嗎？

討論



- 如何教出運算思維？

如何教運算思維



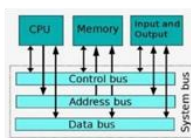
- 運算思維教學宜回歸資訊科技課的本質，重視**資訊科學**的概念理解與應用

實體世界問題



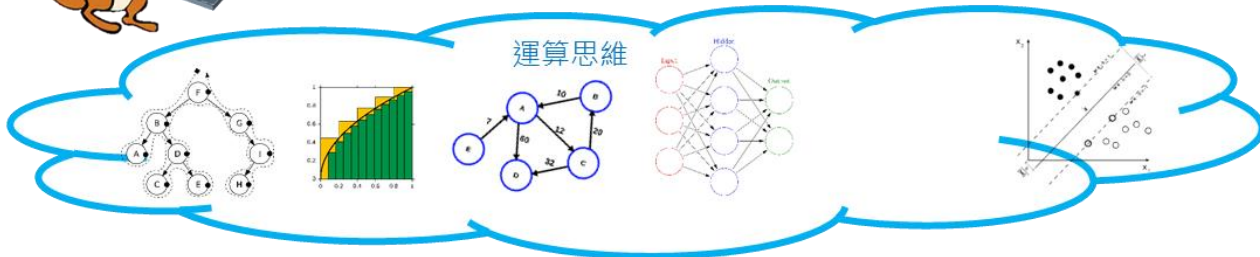
```
public RockPaperScissors(Context context, RockPaperVocabulary, ArrayList<Four>
    this.mContext = context;
    this.layoutsInflater = LayoutInflater.from(mContext);
    this.vocabulay = vocabulary;
    this.examples = examples;
}

@Override
public int getCount() {
    return examples.size() * 4;
}
```



運算工具

運算思維



如何教運算思維



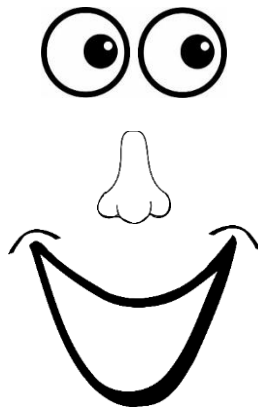
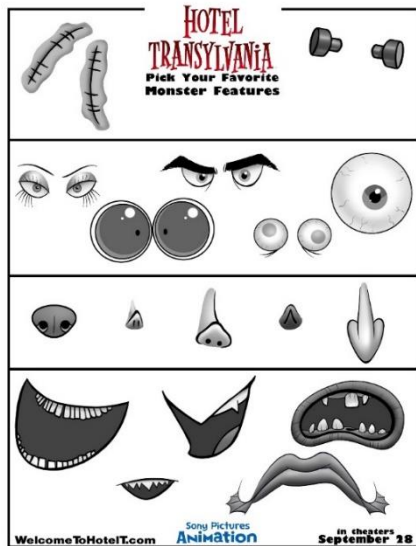
- 不宜與資訊科學分離而只去討論運算思維的各元素

如何教運算思維



■ 抽象化(Abstraction)

- 除去細節以簡化並聚焦於重點
- 辨識與描述普遍化的性質(一般化)

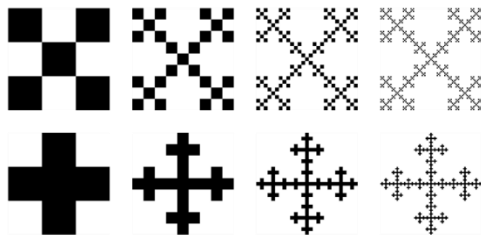
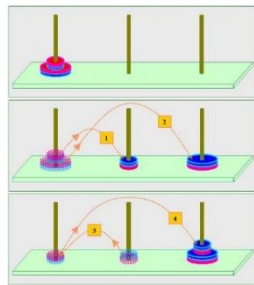
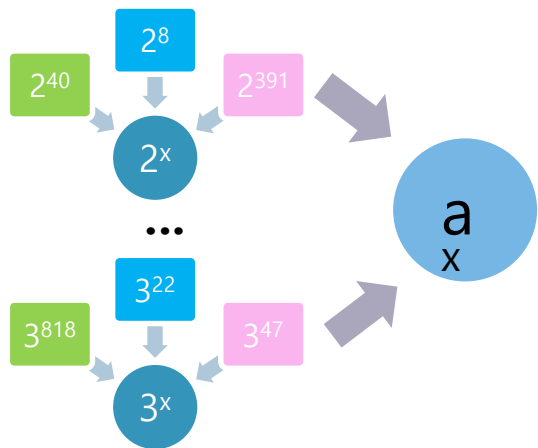


這只是要跟老師們說明什麼是抽象化，但並不是要用這樣的方法教給學生

如何教運算思維



- 樣式一般化(Pattern generalization)
 - 產出共通的模式、規則、原則或理論



遞迴演算法

```
def koch(t, order, size):
```

```
    if order == 0:  
        t.forward(size)
```

基本元素

```
    else:
```

```
        koch(t, order-1, size/3)  
        t.left(60)  
        koch(t, order-1, size/3)  
        t.right(120)  
        koch(t, order-1, size/3)  
        t.left(60)  
        koch(t, order-1, size/3)
```

遞迴關係

如何教運算思維



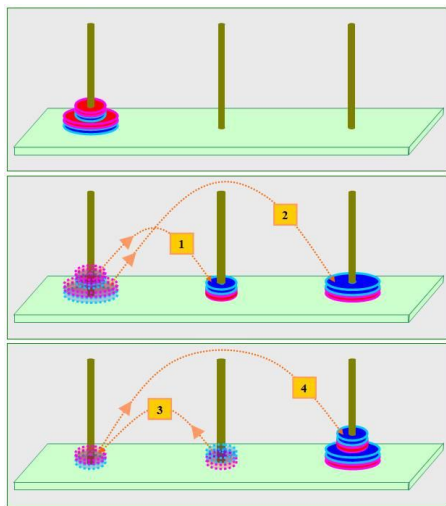
- 樣式辨識(Pattern recognition)
 - 在資料中觀察樣式、趨勢或規則

這只是要跟老師們說明什麼是樣式辨識，但並**不是**要用這樣的方法教給學生

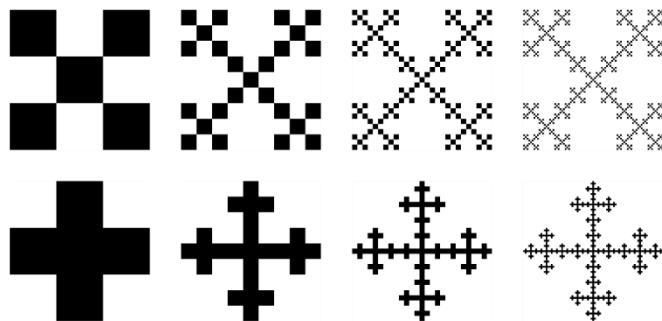
如何教運算思維



■ 樣式辨識與一般化讓運算發揮其能力



河內之塔：透過解題規則的找尋，讓運算幫我們處理人腦難以運作的盤子數量



碎形：透過繪製規則的找尋，讓運算幫我們繪製人類難以處理的複雜圖形。

討論



- 你曾經如何透過資訊科技的知識統整解題中需要的通則？
- 如何將這種能力教給學生？

如何教運算思維



■ 問題解析(Decomposition)

- 將資料、程序、問題拆解成較小、較容易處理的部分
- <http://games.thinkingmyself.com/>

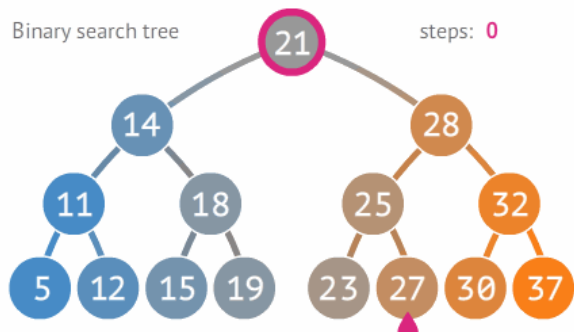


如何教運算思維

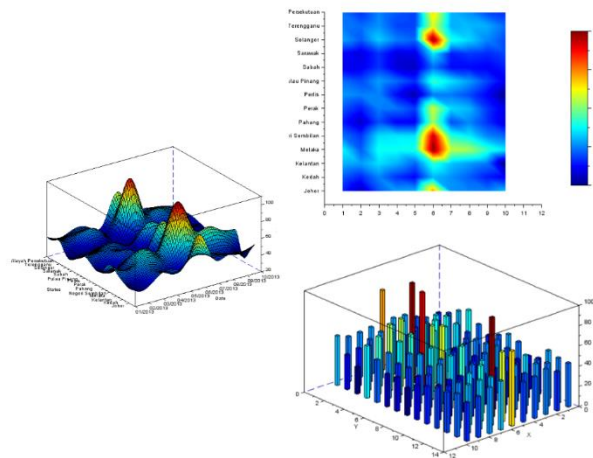


■ 資料表示(Data representation)

- 用適合的圖表、文字或圖片等表達與組織資料



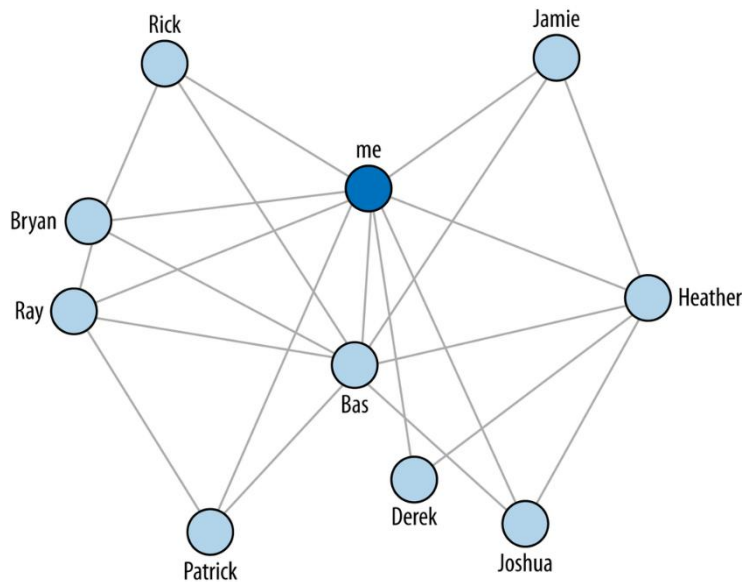
www.penjee.com



如何教運算思維



■ 資料表示(Data representation)



討論

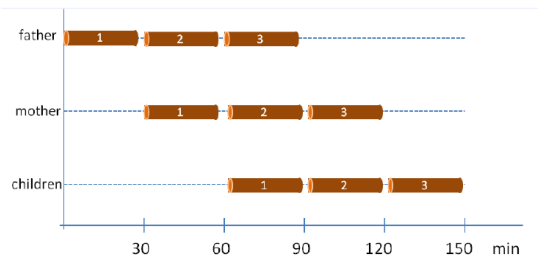
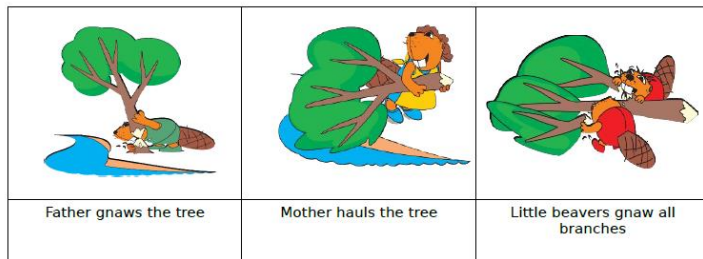


■ 資訊科技領域中有哪些有效的資料表示方法？

如何教運算思維



- 演算法思維(Algorithmic thinking)
 - 產出有序指令以解決問題或完成任務

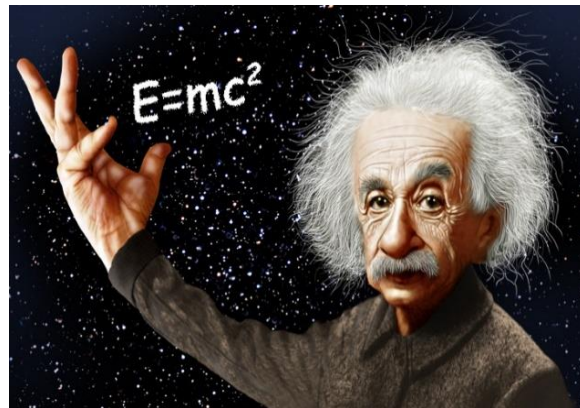


如何教運算思維



■ 建模(Modelling)

- 根據不同需求(為了容易瞭解、定義、量化、視覺化或模擬等)，將複雜的現象以簡化的方式表達
- 可用以將抽象的概念視覺化
- 可作為實驗結果闡釋的依據
- 可作為預測的基礎





教學設計

教學設計-選擇學習內容



■ 選擇學習內容

— 課綱中沒有明列人工智慧內容，那要教嗎？

章	節	對應課綱學習內容參考
第1章 人工智慧 初探	第1節 關於人工智慧的那些事	資 H-IV-6 資訊科技對人類生活之影響
		資 H-IV-7 常見資訊產業的特性與種類
		資 H-V-3 資訊科技對人與社會的影響與衝擊
	第2節 人工智慧的演進	資 S-IV-1 系統平台重要發展與演進
		資 S-IV-4 網路服務的概念與介紹
		資 S-V-2 系統平台之未來發展趨勢
	第3節 人工智慧的道德議題	資 H-IV-1 個人資料保護
		資 H-IV-3 資訊安全
		資 H-IV-4 媒體與資訊科技相關社會議題
		資 H-IV-5 資訊倫理與法律
		資 H-V-1 資訊科技的合理使用原則
		資 H-V-2 個人資料的保護
	第4節 當代人工智慧發展	資 H-V-3 資訊科技對人與社會的影響與衝擊
		資 H-IV-6 資訊科技對人類生活之影響
		資 S-V-2 系統平台之未來發展趨勢
		資 D-V-1 巨量資料的概念
		資 D-V-2 資料探勘與機器學習的基本概念
		資 H-V-3 資訊科技對人與社會的影響與衝擊

第2章 人工智慧 的運作原理	第1節 傳統人工智慧	資 A-IV-1 演算法基本概念 資 D-IV-3 資料處理概念與方法 資 A-V-1 重要資料結構的概念與應用
	第2節 機器學習	資 A-IV-1 演算法基本概念 資 T-IV-1 資料處理應用專題 資 A-V-1 重要資料結構的概念與應用 資 A-V-2 重要演算法的概念與應用 資 D-V-2 資料探勘與機器學習的基本概念
	第3節 類神經網路與深度學習	資 A-IV-1 演算法基本概念 資 D-IV-1 資料數位化之原理與方法 資 D-IV-3 資料處理概念與方法 資 A-V-1 重要資料結構的概念與應用 資 A-V-2 重要演算法的概念與應用 資 D-V-2 資料探勘與機器學習的基本概念

教學設計-選擇學習內容



■ 選擇學習內容

七年級 演算法 (A)	八年級 演算法 (A)	九年級 系統平台 (S)
資 A-IV-1演算法基本概念 問題解決、流程控制	資 A-IV -2陣列資料結構的概念與應用 資 A-IV -3基本演算法介紹（搜尋、排序）	資 S-IV -1系統平台重要發展與演進 資 S-IV -2系統平台組成架構運作原理 資 S-IV -3網路技術的概念與介紹 資 S-IV -4網路服務的概念與介紹
程式設計 (P)	程式設計 (P)	資料表示、處理及分析 (D)
資 P-IV -1基本概念、功能及應用 資 P-IV -2結構化程式設計 循序、選擇、重複結構	資 P-IV -3陣列程式設計實作 資 P-IV -4模組化程式設計的概念 資 P-IV -5模組化程式設計與問題解決實作	資 D-IV -1資料數位化之原理與方法 資 D-IV -2數位資料的表示方法 資 D-IV -3資料處理概念與方法 資料整理與整合、壓縮、轉換
資訊科技應用 (T)		資訊科技應用 (T)
資 T-IV -1資料處理應用專題		資T-IV-2資訊科技應用專題 多媒體應用專題、程式設計應用專題
資訊科技人類社會 (H)	資訊科技與人類社會 (H)	資訊科技與人類社會 (H)
資H-IV-1個人資料保護 資H-IV-2資訊科技合理使用原則 資H-IV-3資訊安全	資H-IV-4資訊科技重要社會議題 資H-IV-5資訊倫理與法律	資H-IV-6資訊科技對人類生活之影響 資H-IV-7常見資訊產業的特性與種類

教學規劃-訂定教學目標



討論: 以下所列各主題的教學目標是否合宜?

程式設計-條件判斷

1. 能理解if條件判斷的概念、運作模式。
2. 能熟悉if條件判斷式的程式撰寫方法。
3. 能了解else, else if的差別與運用。
4. 能利用if條件判斷解決簡單的問題。
5. 能了解在什麼樣的情況下要使用if條件判斷。

演算法-分而治之

1. 能了解運算思維中，拆解問題、尋找規則的重要性
2. 能體驗運算思維中，演算法設計與抽象化的過程。
3. 能在解題過程使用運算思維，理解分治演算法的產出過程。
4. 能比較各組拆解問題的方式、設計出的解題流程有何差異，評估其優劣。
5. 能理解在特定情境下，使用分治演算法的必要性和可行性。

人工智慧簡介

認知：

1. 學生能了解人工智慧的演進
2. 學生能了解機器學習的概念
3. 學生能了解「分類、分群」的概念

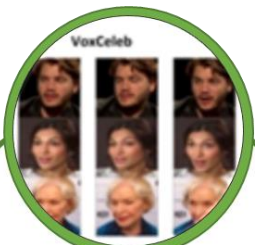
情意：

1. 學生能主動察覺日常人工智慧的演變
2. 學生能主動察覺日常生活「分類、分群」的應用

技能：

1. 學生能運用「分類、分群」解決相關問題
2. 學生能批判性思考人工智慧對社會的影響

教學規劃-設計教學活動



AI人臉秀

以同學生日為發想，使用 Colab 實作 Deepfake 技術，製作慶生影片。



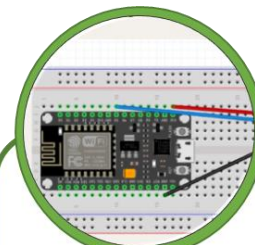
蘋果分類

以分類蘋果為例，藉由 Teachable Machine 實作影像辨識技術。



紀念T-shirt

為了製作校慶紀念T-shirt，可透過 Python 實作機器學習 Kmeans，預測衣服版型。



智慧溫控 妙管家

以開發板實作物聯網，將所偵測到的數據上傳至雲端。



當Siri 遇上物聯網

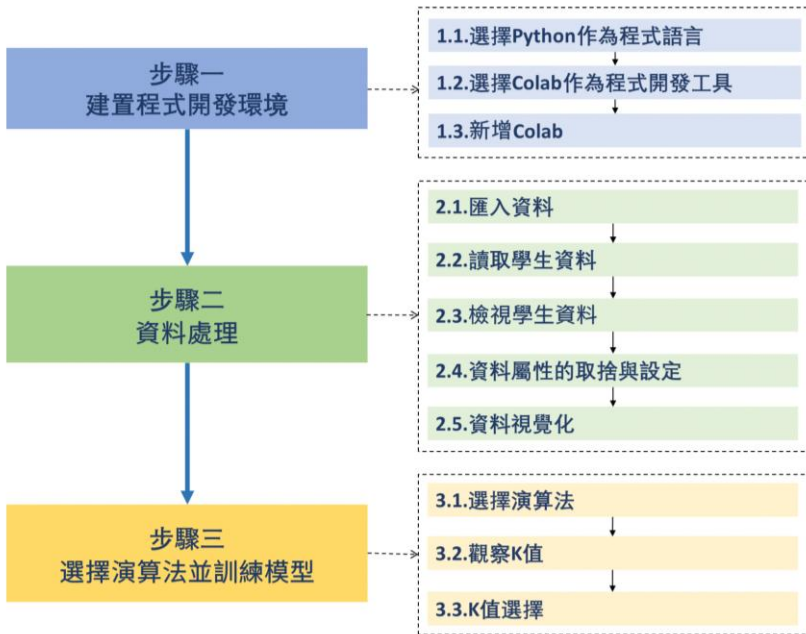
善用語音助理，進一步以更簡易的方式控制物聯網裝置。



紀念T-shirt

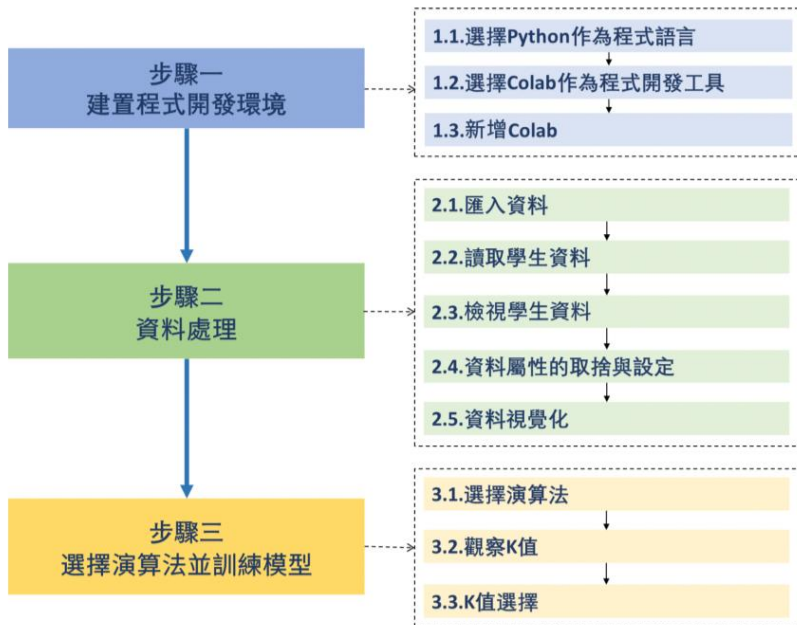


實作流程





紀念T-shirt



步驟三、選擇演算法並訓練模型

3.1. 選擇演算法

根據問題解析，此類問題適合使用非監督式學習，而此專題選用sklearn套件的K-means 函式庫做為機器學習演算法，程式範例如下。

```
# 使用 KMeans 演算法訓練機器
# 資料匯入
from google.colab import files
uploaded = files.upload()
import pandas as pd
df = pd.read_csv('data.csv', encoding='BIG5')

# Kmeans 演算法參數設定
from sklearn.cluster import KMeans
import matplotlib.pyplot as plt
student = df[['資料屬性1', '資料屬性2']]
KM=KMeans(n_clusters=8, init='k-means++')

# 執行 Kmeans 演算法
KM.fit(student)

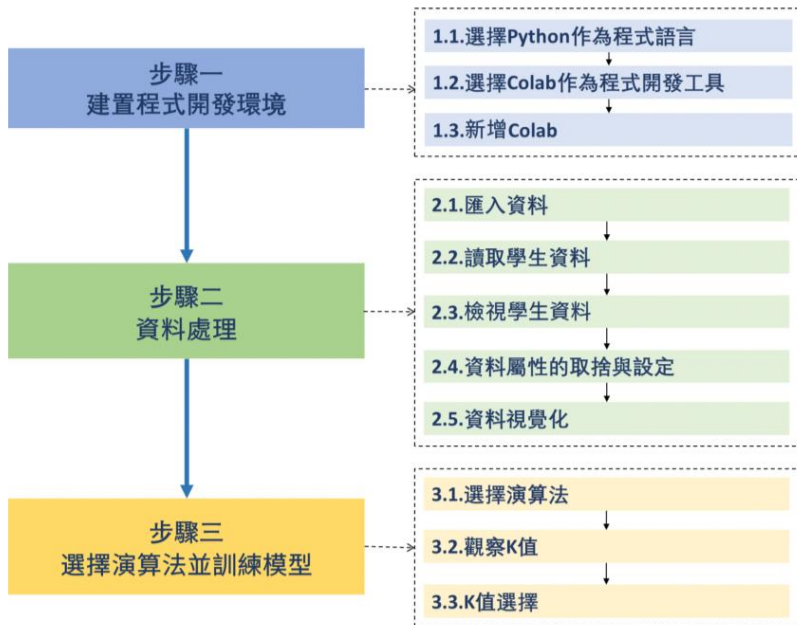
# 繪製圖形
plt.scatter(student.iloc[:,0],student.iloc[:,1],cmap='autumn',c=KM.predict(student))
centers=KM.cluster_centers_
plt.scatter(centers.T[0],centers.T[1],c='black',s=100,alpha=0.5)
plt.xlabel('x 軸(資料屬性1)標籤名稱，必須為英文方能呈現')
plt.ylabel('y 軸(資料屬性2)標籤名稱，必須為英文方能呈現')
plt.title('T-shirt')
plt.show
```

紅框部分：使用者需修改才能正確執行程式。

接下來我們針對程式碼一一做說明。

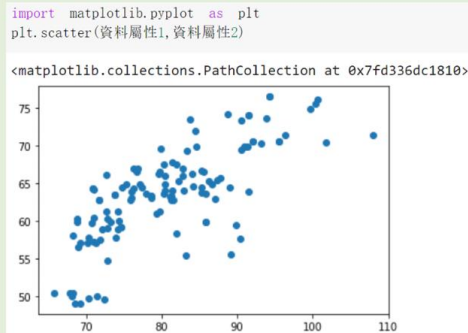


紀念T-shirt



2.5. 資料視覺化

python可以引用Matplotlib函式庫，將資料視覺化，傳遞更易於理解的資訊。為初步了解訓練機器學習的資料屬性分佈情況，故利用scatter()繪製散佈圖，呈現2個維度的資料關係，程式範例如下：



認識scatter(x,y):

1. 依據傳入x軸和y軸的資料，繪製散佈圖，x和y的值個數必須相同。

2. 可使用參數整理如下，

c: 符號顏色，例如，c='black'，設定所有符號顏色為黑色。

s: 符號大小，例如s='100'，設定符號大小為100點。

alpha: 符號透明度，例如，alpha='0.5'，設定符號透明度為0.5。

cmap: 顏色配置，例如，cmap='autumn'，設定符號顏色配置為'autumn'。

linestyle: 線條樣式，例如，linestyle='--'，設定線條樣式為'--'。



運算思維導向教材設計



■ 教學設計步驟



運算思維導向教材設計



選擇課綱
學習內容

對應重要
運算思維

對應主要
學習表現

陣列資料結構
的概念與應用

資料表示、
抽象化

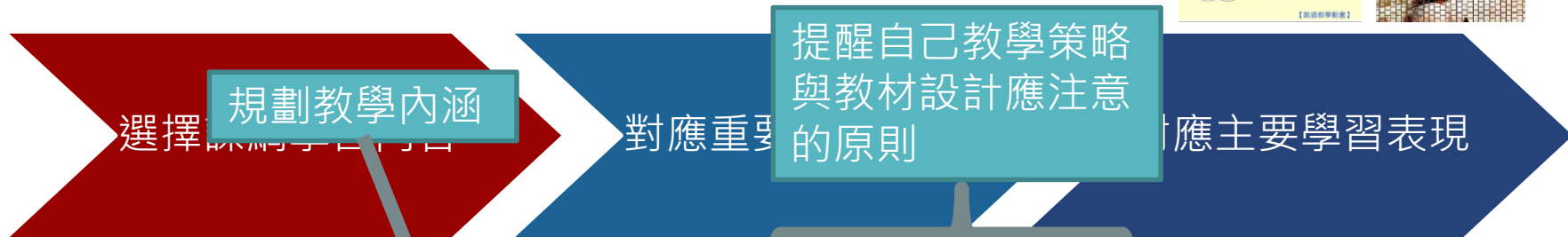
運算思維與
問題解決

結構化
程式設計

演算法思維、
樣式辨識

運算思維與
問題解決

運算思維導向教材設計



學習內容	運算思維	學習表現
資D-IV-1資料數位化之原理與方法 資D-IV-2數位資料的表示方法 資D-IV-3資料處理概念與方法	資料表示與處理 抽象化 演算法思維	資t-IV-4能應用運算思維解析問題。 資p-IV-1能選用適當的資訊科技組織思維，並進行有效的表達。

運算思維導向教材設計



選擇課綱學習內容

對應重要運算思維

對應主要學習表現

學習內容

資D-IV-1資料數位化之原理與方法

資D-IV-2數位資料的表示方法

資D-IV-3資料處理概念與方法

運算思維

資料表示與處理
抽象化
演算法思維

讓學生體驗資料表示的方法與策略

讓學生體會資料如何用符號表示，及有效符號表示對問題解決的影響

運算思維解析

當的資訊科技，並進行有

運算思維導向教材設計



選擇課綱
學習內容

對應重要
運算思維

對應主要
學習表現

- 仔細確認所對應的重要運算思維內涵為何
 - 例如：動態程式規劃的學習需要對應什麼運算思維？若辨識動態程式規劃的演算法樣式是一個重點，則可透過不同的問題(例如旅行銷售員問題與背包問題)所用的動態程式規劃演算法來觀察歸納此一通則

運算思維程式設計教學示例

CT Computational Thinking for K-12 CT內涵 教材 ▾ 課網 ▾ 關於我們

程式參考畫面



生活中的資料檢驗

本專題希望透過商品條碼、身分證字號、銀行帳號、ISBN書碼等這些身邊熟悉，但學生可能從未注意過的小小技術，理解計算機理論中的資料驗證概念。

Scratch 高中 一年級

想想看...



• 這隻恐龍的影像只有黑色和白色，每個像素都可以使用0或1 (1 bit)表示顏色

影像長寬皆為20像素，則儲存這隻恐龍的影像檔案有多大？

$$20 \times 20 \times 1 \div 8 = 50 \text{ byte}$$



照片照「騙」

本專題從照片修圖APP相關議題開始引起學生動機，引導學生思考如何進行影像處理，並在過程中學習資料抽象化，最後產出有序的指令自動進行影像處理。

高中 一年級

<https://ctfork12.ice.ntnu.edu.tw/courseS.html>

運算思維導向程式設計教學示例(國中)

<https://ctfork12.ice.ntnu.edu.tw/courseS.html>

一個PATTERN

一個細胞分裂的樣式 On Scratch1.4



- 重複結構
- 試著做做看
- 可行性與限制

問題

問題討論

實例

教師先以教務處小義工舉例

組自由發想

組說明

歸納此類問題的特點

問與討論解決問題的方法



程式中有四個資料清單

程式執行時存放資料清單

內建的編碼清單



課程進行

遊戲中思考

利用CODE 網站[冰雪奇緣]第4-6關 對應一個正方形、三個正方形、十個正方形



細胞的分裂
《生物樣式的辨識與模擬》

Scratch

本專題配合學生七年級下學期生物課，結合學生對「細胞分裂」的知識概念與迴圈、陣列的運用來進行模擬及重新演繹。

Travelling Salesman Problem
《問題的看見與解決》

Scratch

本專題透過圖形Graph簡化問題描述問題，運用Scratch進行視覺化的思考，試圖發想合理的資料結構並進行演算流程設計，模擬較短路徑的視覺化解題過程。

閃光傳情
《資料表示與驗證》

Scratch

本專題希望讓學生撰寫簡單的模擬程式，可以了解機器如何透過編碼／解碼的方式來表示資料，同時思考在資料傳遞的過程中，應該如何使用簡單的檢查機制以保障資料的正確性。

《幾何繪圖》

Scratch

本專題配合數學課正多邊形單元的外角觀念拆解任務圖形從中發現重複樣式 (finding pattern)，學生透過實作歷程將學會重複結構與函式觀念。

運算思維導向程式設計教學示例(高中)

<https://ctfork12.ice.ntnu.edu.tw/courseS.html>



生活中的資料檢驗

本專題希望透過商品條碼、身分證字號、銀行帳號、ISBN書碼等這些身邊熟悉，但學生可能從未注意過的小小技術，理解計算機理論中的資料驗證概念。



照片照「騙」

本專題從照片修圖APP相關議題開始引起學生動機，引導學生思考如何進行影像處理，並在過程中學習資料抽象化，最後產出有序的指令自動進行影像處理。



便利貼大戰

本專題結合先前已有的二進制表示法以及一維陣列的概念，引入影像編碼的概念，讓學生了解影像的儲存方式，最後透過實作熟悉程式中多維陣列的操作。



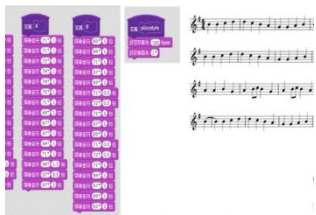
生活中的人工智慧-《chatbot聊天機器人》

本專題引導學生透過生活上的實例與實際使用聊天機器人的經驗，了解電腦模擬人類對話的過程與概念，對人工智慧有初步的認識。從簡單的回覆策略開始逐步加深至機器學習，並融入資料科學的概念

運算思維導向程式設計教學示例(高中)

<https://ctfork12.ice.ntnu.edu.tw/courseS.html>

樂譜上標註樂句積木的名稱



Editor: <https://scratch.mit.edu/projects/editor/>

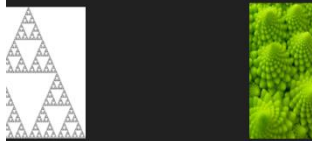
動手玩音樂《我的音樂盒》

Scratch/Python

本專題引導學生思考電腦自動化演奏音樂的創作過程，透過對樂曲中樂句結構的分析，了解模組化程式設計與程式流程的概念。

What is fractal?

碎形 (Fractal) 被定義為「一個零碎的幾何形狀，可以分成數個部分，每一部分 (至少近似地) 是整體縮小後的形狀」，即具有自我相似的性質。



碎形
尋找大自然的密碼

Python Turtle

本專題重點為重複結構與函式的學習。活動中透過重複樣式的辨識與碎形圖形的建構，讓學生體驗分解問題、樣式辨識、模組化與抽象化等運算

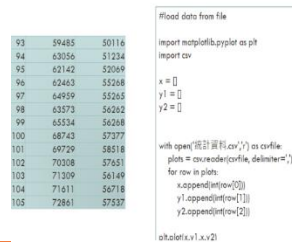
好想抓到所有寶可夢...



寶可夢大師之路

本專題透過Pokémon Go遊戲中常見的問題，引導學生尋找最佳路徑，在此過程中建立將問題抽象化的思維，歸納可能的路徑樣式，最後發展出能動態處理

女生在資訊產業是否會受到不公平待遇



資料分析與視覺化

Python

本專題引導學生探討生活中值得關心的議題(包含教育、經濟、農業、科技、生態、能源與環境等)，運用開放資料集，透過資料分析與視覺化解讀

加入運算思維的開放硬體教學示例

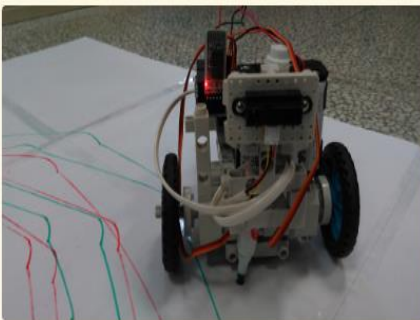
CT Computational Thinking for K-12 CT內涵 教材 ▾ 課綱 ▾ 關於我們



投籃遊戲機

本專題讓學生先觀看影片與作品範例引起動機，接著分析投籃機的功能，設計每項功能的邏輯與規則，再編寫程式模擬遊戲功能，最後選用開放硬體實作投籃遊戲機，以距離感測器感測進球，以七段顯示器顯示分數。

開放硬體 國中 八年級



掃地機器人

本專題讓學生先觀看影片與作品範例引起動機，接著分析掃地機器人的幾種移動路徑，再設計這幾種移動路徑的邏輯與程序，編寫程式模擬之，最後選用機器車實作不同的移動模式。

開放硬體 國中 八年級

<https://ctfork12.ice.ntnu.edu.tw/courseS.html>

開放硬體可訓練的運算思維

抽象化

- 為定義主要概念去識別並萃取相關資訊

問題解析

- 將資料、程序、問題拆解成較小、較容易處理的部分

樣式辨識/一般化

- 在資料中觀察樣式、趨勢或規則
- 產生所觀察樣式的模型、規則、原則或理論以測試預測的結果

資料表示

- 有邏輯的組織和分析資料

演算法設計

- 產出有序指令以解決問題或完成任務

自動化

- 利用電腦或機器重覆任務。

模擬

- 發展模型以模仿真實世界的程序。

運算思維

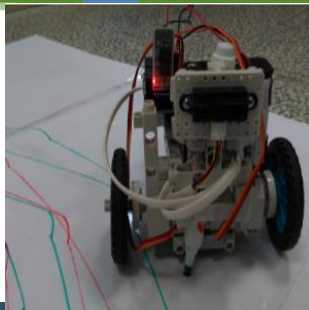
- 學生是否能將資料、程序或問題**拆解**成較小可處理的部分
- 學生是否能**識別並擷取**可表達**主要概念**的重要訊息
- 學生是否能找出資料中可見的**樣式、趨勢與規則**
- 學生是否能產生**可行的程序性指令**以**解決問題**
- 學生是否能讓電腦或機器做**重複性**的工作
- 學生是否能收集與問題相關的可用資料
- 學生是否能藉由尋找資料的樣式規則理解數據
- 學生是否能以**合適的圖表、文字或影像**表達數據
- 學生是否能建立一個可預測結果的**模型**或**規則**

加入運算思維的開放硬體教學示例



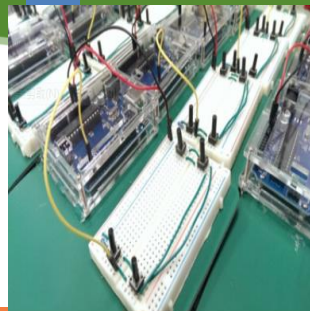
《投籃遊戲機》

☆讓學生先觀看影片與作品範例引起動機，接著分析投籃機的功能，設計每項功能的邏輯與規則，再編寫程式模擬遊戲功能，最後選用開放硬體實作投籃遊戲機，以距離感測器感測進球，以七段顯示器顯示分數。



《掃地機器人》

☆讓學生先觀看影片與作品範例引起動機，接著分析掃地機器人的幾種移動路徑，再設計這幾種移動路徑的邏輯與程序，編寫程式模擬之，最後選用機器車實作不同的移動模式。



《節奏遊戲機》

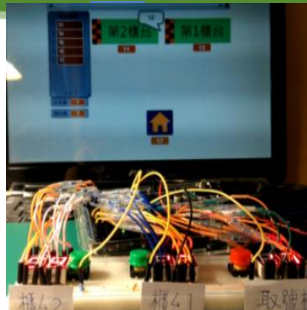
☆讓學生先觀看影片與作品範例引起動機，接著分析節奏遊戲的功能，設計每項功能的邏輯與程序，再編寫程式模擬遊戲的功能，最後選用開放硬體實作節奏遊戲機。使用按鈕，按下一次按鈕表示完成一次打擊。

加入運算思維的開放硬體教學示例



《條碼掃描辨識模擬》

☆先分析條碼掃描辨識功能及步驟，設計每項功能的邏輯與規則，再編寫程式**模擬**該功能，最後用開放硬體實作條碼掃描辨識模擬系統。



《取號叫號機》

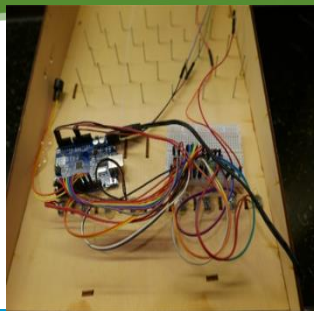
☆讓學生先觀看影片與作品範例引起動機，接著分析取號叫號機的功能，設計每項功能的邏輯與規則，再編寫程式**模擬**取號取號叫號機的功能，最後選用開放硬體實作叫號機或



《電子打地鼠》

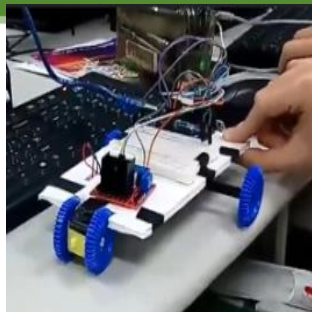
☆以學生的遊戲經驗取材，選自遊樂場常見的遊戲裝置-「打地鼠」遊戲機，玩家在指定的秒數內打中彈出的地鼠可得分，遊戲過程亦可搭配得分音效或配樂，

加入運算思維的開放硬體教學示例



《電子彈珠台》

☆以學生的遊戲經驗取材，選自遊樂場常見的遊戲裝置-「彈珠台」遊戲機，玩家撥動指定數量（或次數）的彈珠，記錄彈珠落點（不同落點的得分不同），遊戲過程亦可搭配得分音效或配樂，遊戲結束後記錄總分及排名。



《智慧生活裝置》

☆運用基本的感測器，設計程式設計及運算思維課程，讓學生可熟悉程式流程及結構，但卻可以產生讓學生有感的物聯網智慧生活裝置，進而引發學生學習興趣。



《智慧避障車》

☆學生將學會拆解的程序。從最終目標智慧避障車往前推，讓學生學會將大任務拆解成一項一項的子功能。整個專題主要是讓學生體會到運算思維中拆解的重要性。把每件小事做好，有意義地組合起來就可以完成一件大事。

加入運算思維的開放硬體教學



教學活動

討論
實例

觀看影片
或實際操作，回答
問題。

解析
問題

解析輸入、處理與
輸出；解析程式的
功能

設計
邏輯

邏輯設計
各項功能

程式
模擬

選擇程式
工具，實
作

開放
硬體

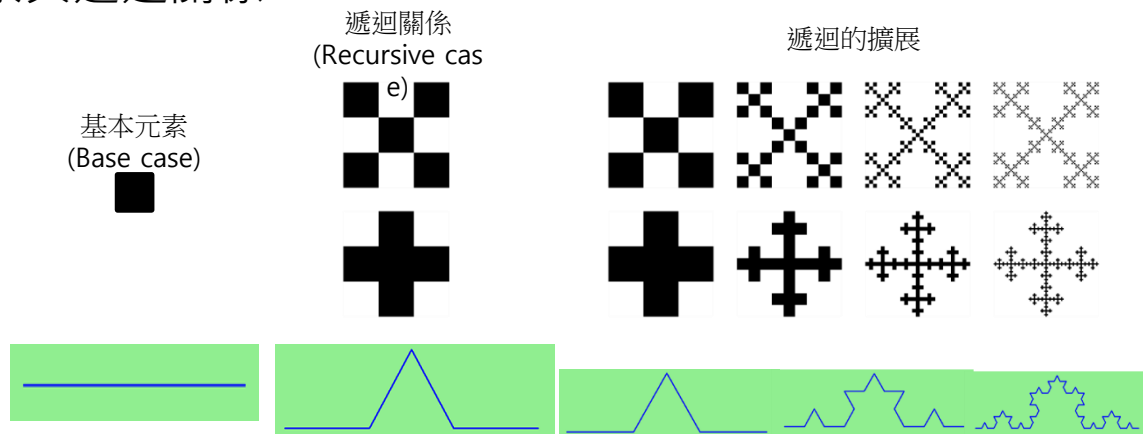
選擇硬體
，實作

運算思維導向教材設計



■ 範例一 樣式辨識(Pattern recognition)、問題解析(Decomposition)

- 透過視覺化圖形的辨識，了解遞迴的樣式(pattern)並透過解析找到基本元素與遞迴關係



運算思維導向教材設計



- 範例一 **樣式辨識(Pattern recognition)**、**問題解析(Decomposition)**
 - 透過觀察與繪製，了解並體驗**遞迴**的樣式(pattern)

程式實作與觀察

- 繪製蕨葉，試修改生長規則另生成不同蕨葉
- 繪製碎形樹，試修改生長規則另生成不同形態的樹



運算思維導向教材設計



■ 範例一 樣式辨識(Pattern recognition)、問題解析(Decomposition) — 循序引導與體驗運算思維

一、分組活動：尋找大自然存在的神秘圖形規律



- 請討論以上圖形的特徵與規律(至少寫 3 個)。

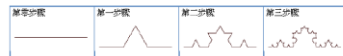
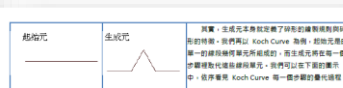
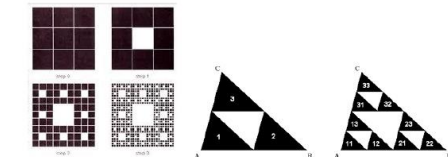
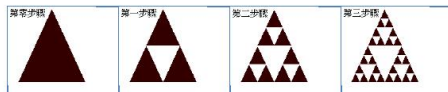
- 請整理寫下各組分享的圖形特徵與規律

一、起始元與生成元疊代法

在繪製碎形的方法中，「起始元與生成元疊代法 (Generator Iteration Method)」是較為直觀與容易操作的，通常用來繪製「完全自我相似」(Strict Self-Similarity) 的碎形。典型的碎形大多是採取這種疊代法，以這種方法製作碎形時，必須指定起始元 (Initiator) 與生成元 (Generator)，其意義分述如下：

* 起始元：是碎形一開始的圖形，起始元是由單一或幾種自我相似的幾何單元(例如線段、三角形或矩形……等等)所組成。

* 生成元：是起始元中的每一圖自我相似的幾何單元下一次疊代的圖形。



任務：建構碎形

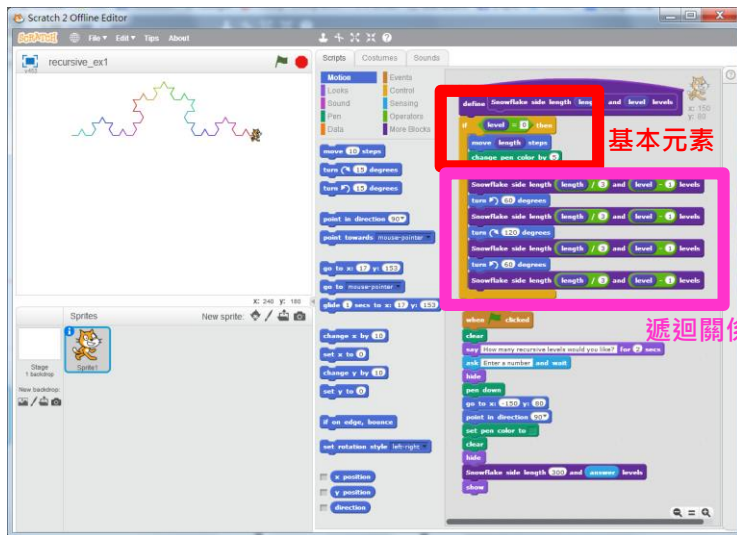
1、設計碎形繪製規則



2、依碎形繪製規則疊代繪製至第 4 代碎形



運算思維導向教材設計



```
def koch(t, order, size):
```

```
    if order == 0:
```

```
        t.forward(size)
```

```
    else:
```

```
        koch(t, order-1, size/3)
```

```
        t.left(60)
```

```
        koch(t, order-1, size/3)
```

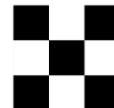
```
        t.right(120)
```

```
        koch(t, order-1, size/3)
```

```
        t.left(60)
```

```
        koch(t, order-1, size/3)
```

遞迴關係



運算思維導向教材設計



範例二 抽象化(Abstraction)、資料表示(Data representation)

■ Data compression (Exploring Computational Thinking, Google for Education)

Activity 1: Smallest amount of data necessary (20 minutes)

Activity Overview: In this activity, students will use pattern recognition and data analysis to determine what is the smallest amount of data necessary to communicate to another person information such as a word, phrase, shape, or visual scene.

Activity:

1. Pair students into groups of two.
2. Have one student think of a word or phrase and write down one of the letters from that word, wait a second, then write another letter from that word, omitting some of the letters. All of the letters should be placed in the order of where they belong e.g. If I were thinking of "HELLO WORLD," I might write down "L," then later write down "LL," and eventually it might look like "H LLO W R D."
3. Have the second student try to guess the word or phrase as quickly as possible.
4. After the second student correctly guesses the word, students should switch roles.
5. After a couple of turns, ask students to try and do the same activity using a shape or a visual scene like "student walking to school," where each student draws the shape or visual one line or curve at a time.

Q1: On average how much information (turns/steps) from the first person was necessary to guess

Word	Phrase	Shape	Visual

運算思維導向教材設計



- 範例二 抽象化(Abstraction)、資料表示(Data representation)
 - 教出壓縮策略與資料表示的精神
 - 用最少位元數精確表達資訊
 - 能夠解碼(解壓縮)

運算思維導向教材設計



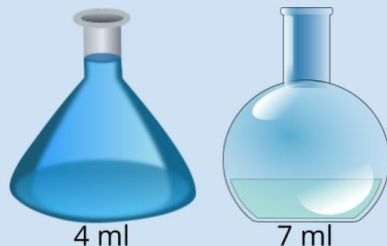
範例三 問題解析(Decomposition)、演算法思維(Algorithmic thinking)

■ Water Water Everywhere! (Exploring Computational Thinking, Google for Education)

Activity:

Ask students to solve the following problem:

You've run into a predicament. You want to do a science experiment. Given two containers, one that can hold 4 millilitres and one that can hold 7 millilitres, how can you get exactly one of these containers to hold exactly 5 millilitres?

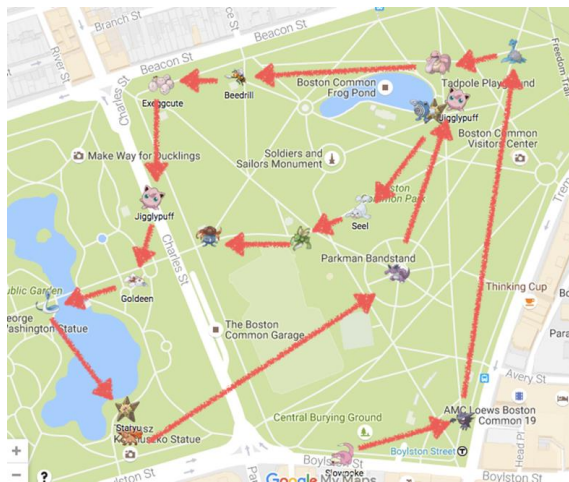


Instruction	Quantity in A (max 4)	Quantity in B (max 7)
START	0	0
Fill A	4	0
A -> B	0	4
Fill A	4	4
A -> B	1	7
Empty B	1	0
A -> B	0	1
Fill A	4	1
A -> B	0	5

運算思維導向教材設計



- 範例四 演算法設計(Algorithm design)
 - 透過視覺化的地圖與路徑，了解**最小成本的路徑**的演算法設計

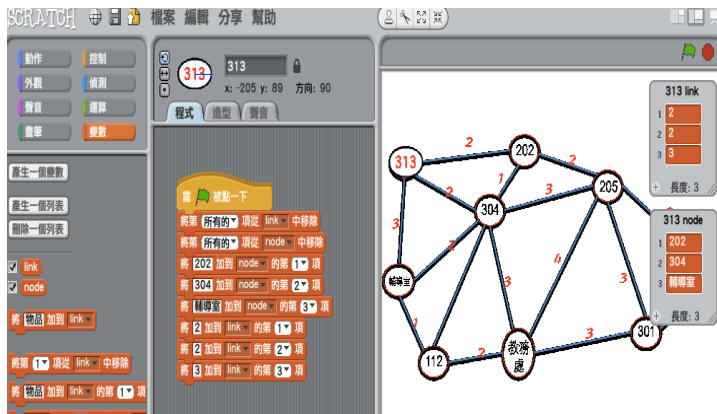


運算思維導向教材設計



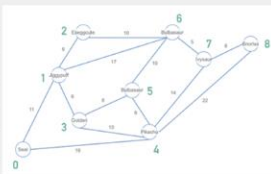
■ 範例五 資料表示(Data representation)

- 透過視覺化的地圖、花費、與陣列的對應，了解陣列資料結構與圖 (Graph) 的資料表示概念



如何將圖Graph儲存下來？

	[0]	[1]	[2]	[3]	[4]	[5]	[6]	[7]	[8]
[0]		11			16				
[1]	11		6	6			17		
[2]	6						10		
[3]	6				10	8			
[4]	16			10		6	14	22	
[5]				8	6		10		
[6]	17	10					10	5	
[7]					14	5		6	
[8]					22		6		



運算思維導向教材設計

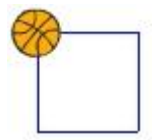


■ 範例六 樣式辨識(Pattern recognition)、模擬(Simulation)

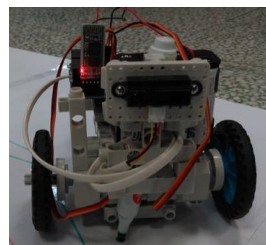
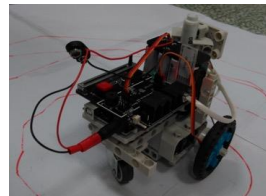
觀察實體掃地機器人



Scratch 軟體模擬



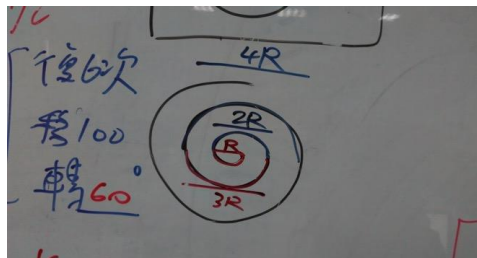
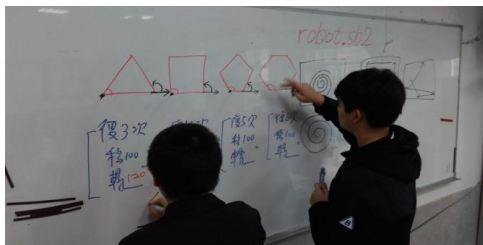
實作機器車螺旋式移動



運算思維導向教材設計



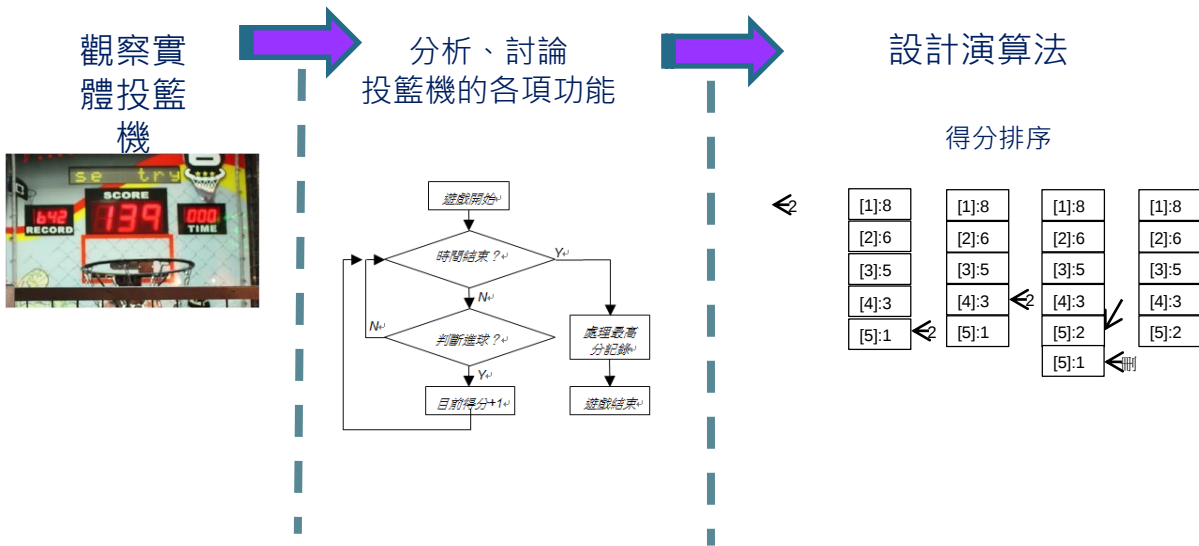
■ 範例六 樣式辨識(Pattern recognition) 、模擬(Simulation)



運算思維導向教材設計



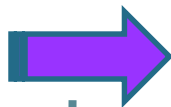
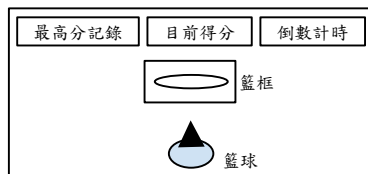
範例七 投籃遊戲機學習--問題分解 (Decomposition) 、演算法設計 (Algorithm Design)



運算思維導向教材設計



Scratch 軟體模擬



實作投籃遊戲機



運算思維導向教材設計

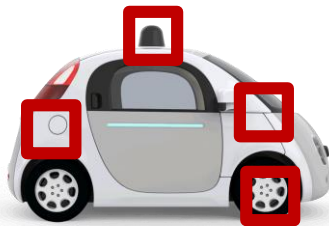


■ 範例八 智慧避障車學習問題分解 (Decomposition)、模擬 (Simulation)

觀看影片



分析、討論
避障車的各項功能



任務卡

請讓你的智慧自走車在遇到障礙物時停下來，避免撞擊到障礙物，然後右轉或左轉，繼續前進。

設計演算法

前進 移動 10 步

遇到障礙物

停下

轉彎

前進

移動 10 步

將 轉 90 度

移動 10 步



動作指令:

1. 啟動後，將「前進」指令拖到「移動」指令後面。
2. 選擇「移動」指令，將「距離」設定為 10。
3. 選擇「移動」指令，將「角度」設定為 90。
4. 選擇「移動」指令，將「距離」設定為 10。

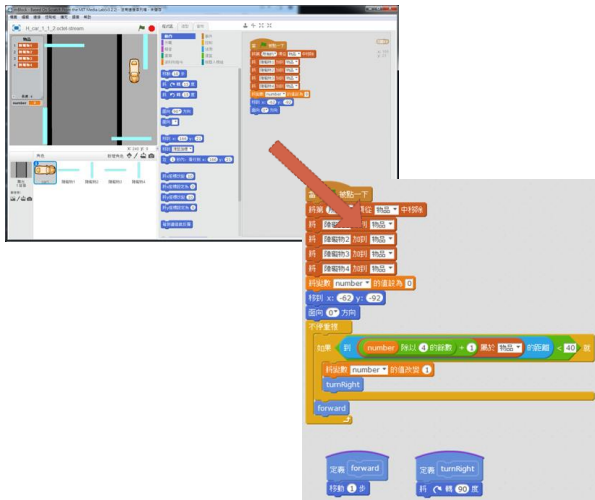
參考指令表:

指令	說明
1. 啟動後，將「前進」指令拖到「移動」指令後面。	參考指令表中的「移動」指令，選擇「前進」。
2. 選擇「移動」指令，將「距離」設定為 10。	選擇「移動」指令，將「距離」設定為 10。
3. 選擇「移動」指令，將「角度」設定為 90。	選擇「移動」指令，將「角度」設定為 90。
4. 選擇「移動」指令，將「距離」設定為 10。	選擇「移動」指令，將「距離」設定為 10。

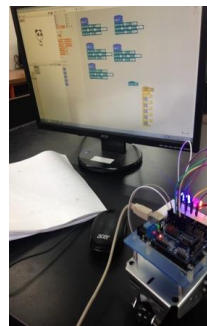
運算思維導向教材設計



mBlock 軟體模擬



實作智慧避障車



運算思維導向教材設計

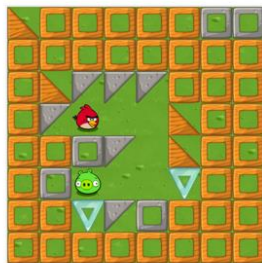


■ 範例九 樣式辨識(Pattern recognition)

80508001E_豐佳燕

二、迴圈-重複循環

1



(1) 要如何找到 ？

用箭頭畫出 路徑(畫在左圖)。



(2) 用積木來表示就像下圖，你發現有重複的地方嗎？試著找出重複指令的區塊，並把它「圈起來」，有()個區塊？



運算思維導向教材設計



- 範例十、照片照騙-資料表示 (Data representation) 、演算法思維 (Algorithmic thinking)
 - 透過影像處理，學習二維陣列的存取與程式設計

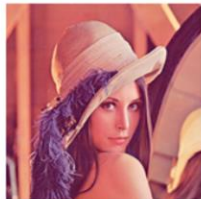
美肌效果



其他特效濾鏡



負片濾鏡功能



顏色顛倒

$\text{invert}(R,G,B)=[255-R,255-G,255-B]$



怎麼告訴別人哪些格子是黑色？



想想海戰棋



對戰中哪A1、B5、分個代表什麼



運算思維導向教材設計



■ 範例十一 資料表示(Data representation)

- 學習影像的資料表示、取樣、與量化的概念

便利貼大戰



便利貼大戰



運算思維導向教材設計



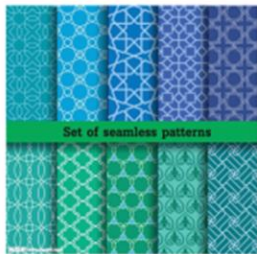
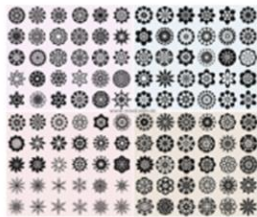
■ 範例十二 樣式辨識(Pattern recognition)

- 透過重複樣式的觀察，學習迴圈的概念

花布設計創作說明

運用**程式**完成**具美感**的花布設計與創作

作品需包含**3**種以上幾何圖案的**規律排列**與組合



- 能夠解構圖形並尋找重複性**規律**
- 能善用自定**函式**與**迴圈**減少重複程式碼
- 能運用變數與運算式進行座標推算、圖形變化**規律**運算與RGB顏色數值運算

運算思維導向教材設計



■ 範例十三 樣式辨識(Pattern recognition)、問題解析(Decomposition)

— 透過聽覺化音樂的辨識，了解**模組化**的樣式(pattern)

Define A

- 彈奏音符 71 1 拍
- 彈奏音符 71 1 拍
- 彈奏音符 72 1 拍
- 彈奏音符 74 1 拍
- 彈奏音符 74 1 拍
- 彈奏音符 72 1 拍
- 彈奏音符 71 1 拍
- 彈奏音符 69 1 拍
- 彈奏音符 67 1 拍
- 彈奏音符 67 1 拍
- 彈奏音符 69 1 拍
- 彈奏音符 71 1 拍
- 彈奏音符 71 1.5 拍
- 彈奏音符 69 0.5 拍
- 彈奏音符 69 2 拍

Define A'

- 彈奏音符 71 1 拍
- 彈奏音符 71 1 拍
- 彈奏音符 72 1 拍
- 彈奏音符 74 1 拍
- 彈奏音符 74 1 拍
- 彈奏音符 72 1 拍
- 彈奏音符 71 1 拍
- 彈奏音符 69 1 拍
- 彈奏音符 67 1 拍
- 彈奏音符 67 1 拍
- 彈奏音符 69 1 拍
- 彈奏音符 71 1 拍
- 彈奏音符 69 1.5 拍
- 彈奏音符 67 0.5 拍
- 彈奏音符 67 2 拍

Define B

- 彈奏音符 69 1 拍
- 彈奏音符 69 1 拍
- 彈奏音符 71 1 拍
- 彈奏音符 67 1 拍
- 彈奏音符 69 1 拍
- 彈奏音符 71 0.5 拍
- 彈奏音符 72 0.5 拍
- 彈奏音符 71 1 拍
- 彈奏音符 67 1 拍
- 彈奏音符 69 1 拍
- 彈奏音符 71 0.5 拍
- 彈奏音符 72 0.5 拍
- 彈奏音符 71 1 拍
- 彈奏音符 69 1 拍
- 彈奏音符 67 1 拍
- 彈奏音符 69 1 拍
- 彈奏音符 62 2 拍

Define playstyle

- 設定節奏為 120 bpm
- 設定樂器為 1

A

A'

B

A'

當按下 空白鍵 鍵

playstyle

A

A'

B

A'

運算思維導向教材設計



■ 範例十三 樣式辨識(Pattern recognition)

- 一 透過聽覺化音樂的辨識，了解**模組化**的樣式(pattern)，進一步進行音樂創作

童話

光良詞/曲/演唱

原來T小邪 0 5 1 7 | 1 - 5 - 3 - | 0 5 1 7 | 1 - 5 - 3 - | 0 5 1 7 |
忘了你 多久 再沒聽到你 對我說

1 - 1 | 1 6 8 5 3 - - - | 0 5 1 7 | 1 - 5 - | 0 5 3 2 |
你 最 愛的那首 我想了 很久 謝謝 你

2 | 1 - | 0 5 1 7 | 1 6 - 8 | 8 1 6 8 5 - - - | 0 2 2 4 |
我不過 我之 後 除了眼淚 你還看

4 3 3 - | 0 3 3 7 | 2 1 1 7 1 1 7 1 | 4 - 9 | 5 4 3 2 |
若我說 承諾 都是騙人的 我不可 能 是 你的王子

7 2 - - - | 0 2 2 4 | 4 3 3 - | 0 3 3 7 | 7 6 7 1 | 1 2 1 |
也許你 不原諒 我昨天 的昨天

6 - - 6 | 6 5 5 3 - - - | 0 5 5 4 | 3 - 3 4 3 - 3 4 |
我願變 成 童話 你愛

3 4 3 2 1 | 1 3 5 | 6 - 6 6 5 2 2 4 3 - - - | 0 1 3 5 |
的承諾 到底 與現實 于 雙城 誰守誰陪 如雲和

6 - 6 6 5 3 2 2 4 | 3 4 3 2 1 - 1 3 | 6 - 6 1 | 1 - 7 - |
你 相信 們會像 童話故事 幸福 和 快樂 是 站

1 - - - 7 | - - - | 6 5 5 小邪 0 2 2 4 | 4 3 3 | 0 3 3 7 |
你笑著 對我說 童話那

2 1 1 7 1 1 7 1 | 4 - 9 | 5 4 3 2 | 2 - - - | 0 2 2 4 |
都是騙人的 我不可 能 是 你的王子

4 3 3 - | 0 3 3 7 | 7 6 7 1 | 1 2 1 | 6 - 6 | 6 5 5 3 -
不原諒 從你眼 裡我以後 我的天 空 是 星星站了

主歌Verse

副歌Chorus



my composition

Title: _____

Composer: _____

A 0 0 0 0 | 0 0 0 0 | 0 0 0 0 | 0 0 0 0 |

B 0 0 0 0 | 0 0 0 0 | 0 0 0 0 | 0 0 0 0 |

A 0 0 0 0 | 0 0 0 0 | 0 0 0 0 | 0 0 0 0 |

© music 1. Weirong 2014

運算思維導向教材設計



範例十四 資料表示與處理(Data representation and processing)

科技領域/資訊科技學習重點		
學習表現	學習內容	科技領域核心素養
資t-IV-4能應用 運算思維 解析問題。 資p-IV-1能選用適當的資訊科技 組織思維 ，並進行 有效的表達 。 資p-IV-2能利用資訊科技與他人進行 有效的互動 。 資p-IV-3能有系統地 整理數位資源 。	資D-IV-1 資料數位化 之原理與方法 資D-IV-2數位資料的 表示方法 資D-IV-3 資料處理 概念與方法 資T-IV-1資料處理應用專題 資T-IV-2資訊科技應用專題 資A-IV-1演算法基本概念 資A-IV-2陣列資料結構的概念與應用 資A-IV-3基本演算法的介紹 資P-IV-1程式語言基本概念、功能及應用 資P-IV-2結構化程式設計 資P-IV-3陣列程式設計實作 資P-IV-4模組化程式設計的概念 資P-IV-5模組化程式設計與問題解決實作 資T-IV-2資訊科技應用專題	科-J-B1 具備運用各種 科技符號 與 運算思維表達溝通 的素養，並理解日常生活中科技與運算的基本概念，應用於日常生活。 科S-U-B1 具備精確掌握各類 科技符號 與 運算思維 表達的能力，能 有效進行思想與經驗的表達 ，與他人溝通並解決問題。

運算思維導向教材設計

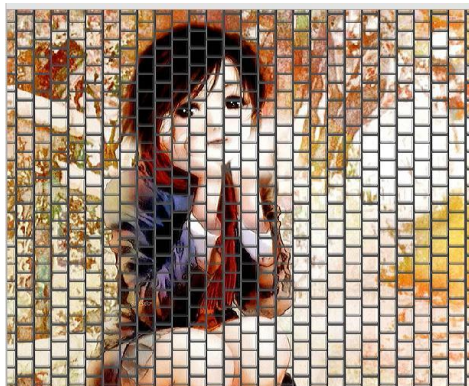


□ 範例十四 資料表示與處理、演算法思維

□ 資料壓縮

□ 活動 拼出間諜的長相

- 你拿到了敵方陣營的間諜拼圖，不幸被打散了，由於捉出間諜之時間有限，只能以十片拼圖將敵方陣營的間諜照片拼起來，你會選哪幾片拼圖拼出最像的間諜？為什麼？



運算思維導向教材設計



□活動 黑白旗

分成兩組，每組再分為發訊者與收訊者，各組先根據字庫，設計好編碼的原則，然後各組發訊者與收訊者帶開，由發訊者用黑白旗發出老師指定的訊息，收訊者負責解碼，看各組收訊者誰的解出來的訊號是正確的。

■ 字庫: ABEFILMNOTUVXYZ

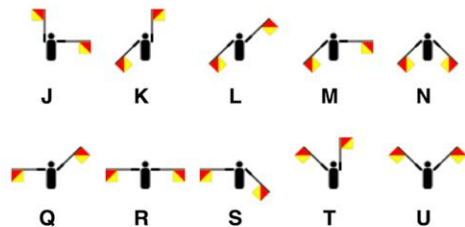
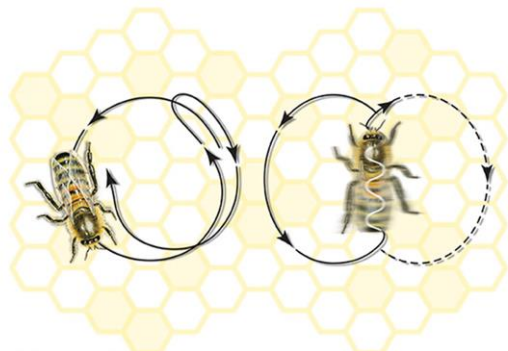


運算思維導向教材設計



■ 範例十五、閃光傳情-資料表示 (Data representation)

- 透過旗語遊戲與蜜蜂的舞蹈，體驗資料表示對資訊傳遞的重要性，並理解資料表示的策略



討論



- 以上教材，不論插不插電，是否都有電腦科學？
- 有問題解決歷程的教學就是有運算思維嗎？
- 什麼樣的教材才能符合運算思維導向的教學？

Barefoot

 **Computing** at School

Recommended for
ages 5-7

World Map Logic -

Logic and Programming

Logic



I know it's raining



I know if I wear
my coat I do not
get wet

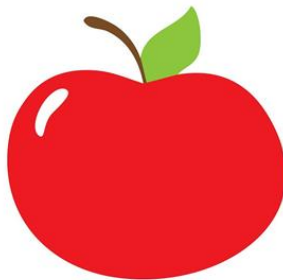


I predict if I wear
my raincoat today
I will not get wet

Logic



I know I am
hungry



I know if I eat
an apple I am
not hungry



I predict if I eat an
apple I won't be hu
ngry anymore

Logic?



I know I am
hungry



I know if I put
my coat on I will
not get wet



I predict if I put my
coat on I will not
hungry anymore

Today we are learning about...



- I can predict what a program will do
- I can explain why I think this



Logic





World Map Bot Sim

World Map Bot Sim

program

(empty)

length: 0

click here to move to x

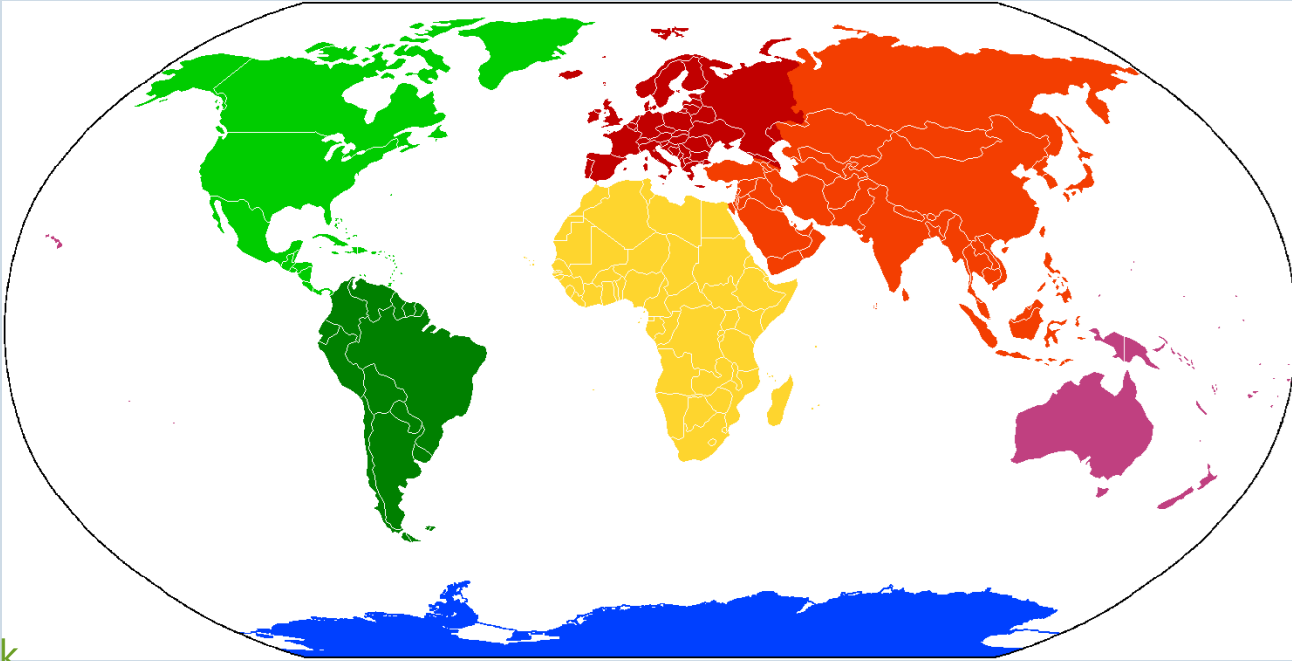
Go

Clear

Pause

The main interface of the World Map Bot Sim. It features a large world map with a grid overlay. The continents are colored: North and South America are green, Africa is yellow, Europe and Asia are red, and Australia is purple. A small yellow robot is positioned on the Atlantic Ocean. A compass rose is in the bottom right corner of the map. Below the map is a control panel with a 'Go' button (a green circle with the word 'Go' in white) and four orange arrows pointing up, down, left, and right. To the left of the 'Go' button is a 'Clear' button (a blue oval with the word 'Clear' in white) and to the right is a 'Pause' button (a blue oval with the word 'Pause' in white). On the left side of the interface is a grey rectangular area labeled 'program' at the top. Inside this area, the text '(empty)' is displayed. At the bottom left of this area is a plus sign icon and the text 'length: 0'. Below the 'program' area is a black rectangular box with the text 'click here to move to x' in white. The entire interface is enclosed in a window with a title bar and standard window controls.

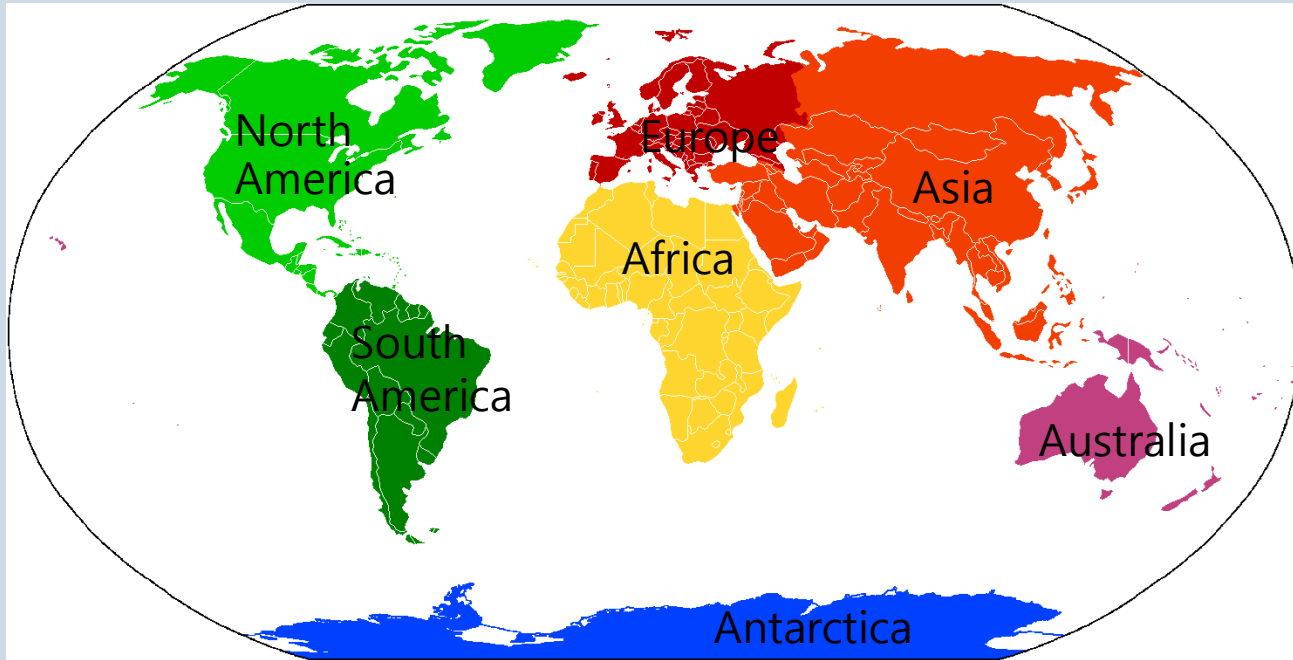
What is this?



[Song](#)

[Continents link](#)

What is this?





Europe

North America

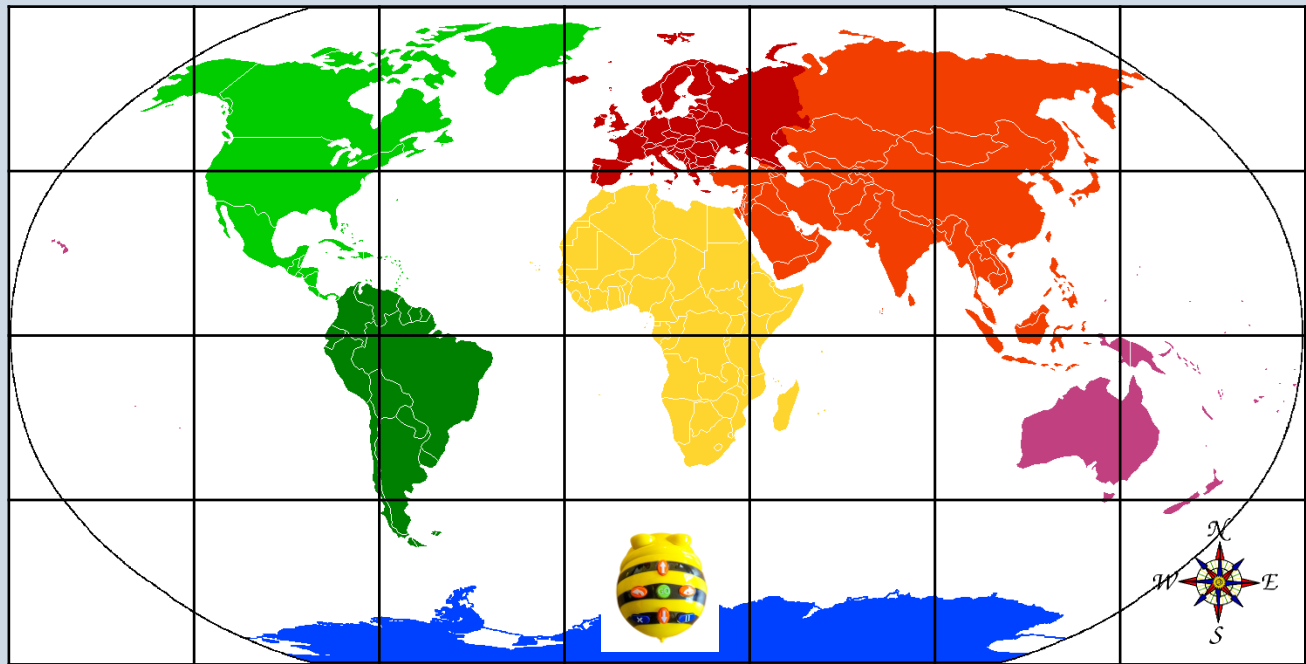
South America

Asia

Australia

Africa

Antarctica



Colours	Red	Pink	Dark Green
Yellow	Orange	Blue	Light Green



Europe

North America

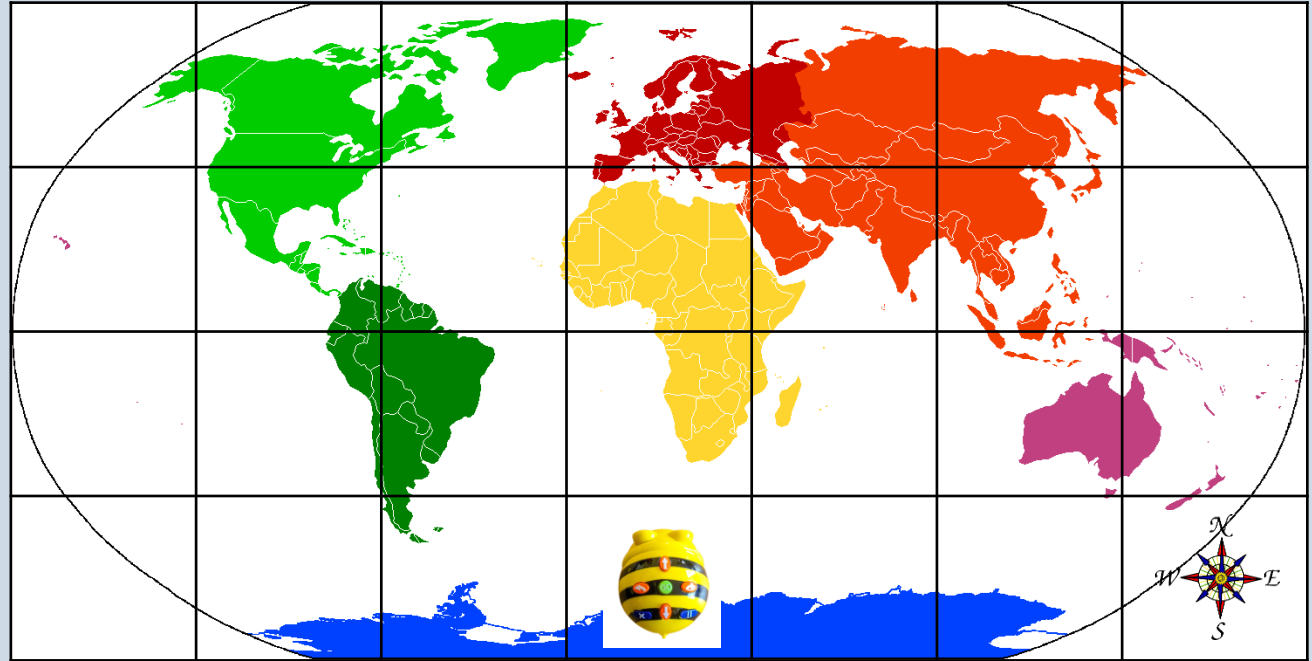
South America

Asia

Australia

Africa

Antarctica



Colours	Red	Pink	Dark Green
Yellow	Orange	Blue	Light Green



Europe

North America

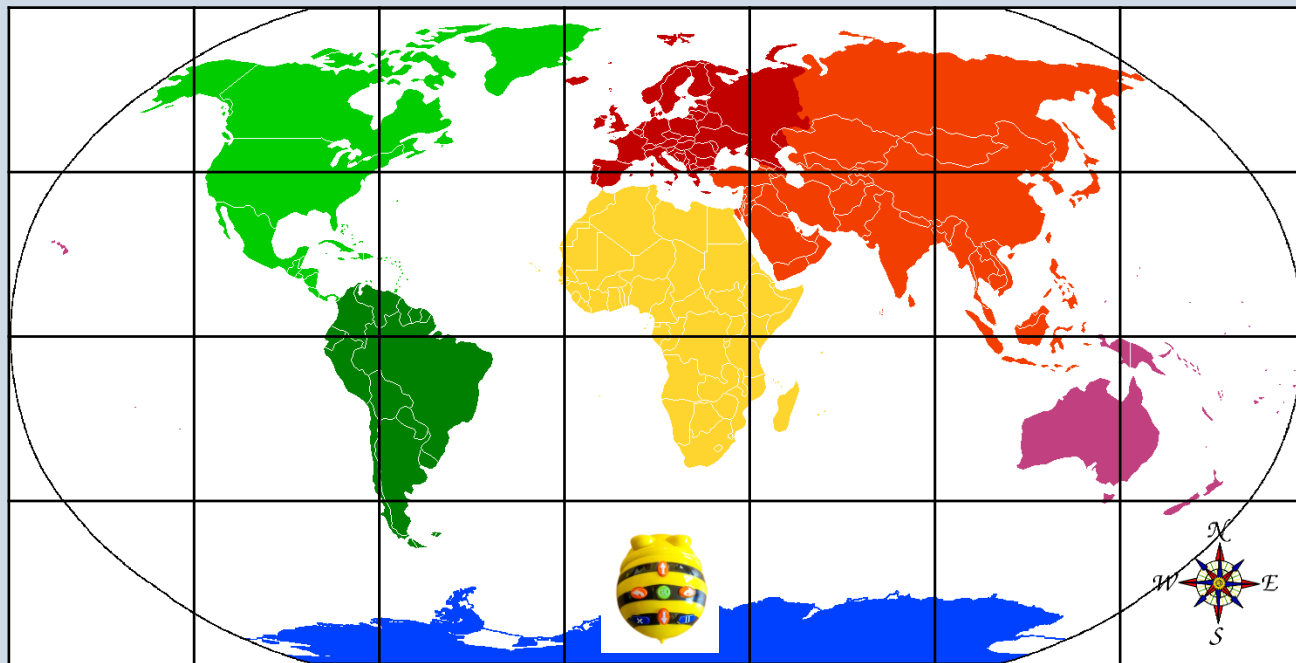
South America

Asia

Australia

Africa

Antarctica



Colours	Red	Pink	Dark Green
Yellow	Orange	Blue	Light Green



Europe

North America

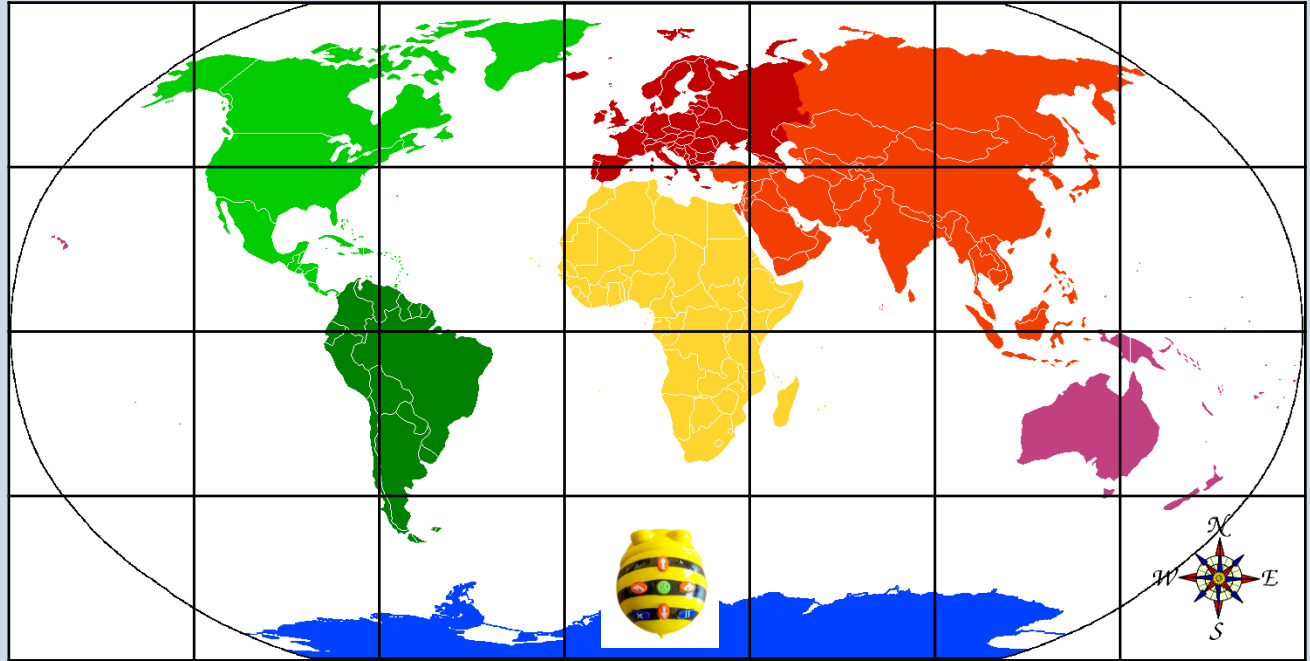
South America

Asia

Australia

Africa

Antarctica



Colours	Red	Pink	Dark Green
Yellow	Orange	Blue	Light Green



Europe

North America

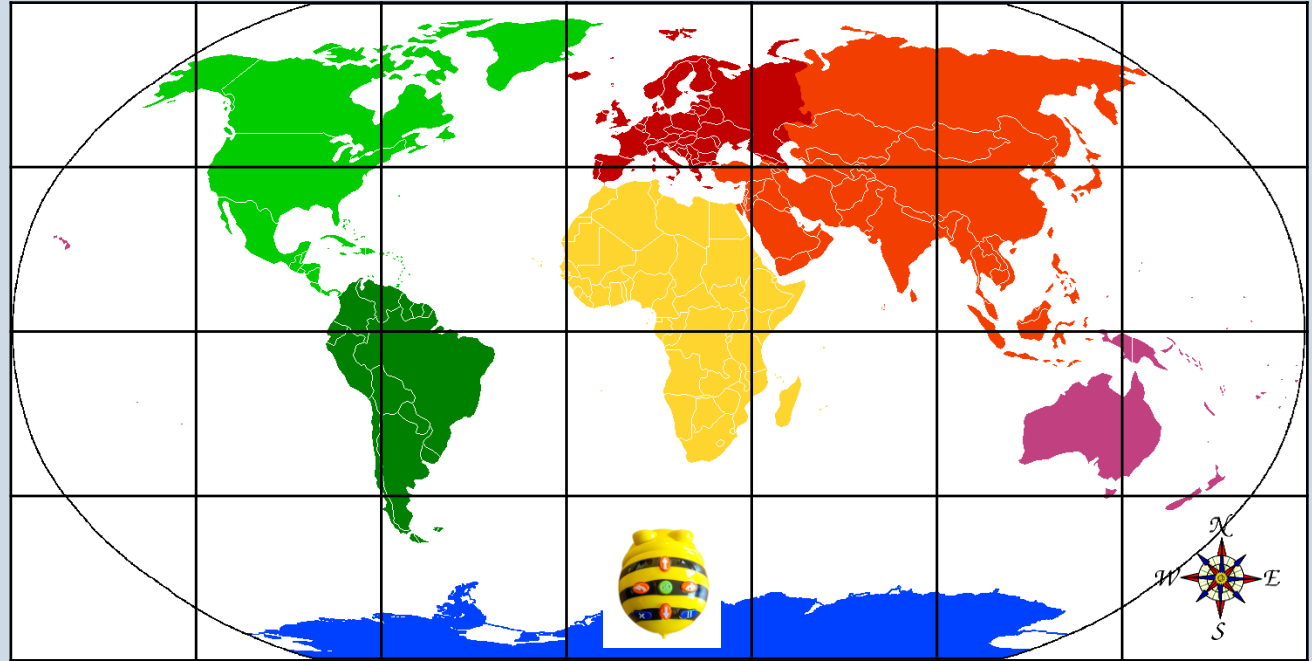
South America

Asia

Australia

Africa

Antarctica



Colours	Red	Pink	Dark Green
Yellow	Orange	Blue	Light Green

Europe

North America

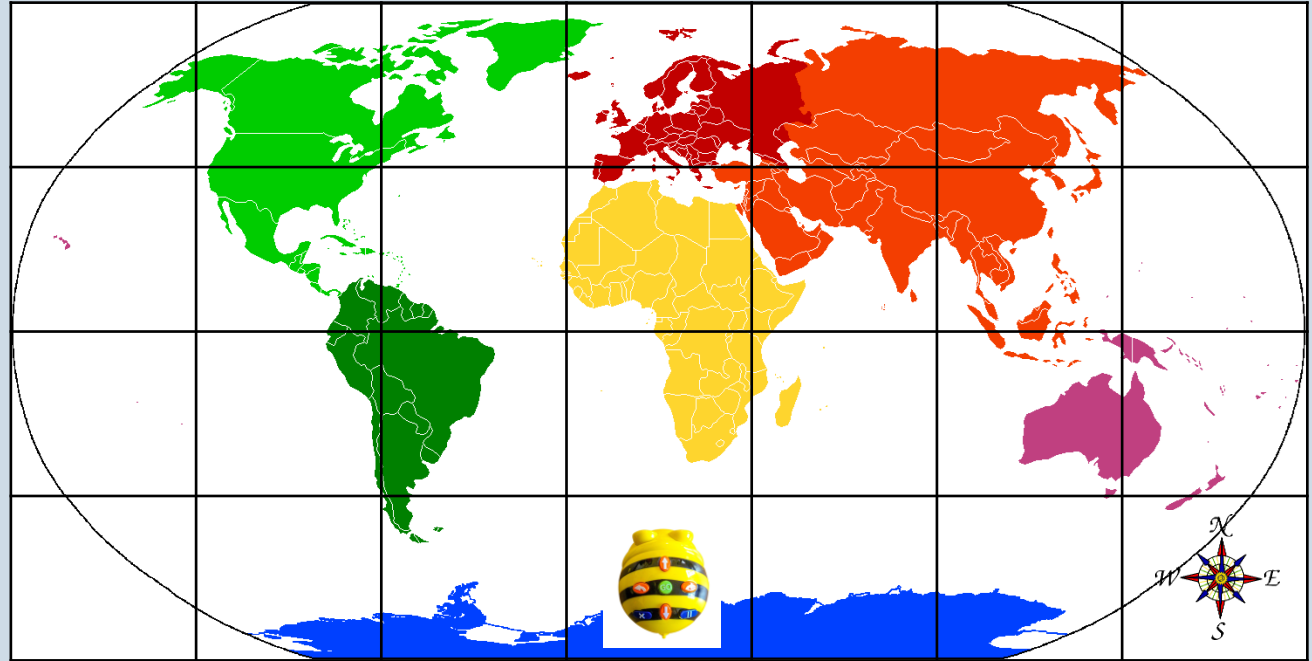
South America

Asia

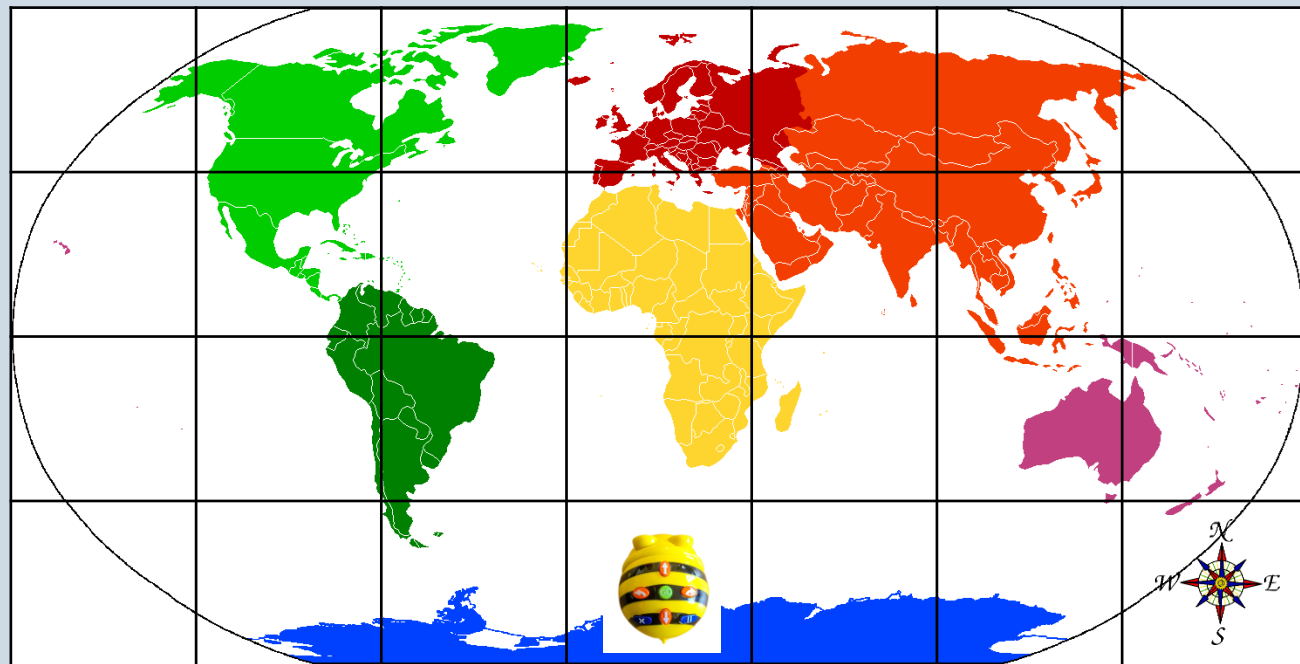
Australia

Africa

Antarctica



Colours	Red	Pink	Dark Green
Yellow	Orange	Blue	Light Green



討論



- 從這份教材學生可以學到什麼？

Barefoot

 Computing at School

Recommended for
ages 7-11

Shapes and Crystal Flowers

Use repetition in programs

Blocks Palette

Scripting Area

Stage

Helpsheet

The screenshot shows the Scratch IDE interface with three blue arrows pointing to specific areas:

- An arrow points from the "Blocks Palette" label to the left sidebar containing various block categories like Motion, Looks, Sound, Events, Control, Sensing, Operators, Variables, and My Blocks. The "Pen" category is currently selected.
- An arrow points from the "Scripting Area" label to the central workspace where scripts are assembled. It shows a sequence of blocks: a blue "move 10 steps" block, a blue "turn 15 degrees" block, an orange "repeat 10" loop containing a blue "move 10 steps" block, and several green "pen" blocks (pen down, set pen color to red, erase all, change pen color by 10, set pen color to 50, change pen size by 1, set pen size to 1).
- An arrow points from the "Stage" label to the right-hand area, which includes the "Sprite1" selection, the "Stage" area with "x" and "y" coordinates, and the "Backdrops" area.

At the bottom of the interface, there is a "Backpack" section.

Which blocks come from which script areas?
Hint: Look at the colour!

Helpsheet 2

The image shows the Scratch programming environment. The top bar includes the Scratch logo, a dropdown menu, and buttons for File, Edit, Tutorials, and an untitled project named 'Untitled-6'. There are also buttons for Share, See Project Page, and Save. The left sidebar shows the 'Code' tab selected, with categories for Motion, Looks, Sound, Events, Control, Sensing, Operators, Variables, and My Blocks. The 'Motion' category is expanded, showing various motion blocks. The main workspace contains a sequence of blocks: 'erase all', 'pen down', 'set pen color to' (purple), followed by a loop of 'move 100 steps', 'turn 90 degrees', 'move 100 steps', 'turn 90 degrees', 'move 100 steps', 'turn 90 degrees', and 'move 100 steps'. A yellow note block is attached to the 'erase all' block, containing the text: 'A sequence based program to draw a square'. A blue callout box with the text 'Can you use the repeat block to make this code draw a square?' points to the sequence of blocks. Below the callout box, a 'repeat 10' block is shown, indicating the goal of the exercise.

Scratch

File Edit Tutorials Untitled-6 Share See Project Page Save

Code Costumes Sounds

Motion

move 10 steps

turn 15 degrees

turn 15 degrees

go to random position

go to x: 100 y: -100

glide 1 secs to random position

glide 1 secs to x: 100 y: -100

point in direction 90

point towards mouse-pointer

Looks

Sound

Events

Control

Sensing

Operators

Variables

My Blocks

Pen

erase all

pen down

set pen color to

move 100 steps

turn 90 degrees

move 100 steps

turn 90 degrees

move 100 steps

turn 90 degrees

move 100 steps

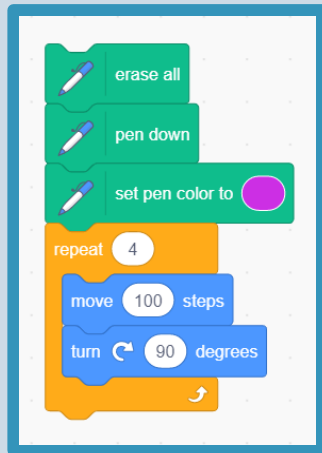
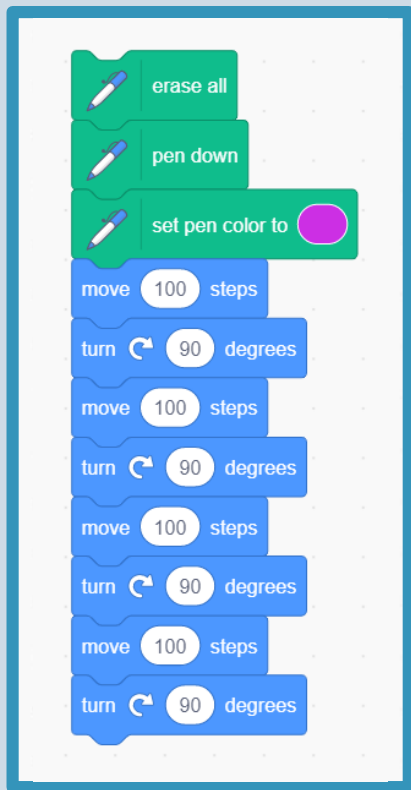
turn 90 degrees

A sequence based program to draw a square

Can you use the repeat block to make this code draw a square?

repeat 10

What is the difference?



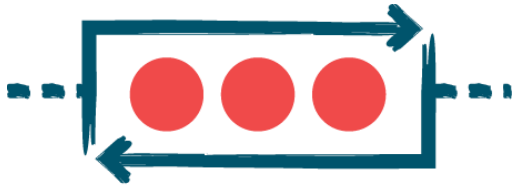
Challenge

**Why are repeats
(or loops) useful
when we program?**

Today we are learning about...



Programs & Repetition:



Repetition

- I can write a program that uses a repeat command
- I can explain what the repeats do in my program

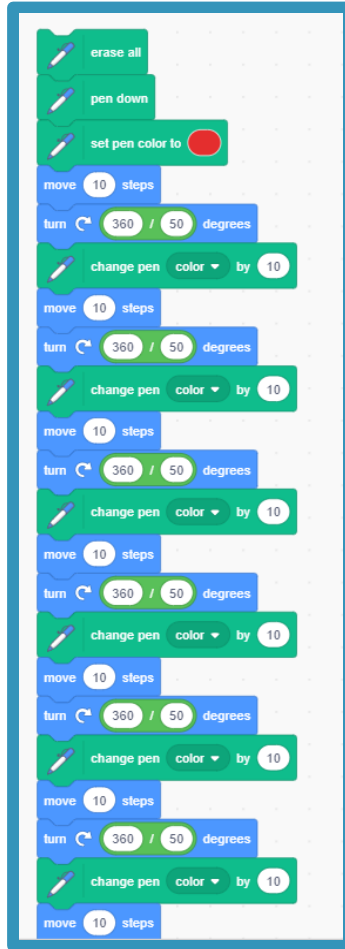


Images from Pixabay CC0 Public Domain

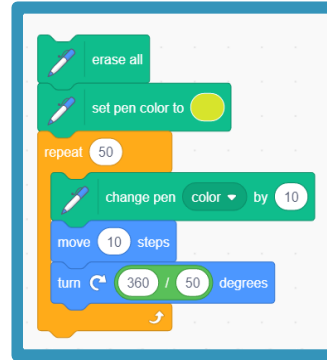


Images from Pixabay CC0 Public Domain

Challenge

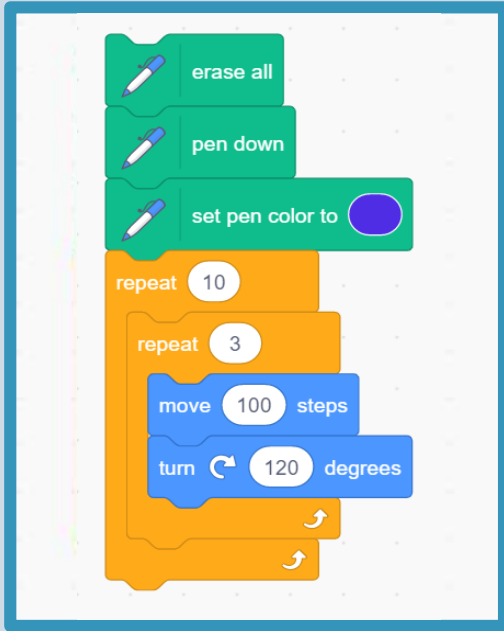


Drawing a Pentacontagon - 50 sided shape
Can you see a mistake?

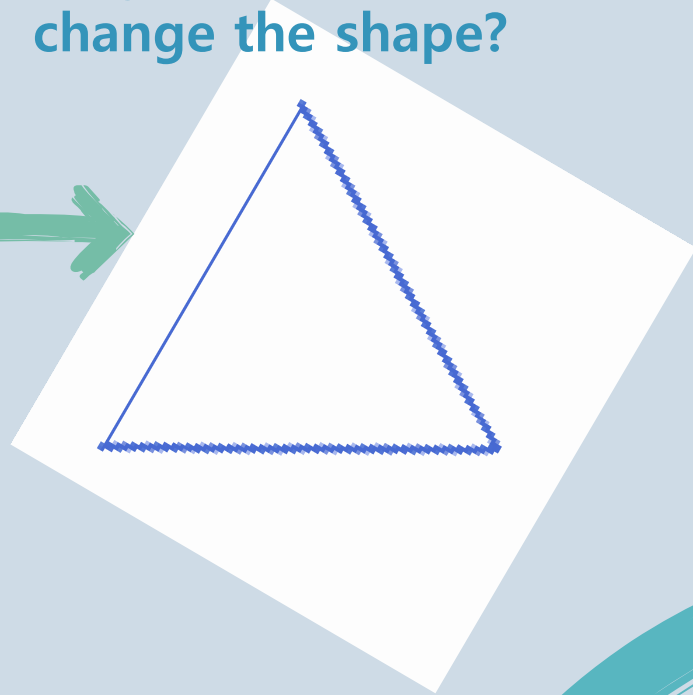


Why are repeats (or loops)
useful when we program?

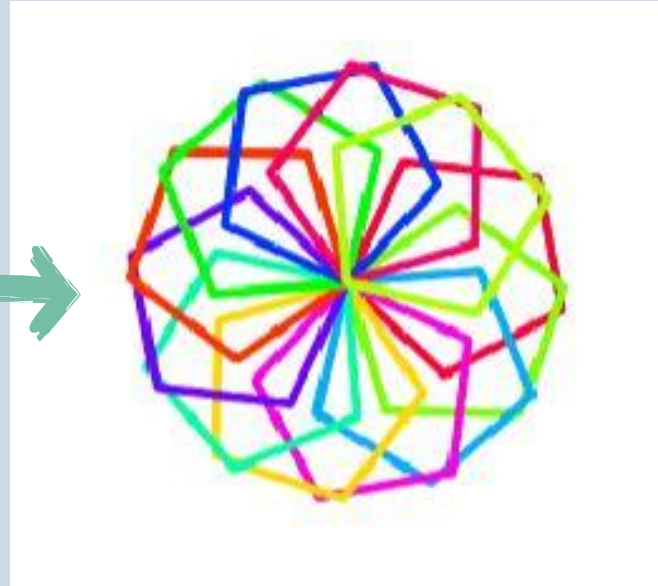
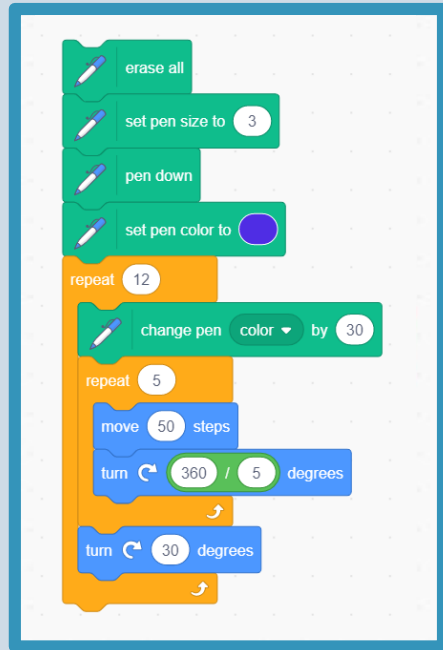
Nested Loop



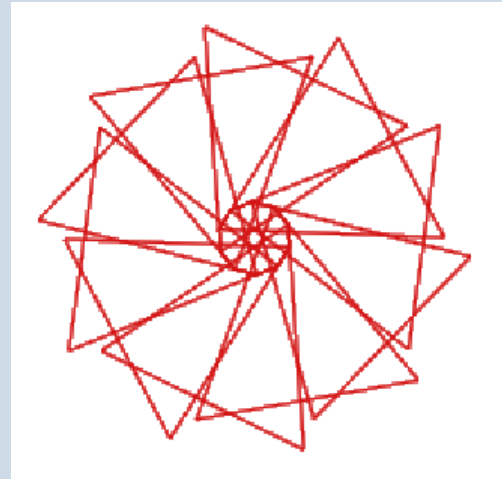
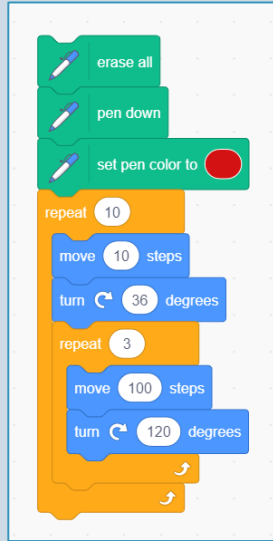
Why does this not change the shape?



Nested Loop

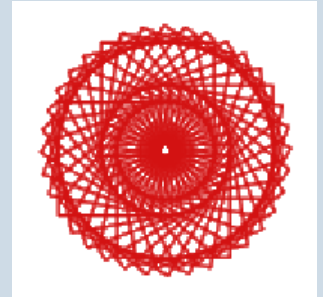
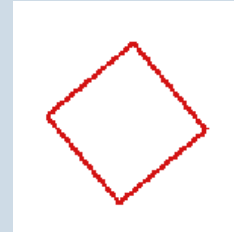
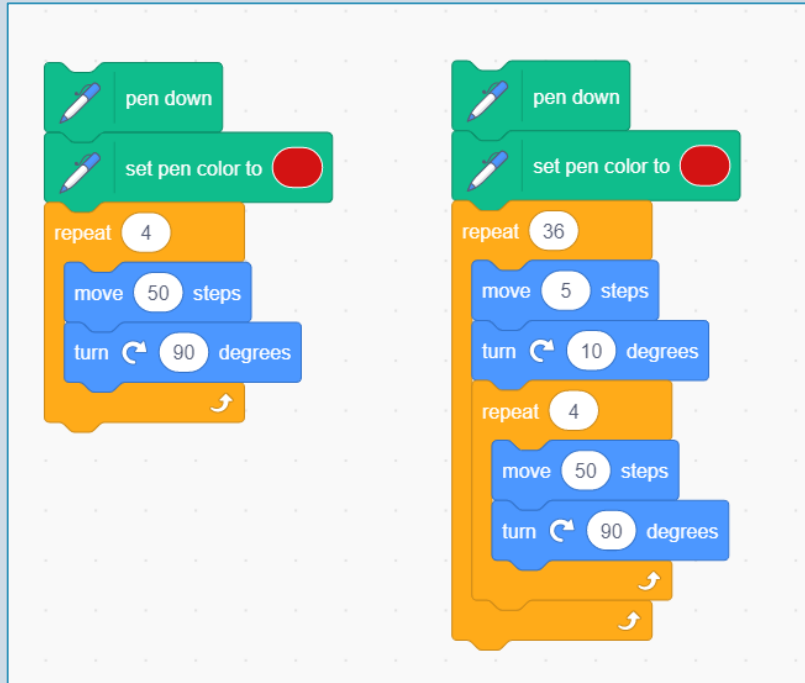


Nested Loop



Challenge:

Nested Loop



Challenge:

討論



- 從這份教材學生可以學到什麼？



■ 這樣教有什麼問題？



資訊科技教學 策略-不插電的 電腦科學

不插電的電腦科學



■ 教學目標

- 能瞭解二進位系統的概念
- 能完成十進位、二進位、與各種進位法的換算
- 認識各種電腦資料的表示
- 能將資料不同的表示法應用於生活中之問題解決

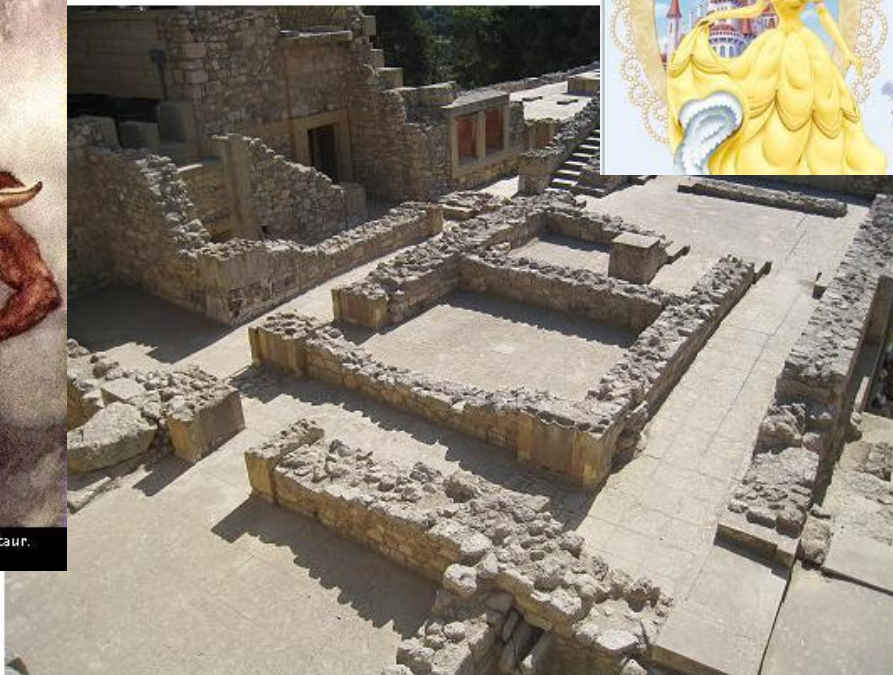


- 二進位系統與電腦資料的表示教學內容的重要概念為何？
→ 教進制轉換運算可以教到這些概念嗎？

希臘神話「黑白帆」-迷宮



W. Russell Flint, 1880-1969: Theseus and the Minotaur.
Photo © Maicar Förlag - GML.



希臘神話「黑白帆」

用二元化的符號表示
欲傳遞的資訊



表達數字



- 伸出你的手指頭
 - 從一數到二十，你會怎麼數？
 - 小叮噹會怎麼數數兒呢？



表達數字



■ 活動 聽聲辨數

分成兩組，第一組先設定三個數字，利用【小叮噹的0&1】FLASH教學工具中的小叮噹二進位教學撥放其二進位的高低音，第二組同學解碼，再反過來由第二組發訊號，而由第一組解碼。

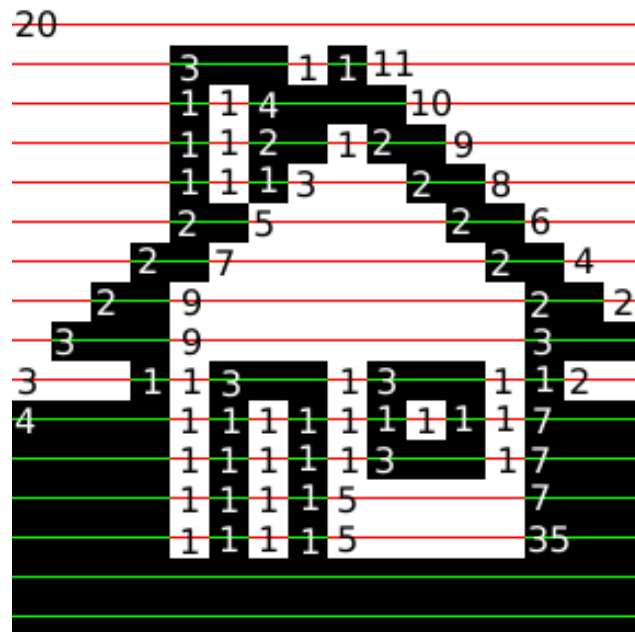


影像怎麼表示

□ 傳話遊戲

影像怎麼表示

□ Run-length coding



再壓多一點



- 各種符號出現的頻率不盡相同

a	b	c	d	e	f	g	h	i	j	k	l	m
8.2	1.5	2.8	4.3	12.7	2.2	2.0	6.1	7.0	0.2	0.8	4.0	2.4
n	o	p	q	r	s	t	u	v	w	x	y	z
6.7	7.5	1.9	0.1	6.0	6.3	9.1	2.8	1.0	2.4	0.2	2.0	0.1

- 知道這些有助於將文章編碼壓縮嗎?

密碼學



■ 你會解碼嗎?

usklp lfxyn mqmsn xisgl tysnq ywwyv zsfkk qnwks wrvqh vswiy rvqps wwsyv olool h
swyv wswmq xsyvw lifpp svsnv ssxln lkcws vsusm msnxn qxvsl hsnxk qulkl fnxsv usr
xs kluqv vswrq nilnu suqis ssnxv soloo lhssx wswis ybuqm rkfus wksrv qpsww syvae
slxwx qnssx kluqm xswws kqzsk lussn wsmok suswq nxxvq fwisw swrvf xwksw rkywq
vfhn lybis ksyvh snsvl xfqnf kwsno kstyf kwwsv lfsnx fmrkf tyswi lhwks iszsk qrrsm
snxiy nsmlu efnsr qyvkf uqifp fulxf qnlyx qmlxf tysis wmsww lhwu styfw svlfy ynsms
nlux svfo ksrqy vkswf nxsvs xwisk loskk spvln usnqy wsxyi fqnwq lqww fofkf xsisp
lfvsi swuqr fswws uvsxs wisus wrkln wsnxw fkwf nxxsw cwxsu sfnhs nfyb ismil hysv
v szsyf kksdv srsvn ivsuq nxlux wfkzq ywrkl fxlzs ukszq ksyvt yfswx wqyru qnsw svx
qy zsvlw qyxel mrxqn fknsp lfxly uyniq yxsty susxv lfxvs iswus ksvlx zlues vuesv lfmr
q wsvwq nrvf usufv skszs isklr kywel yxsfm rqvxl nusnl xfnl kplf wlnx qogsx iskl
w lnxw qnisk qyfw efkfr rskyf mmsx xqyv vlfm snlv vllk flnv tyliv yrksv fusk
xqmol fxsx vsism lyzlf wswml fnwxq yxsuq vswr qnln uspyx yvsuq nusv lnxw rvq
rq wisv llzqf vynuq iszf snsvs rfwy yfksv xfmw wfwk slisv qisv

不插電的電腦科學



- 不插電的電腦科學是系列透過卡片、蠟筆等工具與大地遊戲等方式學習電腦科學概念的教學
- 聚焦於重要概念，而非低階操作
- <https://csunplugged.org/>



CS UNPLUGGED

Computer Science without a computer

CS Unplugged is a collection of free teaching material that teaches Computer Science through engaging games and puzzles that use cards, string, crayons and lots of running around.

Welcome to the new CS Unplugged!

This updated website has unit plans, lesson plans, teaching videos, curriculum integration activities, and programming exercises to plug in the Computer Science concepts they have just learnt unplugged.

The original activities are still available at classic.csunplugged.org.

不插電的電腦科學



- Teaching London Computing 提供許多免費的教學資源
 - Queen Mary University of London 與 King's College London.
- <https://teachinglondoncomputing.org>
- Computing without computers 是Queen Mary University of London的 Paul Curzon 為新學者編寫的教材，介紹程式設計、資料結構、演算法等內容。透過隱喻等方式說明重要概念，而不是直接動手撰寫程式。
- <https://teachinglondoncomputing.org/resources/inspiring-computing-stories/computingwithoutcomputers/>

不插電的電腦科學



■ 不用電腦學程式 — 紙飛機團隊 譯

目錄

譯序	I
前言	VI
第一章. 不再混亂 (簡介)	1
第二章. 人類的語言本能 (程式語言)	9
第三章. 管線工程 (順序性)	30
第五章. 如果這樣那樣的話... (選擇)	77
第六章. 山姆, 再彈一次那首歌 (迭代與遞迴)	95
第七章. 更大又更好 (註解、函式及程序)	134
第八章. 一進一出, 一進一出, 全身搖一搖 (輸入-輸出)	145
第九章. 修理, 再利用, 回收 (物件導向程式設計)	152

第十章. 長幼有序 (列表的觀念與陣列這個資料結構)	169
第十一章. 排隊時, 另一條總是比較快 (佇列與堆疊)	183
第十二章. 見樹又見林 (動態資料結構)	189
第十三章. 開始搜索 (搜尋演算法)	202
第十四章. 來整理各式各樣的東西 (排序演算法)	231
第十五章. 那條正確的道路 (路徑演算法)	243
第十六章. 貪婪演算法	250
第十七章. 真命天子 (選擇演算法)	252
第十八章. 即將結束的旅程	259

不插電的電腦科學



■ 不用電腦學程式

第一部：程式

第一章. 不再混亂 (簡介)

人類是一種討厭混亂的動物。我們嘗試有效規劃我們的生活、城市、家庭甚至有時是我們的桌面和房間。我們將衣服整齊地掛在衣櫥裡，我們利用郵遞區號和街道將城市分割成不同的區塊，且它們彼此相連。我們把書放在書架上，我們在超商排隊結帳，我們用盡一切方法把混亂的東西按照有秩序的方式重新規劃。

有很多種規劃事情的方法，而我們很自然地用不同的方法規劃不同的東西。當然更重要的是，在我們規劃好之後，**我們希望用它們來做些什麼事**。舉例來說：我們把地址按照郵遞區號規劃，是希望可以讓信件更容易被送到；而書店裡的小說通常會按照作者名字的字母開頭順序擺放，使顧客更快速找到想要的書。以我自己來說，我整理客廳的書櫃時會把令人印象最深刻的書放在最顯眼的位置，像是 Stephen Hawkin 的 *A Brief History of Time* 和 John Steinbeck 的書都會被我排在書櫃正中間，原因是我認為他們最能吸引到訪客的目光，而書店則是對於如何快速賣出書本更有興趣。

不插電的電腦科學



■ 不用電腦學程式

同樣的任務，策略上的不同會影響達成方式是否更快、更簡單或更容易成功。這個概念可以透過 *Spit Not So* 這個遊戲來說明（取自 Berlekamp et al, 1982）。*Spit Not So* 是一個雙人遊戲，首先，在九張卡片上寫下 SPIT、NOT、SO、FAT、FOP、AS、IF、IN、PAN。接著，把有字的那面朝上，然後每位玩家輪流選卡，當任一方獲得場上所有同一個字母的卡片（每個字母共有三張）則贏得比賽。舉例來說，如果你挑中了 SPIT、IF 和 IN，則你獲得了所有含有 I 的卡片並贏得了比賽。而如果全部的卡都抽完卻沒有任何一方獲得所有有相同字母的卡片，則雙方平手，因此這個遊戲會像是：

玩家 1: SPIT => 玩家 2: SO => 玩家 1: FAT => 玩家 2: NOT =>

玩家 1: FOP => 玩家 2: IF => 玩家 1: PAN

不插電的電腦科學



■ 不用電腦學程式

在這個情況下玩家 1 贏得了比賽，因為他獲得了所有含有 P 字母的卡：SPIT、FOP 和 PAN！

多玩幾場你就可以更了解這個遊戲，然後翻去下一頁來看看可以怎麼排這些卡來讓你玩得更好（或是你能夠自己發現！）

在開始遊戲之前請先畫好下圖的九宮格，然後將字母填入格子之中。記住如何擺放字母是非常重要的，任一直線都必須要有一樣的字母，舉下圖來說：最底下那排都有 F

NOT	IN	PAN
SO	SPIT	AS
FOP	IF	FAT

不插電的電腦科學



■ 不用電腦學程式

注意不要向你的對手透露你怎麼擺放你的字母，然後再像之前一樣進行這個遊戲。唯一的差別是這次每抽一張卡，你必須在方格內將抽到的那個字母打叉，然後將對手抽到的字母畫圈。舉例來說，用這個版本進行的遊戲會在以下這個情況結束：

NOT	IN	PAN
○		X
SO	SPIT	AS
○	X	
FOP	IF	FAT
X	○	X

不插電的電腦科學



■ 不用電腦學程式

發現了嗎？當你的對手在玩 *Spit Not So* 時，你是在玩圈圈叉叉這個我們小時候就很會玩的遊戲！只要選擇如何排列字母，你就已經把這個很容易出錯的遊戲轉變成一個幾乎不可能輸的遊戲！這就是我所說的使用一個適當的**規劃方法**能給你很大的幫助！

不插電的電腦科學



■ 不用電腦學程式

因此機器只是取代了進行運算的人之前所做的事情，而機器的速度和準確度代表著它們可以更快地處理更大的問題。儘管如此，它們本質上做的事情還是與之前一樣的。電腦是透過一行一行的程式語言的指令來完成任務，而進行運算的人也是依照程式去執行工作的，指令也是照著人類的語言去制定的。我們在進行多位數乘法的時候就是依照小時候學過的步驟，一步一步算出最後的結果，這些步驟就是演算法。演算法包含要做的動作以及執行這些動作的順序。除了算乘法以外，其實到處都可以看到演算法的蹤跡：例如說教你如何 DIY 書架的指令是演算法，破解魔術方塊的步驟也是演算法。事實上，在一系列牽涉到演算法設計的難題中，魔術方塊是其中最複雜的一項。

不插電的電腦科學



■ 不用電腦學程式

某天當我要去商店買東西的時候，一台車開向我然後搖下車窗問：不好意思，請問你知道如何去 Kirkcroft 巷嗎？而我使用了演算法的方式回應他：

先下山；
在第一個路口左轉；
再往山上開；
然後在 Navigation 俱樂部後面第一個彎道左轉之後；
到了！

這為什麼是演算法呢？這是一連串的指令告訴對方如何採取行動，更精確來說，這是一連串的動作跟順序的巧妙安排。如果那個司機嘗試使用不一樣的順序（舉例來說在第一個步驟選擇上山不是下山），那麼他就不能到達正確的地方，而程式只是透過一連串電腦看得懂的指令讓電腦依序去執行而已。

不插電的電腦科學



■ 不用電腦學程式

並不是列了一堆規則便可以叫做演算法。舉例來說，任何一個公園都可能有個看板寫著：

「請走在人行道上」

「禁止踏入草坪」

「禁止腳踏車，滑板，或溜冰鞋」

「公園每日於日落前關閉」

「禁止球類活動」

然而即使這也是一行行的規定所組成，這個卻不是演算法的案例。在飛機裡洗手間內的標語也是一樣的情況：

「禁止吸煙」

「禁止飲食」

「當安全帶警示燈亮起時請勿使用洗手間」

不插電的電腦科學



■ 不用電腦學程式

演算法告訴你如何一步一步執行任務，而上面這些規則告訴你什麼可以做，什麼不能做。因此你應該可以發現演算法是在談論「如何」做，而不是做「什麼」。上面所提到的那些規則即使順序顛倒了，在讀的人看來意義仍然是一樣的，但在演算法裡順序是非常重要的，像是其他在飛機裡可以看到的指令：

緊急逃生發生時：

1. 拉開蓋子
2. 將門把推到開啟位置並鬆手
3. 向外把門推開

這些指令是關於「如何」逃生。當緊急事件發生的時候，沒有人會希望看到一張告示上面寫著我被允許做什麼，或不允許做什麼。因為在緊急情況發生的時候，我們需要的是如何最快離開飛機的方法。所以這些指示必須是清楚，不模糊，且非常精準的，不能夠有任何可能性讓我們對這些指令或是順序感到疑惑。換言之，我們要一個演算法。生活中充滿了演算法，而這些步驟都非常精準地告訴你如何按照順序完成指示，由此可見，順序是非常重要的。

不插電的電腦科學



■ 不用電腦學程式

有一種童謠裡頭包含許多重複的循環，下面這首很明顯就是這樣：

十個小孩擠小床，小小孩兒擠向旁：“睡過去、睡過去”

擠呀擠、滾呀滾，一個小孩滾下床

九個小孩擠小床，小小孩兒擠向旁：“睡過去、睡過去”

擠呀擠、滾呀滾，一個小孩滾下床

八個小孩擠小床，小小孩兒擠向旁：“睡過去、睡過去”

擠呀擠、滾呀滾，一個小孩滾下床

（繼續）

只剩自己睡張床，小小孩兒開心躺：“晚安”

不插電的電腦科學



■ 不用電腦學程式

1. 十個小孩擠小床
2. 當（While）有其他小孩在床上（重複以下步驟）
 - 小小孩兒擠向旁：“睡過去、睡過去”
 - 全部小孩擠呀擠滾呀滾
 - 一個小孩滾下床
3. 小小孩兒說：「晚安」

討論



- 選擇一種程式結構，並設計一段中文的虛擬碼，教學生此程式結構。

不插電的電腦科學



■ 桌遊 — 海霸





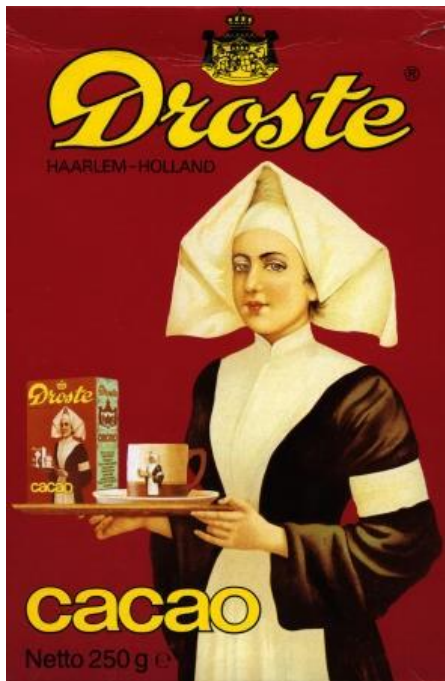
資訊科技教學 策略-視覺化

視覺化

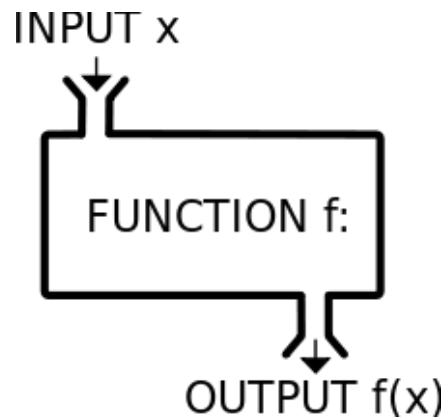


■ 將抽象概念視覺化

■ 遞迴



■ 函式



視覺化模擬輔助教學

- 模擬是利用**引導式的體驗**代替**實際經驗**的學習策略，學生在模擬環境中進行各種**體驗活動**，逐步**歸納概念**，以建立屬於自己的**心智模型**。
- 模擬的過程中，視覺化的呈現可幫助學生**理解抽象資料**以及**發展抽象概念**。



視覺化模擬輔助教學

● 根據教學實驗結果，發現：

✓ 透過觀察演算法動態的模擬效果，以及對照參數調整後的結果，幫助學生將抽象概念進行表徵，以更清楚理解演算法的邏輯、執行順序與變項之間的關聯性，因此可以有效提高學生的學習成就。

✓ 透過模擬觀察複雜的演算法流程或概念架構，能降低學生的學習負擔，因而感受到的課程難度較低，且對抽象主題的理解程度有較正向的感受，因此能提升學習態度。



多媒體學習

- 多媒體學習 (multimedia learning) 是指利用圖片、視訊、聲音與動畫等多媒體元件的輔助進行學習
 - 將二元搜尋樹的動態搜尋歷程以動畫呈現，讓學生可以觀察並追蹤搜尋程序 (<https://www.mathwarehouse.com/programming/gifs/binary-search-tree.php>)
 - 利用河內之塔的互動式遊戲，讓學生直接操作並思考解題程序 (<https://www.mathsisfun.com/games/towerofhanoi.html>)

Discs:

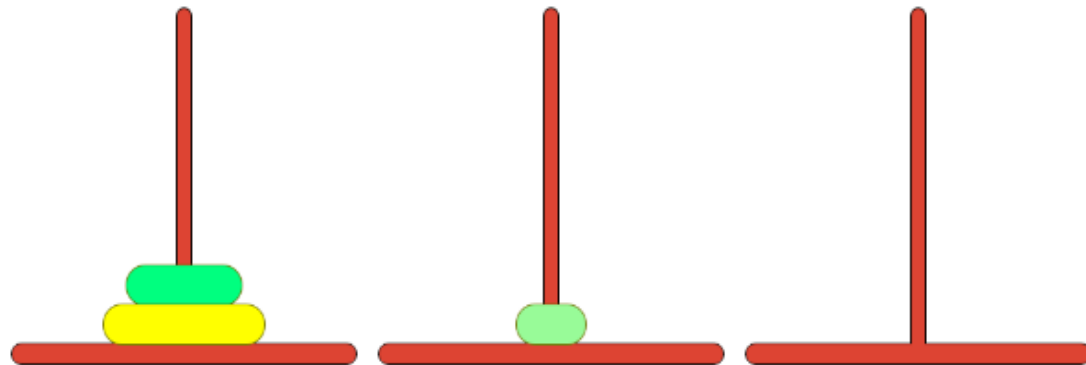
3



Moves: 1

Restart

Solve!



多媒體學習

- 用CPU的模擬軟體http://www.cs.colby.edu/djskrien/CPU_Sim/），讓學生了解程式指令如何從記憶體被取到CPU的暫存器，及如何透過各種不同功能的暫存器進行指令運算等
- 網路平台提供相當豐富的視覺化教學素材，將許多常見的資料結構與演算法進行視覺化，透過動畫、互動工具的操作等方式，讓學生更理解資料結構與演算法的運作原理

<https://visualgo.net/>

Do You Know?

Next Random Tip

Visualgo has two main components: The visualization pages (currently 23) and the Online Quiz component (currently 12 and will be 23 too in the near future). We do not script any of the questions in Online Quiz (and all answers will be graded almost instantly). You can try the online quiz at [our training page](#) or by clicking the 'Training' button on the visualization module.

VISUALGO.NET/EN

visualising data structures and algorithms through animation

Search...



Sorting

Training

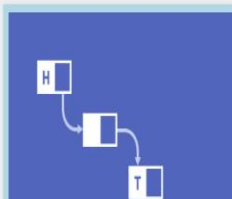
array algorithm bubble select insert selection



Bitmask

Training

bit manipulation set cs3233 array list ds



Linked List

Training

stack queue doubly deque cs1020 cs2020



Hash Table

Training

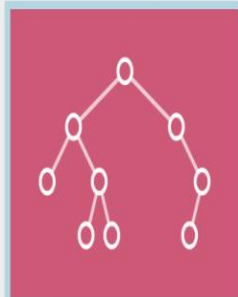
open addressing linear quadratic probing cs1020



Binary Heap

Training

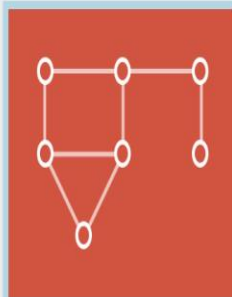
priority queue recursive cs2010 cs2020 cs2040



Binary Search Tree

Training

adelson velicki landis set table AVL cs2010



Graph Structures

Training

tree complete bipartite dag cs2010 cs2020



Union-Find DS

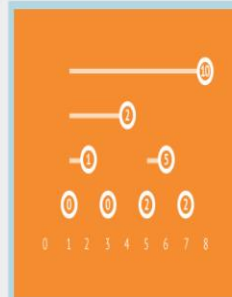
Training

path compression disjoint set data structure



Segment Tree

dynamic range sum min max cs2233



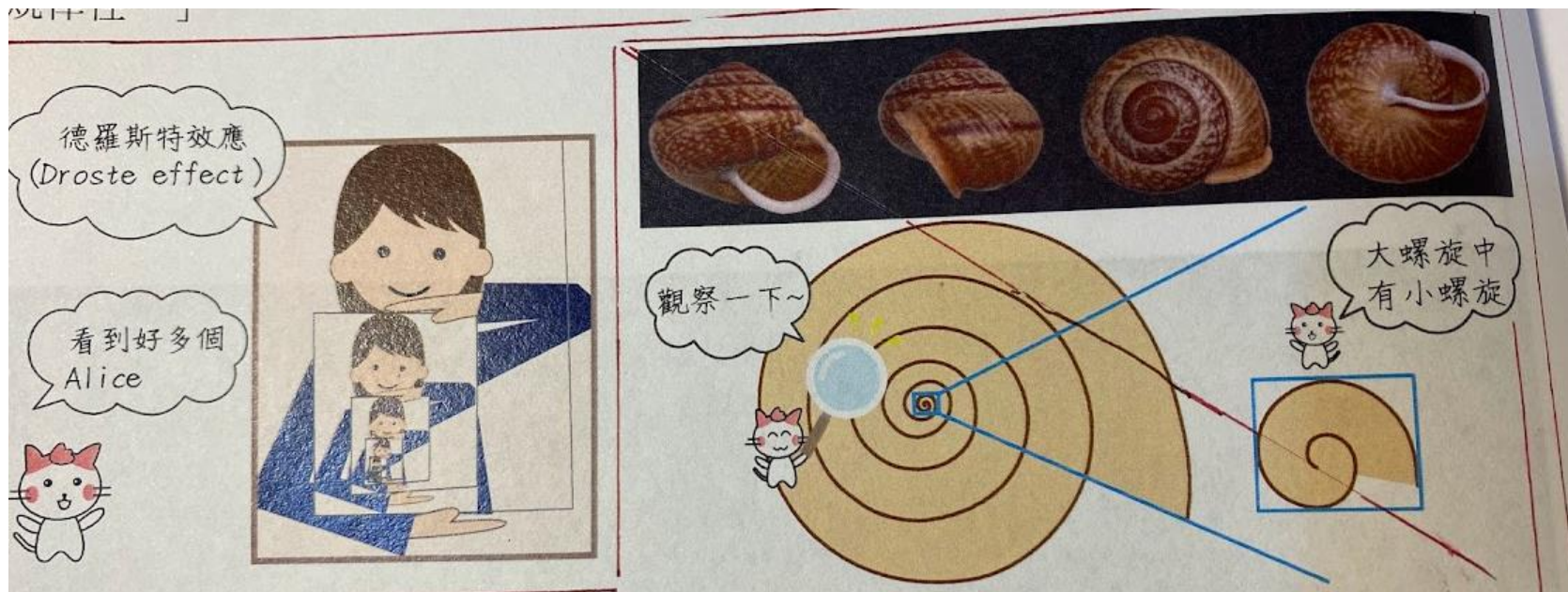
Fenwick Tree

binary indexed tree bit dynamic fenwick range

討論



■ 這是好的遞迴結構教材嗎？

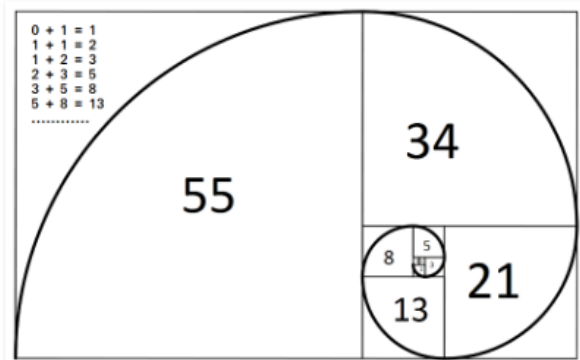


視覺化



■ 這是好的遞迴結構教材嗎？

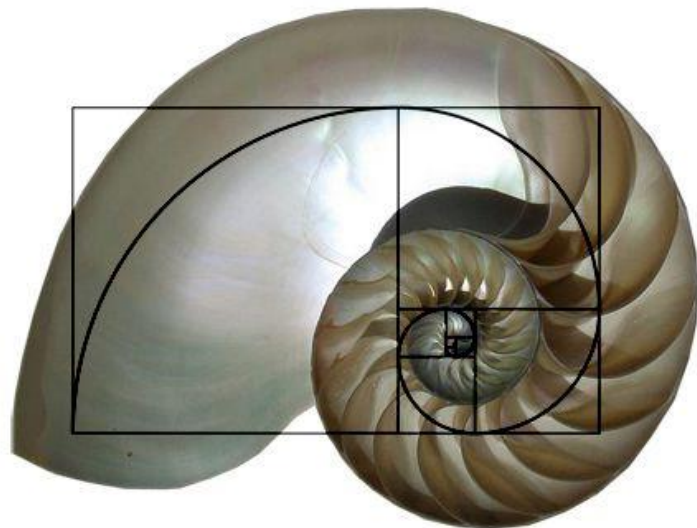
Following is a simple program to print first n [Fibonacci numbers](#).



Examples :

```
Input : n = 3  
Output : 0 1 1
```

```
Input : n = 7  
Output : 0 1 1 2 3 5 8
```



視覺化



■ 視覺化程式執行中的變數變化

Python 3.6

```
1 def listSum(numbers):
2     if not numbers:
3         return 0
4     else:
5         (f, rest) = numbers
6         return f + listSum(rest)
7
8 myList = (1, (2, (3, None)))
9 total = listSum(myList)
```

[Edit this code](#)

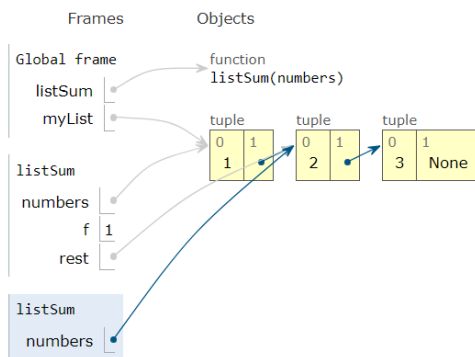
→ line that just executed
→ next line to execute

< Prev Next >

Step 10 of 22

Visualized with pythontutor.com

[Move and hide objects](#)



Αλγόριθμος Συγχώνευσης δύο ταξινομημένων πινάκων

```
1 Αλγόριθμος Συγχώνευση
2 Δεδομένα // A, B, N, M //
3
4 k ← 1
5 j ← 1
6 i ← 1
7 Όσο k ≤ N και j ≤ M Επανάληψη
8   Αν A[i] < B[j] Τότε
9     Γ[k] ← A[i]
10    i ← i + 1
11  Αλλιώς
12    Γ[k] ← B[j]
13    j ← j + 1
14  Τέλος_Αν
15  k ← k + 1
16 Τέλος_Επανάληψης
17 Αντιγραφή του υπολοίπου πινάκα
18 Αποτελέσματα // Γ //
19 Τέλος Συγχώνευσης
20
```

Α

1	2	3	4	5	6	7	8
10	20	30	40	50	60	70	80

Β

1	2	3	4	5	6	7
6	16	28	32	34	36	42

Γ

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
6	10	16	20	28	30	32	34	36	40					

42

Εκτέλεση Βήμα-βήμα Συνέχεια Τερματισμός

Ταχύτητα Εκτέλεσης

Εισαγωγή Προκαθορισμένων Πινάκων

Πίνακας A = [Εισάγετε αριθμούς, διαχωρίζοντάς τους με κενό] Πίνακας B = [Εισάγετε αριθμούς, διαχωρίζοντάς τους με κενό] [Εισαγωγή]

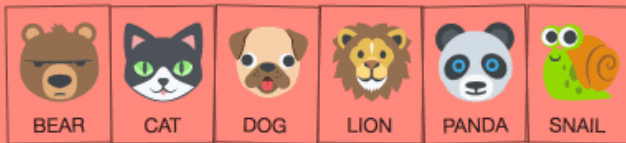
<https://pythontutor.com/>



BINARY SEARCH QUICKSORT BFS

 ABOUT

Find the position of a value
inside a sorted list



press on one of the animals to begin

```
0. function binarySearch(list, item) {  
1.   let low = 0;  
2.   let high = list.length - 1;  
3.  
4.   while (low ≤ high) {  
5.     const mid = Math.round((low + high) / 2);  
6.     const guess = list[mid];  
7.  
8.     if (guess === item) {  
9.       return mid;  
10.    }  
11.    if (guess > item) {  
12.      high = mid - 1;  
13.    } else {  
14.      low = mid + 1;  
15.    }  
16.  }  
17.  
18.  return null;  
19. }
```

視覺化



■ 視覺化演算法程序





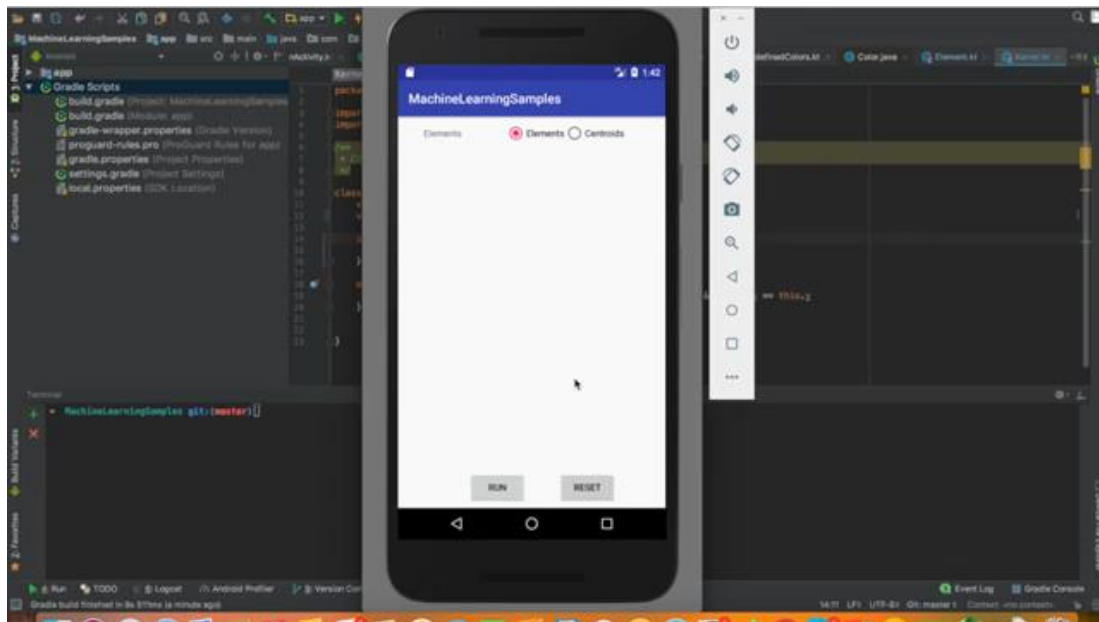
■ 視覺化排序演算法



視覺化



■ 模擬



視覺化



■ 模擬

一、請操作平臺到下方的頁面，並於建立完決策樹後截圖至下方空白處。

概念探索

學習課程 · 決策樹 · 概念探索

請幫幫海盜先生利用表格中的資料拖拉積木區對應顏色的積木，做出與原有決策樹不同的決策樹，來預測下午是否會下雨。
(小提示：決策樹的決策不一定要是正確的)

決策樹①

```
graph TD
    A[現在天氣] -- 晴天 --> B[現在氣溫]
    A -- 多雲 --> C[下雨]
    B -- "≤ 30°C" --> D[下雨機率]
    B -- "≥ 30°C" --> E[沒下雨]
    D -- "≤ 50%" --> F[沒下雨]
    D -- "≥ 50%" --> G[下雨]
```

決策樹②

積木區

- 現在天氣
- 現在氣溫
- 下雨機率
- 下雨
- 沒下雨

確定建立 重設

```
weather = str(input("現在天氣"))
if weather == "晴天":
    temp = int(input("現在氣溫"))
    if temp >= 30:
        print("沒下雨")
    else:
        rain = int(input("下雨機率"))
        if rain < 50:
            print("沒下雨")
        else:
            print("下雨")
else:
    print("下雨")
```

編號	現在天氣	下雨機率	現在氣溫	下午下雨
1	多雲	30%	22°C	是
2	晴天	70%	33°C	是
3	晴天	40%	27°C	否
4	晴天	20%	32°C	是
5	多雲	90%	18°C	是
6	多雲	20%	23°C	否
7	晴天	30%	31°C	否
8	晴天	10%	27°C	否

上一頁 下一頁

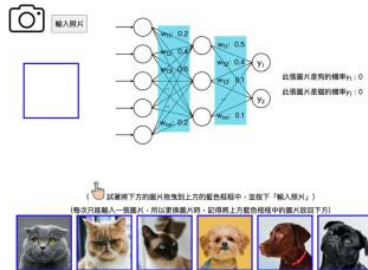
不能只提供視覺化工具，
重要的是思考的引導

二、承上題，你認為哪棵決策樹較好？

視覺化



C. 應用類神經網路



1. 請在資料搜集的頁面中，將圖片拖曳至訓練資料集並按下「資料搜集完成」，觀察頁面中呈現了什麼，並且記錄下來：

2. 請在資料搜集的頁面中，試著將「貓味的圖片」拖曳到「狗的類別」中，按下「資料搜集完成」，觀察頁面中呈現了什麼，並且記錄下來：

3. 承上題，思考看看，如果將訓練資料集當中的類別標示錯誤，對於訓練類神經網路有什麼影響？

4. 請在訓練類神經網路的頁面中，按下「進行第一次迭代」且接續按下「讀取圖片」，觀察頁面中呈現了什麼，並且記錄下來：

5. 請在訓練類神經網路的頁面中，按下「進行第一次迭代」且接續按下「讀取圖片」，觀察頁面中呈現了什麼，並且記錄下來：

6. 請在應用類神經網路的頁面中，將貓咪的圖片拖曳到左上角的方格中，並按下輸入照片，觀察頁面中呈現了什麼，並且記錄下來：

7. 承上題，思考看看，如果今天輸入了一個「很像貓咪的狗圖片」，類神經網路所輸出的機率值可能會有什麼狀況？

8. 請重新簡述，這三個頁面所學習到的相關概念：

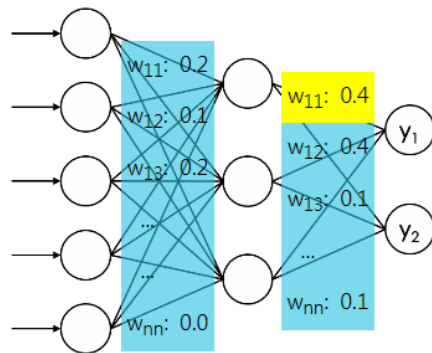
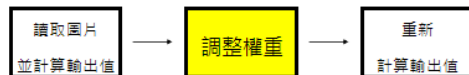
不能只提供視覺化工具，重要的是思考的引導

( 按下進行迭代後，請仔細觀察訓練類神經網路的過程)

編號	圖片	類別	讀取
1		貓	✓
2		貓	
3		貓	
4		貓	
5		貓	
6		狗	
7		狗	
8		狗	
9		狗	
10		狗	

訓練
類神經網路

讀取圖片



此張圖片是狗的機率 $y_1: 0.31$

此張圖片是貓的機率 $y_1: 0.31$

以動畫模擬迭代

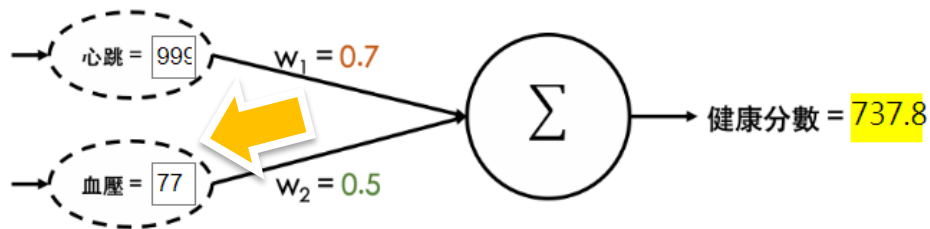
釐清概念

( 試著在方格中輸入數字，並按下計算)



(試著打開聲音並點擊心臟)

	平均心跳	平均血壓	健康分數
小東	60	80	80
國裕	70	100	100
小恩	90	70	97
祺祺	85	120	75



測量血壓

收縮壓為 101 mmHg

舒張壓為 60 mmHg

計算

聽覺製造臨場感

手動調整參數

視覺化



運算思維導向程式設計視覺化輔助學習平台

條件

- 計算期末的總成績
- 計算百貨公司折扣
- 判別直角三角形
- 測謊機
- 電影售票機

[回首頁](#)

單元範例-例題5

程式視覺化

模擬執行



假日，美伢決定帶小新去看電影。來到電影院後，看著人工窗口大排長龍的隊伍，於是便決定去空無一人的電子售票機購票。這台電子售票機的功能並不完善，每次只能販售一張電影票。請同學試著寫出，能夠讓機器自動進行條件判斷的程式，亦即：消費者只要輸入年齡，電子售票機就會顯示相對應的票價。開心電影院的票價：60歲以上(敬老票)，票價為150元。12歲到59歲(一般票)，票價為200元。低於12歲，票價為170元。(EX:輸入-10 輸出-票價為170)

程式碼

```
age = int(input("請輸入年齡："))
if (age >= 60):
    print("票價150元")
elif (age >= 12):
    print("票價200元")
else:
    print("票價170元")
```

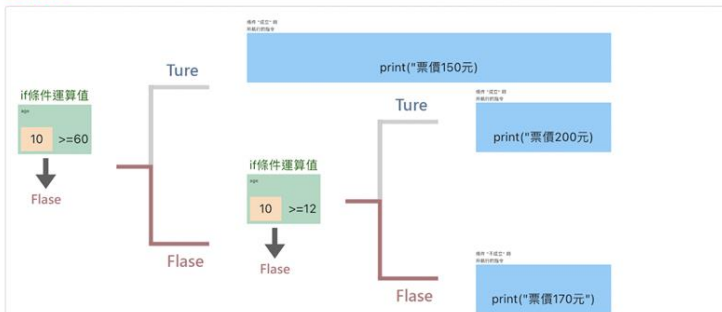


程式解釋

程式語法

```
if 條件運算值1:
    條件1"成立"時所執行的指令
elif 條件運算值2:
    條件2"成立"時所執行的指令
else:
    條件1、2皆"不成立"時所執行的指令
```

變數記憶



執行結果

請輸入年齡： 10
票價170元



DFS

先看到的，後搜（找便利商店）



■ 步驟一：

- 從我家開始，（並標記為已走訪過）。

■ 步驟二：

- 相鄰我家的地點，(依逆時針方向)有博物館、加油站。
- 選擇排在最後的加油站來出發。



先看到的，後搜（找便利商店）



- 步驟三：
 - 從加油站出發，並標記（已走過）。
- 步驟四：
 - 相鄰加油站，有四個地點。分別為醫院、我家（已走過）、主題樂園和科技公司。
 - 選擇排在最後的科技公司來出發。



科技公司
主題樂園
醫院
~~加油站~~
博物館

先看到的，後搜（找便利商店）



- 步驟五：
 - 從科技公司出發，並標記（已走過）。
- 步驟六：
 - 相鄰科技公司，有四個地點。分別為便利商店、加油站（已走過）、大賣場和電影院。
 - 選擇排在最後的電影院。



？該放什麼地點

先看到的，後搜（找便利商店）



■ 步驟七：

- 從電影院出發，並標記（已走過）。

■ 步驟八：

- 相鄰電影院的地點--無，而科技公司（已走過）。
- 無路可走，只好循路，退回到科技公司。相鄰科技公司，未走訪的還有二處，便利商店和大賣場。
- 選擇排在最後的大賣場出發。



先看到的，後搜（找便利商店）



■ 步驟九：

- 從大賣場出發，並標記（已走過）。

■ 步驟十：

- 相鄰大賣場，只有科技公司（已走過）。
- 無路可走，只好循路，退回到科技公司。
- 相鄰科技公司，未走訪的有便利商店。
- 選擇便利商店出發



先看到的，後搜（找便利商店）

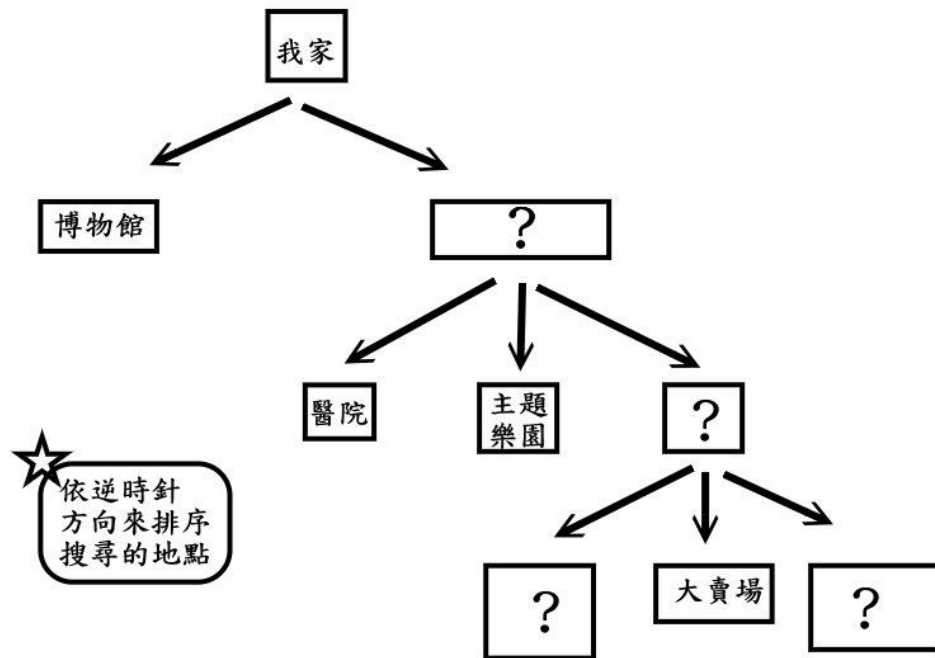


■ 步驟十一：

- 從便利商店出發
- 找到了

電影院
大賣場
便利商店
科技公司
主題樂園
醫院
博物館

請問找到便利商店，經過哪些地點呢？



總結



1. 從起點出發，並設定已走過。
2. 持續走訪任一相鄰的地點(未走過)，若不是目標，設定已走過。若是目標，結束。反覆步驟2。
3. 若前面的相鄰地點都走過，則持續往後退，往其他相鄰地點(未走過)。反覆步驟2。若都沒有符合目標，表示無此路徑存在。
4. 上述的方法為DFS演算法。

加油站

博物館





BFS

先看到，先搜尋（找便利商店）（依順時針方向）



■ 步驟一：

- 從我家開始。

■ 步驟二：

- 相鄰我家的地點
有加油站和博物館。
並把我家標記（已走過）。



博物館
加油站
我家

先看到，先搜尋（找便利商店）（依順時針方向）



■ 步驟三：

- 從加油站開始。

■ 步驟四：

- 相鄰加油站的地點有科技公司、主題樂園、我家（已走過）、醫院。
- 並把加油站標記（已走過）。



醫院
主題樂園
科技公司
博物館
加油站

先看到，先搜尋（找便利商店）（依順時針方向）



- 步驟五：
 - 從博物館開始。
- 步驟六：
 - 相鄰博物館的鄰居有學校、我家（已走過）、機場。
 - 並把博物館標記（已走過）。



機場
學校
醫院
主題樂園
科技公司
博物館

先看到，先搜尋（找便利商店）（依順時針方向）



■ 步驟七：

- 從科技公司開始。

■ 步驟八：

- 相鄰科技公司的地點有電影院、大賣場、加油站（已走過）、便利商店。
- 並把科技公司標記（已走過）。



便利商店
大賣場
電影院
機場
學校
醫院
主題樂園
科技公司

先看到，先搜尋（找便利商店）（依順時針方向）



■ 步驟九：

- 從主題樂園開始。

■ 步驟十：

- 主題樂園無相鄰地點，略過。
- 並把主題樂園標記（已走過）。



便利商店
大賣場
電影院
機場
學校
醫院
主題樂園

先看到，先搜尋（找便利商店）（依順時針方向）



■ 步驟十一：

- 從醫院開始。

■ 步驟十二：

- 醫院的鄰居有銀行、加油站（已走過）、學校。
- 並把醫院標記（已走過）。



?

便利商店
大賣場
電影院
機場
學校
醫院

先看到，先搜尋（找便利商店）（依順時針方向）



■ 步驟十三：

- 從學校開始。

■ 步驟十四：

- 相鄰學校的地點有醫院（已走過）、博物館（已走過）、圖書館。
- 並把學校標記（已走過）。



先看到，先搜尋（找便利商店）（依順時針方向）



■ 步驟十五：

- 從機場開始。

■ 步驟十六：

- 機場無鄰居，略過。
- 並把機場標記（已走過）。



先看到，先搜尋（找便利商店）（依順時針方向）



- 步驟十七：
 - 從電影院開始。
- 步驟十八：
 - 電影院無鄰居，略過。
 - 並把電影院標記（已走過）。
- 步驟十九：
 - 從大賣場出發。
- 步驟二十：
 - 大賣場無鄰居，略過。
 - 並把大賣場標記（已走過）。



圖書館
學校
銀行
便利商店
大賣場
電影院
機場

先看到，先搜尋（找便利商店）（依順時針方向）



■ 步驟二十一：

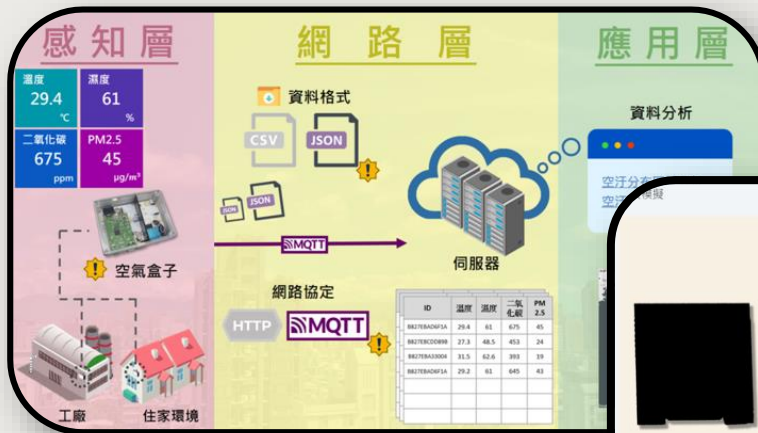
- 從便利商店出發。
- 找到囉。



圖書館
學校
銀行
便利商店

物聯網之教學主題

物聯網





人工智慧



機器學習

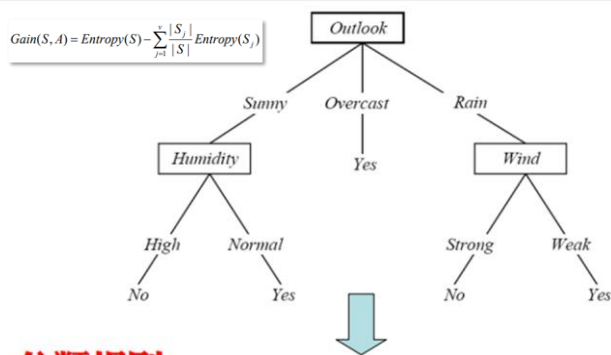


資訊安全



傳統的人工智慧教學

資料來源：決策樹學習、聚類分析

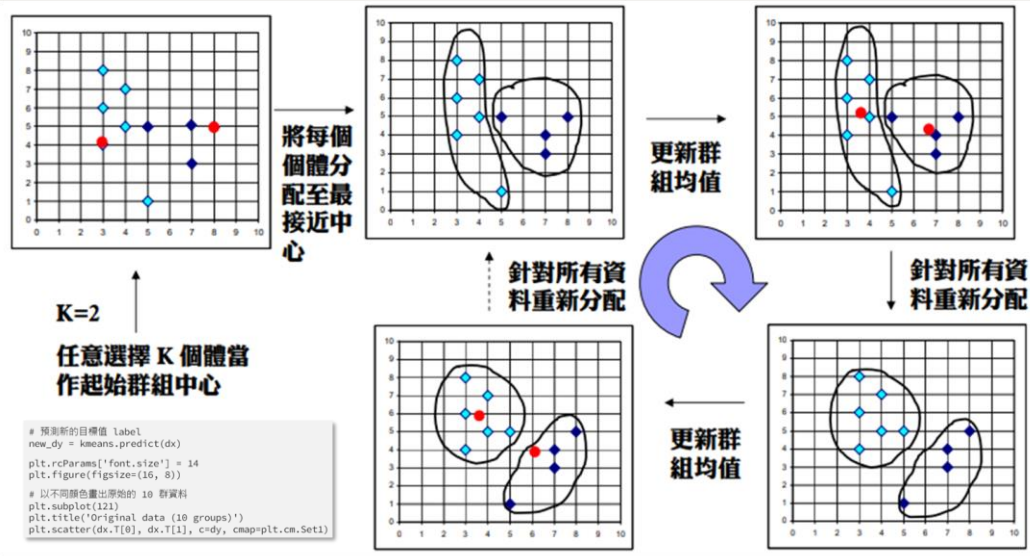


分類規則:

If Outlook = Sunny and Humidity = High Then Play Tennis = No
If Outlook = Sunny and Humidity = Normal Then Play Tennis = Yes
If Outlook = Overcast Then Play Tennis = Yes
If Outlook = Rain and Wind = Strong Then Play Tennis = No
If Outlook = Rain and Wind = Weak Then Play Tennis = Yes

【Decision tree】

學生較難將特徵的選擇的公式
與實際程式碼產生連結



【K- means】

直接進入程式碼的分群實作
較容易混淆演算法的分類概念

傳統人工智慧教學的困境

- ✓ 目前較缺乏針對學生設計的人工智慧教材。
- ✓ 以往針對K-12的人工智慧教育多著重在使用現有工具動手操作，而不介紹人工智慧理論，學習者難以瞭解其背後原理。
- ✓ 人工智慧涵蓋許多抽象與複雜的概念，學生不易理解。

K-means演算法步驟

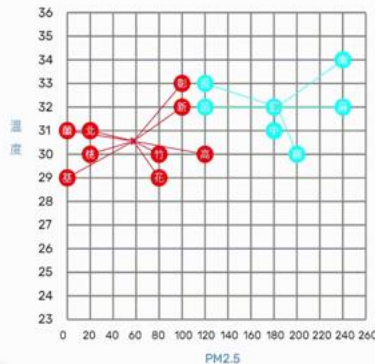
1. 選取 K 個中心
2. 計算各資料與各中心的距離
3. 將各資料分入最近距離的群
4. 重新計算各群中心
5. 重複步驟2至4，直到各群中心不再改變

K為要分成的群數

歐幾里得距離公式：

$$d = \sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2}$$

$$\text{中心公式：} C = \left(\frac{x_1 + x_2 + \dots + x_k}{k}, \frac{y_1 + y_2 + \dots + y_k}{k} \right)$$



請選擇K值:

2

請輸入初始的分群中心點(輸入範例: 0,30)
提醒: 中心點請輸入方格上的座標

100.28

200.32

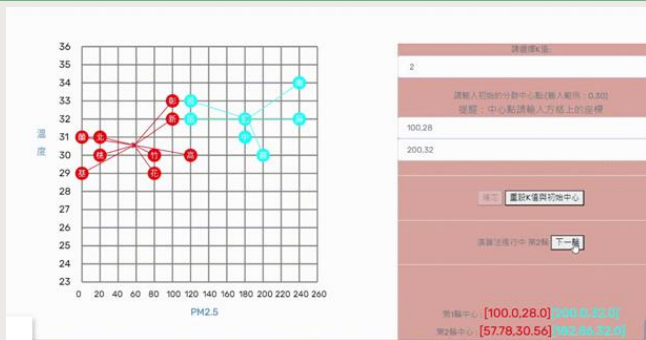
確定 重新K值與初始中心

演算法進行中 第2步 下一節

第1個中心: [100.0,28.0] [200.0,32.0]
第2個中心: [57.8,30.56] [182.96,33.0]

視覺化模擬平台可以協助

- ✓ 降低學生的學習負擔。
- ✓ 讓學生依照自己的進度與需求進行不同階段的模擬操作及互動。
- ✓ 在平臺上修改演算法的參數以及實作練習題填答完後給予的即時回饋，皆有助於其對演算法之理解。
- ✓ 讓學生以自己的步調進行學習，並藉由即時回饋隨時檢視學習成效，因此能擁有較好的自我效能。
- ✓ 專題製作：人工智慧概念與實作產生更多連結，更深入瞭解程式設計的意義與重要性。



討論



- 設計一種**視覺化**的教學呈現，幫助學生理解複雜或抽象的概念或演算法



資訊科技教學 策略-動手實作



■ Teaching Physics by doing experiments in lectures

– Professor Walter Lewin, MIT

- Potential Energy (勢能)

<http://www.youtube.com/watch?v=CgqBg44azYk&feature=relmfu>

44:00

- Simple Harmonic Oscillator (簡諧運動)

<http://ocw.mit.edu/courses/physics/8-01-physics-i-classical-mechanics-fall-1999/video-lectures/lecture-10/>

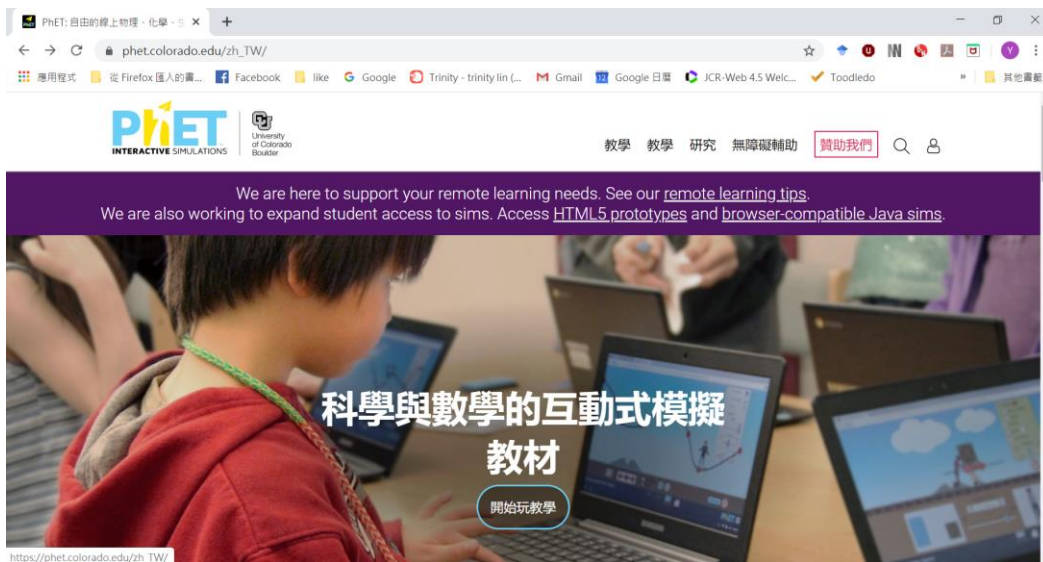
45:00

動手實作

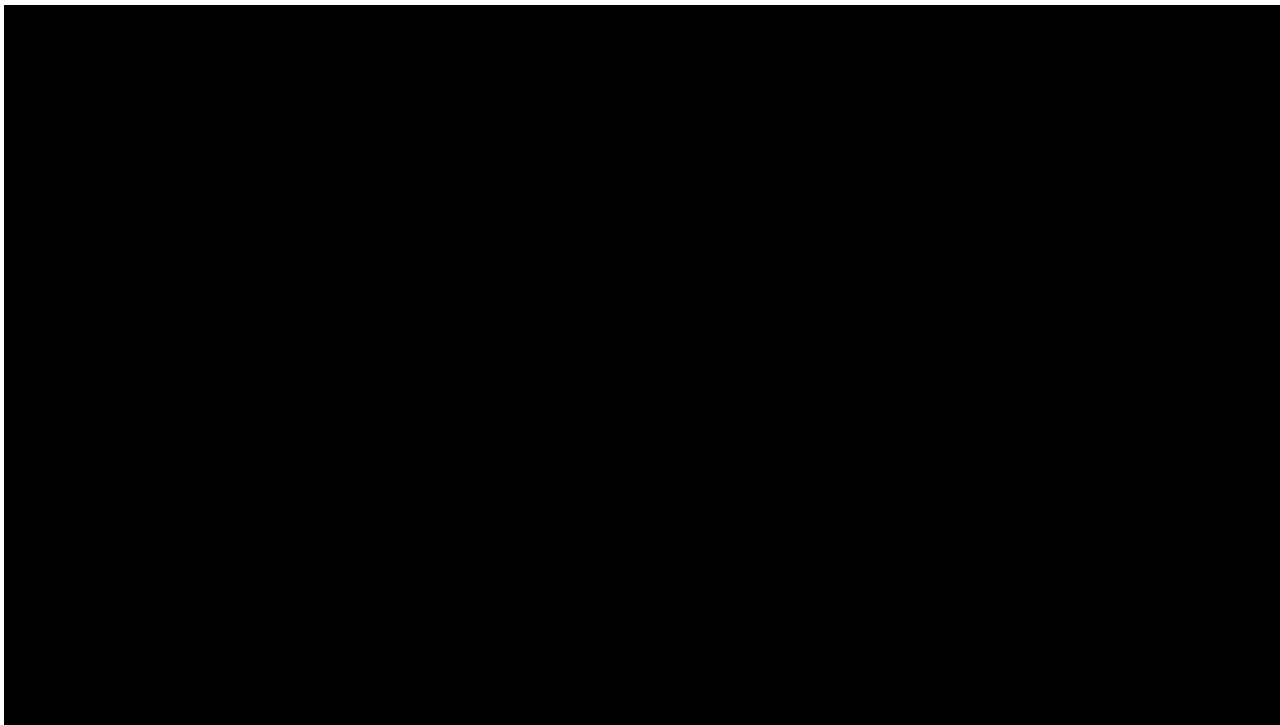


■PhET 模擬教學

— <https://phet.colorado.edu/>



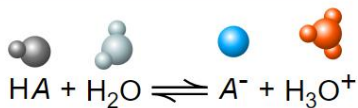
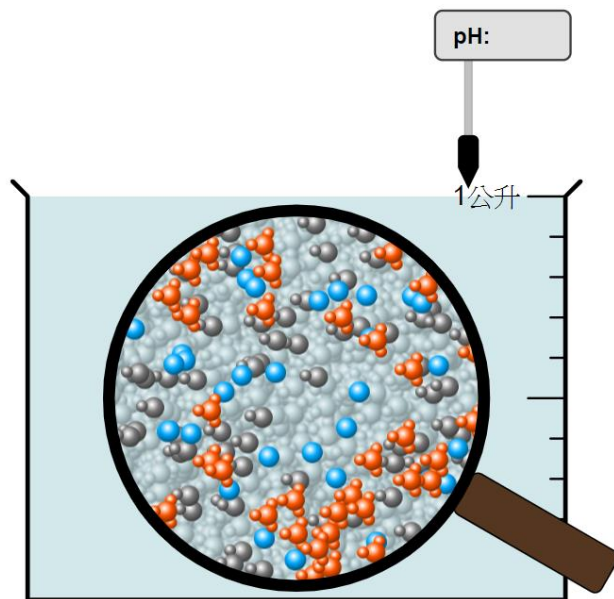
動手實作



動手實作



■PhET 模擬教學



溶液

酸 ☐ 鹼

起始濃度:

0.022

0.001 0.01 0.1 1

酸鹼的強度:

弱 ☐ 強

軟弱 軟強

顯示

☒ 分子 

☒ 溶劑 

☐ 圖表 

☐ 隱藏放大鏡 

工具



驗溶液



介紹



自訂溶液



動手實作



– VISUALGO

- <https://visualgo.net/>

The screenshot displays the Visualgo website interface. At the top, there's a navigation bar with 'VISUALGO.NET', a language dropdown set to 'en', and links for 'Training', 'Translation', and 'Login'. A 'Do You Know?' message states that the site is the English default interface and will gradually become multilingual. A 'Next Random Tip' button is also present. The main content area features a grid of algorithm visualizations, each with a title, a brief description, and a 'Training' button. The algorithms shown include: Sorting (bar chart), AND (truth table), Hash Table (hash function), Binary Heap (tree diagram), Binary Search Tree (tree diagram), Graph Structures (graph diagram), Union-Find DS (tree diagram), Segment Tree (tree diagram), and Fenwick Tree (array diagram). Each visualization is accompanied by a small diagram illustrating the algorithm's concept.

Do You Know?
This is the English (default) interface for <https://visualgo.net>. Visualgo will gradually grow into a multilingual site. Try visiting the other versions of Visualgo, e.g. Chinese: [Indonesia: \[Russian\]\(#\)](#), etc (see [statistics](#) page).

Next Random Tip

visualizing data structures and algorithms through animation

Search

Featured story: Visualizing Algorithms with a Click

Sorting
array | algorithm | bubble | quick | merge | selection | insertion

AND
0 0 1 1
0 0 0 1
0 0 0 1

Hash Table
array | addressing | linear | quadratic | random | random | random

Binary Heap
priority queue | insertion | extract | insert | delete

Binary Search Tree
priority queue | insertion | delete | insert | delete

Graph Structures
tree | spanning | shortest | path | cycle | cycle

Union-Find DS
path | compression | disjoint | set | data structure | union find

Segment Tree
segment | range | sum | min | max | random | range

Fenwick Tree
range | range | sum

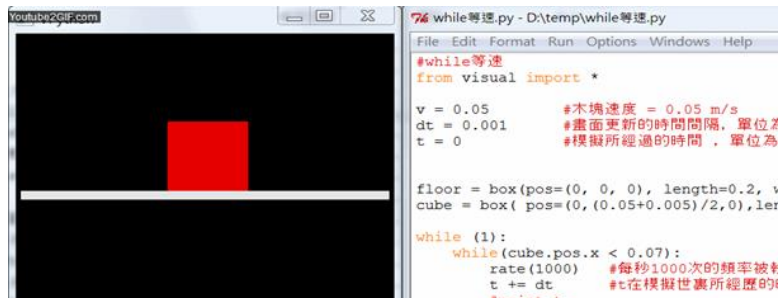


資訊科技教學策略- STEM/STEAM教學

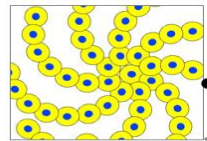
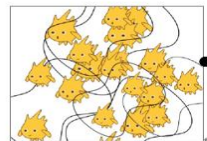
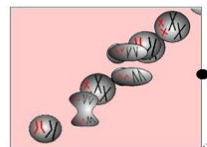
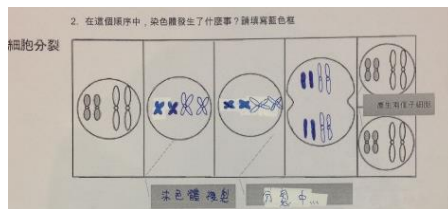
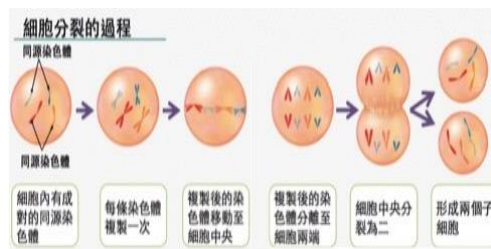
STEM舉例

- 透過撰寫程式模擬物理現象

- 學生可體驗程式設計如何應用在實務的科學問題上。由於必須寫程式模擬物理行為並建立物理模型，學生能深刻體會如何透過程式的邏輯模擬出物理現象，且在調整程式碼中物理公式的參數過程中，可觀察物理現象是否正確以檢驗程式設計的邏輯，並更加明白物理現象

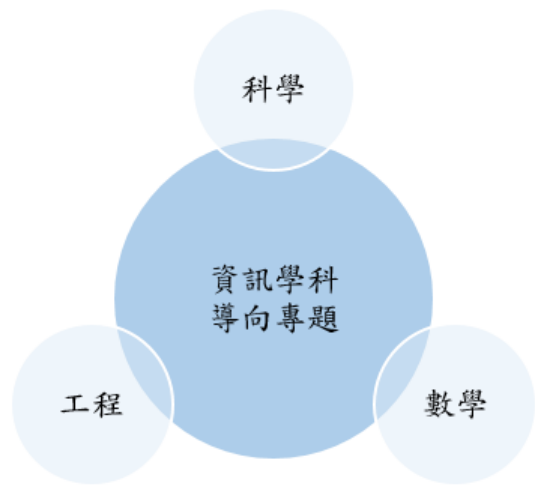


- 資訊科技亦可與生物科進行科際整合，透過模擬細胞分裂，學習不斷重複的迴圈、遞迴等概念

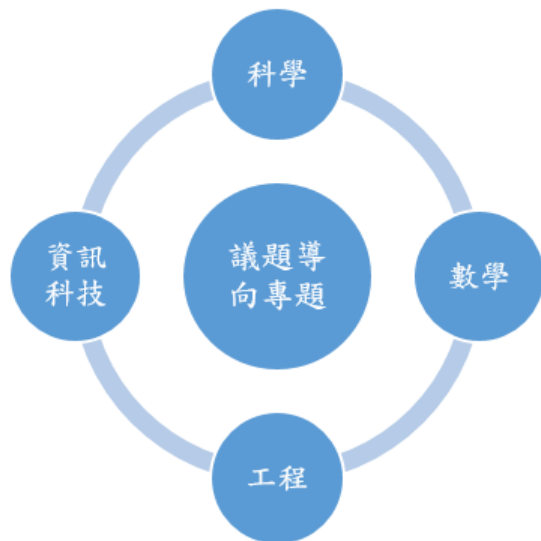


內涵重點

主題	重點
STEM科際 整合資訊 科技教學	• STEM教學設計 • 學科導向STEM教學設計 • 議題導向STEM教學設計 • STEM專題實例 • 範例1：以運算方式建構生態系統模型 • 範例2：太空探索 Space Exploration Robo Blockly project • 範例3：機器人學代數 Algebra 1 with Computing and Robotic • 範例4：科學與電腦科學跨領域課程 • STEAM • 課程範例：人文藝術裡的coding • 開放硬體



學科導向 STEM 教學設計



議題導向 STEM 教學設計

以運算方式建構生態系統模型

一、STEM專題實例1：以運算方式建構生態系統模型。

本專題讓學生在獵物與捕食者的相互作用中建構生態系統的運算模型(如圖 8-3)，同時學會運用程式設計工具進行邏輯表達與問題解決。

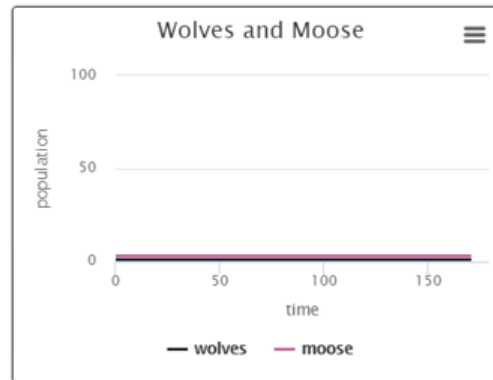
- 教學設計模式：資訊學科導向 STEM 專題。
- 資訊科技學習內容：文字式程式設計、資料處理表示與分析。
- 運算思維：模式化與模擬。
- STEM 領域知識：生物進化（統一與多樣性）、生態系統（相互作用與動力學）。

以運算方式建構生態系統模型

Isle Royale



Powered by NetLogo



setup



太空探索

二、STEM專題實例2：太空探索

本專題（**Space Exploration Robo Blockly Project**）透過日食圖的繪製，幫助於學習物理科學中的運動學、位置計算與單位轉換（如圖 8-4）。運用程式設計與機器人活動讓學生學習以運算思維來解決現實世界中的複雜問題。

- 教學設計模式：資訊學科導向 STEM 專題
- 資訊科技學習內容：結構化程式設計（視覺化程式設計）
- 運算思維：抽象化-以運算模型描述太陽、地球與月球之間的關係
- STEM 領域知識：地球科學-日食原理、物理-運動學、數學-單位轉換

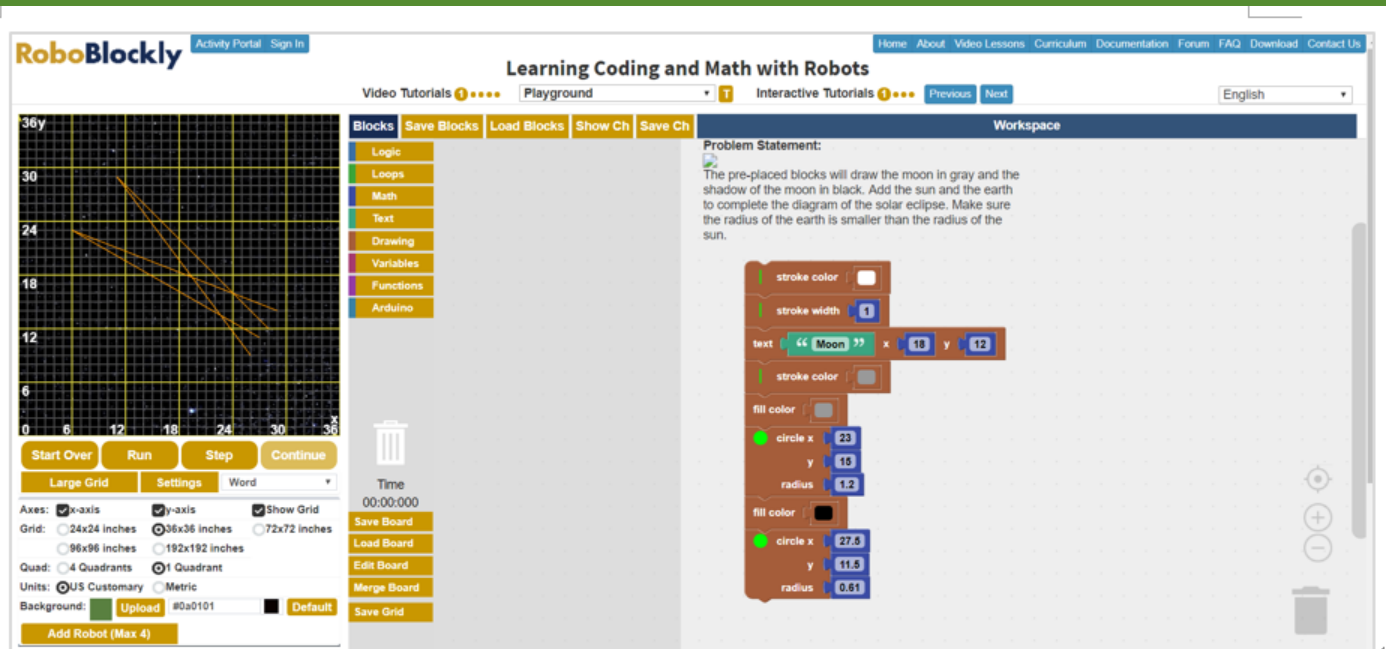


圖 8-4 太空探索專題- 《Draw a Solar Eclipse》| RoboBlockly

三、STEM專題實例3：機器人學代數

本專題（**Algebra 1 with Computing and Robotic**）以機器人為主角，設定視覺化情境，讓機器人重現物理現象，引導學生完成代數 1 的學習主題，同時教授學生程式設計與運算思維。學生透過分析真實狀況規畫演算流程與解決方案，並利用程式實作與測試修正分析結果正確性，以強化數學概念的知識。

- 教學設計模式：資訊學科導向 STEM 專題
- 資訊科技學習內容：程式設計、資料表示與分析
- 運算思維：問題拆解、樣式辨識、抽象化、演算法設計
- STEM 領域知識：數學（線性方程組和不等式、多項式函數、指數函數、機率、統計）、物理

四、STEM專題實例4：科學與電腦科學跨領域課程

本專題以 Python 為程式設計工具，運用科學相關運算工具與方法，納入電腦科學議題，進行一系列跨領域專題活動，包含數位語音處理、網格滲透運算、物理模擬系統、蛋白質交互作用分析。

- 教學設計模式：議題導向 STEM 專題
- 資訊科技學習內容：程式設計、資料表示與分析
- 運算思維：問題拆解、抽象化、樣式辨識
- STEM 領域知識：數學、物理、生物、物件導向設計、科學史、運算的限制、未來的運算方法-DNA 運算與量子運算

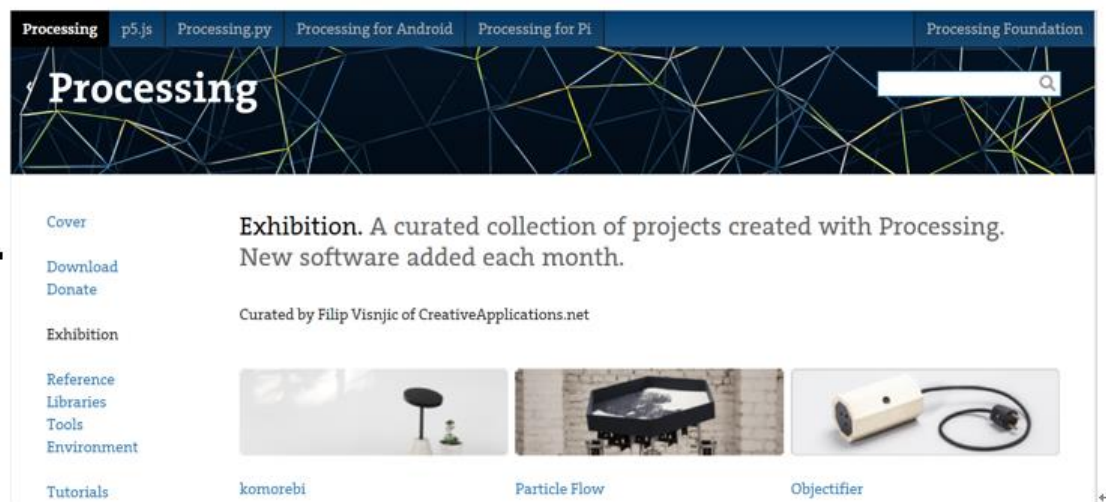


圖 8-5 Processing Exhibition (<https://processing.org/exhibition/>)

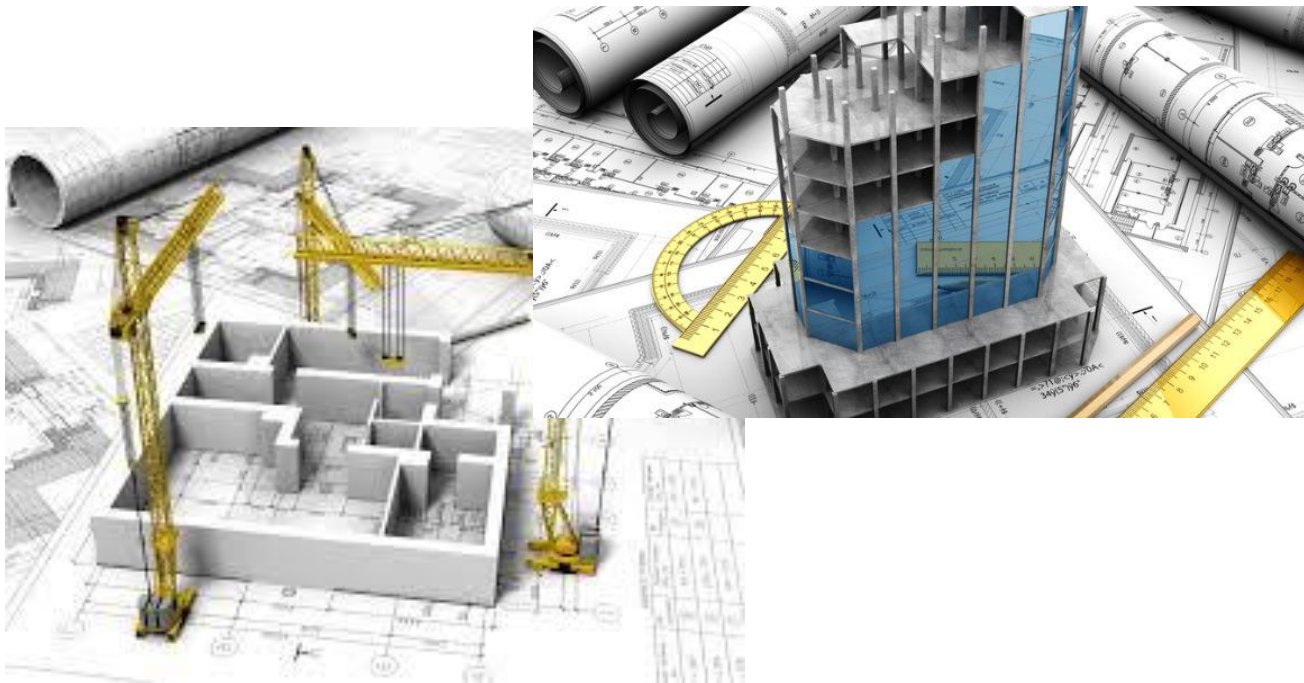
本專題使用 Processing 互動圖像程式設計語言，結合 Arduino 互動感測器與 Kinect 空間互動媒介進行實體混和媒體的互動作品設計，培養學生跨域創意設計的能力。從「應用目的思考」、「創意修改」到「深入探索」方式逐步啟發學生對 Algorithmic Art 的興趣。Algorithmic Art 是一種新興影像藝術，能運用電腦圖學與數學函式產生一些難以畫出或難以拍攝的影像。Processing 相關應用如圖 8-5。

STEM/STEAM教學



■ 建築

- 建築學
- 美學
- 物理學
- 結構力學
- 工程力學
- 材料力學



STEM/STEAM教學



■ 西北大學發展之CT-STEM教學

- 美國國家科學基金會(National Science Foundation, NSF)與西北大學執行了一個Computational Thinking in STEM (CT-STEM)計畫，設計了一系列STEM的課程來教導高中生運算思維，同時也舉辦了許多教師工作坊，以進行CT-STEM教學之訓練，每個課程單元皆包含STEM四種領域的知識，以及運算思維元素
- <https://ct-stem.northwestern.edu/>

STEM/STEAM教學



■ 西北大學發展之CT-STEM教學範例

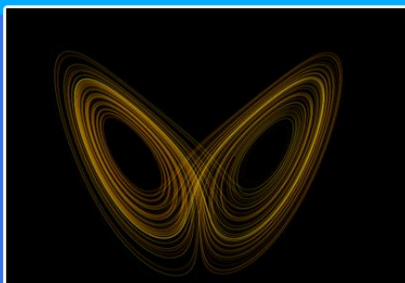
DA (Data Analysis), MS (Modeling & Simulation), CPS (Computational Problem Solving), ST (Systems thinking)

單元名稱 知識領域	The Physics of Angry Birds	Determination of Ohm's Law from Physical Data	Investigation of Resonance Using Computer Simulated Experiments	Modifying Code Introduction to Netlogo: A Newton's Law of Gravity Simulation
S	動量守恆、牛頓力學	歐姆定律、理解原子分子	駐波、諧音	牛頓力學
T	使用試算表，推估方程式	使用麵包板、電阻器	電腦	電腦
E	使用電腦模擬產生資料	使用試算表	使用JAVA模擬與設計	電腦模擬，Netlogo 簡易 coding
M	計算方程式	計算資料，推出歐姆定律	計算方程式	公式理解與計算
CT	DA, MS, CPS	DA, MS	DA, MS, ST	DA, MS, CPS

STEM/STEAM教學

[Curricula](#)[Intro to CT](#)[Teacher Support ▾](#)[About ▾](#)[Register](#)[Login](#)

Browser-friendly CT lesson plans for your classroom



LESSON PLAN - ENVIRONMENTAL SCIENCE

40-50 min

Intro to learning with Computational Models

by Sugat Dabholkar
Standards: NGSS

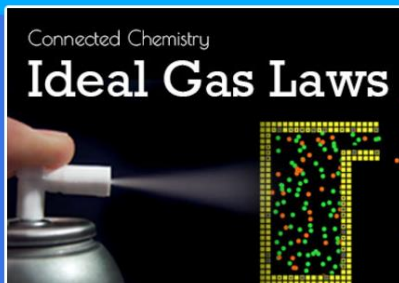


UNIT - BIOLOGY

6 Lessons

Evolution of Populations

by Sugat Dabholkar
Standards: NGSS, CT-STEM



Connected Chemistry

Ideal Gas Laws

UNIT - CHEMISTRY

6 Lessons

Ideal Gas Laws - Connected Chemistry 2019

2 Authors
Standards: NGSS, CT-STEM



STEM/STEAM教學



CT-STEM Preview - Ideal Gas Laws - Connected Chemistry 2019

powered by NetLogo CC2019-M2-Pressur...

File: New
Export: NetLogo HTML

Mode: Interactive Commands and Code: Bottom

model speed
ticks: 249

initial-number 97
number-to-add 71
+++ add particles
show-wall-hits? ☒

Command Center

CT-STEM Preview - Ideal Gas Laws - Connected Chemistry 2019

observer v Clear

NetLogo Code

Jump to Procedure Recompile Code ☐ Auto Complete

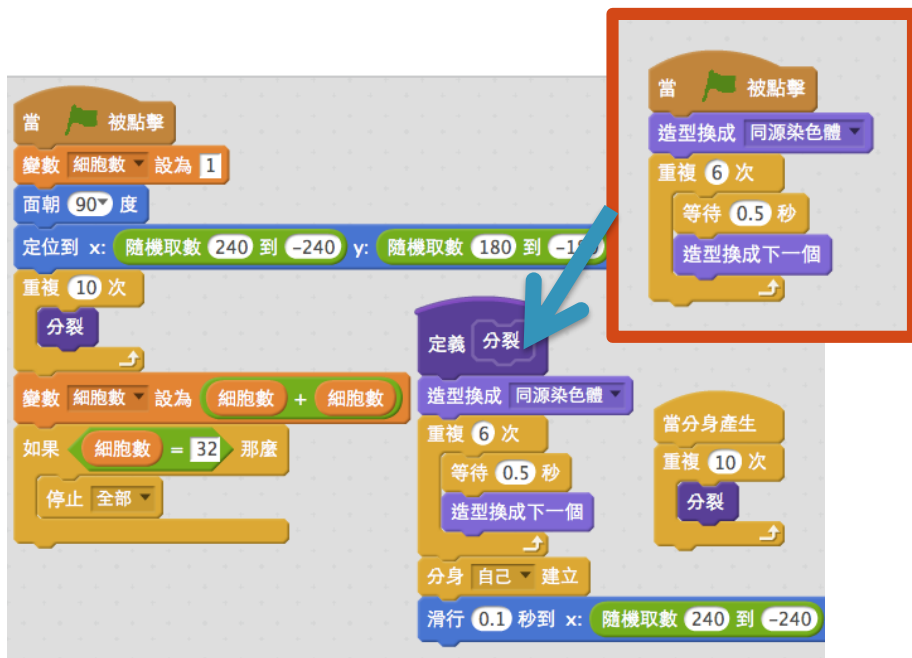
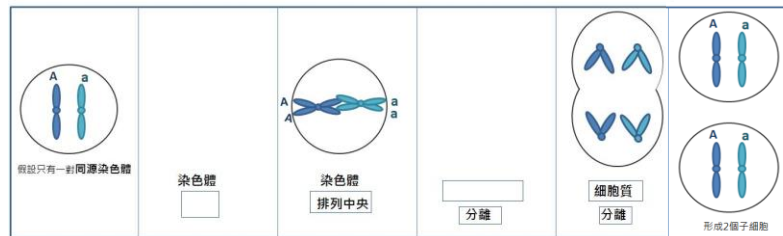
Disabled

```
66 to go
67   let old-clock 0
68
69   ask particles [ bounce ]
70   ask particles [ move ]
71   if collisions? [
72     ask particles
73       [ check-for-collision ]
74
75   ]
76   set old-clock ticks
77   tick-advance tick-advance-amount
78
79   if floor-ticks > floor (ticks - tick-advance-amount) [
80     ifelse any? particles
81       [ set wall-hits-per-particle mean [wall-hits] of particles ]
82       [ set wall-hits-per-particle 0 ]
83     ask particles [ set wall-hits 0 ]
84
85   ]
86   calculate-tick-advance-amount
87   ask flashes with [ticks - birthday > 0.4] [
88     die
89   ]
90
91   if (show-wall-hits? = false) [ ask flashes [ die ] ]
92   display
93 end
94
95
96 to calculate-tick-advance-amount
```

STEM/STEAM教學



■ 細胞分裂



data 換算方法

data 索引方法

重複樣式

	重複樣式		
	Data 1	Data 2	Data 3
1	(0,0)	100%	
2	(50,0)	98%	
4	:	96%	
8	:	94%	
16	:	92%	
32	:	90%	
64	:	88%	
128	:	86%	
256	:	84%	
—	—	—	

方法
 方法
 共 8 次
 一次轉 45 度
 →

↓
共
8
次
一次轉
45
度

無法再分裂→

分裂時: 1次 →

很多次 →

右轉45度
移動50步

2ⁿ個

2ⁿ⁺¹個

STEM/STEAM教學



■ 模擬與建模

- 2-11:請觀察速率、時間、距離之間的關係〈選擇題〉

(A) 相同距離的情況下，速率與時間成正比

(B) 相同距離的情況下，速率與時間成反比

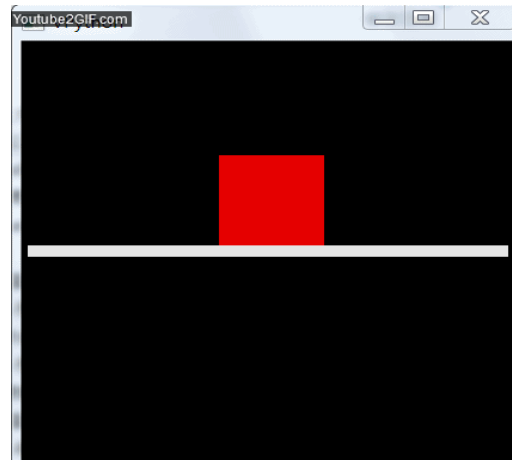
答案：_____

- 2-12:想一想，攀岩高手爬一趟的平均速率為多少？

◇ 平均速率=總距離（上山+下山）／總時間（上山時間+下山時間）

◆ 平均速率=（30+30）／（6+3）=60／9 = 20／3

- 2-13:請觀察平均速率的時間與距離



7% while等速.py - D:\temp\while等速.py

File Edit Format Run Options Windows Help

```
#while等速
from visual import *
```

```
v = 0.05          #木塊速度 = 0.05 m/s
dt = 0.001        #畫面更新的時間間隔，單位為
t = 0             #模擬所經過的時間，單位為
```

```
floor = box(pos=(0, 0, 0), length=0.2, v
cube = box( pos=(0, (0.05+0.005)/2,0), ler
```

```
while (1):
    while (cube.pos.x < 0.07):
        rate(1000)    #每秒1000次的頻率被執
        t += dt        #t在模擬世界裏所經歷的t
        #print t
        cube.pos.x += v*dt    #位移=速度*時
        #print cube.pos.x
    print t
```

速率(藍球)第_____小時，距離經過了40.00 公里

速率(藍球)第8 小時，距離經過了_____ 公里

```
>>>
上山 (紅球) 第1小時，距離走了5公里
上山 (紅球) 第2小時，距離走了10公里
上山 (紅球) 第3小時，距離走了15公里
上山 (紅球) 第4小時，距離走了20公里
上山 (紅球) 第5小時，距離走了25公里
上山 (紅球) 第6小時，距離走了30公里
下山 (綠球) 第1小時，距離走了10公里
下山 (綠球) 第2小時，距離走了20公里
下山 (綠球) 第3小時，距離走了30公里
```

```
平均速率 (藍球) 第1小時，距離走了6.67公里
平均速率 (藍球) 第2小時，距離走了13.33公里
平均速率 (藍球) 第3小時，距離走了20.00公里
平均速率 (藍球) 第4小時，距離走了26.67公里
平均速率 (藍球) 第5小時，距離走了33.33公里
平均速率 (藍球) 第6小時，距離走了40.00公里
平均速率 (藍球) 第7小時，距離走了46.67公里
平均速率 (藍球) 第8小時，距離走了53.33公里
平均速率 (藍球) 第9小時，距離走了60.00公里
>>> |
```

教學活動

根據科學建模歷程(Sins et al., 2005b)與物理建模歷程(Ramaila, 2014)發展出適合本研究之STEM程式設計教學建模學習程序。學習者於學習活動階段與專題製作階段，經歷「分析」、「演算法設計」、「程式化」與「解釋」四個建模程序。

■ 題目 2

有一架噴射機在跑道上，如果從靜止開始滑行 5 秒後，達到 180 公里/時的速度而起飛，則此飛機在跑道上滑行時，平均加速度量值為多少 公尺/秒²，前進多少公尺。請用程式模擬噴射機在跑道上速度與位置。

✧ 小提示：

◆ 平均加速度：單位時間內的速度變化量。

◆ 公式 1: $\bar{a} = \frac{\Delta v}{\Delta t} = \frac{v_2 - v_1}{t_2 - t_1}$

◆ 公式 2: $\Delta x = \frac{1}{2} (v_{end} + v_{start}) * t$

◆ 公式 3: $180(\text{ km/h }) = \frac{180 \times 1000}{60 \times 60} = 50 (\text{ m/s })$

2-1:想一想，如何從空白的畫面，畫出題目要求的動畫？

2-2:想一想，要如何完成任務呢？

Q：需要知道哪些條件？

A：噴射機初始速度是靜止，為 0

噴射機在跑道上起飛前的速度是 50 m/s

時間從 0 開始經過 5 秒

2-3:根據以上的條件，我們可以先設定畫面與變數，噴射機初始位置假設在(-125.0,0.5,0.0)：

```
from visual import *
#變數設定-----
v_start=0 #m/s #初速度
v_end=50 #m/s #末速度
t_start=0 #初時間
t_end=5 #末時間
dt = 0.001
#畫面設定-----
scene = display(width=2000, height=800, background=(0.5,0.5,0)) #設定畫面
floor = box(length=250, height=0.01, width=4, color=color.blue) #跑道
ball = sphere(pos=(-125.0,0.5,0.0), radius = 0.5, color=color.red) #噴射機
```

While 單元加速度之學習單

分析

教學活動

While 單元加速度之學習單

演算法設計

2-4: 已經知道噴射機的初始速度、在跑道上起飛前的速度、時間，請計算噴射機的加速度：

✧ 小提示：

(A) 公式 1: $\bar{a} = \frac{\Delta v}{\Delta t} = \frac{v_2 - v_1}{t_2 - t_1}$

(B) 公式 2: $\Delta x = \frac{1}{2} (v_{end} + v_{start}) * t$

#計算加速度與位置

a = (v_end - v_start) / (t_end - t_start) #加速度

print "噴射機的加速度是", a

x = (v_end + v_start) * t_end / 2

#噴射機從靜止到起飛前的距離

print "噴射機從靜止到起飛前的距離", x

2-5: 已經知道噴射機加速度與時間，想一想怎麼轉換程式碼顯示動畫？

因為 vpython 顯示動畫是 3D 空間，速度是有方向與大小的，在這邊先將速度轉換成 3D 向量。

#速度有方向與大小，在 vpython 中用向量表示

ball.v = vector(0.0, 0.0, 0.0) #噴射機初始速度向量

2-6: 已經知道噴射機加速度與時間，想一想如何用程式碼模擬噴射機起飛前的狀態：

當噴射機的位置小於等於噴射機起飛前的位置時

1. 噴射機每秒結束後都會有新的速度值

2. 噴射機每秒都會前進

#物體運動-----

while ball.pos.x <= x:

rate(1000)

ball.pos = ball.pos + ball.v * dt * 0.5

ball.v.x = ball.v.x + a*dt

#當噴射機的位置小於等於起飛位置時

#每秒動1000次

#噴射機每秒的距離 (V+Vo)*t/2

#噴射機每秒速度 V=Vo+a*t

教學活動

While 單元加速度之學習單

程式化

```
# -*- coding: utf8 -*-
```

```
#飛機起飛  
from visual import *
```

```
#初始速度  
v_start=0
```

```
#終止（起飛那一瞬間）速度  
v_end=50
```

```
#初始時間  
t_start=0
```

```
#終止（起飛那一瞬間）時間  
t_end=5
```

```
#設定間隔時間長短以便觀察  
dt=0.01
```

```
#設定畫面  
scene=display(width=2000, height=800, background=(0.5,0.5,0))
```

```
#以地板代表跑道  
floor=box(length=250, height=0.01, width=4, color=color.blue)
```

```
#以球代表飛機  
airplane=sphere(radius=0.5, color=color.red)
```

```
#設定飛機初始位置x,y,z都要設定  
airplane.pos.x=0  
airplane.pos.y=0.5  
airplane.pos.z=0
```

```
#設定飛機初始速度  
airplane.velocity = vector(0,0,0)
```

```
#利用公式計算加速度(末速-初速)/(末時間-初時間)  
a=(v_end-v_start)/(t_end-t_start)  
print "加速度",a
```

```
#利用公式計算靜止到起飛的距離(末速+初速)*所花時間*0.5  
x=(v_end+v_start)*5*0.5  
print "從靜止到飛行前距離",x
```

```
#利用while迴圈一步一步畫出運行的過程  
#飛機準備從第0秒開始繪製這5秒的過程所以時間歸零  
t=0  
#當 飛機的x位置 還未超過 滑行距離 的前提下要執行迴圈  
while airplane.pos.x < x: #記得while語法後面有一個冒號
```

```
#把動作切的很細0.01秒一步一步請電腦畫上去  
print "現在計算",t,"秒到",t+dt,"秒的情況"
```

```
#初速就是當下飛機的速度  
v0=airplane.velocity.x
```

```
#末速就是預算0.01秒的情況 末速=初速+加速度*這段小小的時間  
v=v0+a*dt
```

```
#計算這0.001秒的移動距離 (末速+初速)*所花0.01秒*0.5  
dx=(v0+v)*dt*0.5
```

```
#把計算結果印出來看一看  
print "初速",v0,"末速",v,"移動距離",dx  
print "從原點到目前距離",airplane.pos.x  
print "-----"
```

```
#真正的去改變飛機的現況  
#1.改變飛機的x座標(距離累加上)  
airplane.pos.x=airplane.pos.x+dx
```

```
#2.改變飛機的速度(初速再加上這次的增加)  
airplane.velocity.x=airplane.velocity.x+a*dt
```

```
#準備下一回合時間累增0.01秒例如3.45秒進到3.46秒  
t=t+dt
```

```
#為了畫面更新速度  
rate(10000)
```

教學活動

While 單元加速度之學習單

解釋與評估

```
*Python 2.7.10 Shell*
File Edit Shell Debug Options Window Help
初速 33.3 末速 33.4 移動距離 0.3335
從原點到目前距離 55.4445
-----
現在計算 3.34 秒到 3.35 秒的情況
初速 33.4 末速 33.5 移動距離 0.3345
從原點到目前距離 55.778
-----
現在計算 3.35 秒到 3.36 秒的情況
初速 33.5 末速 33.6 移動距離 0.3355
從原點到目前距離 56.1125
-----
現在計算 3.36 秒到 3.37 秒的情況
初速 33.6 末速 33.7 移動距離 0.3365
從原點到目前距離 56.448
-----
現在計算 3.37 秒到 3.38 秒的情況
初速 33.7 末速 33.8 移動距離 0.3375
從原點到目前距離 56.7845
-----
現在計算 3.38 秒到 3.39 秒的情況
初速 33.8 末速 33.9 移動距離 0.3385
從原點到目前距離 57.122
-----
現在計算 3.39 秒到 3.4 秒的情況
初速 33.9 末速 34.0 移動距離 0.3395
從原點到目前距離 57.4605
-----
現在計算 3.4 秒到 3.41 秒的情況
初速 34.0 末速 34.1 移動距離 0.3405
從原點到目前距離 57.8
-----
現在計算 3.41 秒到 3.42 秒的情況
初速 34.1 末速 34.2 移動距離 0.3415
從原點到目前距離 58.1405
-----
現在計算 3.42 秒到 3.43 秒的情況
初速 34.2 末速 34.3 移動距離 0.3425
從原點到目前距離 58.482
```

2-7:請問此題目噴射機的平均加速度量值為 公尺/秒²，前進 公尺。

2-8:試試看 `print ball.pos`，觀察噴射機每秒的距離。

2-9:試試看 `print ball.v.x`，觀察噴射機每秒的速度。

2-10 ☐ 我可以完成任務 ☐ 我不能完成任務 (請圈起來)

STEM/STEAM教學



• 作業

電腦+物理課後作業

班級：_____ 座號：_____ 姓名：_____

#同步衛星與繞極衛星 (衛星的大小不符實際比例)

from visual import *

G = 6.7e-11 #重力常數

t = 0

dt = 0.0001

scene = display(width=800, height=800)#場景

earth = sphere(radius=6378.1, material=materials.BlueMarble)#地球半徑,球的材質

earth.mass = 5.972e24#地球質量

mu = G*earth.mass#重力常數*地球質量

ge_satellites=[]#三個同步衛星

for i in range(0,3,1):

ge_satellites.append(sphere(radius=1000, material=materials.rough))#增加同步衛星的半徑與材質

ge_satellites[i].r = 42164#同步衛星與地球的距離

ge_satellites[i].av = (mu/ge_satellites[i].r**3)**0.5

arrows=[]#繞極衛星

for i in range(0,3,1):

arrows.append(arrow(shaftwidth=100))#增加繞極衛星軸寬度

PO_satellite = sphere(radius=1000, make_trail=True, retain=1000, material=materials.rough, color=(0.7,0.7,0.7))#繞極衛星的半徑顏色等資訊

PO_satellite.r = 6378.1+1500#繞極衛星與地球的距離

PO_satellite.av = (mu/PO_satellite.r**3)**0.5

earth.av = (mu/(ge_satellites[0].r)**3)**0.5

while True:

rate(1000)

t += dt#每秒

for i in range(0,3,1):

ge_satellites[i].x = ge_satellites[i].r*math.cos(-ge_satellites[i].av*t+i*120*pi/180)#同步衛星x軸

ge_satellites[i].y = ge_satellites[i].r*math.sin(-ge_satellites[i].av*t+i*120*pi/180)#同步衛星y軸

ge_satellites[i].pos = Vector(ge_satellites[i].x, 0, ge_satellites[i].z)#同步衛星位置,固定

arrows[i].pos = ge_satellites[i].pos#同步衛星位置的座標置到繞極衛星位置

arrows[i].axis = -0.85*ge_satellites[i].pos

PO_satellite.y = PO_satellite.r*math.cos(-PO_satellite.av*t)#繞極衛星y軸

PO_satellite.z = PO_satellite.r*math.sin(-PO_satellite.av*t)#繞極衛星z軸

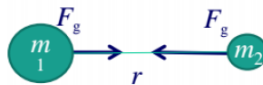
4. for i in range(0,3,1):

print i

i 會是：_____

5. 宇宙間的一切物體都是互相吸引的，
兩個物體間的引力大小，跟他們的
質量的乘積成_____比，跟他們的
_____成反比。

2.請看物理課本，萬有引力定律公式為：



3. 請看物理課本，

m1、m2 為：_____

r 為：_____

Fg 為：_____

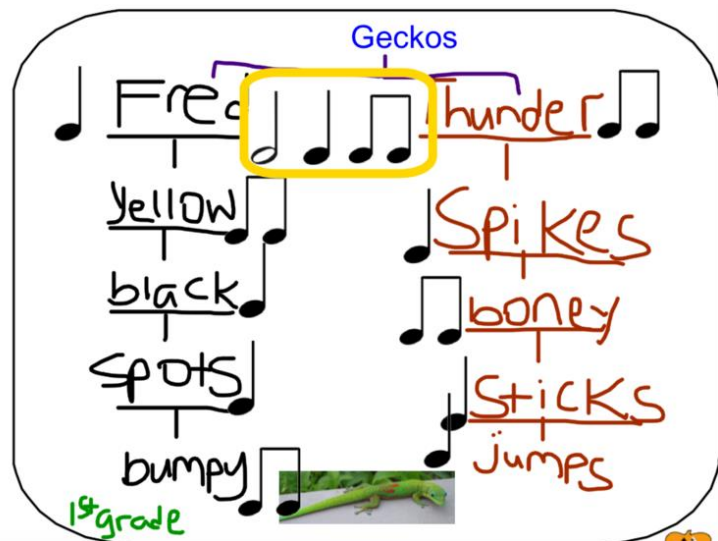
STEM/STEAM教學



■ 音樂科技

- <http://amymburns.com/>
- <http://amymburns.com/blogcc>
- 一年級學生在科學課中觀察兩隻壁虎，名叫弗雷德和雷霆，並為牠們作曲。學生必須建立一張思維圖（樹狀圖），顯示了每隻壁虎的特徵，接著選擇與特徵匹配的節奏作曲，並在課堂用樂器演奏。

■ <https://www.youtube.com/watch?v=ViQawYvAvnI>



STEAM



- 用機器人Ozobot敘說謀殺之謎的故事
 - 程式設計與寫作是很類似的
 - 匹茲堡蒙圖爾高中十年級英語文學老師Gina Ligouri要求學生創作謀殺之謎故事，並需具有清晰的情節、人物、動機和結果。學生們編寫好故事後，須製作視覺故事板，並使用 Ozobots 在板上移動並與學生分享他們的故事。



<https://ozobot.com/blog/creator-of-the-month-ms-ligouris-murder-mystery-stories>

運算思維導向的教材教法

透過**探索**與**思考**的學習活動讓學生體驗運算思維，且每種教學活動皆有其所欲強調的主要運算思維，以引導此教學活動的設計理念與教學內涵的主軸



- 影音操作
- 任務探索

運算思維

- 將學習概念對應到主要的運算思維
- 體驗運算思維

學習內容

教師透過範例程式進行程式概念與語法解說，或是講解資訊科學內容

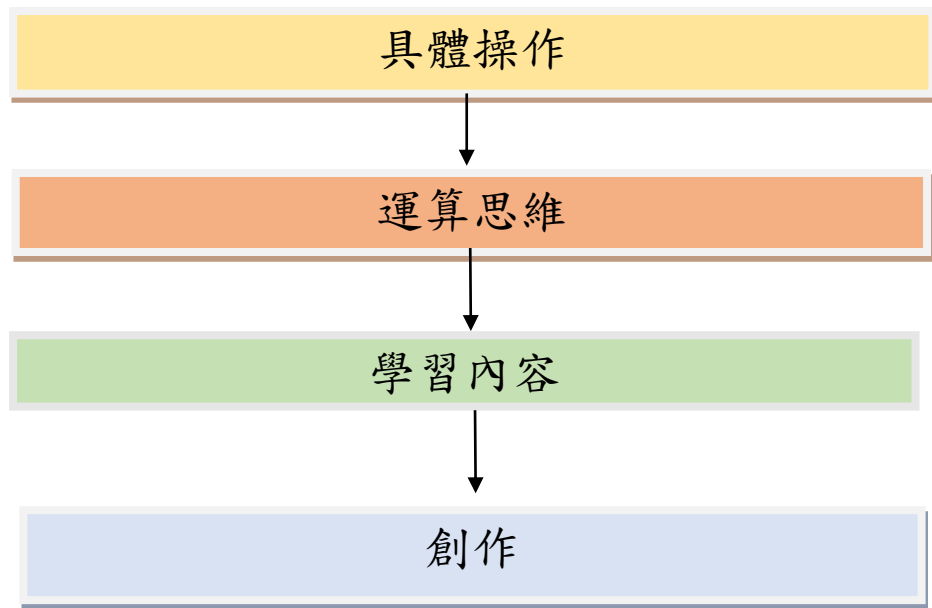
專題實作

- 整合知識、能力（包含技能）與態度
- 重視情境與脈絡的學習
- 重視學習的歷程、方法及策略
- 強調實踐力行的表現



圖 5-12 「模組化程式設計-動手玩音樂」運算思維教學活動流程

運算思維導向之教學活動流程設計



資訊科技學習表現



運算思維與問題解決

能具備運用運算工具之思維能力，藉以分析問題、發展解題方法，並進行有效的決策。



資訊科技與溝通表達

能利用資訊科技表達想法並與他人溝通。



資訊科技與合作共創

能利用資訊科技與他人合作並進行創作。



資訊科技的使用態度

能建立康健、合理與合法的資訊科技使用態度與習慣，並樂於探索資訊科技。

示例 1：迴圈程式設計

- 傳統教學
 - ✓ 我們可能直接告訴學生迴圈的指令與撰寫方式
- 運算思維導向的教學
 - ✓ 便可透過讓學生去觀察指令的重複現象
 - ✓ 並辨識出重複的指令樣式，以培養其歸納重複指令的樣式來進行迴圈程式設計的能力。
 -
 - ✓ 可在學習單中加入圈選重複指令的活動，以明確引導學生思考如何進行樣式辨識。

示例 2: 資料數位化

- 傳統教學

- ✓ 經常直接教導學生二進制、十進制、與十六進制等不同進制之間的換算，學生可能會誤以為資料數位化是數學運算。

- 在運算思維導向的教學中

- ✓ 應更強調為了儲存、傳播與運算資訊
- ✓ 必須用合適的資料表示格式或演算法
- ✓ 因此可設計一些教學活動，讓學生體驗資料的表徵方式，以引導學生思考如何進行資料表示

十二年國教資訊科技教學

- 強調透過運算思維的培養讓學生具備問題解決的高階能力
- 如何幫助學生在實作任務中學會問題解決所需的程序與策略，培養運算思維能力並理解學習內容，是資訊科技教學設計的重點
- 而如何引導學生從分析問題、理解概念、仿作到獨立思考解決問題，也是教學實施階段的重要任務

動機引導與探討 ↗	播放範例音樂盒引起動機，鼓勵學生製作自己喜歡的兒歌音樂盒。 ↗ 自選曲目：引導學生選擇喜愛的樂曲。 ↗ 1. 2 人一組解決部分學生識譜能力不足的問題。教師可提供範例樂譜，或依課堂時間而定讓學生自行找譜。 ↗ 2. 識譜練唱。 ↗	■ 範例樂曲 ↗ ■ 範例樂譜 ↗
運算思維歷程 ↗	1. 問題拆解、樣式辨識。 ↗ 樂句結構分析：在樂譜學習單上寫出樂句，並標示出樂句代號。 ↗ 2. 抽象化：樂句函式定義音符數值化。 ↗ 定義每個樂句內容：對應樂譜，將音高與節拍數值化。 ↗ 3. 演算法設計：樂曲描述。 ↗ 安排樂句流程控制，並以代號與程式流程描述樂曲演奏。 ↗	■ 學習單 ↗

程式概念解說
程式實作

1. 樂曲程式寫作。
 - (1) 函式概念解說。
 - (2) 定義樂句函式-依據對應的數值描寫樂句函式。
 - (3) 以程式流程描述樂句演奏順序。
 - (4) 試聽並調整，完成兒歌音樂自動播放程式，產出完成檔。



個性化兒歌自選曲 - 樂曲程式寫作

2. 變奏曲實作。
 - (1) 修改樂曲演奏流程、音符、樂器與演奏速度。
 - (2) 分組合作共同創作編寫音樂盒程式。
 - (3) 兒歌變奏程式實作與測試修正。

- 學習單
- 程式實作檔

賞析

1. 線上音樂廳公開分享。
2. 小組互評與賞析：評析程式模組化與流程控制的優點。

- 線上音樂廳

領域/科目		資訊科技		設計者			
實施年級		國中 ☐ 七年級 ☐ 八年級 ☐ 九年級 高中 ☐ 十年級 ☐ 十一年級 ☐ 十二年級		總節數		共_____節，_____分鐘	
單元名稱							
設計依據							
學習 重點	學習表現						
	學習內容						
運算思維							
教材來源							
教學設備/資源							
學習目標							
教學活動設計							
教學活動內容及實施方式				時間	備註		

教學單元規劃重點

- 單元名稱
- 實施年級
- 總時數
- 學習內容
- 學習表現
- 情境引導
- 教學活動
- 學習工具
- 運算思維（必須凸顯運算思維精神）