運算思維簡介

Introduction to Computational Thinking

大綱

- 什麼是運算思維
  - 運算思維的定義
  - 運算思維的內涵
  - 運算思維與數學思維
- 運算思維的重要性
  - 運算思維與日常生活
  - 運算思維與職涯發展
- 如何培養運算思維
  - 運算思維與資訊科學
  - 運算思維與資訊科技課綱
  - 運算思維與其他學科領域
- 資訊科技課程綱要簡介

什麼是運算思維

什麼是運算思維

- 運算思維的定義
- 運算思維的內涵
- 運算思維與數學思維

什麼是運算思維-定義

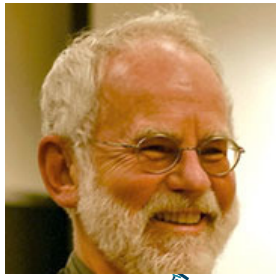
- 運算思維 ≠ 資訊科技使用能力
- 運算思維 ≠ 程式設計
- 運算思維 ≠ 資訊科學

→ 那麼什麼是運算思維?

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.

什麼是運算思維-定義

- 各國課綱所提及的運算思維定義
  - 美國CSTA[1]「電腦科學」課程
    - 一種能利用電腦解決問題的思維，包含使用如抽象化、遞迴、迭代等概念來處理與分析資料，並產出實體與虛擬作品的能力(CSTA, 2011)
  - 澳洲「數位科技」課程
    - 利用數位科技設計與實作演算法解決問題的思維(ACARA, 2013)
  - 英格蘭「運算」課程
    - 一種凌駕於電腦軟硬體之上，能針對系統與問題提出思考架構的思維模式(DOEE, 2013)
  - 我國「資訊科技」課程
    - 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(國教院，2015)

註1：美國資訊科學教師組織（Computer Science Teachers Association, CSTA）

什麼是運算思維-定義

- ISTE[2]與CSTA對運算思維的操作型定義
  - 運算思維是一個包含(但不限於)以下特色的問題解決歷程
    - 把問題表示為一個可以讓我們用電腦和其他工具去解決的形式
    - 邏輯地組織與分析資料
    - 將資料透過模型化或模擬等方式進行抽象化表示
    - 透過演算法思維將解法自動化
    - 能識別、分析與實作可行的解法，以達到有效整合解題程序與資源的目的
    - 一般化和轉換這個問題解決程序到適合各種不同的問題情境

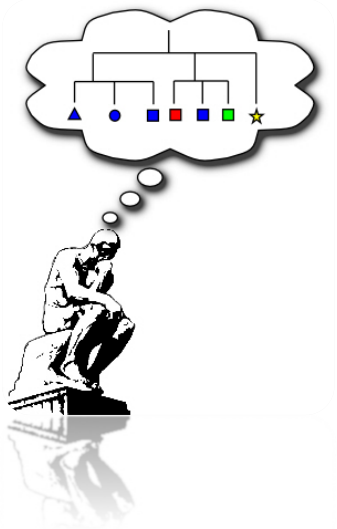
註2：美國國際科技教育應用協會（The International Society for Technology in Education, ISTE）

資料來源

- 美國CSTA「電腦科學」課程
  - CSTA (2011). CSTA K–12 computer science standards. The ACM K-12 Education Task Force. Retrieved from http://www.csta.acm.org/Curriculum/sub/CurrFiles/CSTA_K-12_CSS.pdf
- 澳洲「數位科技」課程
  - Australian Curriculum, Assessment, Reporting Authority (ACARA) (2013). Draft Australian curriculum technologies. Retrieved from http://consultation.australiancurriculum.edu.au/Static/docs/Technologies/Draft%20Australian%20Curriculum%20Technologies%20-%20February%202013.pdf
- 英格蘭「運算」課程
  - Department for Education in England (DOEE) (2013, September 11). National curriculum in England: Computing programmes of study. Retrieved from https://www.gov.uk/government/publications/national-curriculum-in-england-computing-programmes-of-study/national-curriculum-in-england-computing-programmes-of-study
- ISTE與CSTA的運算思維操作型定義
  - ISTE (2011). Operational Definition of Computational Thinking. Retrieved from http://www.iste.org/docs/ct-documents/computational-thinking-operational-definition-flyer.pdf?sfvrsn=2

什麼是運算思維-定義

- Google 對運算思維的定義
  - 運算思維是一種利用運算解決問題所需之「心智歷程」(抽象化、演算法設計、解析、樣式辨識等)與「具體產出」(自動化、資料表示、樣式一般化等)



什麼是運算思維-定義

- 根據Google的定義，運算思維包含
  - **抽象化**: 為定義主要概念去識別並萃取相關資訊
  - **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
  - **資料分析**: 透過歸納模式或發展深入分析方法以理解資料
  - **資料蒐集**: 蒐集與問題解決相關的資料
  - **資料表示**: 用適合的圖表、文字或圖片等表達與組織資料
  - **解析**: 將資料、程序、問題拆解成較小、較容易處理的部分
  - **平行化**: 同時處理大任務中的小任務以有效達到解題目的
  - **樣式一般化**: 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
  - **樣式辨識**: 在資料中觀察樣式、趨勢或規則
  - **模擬**: 發展模型以模仿真實世界的程序

資料來源：Google (2015). Exploring Computational Thinking. Retrieved from https://www.google.com/edu/resources/programs/exploring-computational-thinking/

什麼是運算思維-定義

- Grover & Pea認為運算思維應包含
  - 抽象化與樣式一般化
    - 模型化
    - 模擬
  - 系統化資訊處理
  - 符號系統與表示
  - 演算法與流程控制
    - 迭代、遞迴與平行思考
    - 條件式邏輯
  - 結構化問題解析
  - 效能分析
  - 除錯與系統化偵錯

什麼是運算思維-定義

Google (2010)

- Data Analysis
- Data Collection
- Data Representation
- Abstraction
- Decomposition
- Algorithm Design
- Simulation
- Pattern Generalization
- Pattern Recognition
- Automation
- Parallelization

ISTE, CSTA& NSF (2011)

- Data Collection
- Data Analysis
- Data Representation
- Abstraction
- Problem Decomposition
- Algorithms & Procedures
- Simulation
- Automation
- Parallelization

Selby (2013)

- Abstraction
- Decomposition
- Algorithmic thinking
- Generalization
- Evaluation

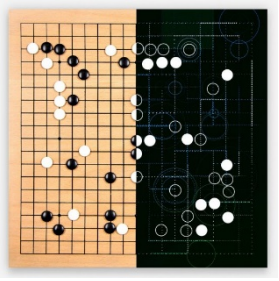
什麼是運算思維-定義

- 何謂運算思維?
- 運算思維與資訊科學的關係?

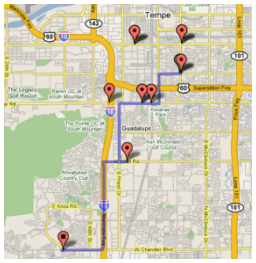
什麼是運算思維-定義

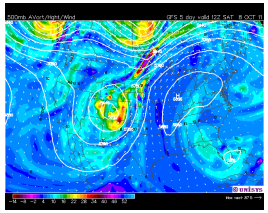
- 運算思維與問題解決

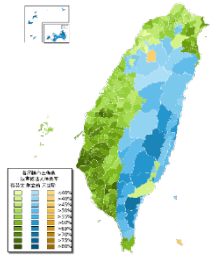
實體世界問題



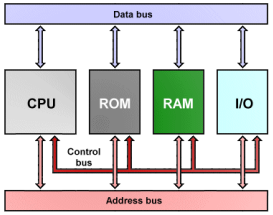
$$f'(x_0) = \lim_{h\to 0}\frac{f(x_0+h)-f(x_0)}{h} = \lim_{h\to 0}\frac{(x_0+h)^{1/2}-(x_0)^{1/2}}{h}$$
$$= \lim_{h\to 0}\frac{[(x_0+h)^{1/2}-(x_0)^{1/2}][(x_0+h)^{1/2}+(x_0)^{1/2}]}{h[(x_0+h)^{1/2}+(x_0)^{1/2}]}$$
$$= \lim_{h\to 0}\frac{(x_0+h)-x_0}{h[(x_0+h)^{1/2}+(x_0)^{1/2}]} = \lim_{h\to 0}\frac{h}{h[(x_0+h)^{1/2}+(x_0)^{1/2}]}$$
$$= \lim_{h\to 0}\frac{1}{[(x_0+h)^{1/2}+(x_0)^{1/2}]} = \frac{1}{x_0^{1/2}+x_0^{1/2}} = \frac{1}{2x_0^{1/2}} = \frac{1}{2}x_0^{-1/2}$$





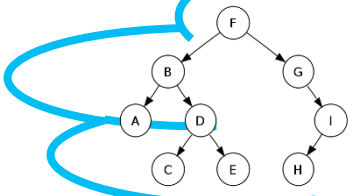


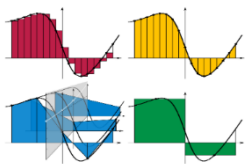
```
fun append (xs, ys) =
    if null xs
    then ys
    else (hd xs):: append (tl xs, ys)

fun map (f, xs) =
    case xs of
        [] => []
      | x :: xs' => (f x)::(map (f, xs'))

val a = map (increment, [4,8,12,16])
val b = map (hd, [[8,6],[7,5],[3,0,9]])
```

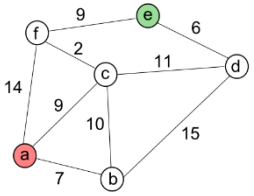


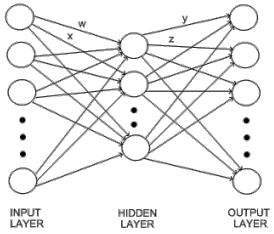
運算工具

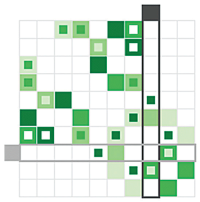
運算思維

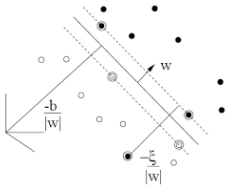






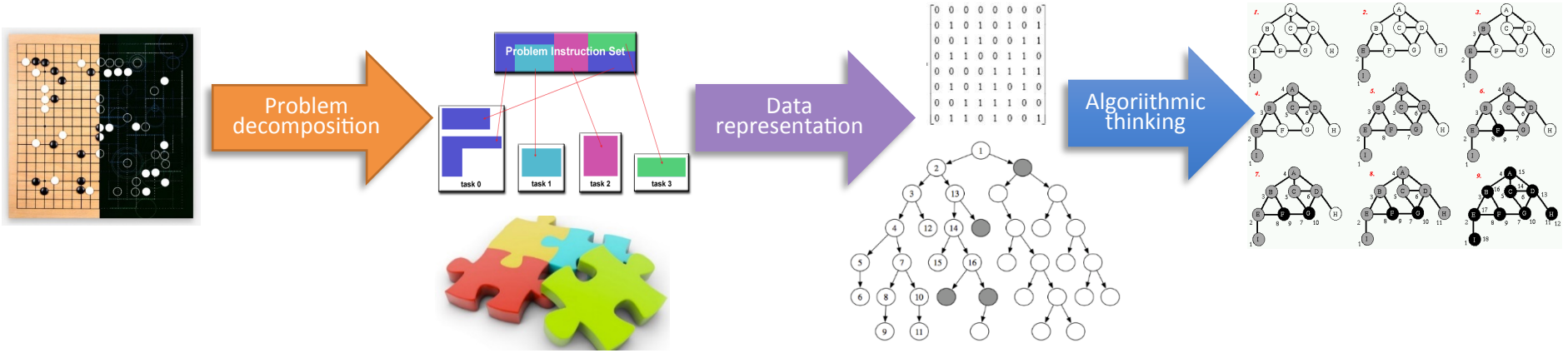






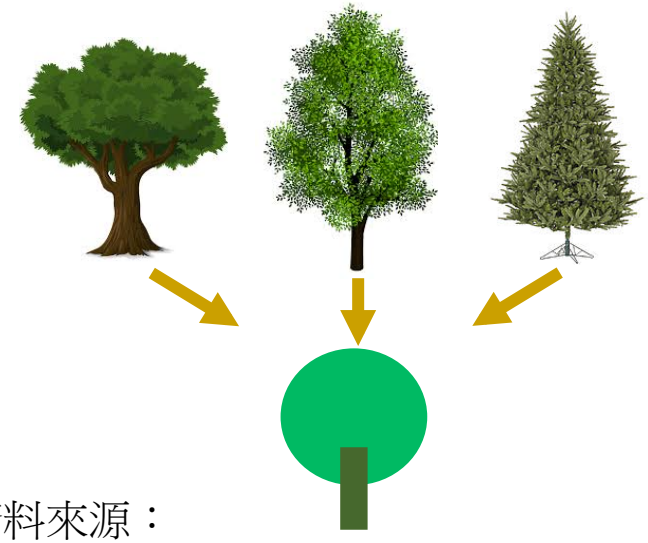
什麼是運算思維-定義

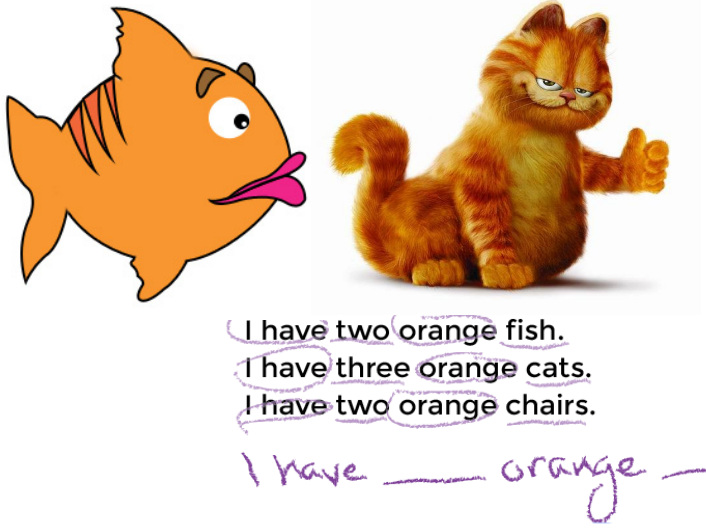
- 運算思維在運算歷程中處處可見



什麼是運算思維-內涵

- 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)





資料來源：

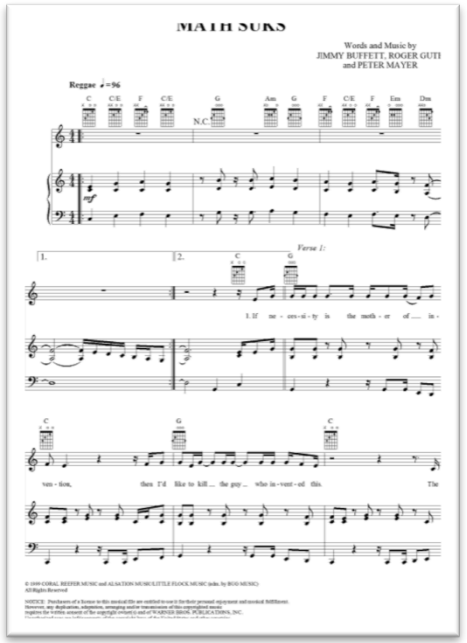
Kramer, J. (2007). Is abstraction the key to computing?. Communications of the ACM, 50(4), 36-42.

什麼是運算思維-內涵

- 藝術的抽象化

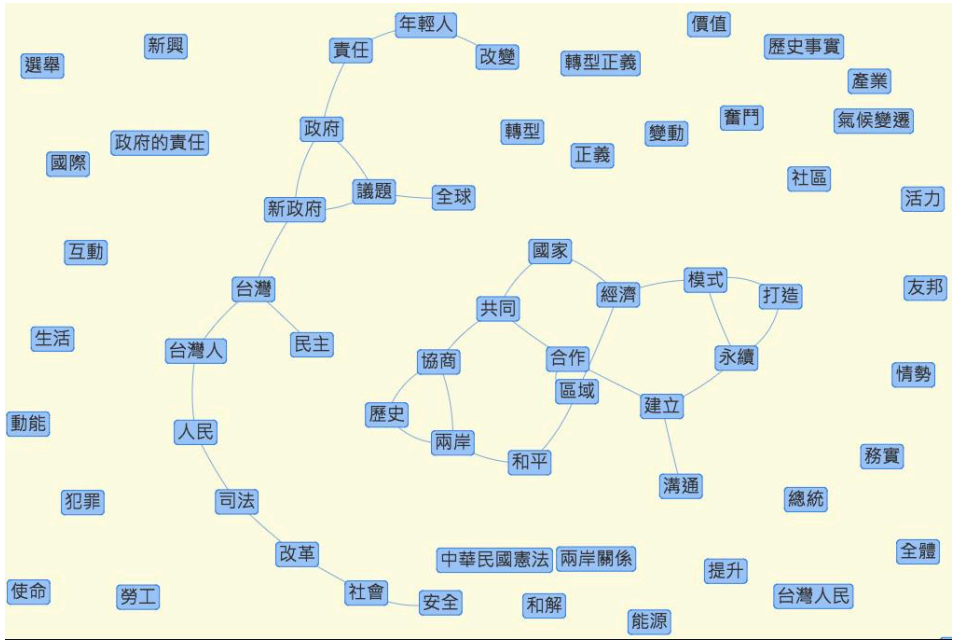
國王的哀愁

Icarus的墜落



什麼是運算思維-內涵

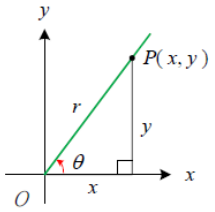
- 語文的抽象化

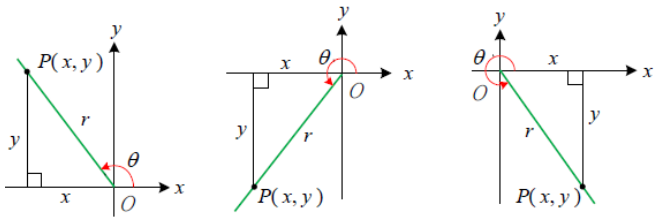


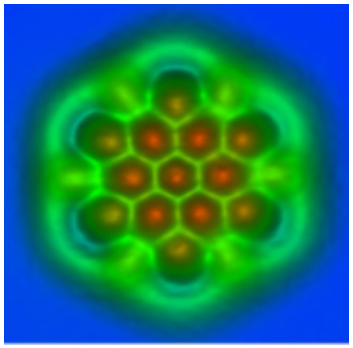
什麼是運算思維-內涵

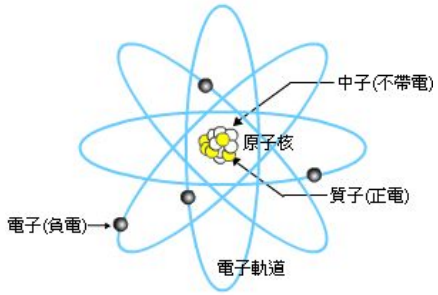
- 數學與科學的抽象化

$$\sin\theta=\frac{y}{r}\quad;\quad \csc\theta=\frac{r}{y}$$
$$\cos\theta=\frac{x}{r}\quad;\quad \sec\theta=\frac{r}{x}$$
$$\tan\theta=\frac{y}{x}\quad;\quad \cot\theta=\frac{x}{y}$$



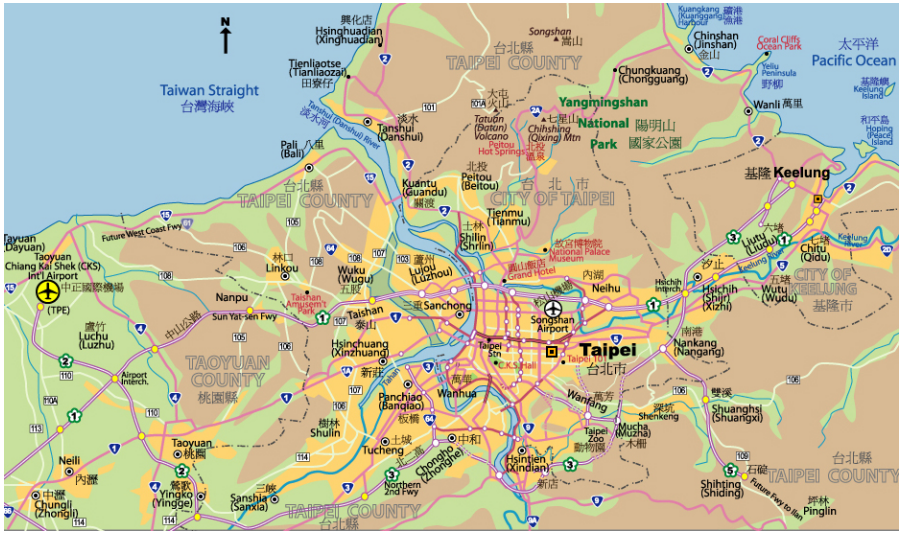






什麼是運算思維-內涵

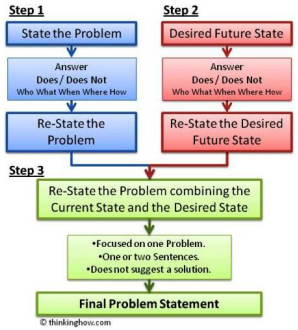
- 日常生活中的抽象化

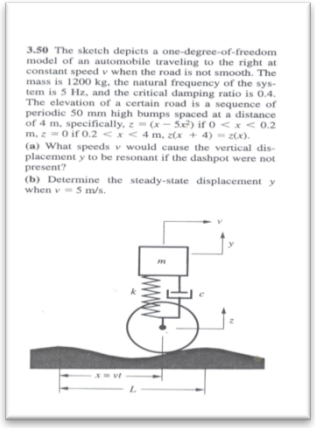


什麼是運算思維-內涵

- 運算哪裡有抽象化?

```
using namespace std;

int main() {
int number, reverse = 0;
 cout<<"Input a Number to Reverse:  ";
 cin>> number;

   for( ; number!= 0 ; )
   {
      reverse = reverse * 10;
      reverse = reverse + number%10;
      number = number/10;
   }
   cout<<"New Reversed Number is:  "<<reverse

   return 0;
}
```





3.50 The sketch depicts a one-degree-of-freedom model of an automobile traveling to the right at constant speed v when the road is not smooth. The mass is 1200 kg, the natural frequency of the system is 5 Hz, and the critical damping ratio is 0.4. The elevation of a certain road is a sequence of periodic 50 mm high bumps spaced at a distance of 4 m, specifically, $z = (x - 5x^2)$ if $0 < x < 0.2$ m, $z = 0$ if $0.2 < x < 4$ m, $z(x + 4) = z(x)$.

(a) What speeds v would cause the vertical displacement y to be resonant if the dashpot were not present?

(b) Determine the steady-state displacement y when $v = 5$ m/s.

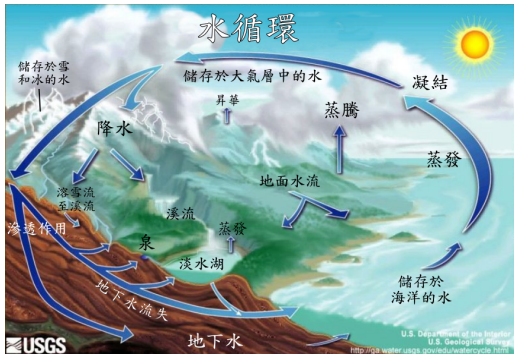
什麼是運算思維-內涵

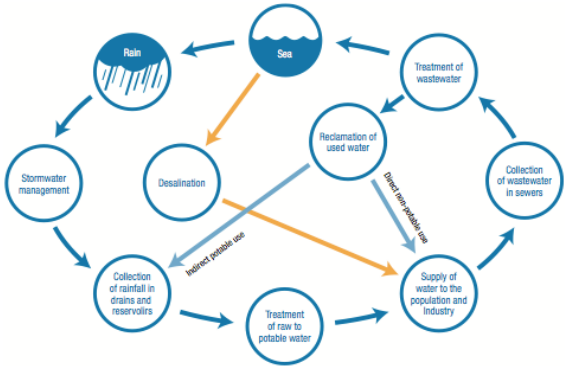
- 抽象化包含哪些歷程?
  - 辨識並擷取與解題相關的關鍵部分，抽取基本的解題單元，以重複利用此一解題單元，並擴展解題領域
    → 樣式辨識/一般化
  - 從複雜的現實世界映射到簡化的抽象模型
    → 模型化、模擬

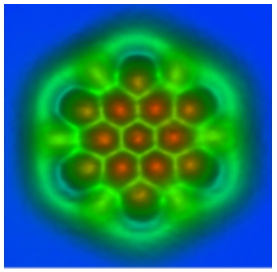
什麼是運算思維-內涵

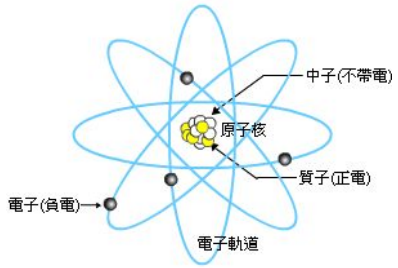
- 樣式一般化(Pattern generalization)
  - 產出共通的模式、規則、原則或理論

2^{10} → 2^{x} → a^{x}





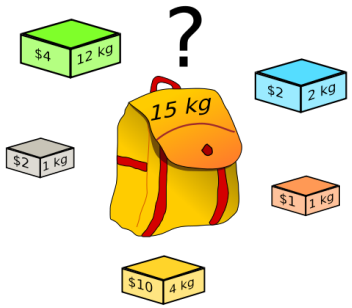




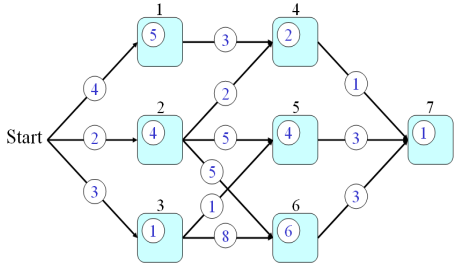
什麼是運算思維-內涵

- 樣式辨識(Pattern recognition)

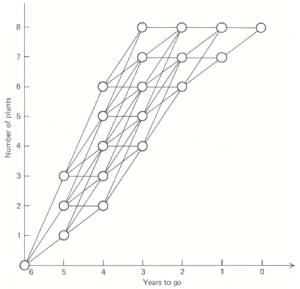
問題



辨識資料表示樣式



$$v_n(j) = \text{Min}\,\{d_{ij} + v_{n-1}(i)\}, \quad 1 \le j \le N,$$



maximize $\sum_{i \in 1..j} v_i \; x_i$

subject to $\sum_{i \in 1..j} w_i x_i \le k$

$x_i \in \{0,1\} \quad (i \in 1..j)$

辨識演算法樣式

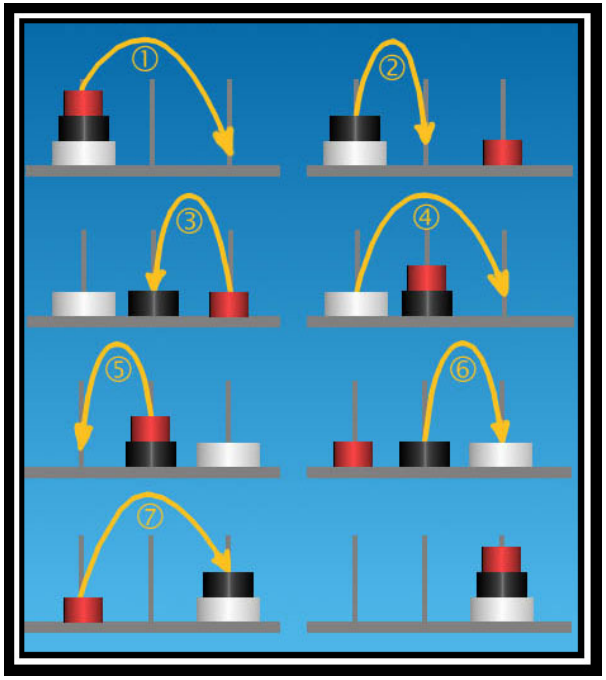
		A	B	C	X	Y	Z	A	Y
	0	0	0	0	0	0	0	0	0
X	0	0	0	0	1	0	0	0	0
Y	0	0	0	0	0	2	0	0	1
Z	0	0	0	0	0	0	3	0	0
A	0	1	0	0	0	0	0	4	0
B	0	0	2	0	0	0	0	0	0
C	0	0	0	3	0	0	0	0	0
B	0	0	1	0	0	0	0	0	0

Capacity	0	1	2	3
0	0	0	0	0
1	0	0	0	0
2	0	0	0	3
3	0	0	0	3
4	0	5	5	5
5	0	5	6	6
6	0	5	6	8
7	0	5	6	9
8	0	5	6	9
9	0	5	11	11

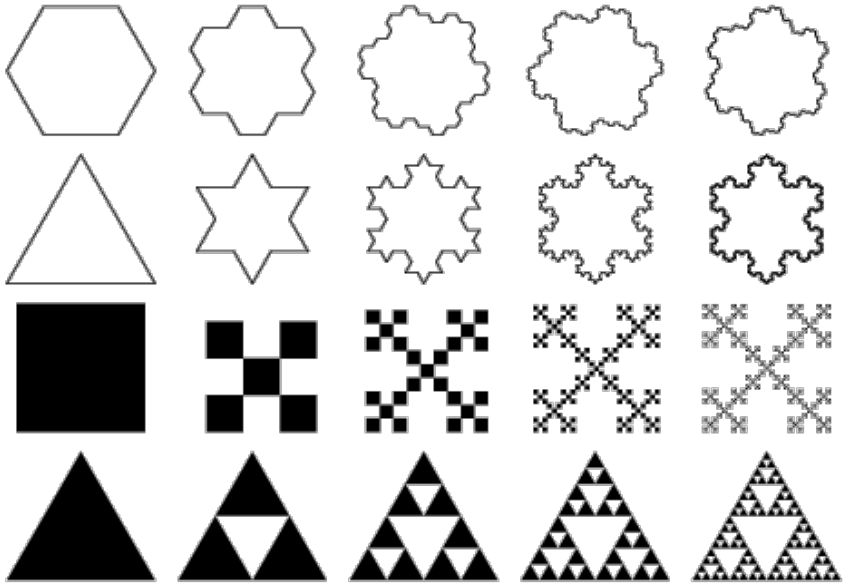
$v_1 = 5 \quad v_2 = 6 \quad v_3 = 3$
$w_1 = 4 \quad w_2 = 5 \quad w_3 = 2$

什麼是運算思維-內涵

- 樣式辨識與一般化讓運算發揮其能力



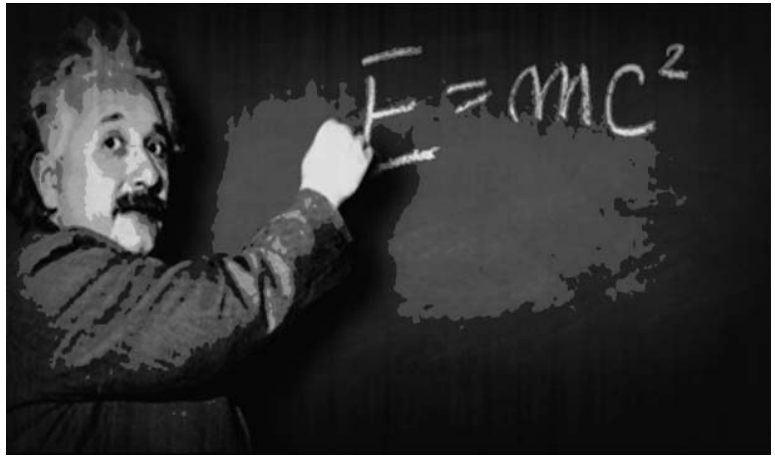
河內之塔：透過解題規則的找尋，讓運算幫我們處理人腦難以運作的盤子數量。



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。

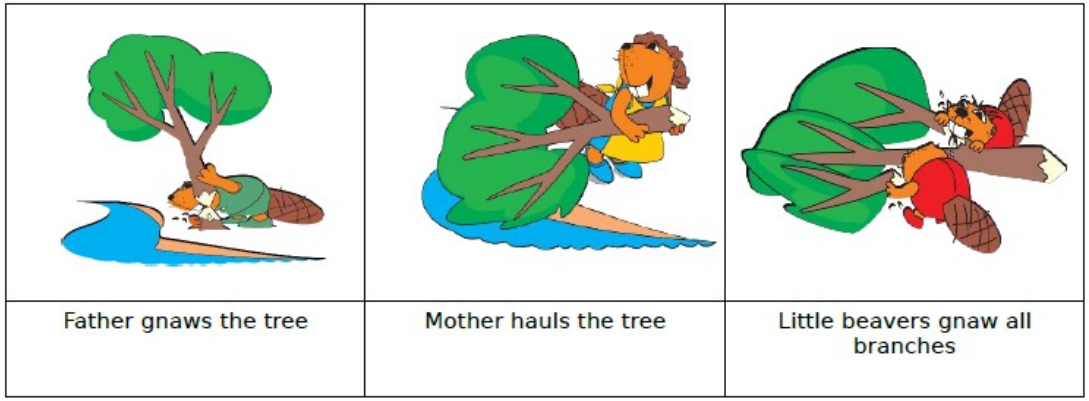
什麼是運算思維-內涵

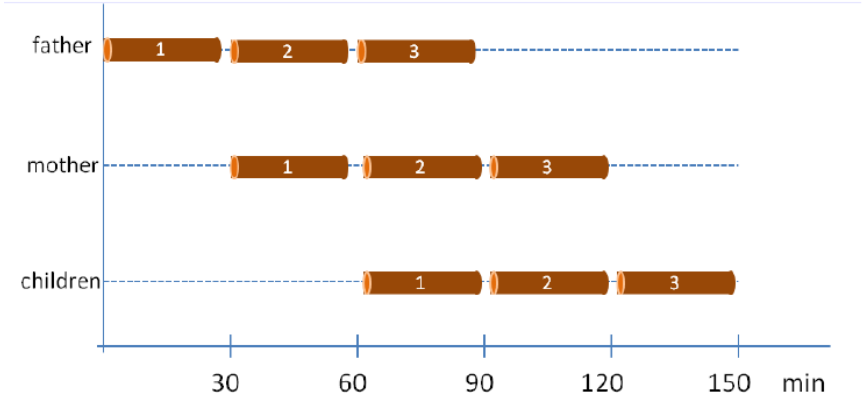
- 模型化(Modelling)
  - 根據不同需求(為了容易瞭解、定義、量化、視覺化或模擬等)，將複雜的現象以簡化的方式表達
  - 可用以將抽象的概念視覺化
  - 可作為實驗結果闡釋的依據
  - 可作為預測的基礎



什麼是運算思維-內涵

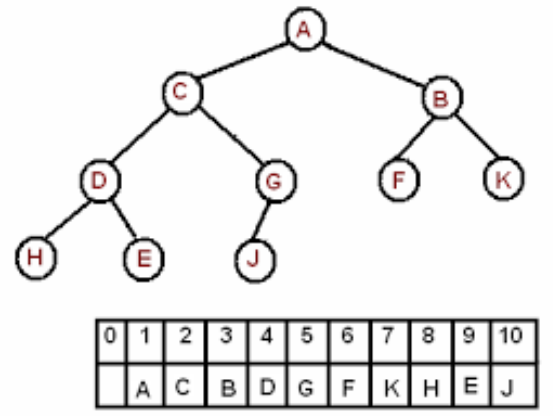
- 演算法思維(Algorithmic thinking)
  - 產出有序指令以解決問題或完成任務

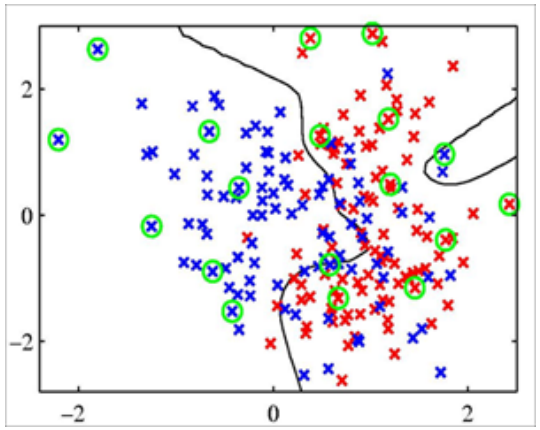


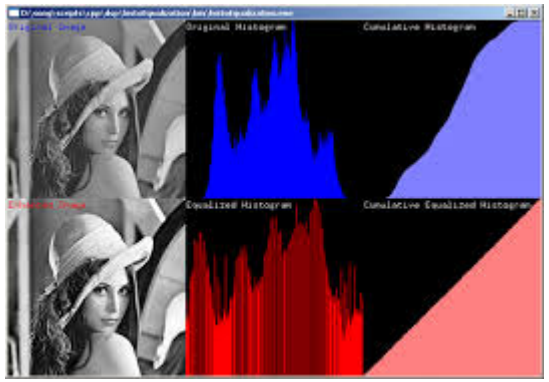


什麼是運算思維-內涵

- 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







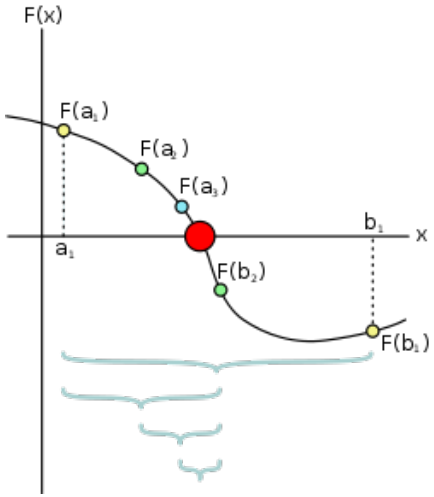
什麼是運算思維-運算思維與數學思維

?

- 運算思維 = 數學思維

求多項式的解：

$f(x)=x^3-4x-9$



用二分法求解：

C Program for Bisection Method Source Code

```c
#include<stdio.h>
#include<math.h>
float fun (float x)
{
    return (x*x*x - 4*x - 9);
}
void bisection (float *x, float a, float b, int *itr)
/* this function performs and prints the result of one iteration */
{
    *x=(a+b)/2;
    ++(*itr);
    printf("Iteration no. %3d X = %7.5f\n", *itr, *x);
}
void main ()
{
    int itr = 0, maxmitr;
    float x, a, b, allerr, x1;
    printf("\nEnter the values of a, b, allowed error and maximum iterations:\n");
    scanf("%f %f %f %d", &a, &b, &allerr, &maxmitr);
    bisection (&x, a, b, &itr);
    do
    {
        if (fun(a)*fun(x) < 0)
            b=x;
        else
            a=x;
        bisection (&x1, a, b, &itr);
        if (fabs(x1-x) < allerr)
        {
            printf("After %d iterations, root = %6.4f\n", itr, x1);
            return 0;
        }
        x=x1;
    }
    while (itr < maxmitr);
    printf("The solution does not converge or iterations are not sufficient");
    return 1;
}
```

http://www.codewithc.com/c-program-for-bisection-method/

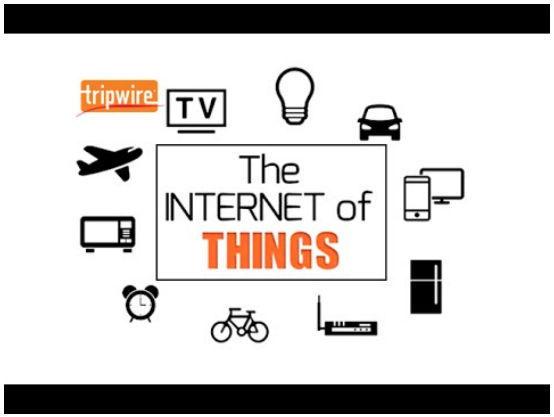
運算思維的重要性

這是一個運算的時代

Paul Schottmiller - March 19, 2013

GHETTO UBER DRIVER頻道建立者：Latrell G.

The Apple Watch, released in 2015.



Risks of The Internet of Things頻道建立者：Tripwire, Inc.

Amazon warehouse robots

運算思維

運算思維的重要性-日常生活

- 美國卡內基梅隆大學教授 Jeannette M. Wing 認為在基礎語言能力中應該加入電腦運算的因素，在讀、寫和算數之外，還需要該加上電腦運算的概念：「電腦運算思考的技巧，並不是只有電腦科學家的專利，而是每個人都應該具備的能力及素養。」
- 日常生活與運算的關係愈來愈密切
  - 社交網路
  - 智慧型居家
  - 醫療
  - 交通
  - 購物
  - …
- 具備運算思維能更善用運算解決日常生活問題

運算思維的重要性-職涯發展

- 科學與工程領域
  - 利用運算模擬建築結構，以確認安全性
  - 利用運算預測氣象，以增加準確性
- 人文與社會領域
  - 利用運算分析並優化廣告投放策略
  - 利用運算分析人口老化趨勢與醫療資源分布
- 藝術領域
  - 利用運算建構三維動畫
  - 利用運算創作數位音樂

如何培養運算思維

如何培養-運算思維與資訊科學

- 既然運算思維不等同於資訊科學，那麼欲培養運算思維是否需要學習資訊科學?
  - 是否可不透過數學(物理、化學、藝術...)的學習培養數學(物理、化學、藝術...)思維?

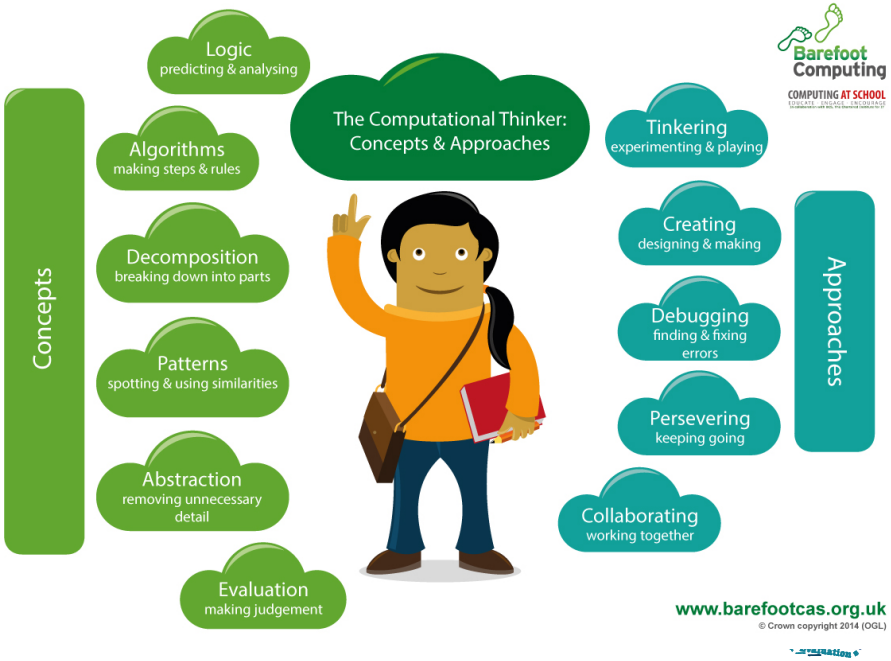
如何培養-運算思維與資訊科技課綱

- 各國課綱與運算思維

美國CSTA電腦科學課程

CSTA (2011). CSTA K–12 computer science standards. The ACM K-12 Education Task Force. Retrieved from http://www.csta.acm.org/Curriculum/sub/CurrFiles/CSTA_K-12_CSS.pdf

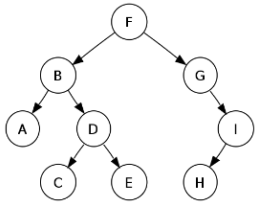
英格蘭運算課程

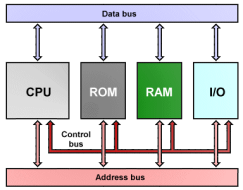


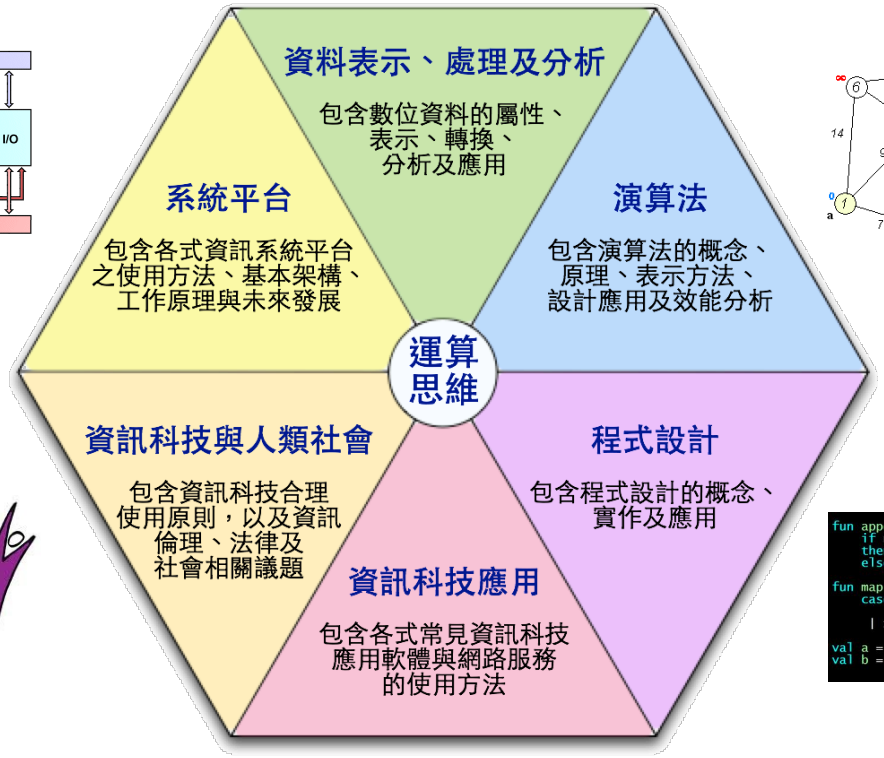
http://www.barebootcas.org.uk

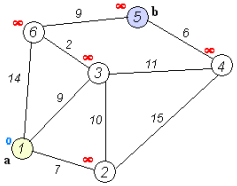
如何培養-運算思維與資訊科技課綱

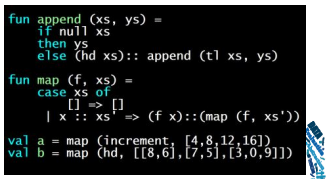
- 我國課綱與運算思維











如何培養-運算思維與其他學科領域

- 運算思維元素於各領域之應用範例

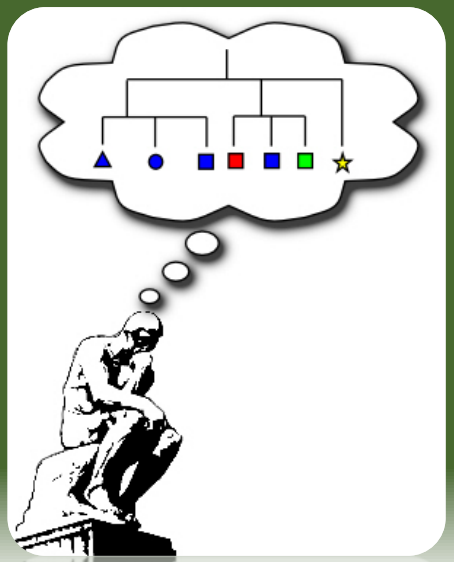
運算思維元素	各領域應用範例				
	資訊科學	數學	科學	社會研究	語言藝術
抽象化	使用程序來封裝一組經常重複使用的指令；使用函數；使用條件敘述、迴圈、遞迴等	使用代數的變數；辨識應用問題中的基本事實；研究代數函數並與程式函數比較；使用迭代(iteration)來解決應用問題	為一個物理的實體建立模式	總結事實，從事實中演繹結論	使用明喻和隱喻；寫有分支的故事
問題解析	定義物件和方法；定義main和functions	利用表示式表達運算順序	對物種進行分類		撰寫大綱
資料表示	使用資料結構，例如：陣列(array), 鏈結串列(linked list), 堆疊(stack), 佇列(queue), 圖(graph), 雜湊表(hash table)等	用長條圖、圓餅圖表示資料；使用集合、數列、圖等表示資料	從實驗資料做出結論	總結並表達趨勢	為不同句型呈現其樣式
模式化與模擬	利用動畫呈現演算法，參數掃值(parameter sweeping)	繪製笛卡爾平面上的函數並修改變數的值	模擬太陽系運動	玩世紀帝國、Oregon trail	重現一個故事
演算法思維	學習經典演算法；針對某一領域的問題實作演算法	做長除法、因數分解；作加減法的進位	進行實驗程序		撰寫操作說明

資料來源：Barr, V., & Stephenson, C. (2011). Bringing computational thinking to K-12: What is involved and what is the role of the computer science education community? *ACM Inroads*, *2*, 48–54.

資訊科技課程綱要簡介

未來人才需求

- 具備善用運算方法與工具解決問題的能力→運算思維



- 具備創新與動手實作的能力→程式設計
  - 問題解決
  - 溝通表達
  - 合作共創

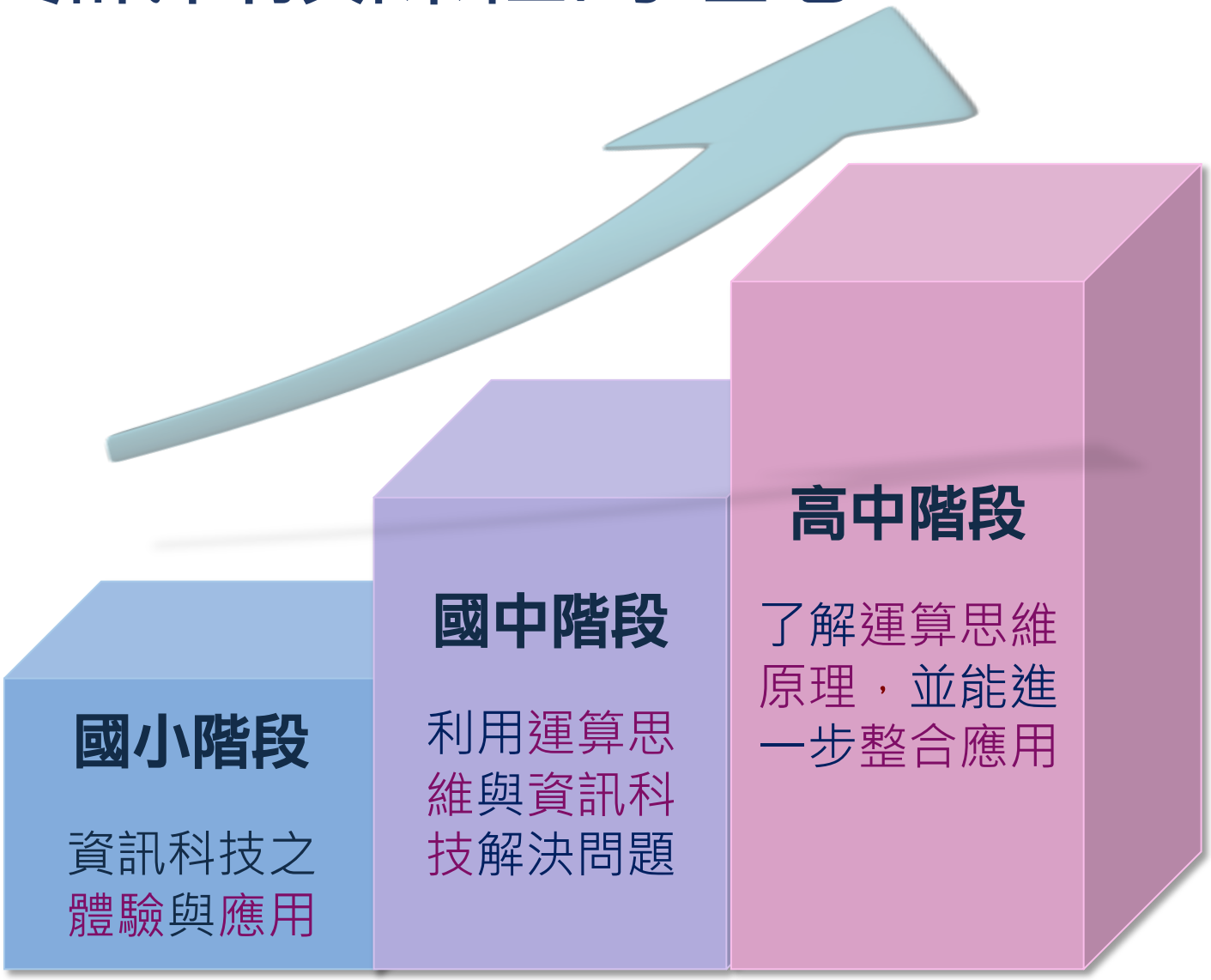
```c
#include<stdio.h>
#include<math.h>
float fun (float x)
{
    return (x*x*x - 4*x - 9);
}
void bisection (float *x, float a, float b, int *itr)
/* this function performs and prints the result of one iteration */
{
    *x=(a+b)/2;
    ++(*itr);
    printf("Iteration no. %3d X = %7.5f\n", *itr, *x);
}
void main ()
{
    int itr = 0, maxmitr;
    float x, a, b, allerr, x1;
    printf("\nEnter the values of a, b, allowed error and maximum iterations:\n");
    scanf("%f %f %f %d", &a, &b, &allerr, &maxmitr);
    bisection (&x, a, b, &itr);
    do
    {
        if (fun(a)*fun(x) < 0)
            b=x;
        else
            a=x;
        bisection (&x1, a, b, &itr);
        if (fabs(x1-x) < allerr)
        {
            printf("After %d iterations, root = %6.4f\n", itr, x1);
            return 0;
        }
        x=x1;
    }
    while (itr < maxmitr);
    printf("The solution does not converge or iterations are not sufficient");
    return 1;
}
```

美國二十一世紀關鍵能力聯盟訂定二十一世紀的關鍵能力包含：批判性思考與解決問題、溝通、合作共創、以及創造力（Partnership for 21st Century Skills-P21, 2007）

資訊科技課程的理念

- 資訊科技課程是以運算思維為主軸，透過電腦科學相關知能的學習，培養邏輯思考、系統化思考等運算思維，並藉由資訊科技之設計與實作，增進運算思維的應用能力、解決問題能力、團隊合作以及創新思考。

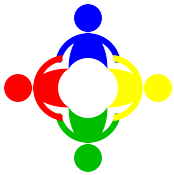
資訊科技課程的理念


國小階段
資訊科技之體驗與應用
國中階段
利用運算思維與資訊科技解決問題
高中階段
了解運算思維原理，並能進一步整合應用

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。

各學習階段之學習內容

	三至六年級	七年級	八年級	九年級	普通高中
演算法(A)	■ 程序性的問題解決方法 ■ 簡單的問題解決表示方法 ■ 抽象化 ■ 指令化	■ 演算法基本概念	■ 陣列資料結構的概念與應用 ■ 基本演算法的介紹		■ 重要資料結構的概念與應用 ■ 重要演算法的概念與應用 ■ 演算法效能分析
程式設計(P)	■ 程式設計之功能、操作及應用	■ 程式語言基本概念功能及應用 ■ 結構化程式設計	■ 陣列程式設計實作 ■ 模組化程式設計的概念 ■ 模組化程式設計與問題解決實作		■ 陣列資料結構的程式設計實作 ■ 重要演算法的程式設計實作
系統平台(S)	■ 常見系統平台之基本功能 ■ 常見系統平台之使用與維護 ■ 常見網路設備與行動裝置之功能			■ 系統平台重要發展與演進 ■ 系統平台之組成架構與基本運作原理 ■ 網路技術的概念與介紹 ■ 網路服務的概念與介紹	■ 系統平台之運作原理 ■ 系統平台之未來發展趨勢
資料表示、處理及分析(D)	■ 常見的數位資料類型與儲存架構 ■ 數位資料的表示方法 ■ 系統化數位資料管理方法			■ 資料數位化之原理與方法 ■ 數位資料的表示方法 ■ 資料處理概念與方法	■ 巨量資料的概念 ■ 資料探勘與機器學習的基本概念

	三至六年級	七年級	八年級	九年級	普通高中
資訊科技應用(T)	■ 繪圖軟體的使用 ■ 文書處理軟體的使用 ■ 瀏覽器的使用 ■ 資料搜尋的基本方法 ■ 數位學習網站與資源的使用 ■ 簡報軟體的使用 ■ 影音編輯軟體的操作與應用 ■ 網路通訊軟體的使用 ■ 雲端服務或工具的使用	■ 資料處理應用專題		■ 資訊科技應用專題	■ 數位合作共創的概念與工具使用
資訊科技與人類社會(S)	■ 康健的數位使用習慣 ■ 資訊科技之使用原則 ■ 資訊安全基本概念及相關議題	■ 個人資料保護 ■ 資訊科技合理使用原則 ■ 資訊安全	■ 資訊科技重要社會議題 ■ 資訊倫理與法律	■ 資訊科技對人類生活之影響 ■ 資訊科技相關職業類科之升學進路 ■ 資訊科技相關職業之生涯發展	■ 資訊科技的合理使用原則 ■ 個人資料的保護 ■ 資訊科技的重要社會議題 ■ 資訊科技對人類社會之影響 ■ 資訊科技領域性向之自我理解 ■ 資訊科技相關行業之進路與生涯發展

各學習階段之教學-示例

國小階段

- 演算法：透過遊戲與軟體的操作體驗問題解決的流程設計
- 程式設計：透過視覺化程式設計體驗運算的能力

國中階段

- 演算法：搭配程式設計的實作，了解如何透過運算解決問題
- 程式設計：實作問題解決的策略，以了解透過程式設計解決問題的基本方法

高中階段

- 演算法：了解運算的原理，藉以設計問題解決的策略，並搭配程式設計的實作以解決問題
- 程式設計：整合運算思維、程式設計與其他資訊科技工具，創作資訊作品