新北市資訊師資輔導團
兒童程式教學實務分享

橘子蘋果程式設計學苑
Kevin

2016/3/1

束凱文 Kevin Shu

橘子蘋果程式設計學苑

不喜歡複雜, 喜歡用簡單的思維解決問題。

喜歡分享, 剛出社會的兩年期間累積了80多篇的技術網誌, 獲得7萬多閱覽率。

2011/5 與 Google 台灣總經理同台, 對談網路趨勢

2012/8 - 2013/2　系統工程師

2013/3 - 2013/5　前端工程師

2013/5 開始創業

2014/4 開發12年國教檢測平台, 接受UDN訪問

2014/6　　接任橘子蘋果程式設計學苑 營運長

橘子蘋果程式設計學苑

台灣目前最具指標性及影響力之兒童程式設計教育機構。

2011年

開始作教材

2012年

開始進□免費教學

2013年

進入宜蘭**人文國小**

2014年

進入台北**光復國小**

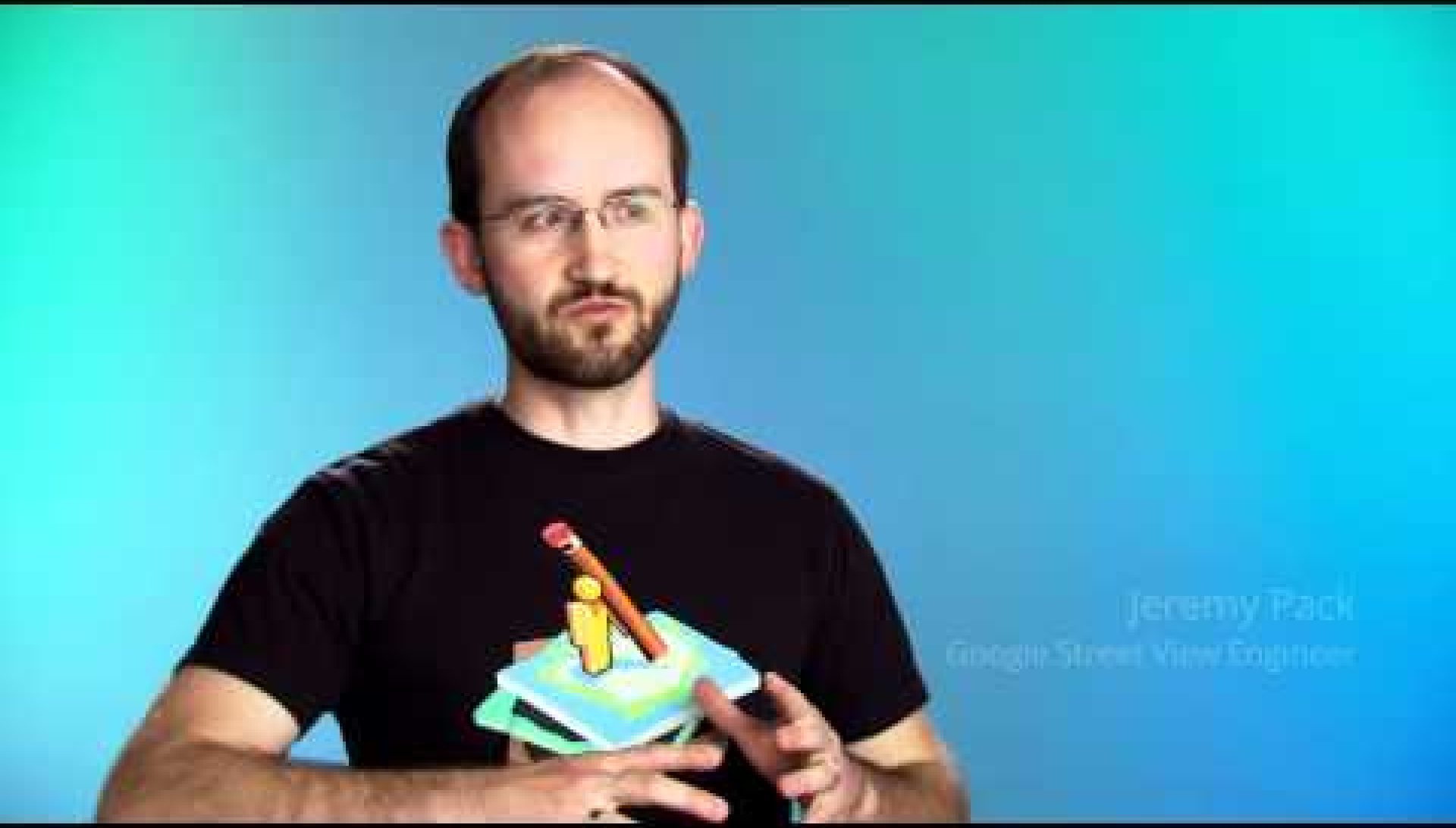
協助**建國中學**成立資訊創業性質社團 - TWCL

2015年

什麼是計算性思維？

計算性思維
Computational Thinking

用電腦的邏輯來解決問題的思維


Jeremy Pack
Google Street View Engineer

計算性思維
Computational Thinking

1. **拆解** (Decomposition):
   將一個任務或問題拆解成數個步驟或部分。

2. **找出規律** (Pattern Recognition):
   預測問題的規律，並找出模式做測試。

3. **歸納與抽象化** (Pattern Generalization and Abstraction):
   找出最主要導致此模式的原則或因素。

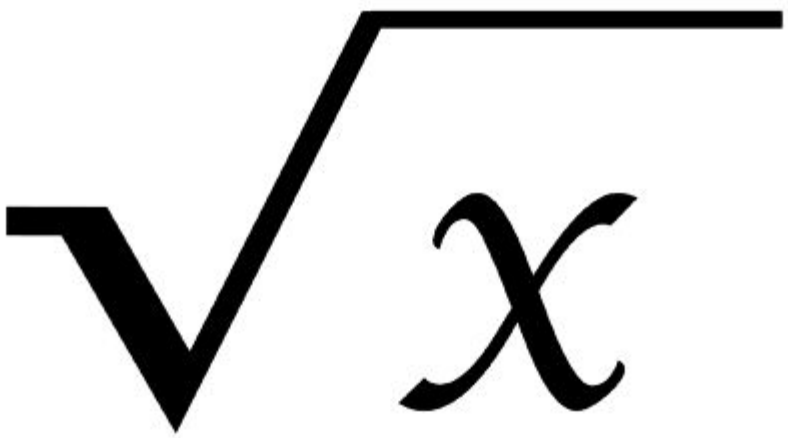
4. **設計演算法** (Algorithm Design):
   設計出能夠解決類似問題並且能夠被重複執行的指令流程。

CT是一種哲學，
是一種不用執著於「一步到位」，
能夠漸進、遞迴地達成需求的思維！

在倚天屠龍記中...

在張三豐演練完劍法給張無忌看後...
只聽張三豐問道:「孩兒, 你看清楚了沒有？」
張無忌道:「看清楚了。」
張三豐道:「都記得了沒有？」
張無忌道:「已忘記了一小半。」
張三豐道:「好, 那也難為了你。你自己去想想罷。」
張無忌低頭默想。過了一會, 張三豐問道:「現下怎樣了？」
張無忌道:「已忘記了一大半。」
最後, 張無忌將太極劍法忘光光了, 但卻把敵人打得落花流水, 正所謂「**劍意勝劍招**」!

—



舉例來說，我們要如何教小學生「開根號」？

用 CT 進行思考

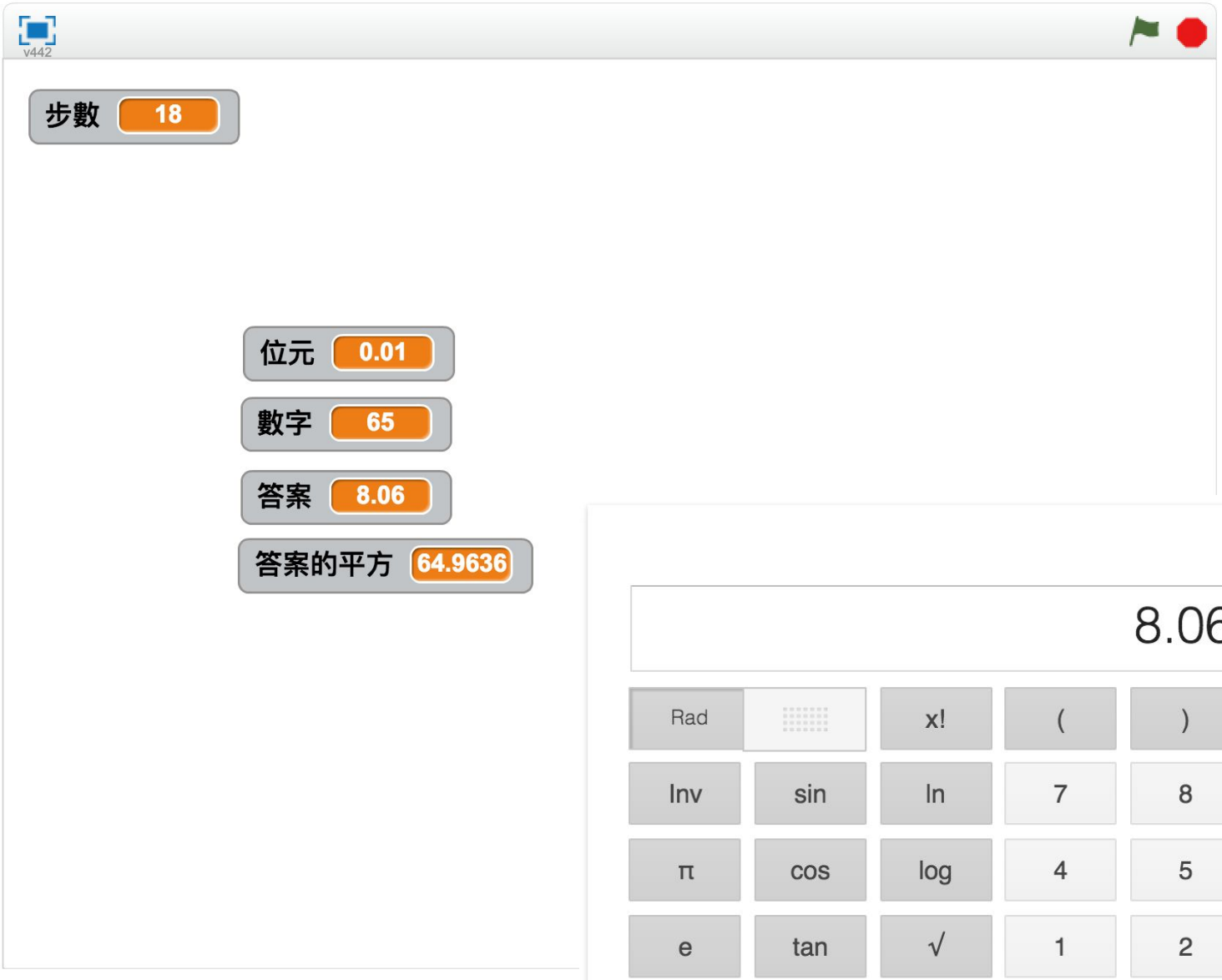
1. **拆解**:
   瞭解到「任意正整數都可以看作是『另一較小數與自己的乘積』, 而此數即為該數之平方根」

2. **找出規律**:
   觀察出「任意正整數的平方根, 必定會在0與該數之間」

3. **抽象化&歸納**:
   「用X來代表所求數之平方根」(抽象化)、
   「只要盡量求到精準即可, X與X的乘積與所求數相差0.1以內即可」(歸納)

4. **設計演算法**:
   設計一個「從0開始慢慢找出X」的流程

用 MIT Scratch 來表達演算法


v442
步數 18
位元 0.01
數字 65
答案 8.06
答案的平方 64.9636
square root(65) =
8.0622577483
Rad
x!
(
)
%
AC
Inv
sin
ln
7
8
9
÷
π
cos
log
4
5
6
×
e
tan
√
1
2
3
−
Ans
EXP
xy
0
.
=
+

詳細內容

講完大觀念了，
現在回歸實務面～

教學實務框架與經驗分享

學生會遇到的問題

1. 學生沒有頭緒, 不知從何開始
2. 學生亂拉方塊
3. 學生亂命名變數
4. 太多程式碼串, 難以 Debug!
5. 忘記初始化
6. 持續判斷條件時忘記加迴圈

限制是好事！

有了框架, 效率反而更好？

1. 不用跳脫到更高的思維思考,
   只要專注於真正重要的邏輯
2. 跟隨最佳實踐, 吸取前人的智慧
3. 跟隨慣例, 減少溝通上的歧異

MIT Scratch「只有」100多個方塊...
但是能做的事情，很多！

一原則

一個角色只有一串程式碼

原因

1. 架構簡單, 學生不容易出錯
2. 能夠用流程圖表示邏輯, 方便講解
3. 邏輯清晰, 老師更容易 Debug!
4. 難度分層容易, 如果一個專案非得要用到平行化, 就代表這個專案可能太難囉

三部分

開始、初始化、迴圈

原因

1. 架構很清楚, 比較不會一開始毫無頭緒
2. 結構清楚, 容易追蹤錯誤
3. 口訣好記、減少學生出錯的機會
   (忘記初始化、忘記用迴圈包住條件式)

三步驟

拆、找、試

原因

1. 學生不易理解CT的觀念, 給個明確的口訣比較容易讓學生有所依循
2. **拆**:讓學生先別急著動手做, 限思考如何將大問題拆成小問題
3. **找**:通常拆成小問題後就有辦法找到合適的指令來解決, 如果找不到, 再拆！
4. **試**:試試看找到的指令是否成功運作。如果是, 將這些小問題的解決方案組合起來;如果不是, 繼續找！

還有很多限制可以試試...

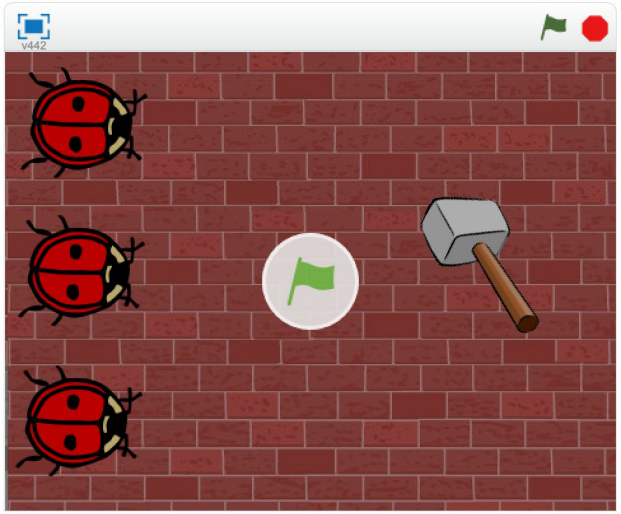
1. 限制使用的積木
2. 限制使用的角色數量
3. 限制使用的素材
4. 限制使用的變數

實作時間

專案實作 - 蟲蟲危機 (By 高慧君老師)

使用滑鼠移動榔頭，瞄準蟲蟲後，按下空白鍵敲打蟲蟲，看看你在 30秒內能打到幾隻。

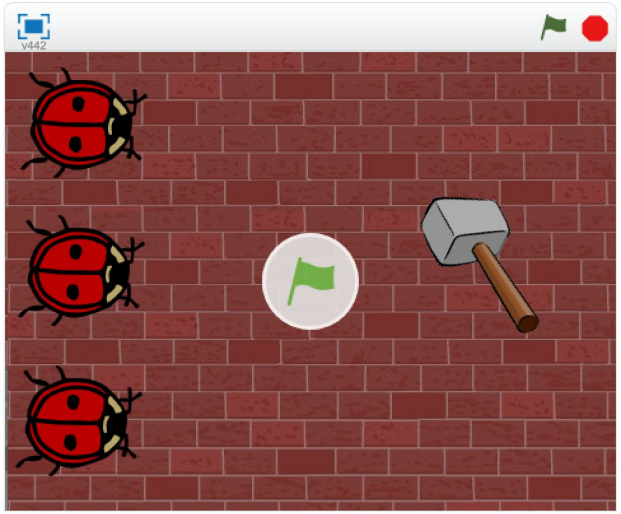
範本連結：http://bit.ly/oa-bugs



專案實作 - 蟲蟲危機 (By 高慧君老師)

使用滑鼠移動榔頭，瞄準蟲蟲後，按下空白鍵敲打蟲蟲，看看你在 30秒內能打到幾隻。

範本連結：http://bit.ly/oa-bugs



Thanks~