第2章 基礎程式設計(1)

- 認識演算法與程式語言
- Scratch 程式設計 - 基礎篇
- Scratch 程式設計 - 計算篇
- Scratch 程式設計 - 繪圖篇

翰林出版

目　錄

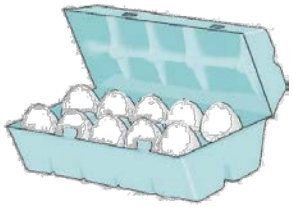
基礎程式設計

認識演算法與程式語言

- 演算法是指運算的具體步驟，就是為了解決某一問題所設計的一套有限運算規則的集合。
- 程式設計就是依邏輯順序安排一組指令以實踐運算規則。
- 演算法是一種解決問題的方法，程式語言是實踐演算法的工具。

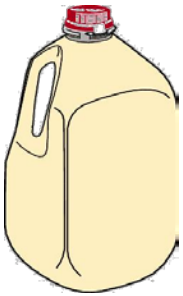
煎荷包蛋看演算法

輸入 ＋ 演算法 ＝ 輸出



雞蛋

＋

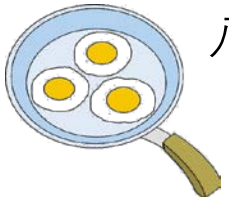


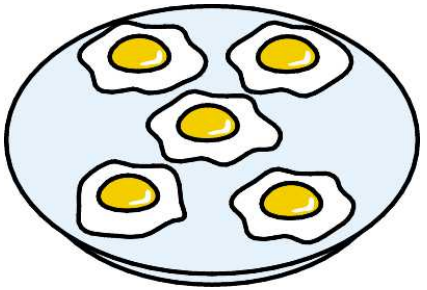
油

將油倒進
鍋中加熱

打蛋

小火煎
八分鐘





荷包蛋

認識演算法與程式語言

- 食譜類似是一種演算法，把製作的步驟描述得很詳盡，但是每個人不同的時間實作，結果會產生不同的品質。

- 食譜的演算法不夠精準，但是電腦的演算法必須表示非常精確，不容許有模糊的空間。

演算法基本概念

- 演算法是用來解決問題的方法，我們可以把解決問題的方法與過程寫成精確的步驟，再依照這些步驟去執行。

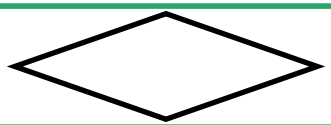
- 運用流程圖符號，可以清晰的表示演算法，也可以方便判讀與相互交流解決問題的步驟。

製作蛋炒飯看演算法

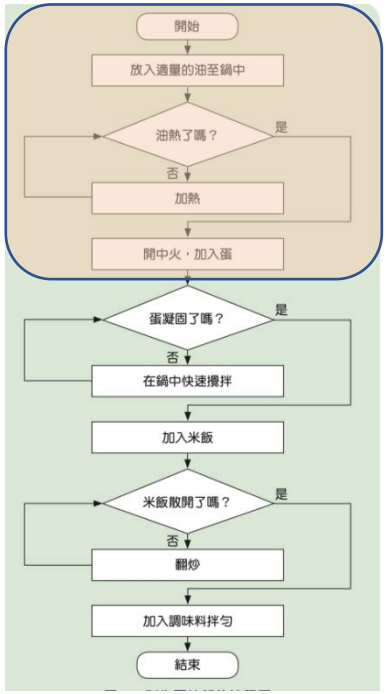
蛋炒飯製作的演算法可能類似：

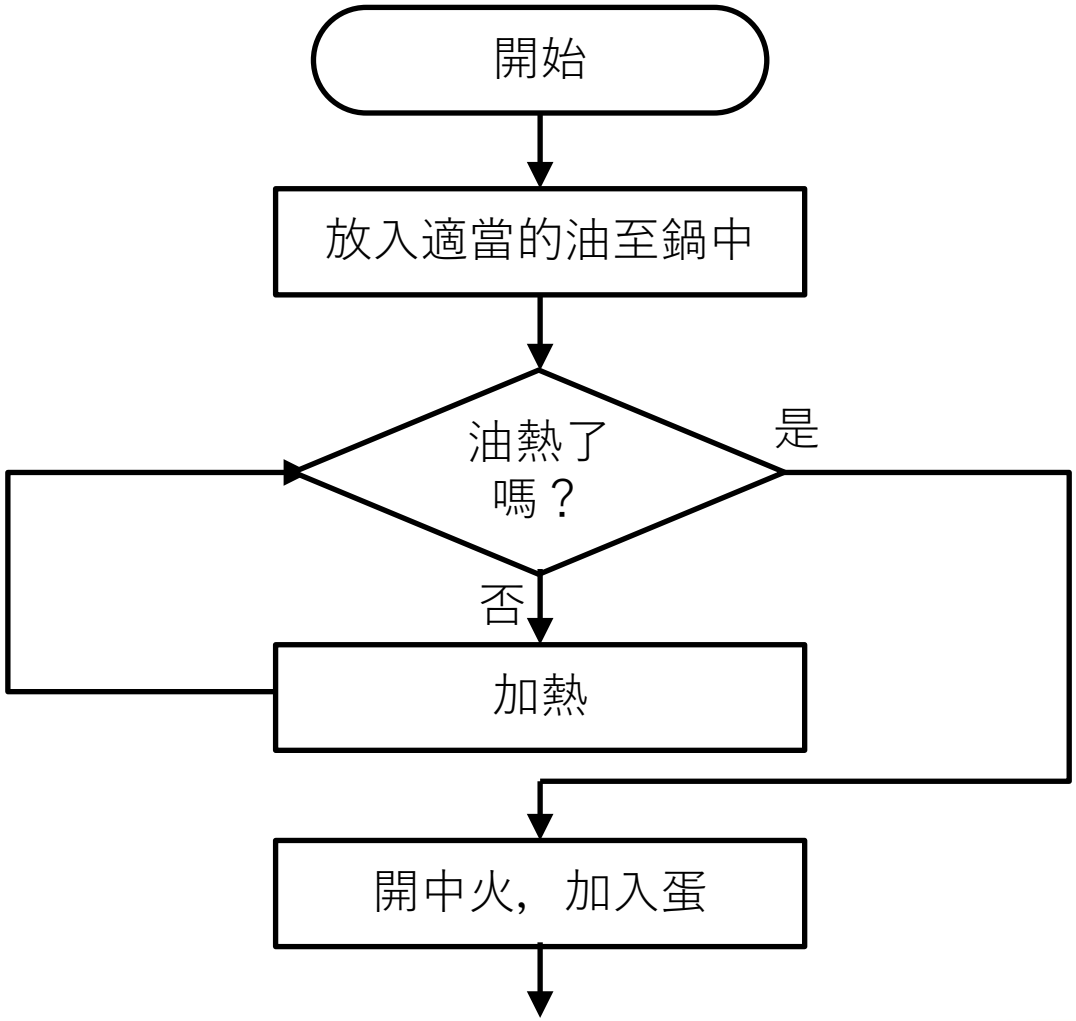
1. 放入適量的油至鍋中
2. 熱油
3. 把蛋打下去炒
4. 蛋炒至金黃色後加入米飯
5. 加一些調味料翻炒至均勻

常用流程圖符號

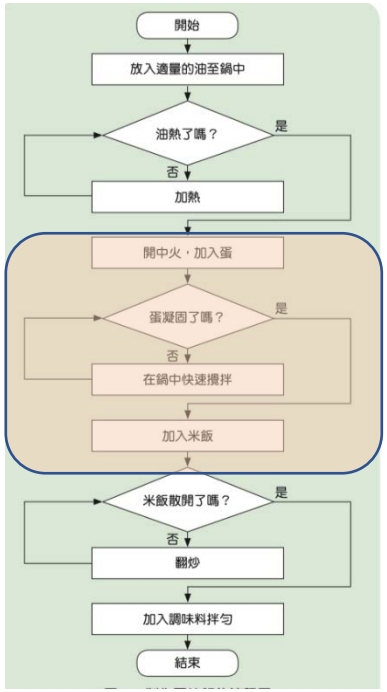
符　　號	意　　義	說　　明
	開始／結束	流程圖開始或結束
	處理	處理一項工作
	流程方向	流程進行的方向
	輸入／輸出	進行資料輸入或輸出
	決策	依條件比較結果進行不同處理
	迴圈	迴圈變數初值與終值的描述
	連接	流程的連結點

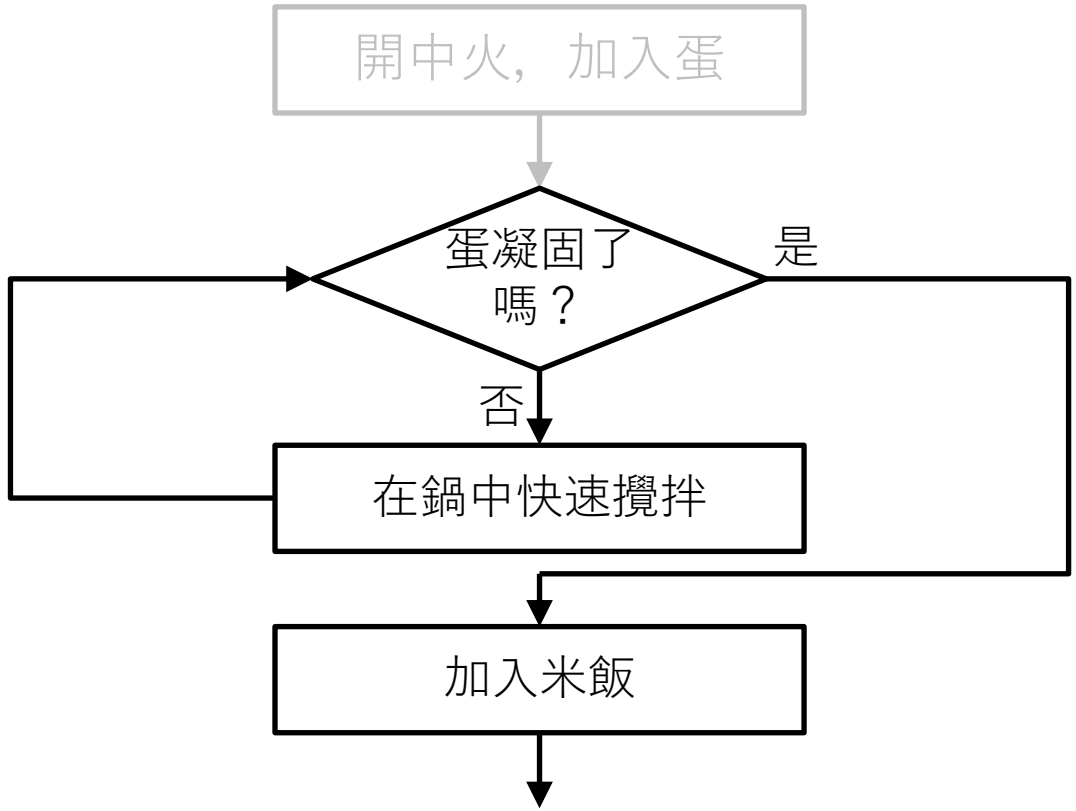
蛋炒飯製作流程



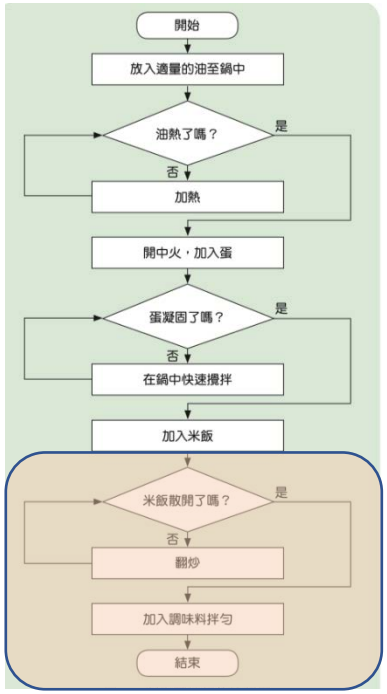


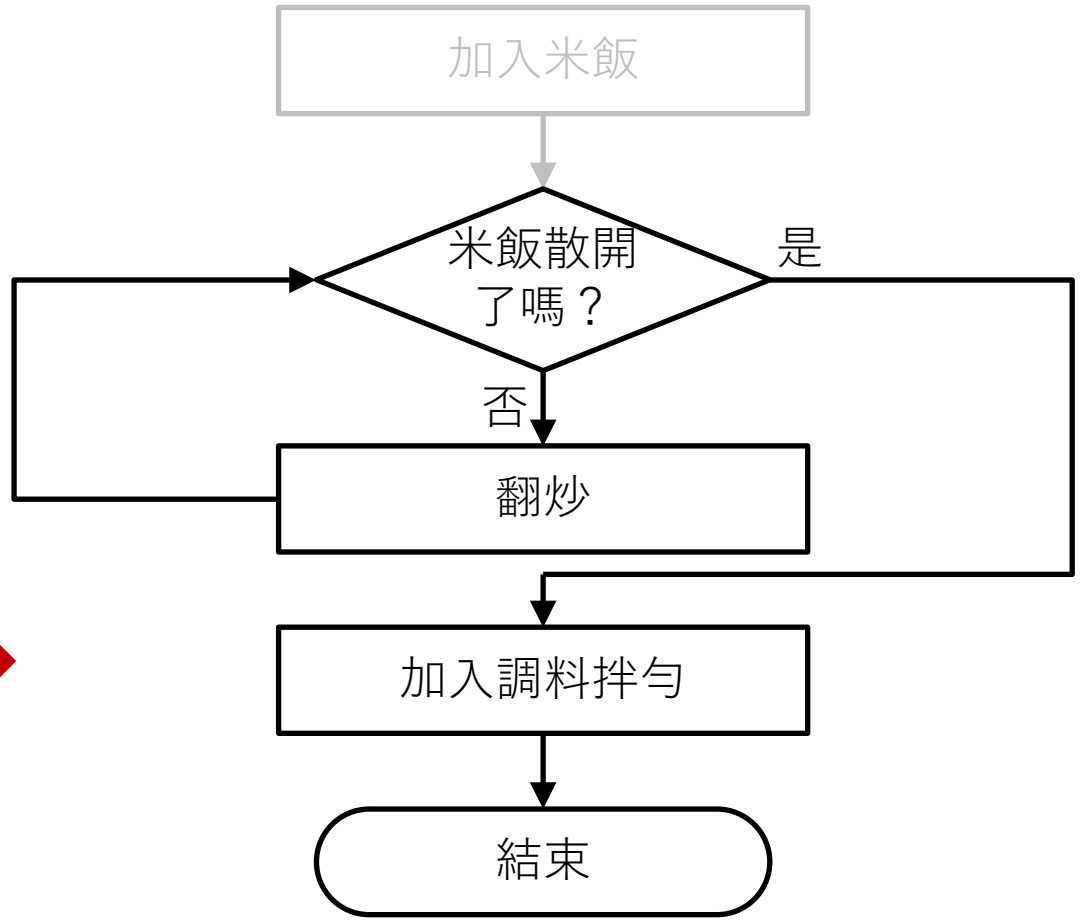
蛋炒飯製作流程





蛋炒飯製作流程





求任意數的所有因數

問題分析

本案例如果用窮舉法來解決，就是從 1 開始，針對每一個整數依序進行測試，以找出符合條件的數字。

執行迴圈	處理	判斷	輸出
第一次	x＝1	20 除以 1整除	1
第二次	x＝2	20 除以2整除	2
第三次	x＝3	20 除以3沒有整除	沒有執行任何敘述
第四次	x＝4	20 除以4整除	4
⋮			
第二十次	x＝20	20 除以20整除	20

求任意數的所有因數

小知識

窮舉法

是一種反證法論證，　密碼學上又稱為「暴力破解法」，依照逐個可能性——推算直到找出解答。

求任意數的所有因數

畫流程圖

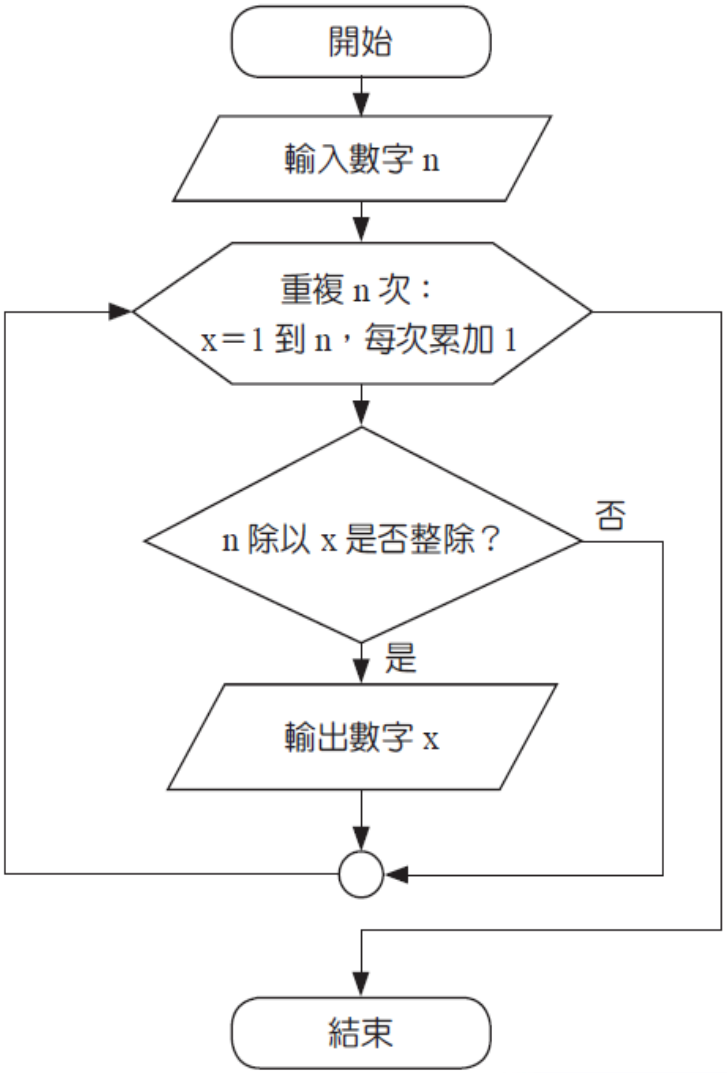
用窮舉法，從 1 開始，針對每一個整數依序進行測試，以找出符合條件的數字。

輸入

處理

判斷

輸出



程式語言基本概念

- 電腦是一部機器，只要給予指令，它就會照指令執行工作，然後將結果輸出，而這些指令的組合就成為程式。
- 複雜的設計可編譯成機器能了解的程式碼。
- 依照程式碼指示，機器可不斷重複工作直到完成。

程式語言的種類

- 低階語言
  - 機器語言
  - 組合語言
- 高階語言
  - 程序導向
  - 物件導向

低階語言

機器語言：

是早期所使用的低階語言，也是電腦唯一能識別的語言，由二進位（0與1）組成指令，可以直接在電腦中執行，因此執行效能最好。

但因為 不易學習與使用，

編寫錯誤也不易發現，

後期發展出組合語言。

低階語言

組合語言：
使用一些簡單且有意義的文字來撰寫，例如ADD(加)，組合語言必須經過組譯的處理，才可以在機器上執行；執行速度雖快，但必須對硬體結構瞭解的人才能撰寫。

組譯

機器語言：
是由0與1兩種符號組成的代碼，它能夠直接被電腦理解並執行，是電腦硬體能夠直接識別的指令集合。但不同電腦硬體架構使用不同機器語言。

高階語言：
接近人類語言的程式語言，必須先轉譯成機器語言，才能被電腦硬體執行。

轉譯

低階語言

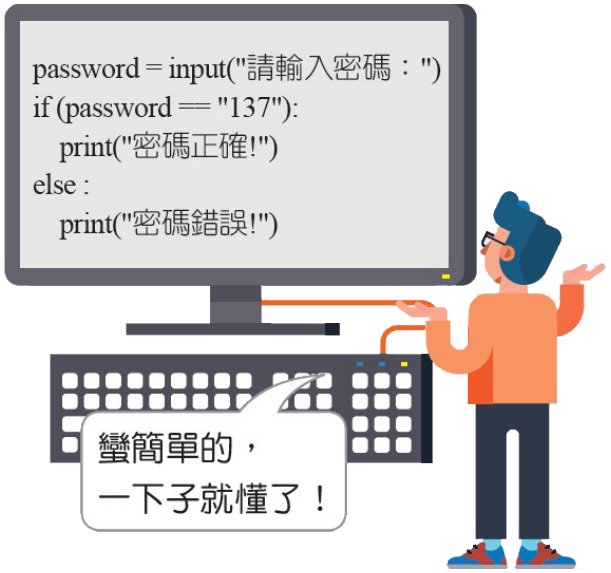
- 低階語言是以硬體為導向的描述指令語言(如:機器語言與組合語言）。
- 雖比較難看得懂，但它可以直接與硬體(CPU)溝通，有較好的執行效能。

組合語言

機器語言

高階語言

高階語言是一種語法接近人類語言的程式語言。易於閱讀與學習，但以高階語言設計的程式，須先轉譯成機器語言，才能被電腦硬體執行。



python

scratch

高階語言

■程序導向：

根據程式目標編寫處理過程的程式，再依照指令的邏輯順序執行。

■物件導向：

以物件觀念，編寫物件與物件組合或互動的程式，以達成程式目標，具有封裝、繼承、多型三大特性。

程式語言的功能

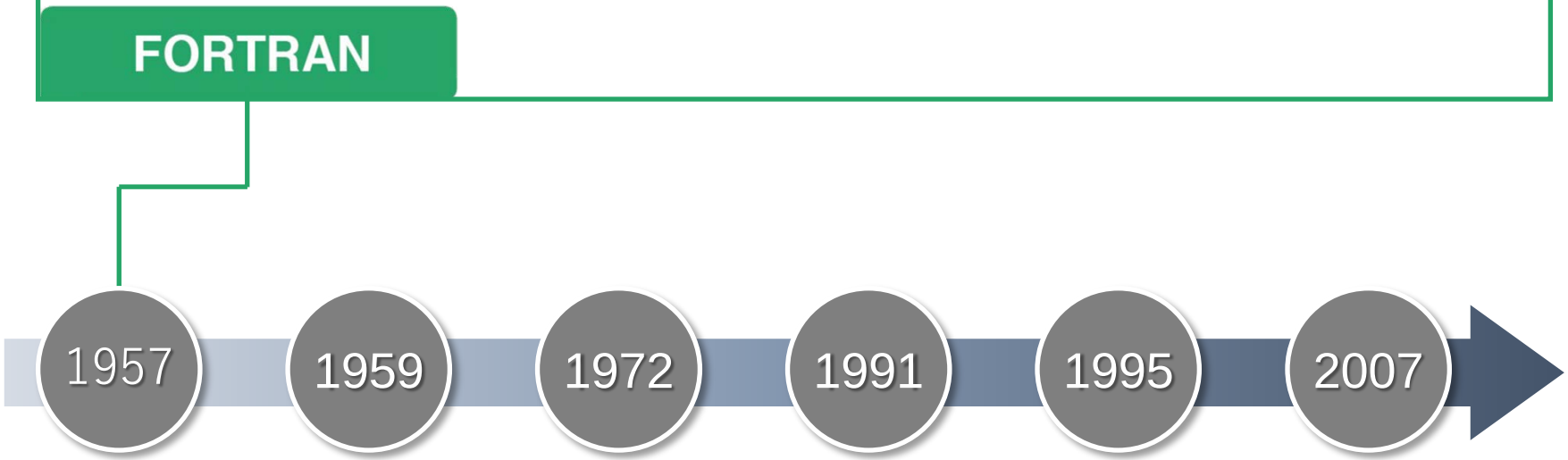
- 最重要的功能就是啟動（Booting）電腦並分配資源，指揮電腦運作。
- 讓使用者可以透過程式來操作電腦硬體，使用者與電腦就可產生互動。
- 網路雲端是以各種硬體所建構的環境，需要靠多種程式發揮功能，將異地的電腦串連，讓眾多使用者可以同時在線上互動與溝通。

程式語言的應用

程式語言種類非常多，但因用途不同，功能也不同。可以分為一般用途及特殊用途。雖不同程式語言的語法不一樣，但是基本邏輯是類似的。

常見的程式語言

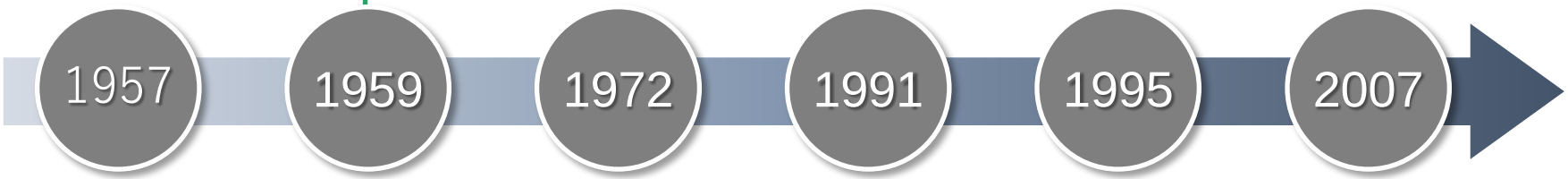
1957年IBM公司推出世界上第一個高階程式語言，廣泛的被應用在科學與工程等高效能計算領域。



常見的程式語言

針對商業數據處理所制訂的程式語言，主要應用於金融、銀行、和會計等商業數據處理。

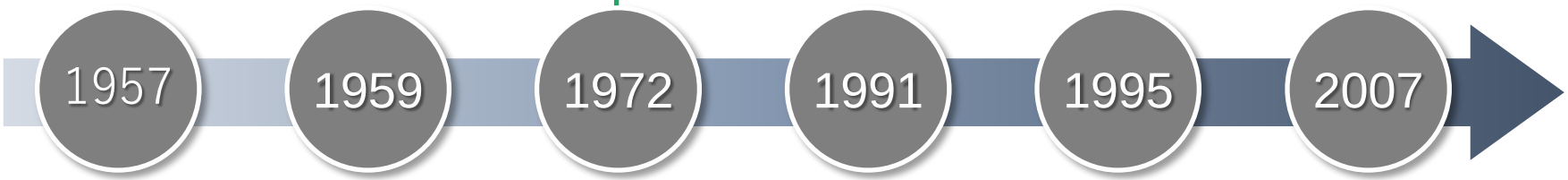
COBOL



常見的程式語言

常用來撰寫電腦系統，目前使用很廣，強而有力的一般用途程式語言。

C / C++



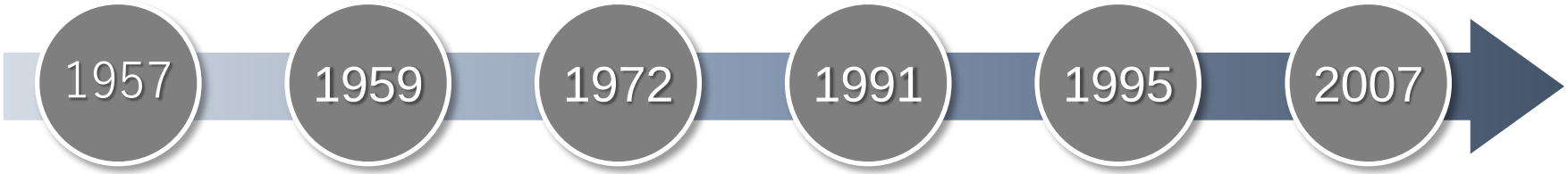
常見的程式語言

視覺化介面的開發工具，可快速建立各種應用程式。

Visual Basic

簡單又便於閱讀，擁有豐富且功能完備的函式庫，可以輕鬆完成很多常見的任務。

Python



常見的程式語言

主要是要在瀏覽器上執行程式而推出，後來延伸到遊戲及伺服器環境

Java Script

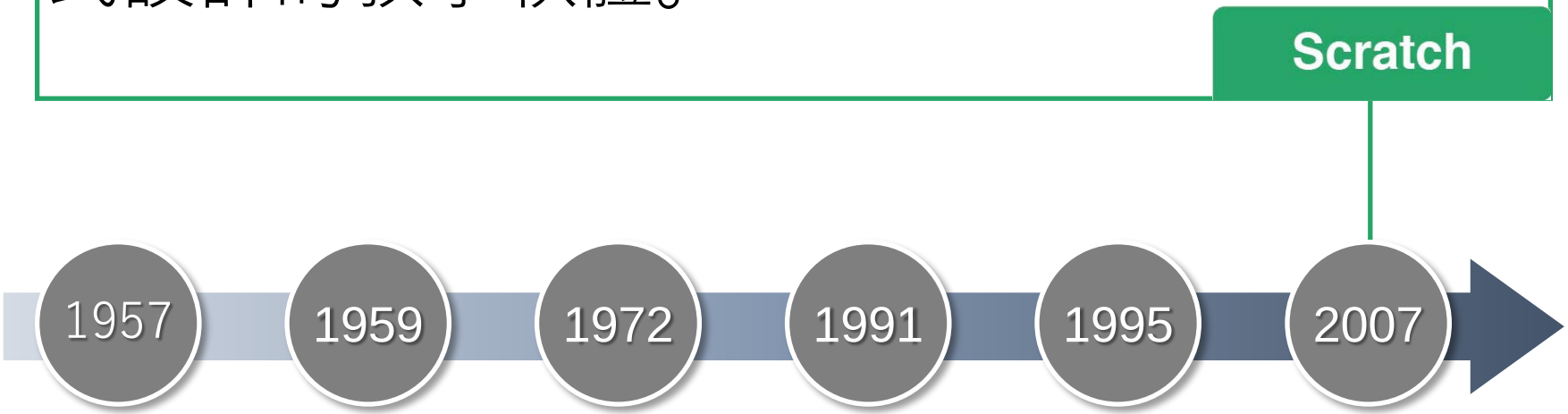
電腦、手機、平板電腦上都能夠執行的跨平臺程式語言

Java

1957 1959 1972 1991 1995 2007

常見的程式語言

用拖曳積木的方式來撰寫程式，可輕易完成各式各樣的動畫或互動遊戲，很適合入門程式設計的教學軟體。



問題與討論

你還想知道什麼?