

# C++資料結構與程式設計

*C語言概觀*

NTU CSIE 張傑帆

# 程式語言的分類

- 第一代語言：機器語言 (Machine Language)
- 第二代語言：組合語言
- 第三代語言：高階語言
- 第四代語言：查詢語言(Query Language)
- 第五代語言：物件導向與自然語言

# 程式語言的分類

## 第一代：機器語言 (Machine Language)

以連續0、1來編寫程式，執行速度最快。

屬機器導向語言。CPU 的架構不同，此種語言與機器相依度高，可攜性極低。

0、1組合而成，費時費力，實用性差且難維護。

記憶位址	內容(2 進制)	內容(16 進制)
1000	1010 0011 0000 0001	A301
1002	0000 0001 1011 0010	01B2
1004	0001 0011 1101 0101	13D5



# 第二代語言：組合語言 (Assembly Language)

亦稱低階語言，屬於一種符號式語言。

是使用助憶碼，由字母和數字組合而成。

$sum = 10 + 20$ ，組合語言寫法：

```
mov ax, 10 ;  
add ax, 20 ;  
mov sum, ax ;
```

屬於機器導向語言，和電腦硬體相依性高，不同CPU，語法不同，可攜性低。

適用於電腦專業人員編寫有關電腦系統或輸出入介面的驅動程式。

使用組譯器將撰寫的程式碼逐行翻譯成機器語言才能執行。

## 第三代語言：高階語言 (High-Level Language)

語法更接近人類語言與數學表示式，程式稍加修改，可在不同電腦系統上執行，可攜性高。

屬於程序導向語言

如：BASIC(交談式操作環境)、FORTRAN(工程)、COBOL(商業應用)、PASCAL、C 均屬之，由於都是屬於傳統高階語言，共同特點就是按照指令的邏輯順序執行。

## 第四代語言：查詢語言(Query Language)

屬於非程序語言以問題為導向，只描述問題不必敘述解決問題的步驟。

先透過前置處理器轉換成第三代的程序語言才能編譯成可執行碼。

包括結構化查詢語言SQL(Structural Query Language)適用於資料庫查詢或 AutoCAD 適用於工程繪圖。



# 第五代語言：物件導向與自然語言

物件導向語言，是一種比程序導向更進階的語言。

C++ 是在 C 中加入物件導向語法的程式語言。

此種語言每個物件擁有自己的屬性和方法，具有下列特性：

- 再利用(Reused)
- 繼承(Inheritance)
- 封裝(Encapsulation)
- 多形(Polymorphism)特性

使得物件有如積木一樣都具有某些小功能，物件與物件間利用呼叫可互傳資訊或兜成一個大程式。

由於網際網路蓬勃發展、超媒體與網路資訊服務充斥全球資訊網、以及智慧型裝置(智慧型手機與平板電腦)硬體設備愈來愈進步。

因此許多大廠紛紛提供能開發Web 應用程式(網頁程式設計)與智慧型裝置應用程式的程式語言，像這類的程式語言有：Java、VB、C#、ActionScript 3.0...等物件導向程式語言。

自然語言(Natural Language)屬於人工智慧語言，近似人類的語言是程式語言的終極目標。如：LISP(List Processing)、PROLOG(Logic Programming)。



# 翻譯器的分類

- 編譯器(Compiler)
- 直譯器(Interpreter)
- 組譯器(Assembler)

# 編譯器 (Compiler)

是電腦廠商提供的系統軟體(程式)。

將高階語言所寫的程式碼轉換成能直接被機器接受之目的程式。

優點：

是程式經編譯過存成目的檔，下次執行時程式若未修改過可馬上執行，較節省編譯和執行時間。

缺點：

編譯和連結時間較長而且程式有修改過必須重新編譯程式執行時必須將整個執行檔一次載入，需要較大的記憶體、程式存檔時亦需要較大的輔助儲存體空間、執行階段發生錯誤時除錯較難處理。

# 直譯程式 (Interpreter)

亦是電腦廠商提供的系統程式之一。

將高階語言所編寫的程式碼，依其敘述的邏輯順序，將指令逐一轉為機器語言指令後執行。

優點：

執行時所需記憶體空間和存檔時所需磁碟空間較小，且程式較易除錯適合初學者。

缺點：

每次執行均須重新翻譯，執行所需的時間較長，程式若供多人使用時效率較差。



# C 語言的沿革

C 語言的前身追溯到 1960 年以解決問題為導向的高階語言- ALGOL 60，當時博得好評價，不適合撰寫系統軟體。

1963 年英國劍橋和倫敦大學以 ALGOL60 為基礎，共同推出與硬體有關 CPL (Combined Programming Language) 語言，由於當時考慮層面過於寬廣，造成不方便撰寫系統軟體。

1968年 Martin Richards (世界公認C語言的鼻祖) 在英格蘭劍橋簡化CPL語言而發展出BCPL(Basic Combined Programming Language)。

1970年

Ken Thompson於美國Bell 實驗室再度精簡BCPL語言設計出接近硬體的 B 語言。

1972~1973 年間

Dennis Ritchie於美國 Bell 實驗室，為新型 PDP-11 電腦重新改寫 Unix 作業系統，結合B 語言和 BCPL語言重要觀念加上資料型別以及一些其他概念發展成 C 語言。

1973年

K. Thompson 和 D. M. Ritchie 兩人合作把 UNIX的 90% 以上用 C 改寫成即 UNIX-5，直到 1975 年 UNIX-6公佈後，才引起注意。

1978年

Ritchie 和 Brian Kernighan於出版「The C Programming Language」一書，奠定 C 語言完整架構，將此版本的 C 語言稱為 K&R C 語言。

隨後 C 語言百家爭名，產生出多種版本的C語言如：  
Lattic C、MS-C、Quick C 等，

為使 C 語言標準化，1983年夏，美國國家標準協會  
(ANSI) 制定一套 ANSI 的 C 語言標準。

標準化過程達六年之久，最後於 1989/12 ANSI 標準  
終於完成，1991年初 ANSI C 第一版 終於出現。

1987年美國寶蘭(Borland)公司結合了 K&R C 和ANSI C  
推出Turbo C，深受當時程式設計者的喜愛隨著資訊  
科技的進步，導致物件導向程式設計的流行。



1999 年3 月微軟公司推出Visual C++ 6.0 。

2002 年微軟公司推出Visual Studio .NET，此時將Visual Basic .NET、Visual C# .NET 和Visual C++ .NET 等程式語言整合至Visual Studio .NET 整合開發環境內。

2003 年微軟公司推出Visual C++ .NET 2003 的使用更加容易及更具親和力。

2005 年微軟公司推出Visual C++ 2005 。

2008 年微軟公司推出Visual C++ 2008 。

2010 年微軟公司推出Visual C++ 2010，此時將Visual C++ 2010 包含於Visual Studio 2010 Express 版中，並放在微軟網站上，以提供給使用者下載使用。

2012年微軟公司推出Visual C++ 2012 / 2013

# C 語言的特色

程式具有區塊結構及不嚴謹的資料型別檢查

為UNIX 作業系統所採用的程式語言

介於低階和高階語言中階程式語言

可呼叫處理硬體函式庫或自行設計需要函式庫來直接控制硬體，以提升硬體執行速度。另一方面C 語言可用來發展高階軟體介面

C 語言具有高階架構和低階功能

為一種可攜性的系統程式發展語言

具可攜性及高跨平台功能

結構化程式設計

提供指標及位址運算能力

允許使用動態資料結構



# 設計程式的步驟

問題定義

問題分析

設計演算法

撰寫程式

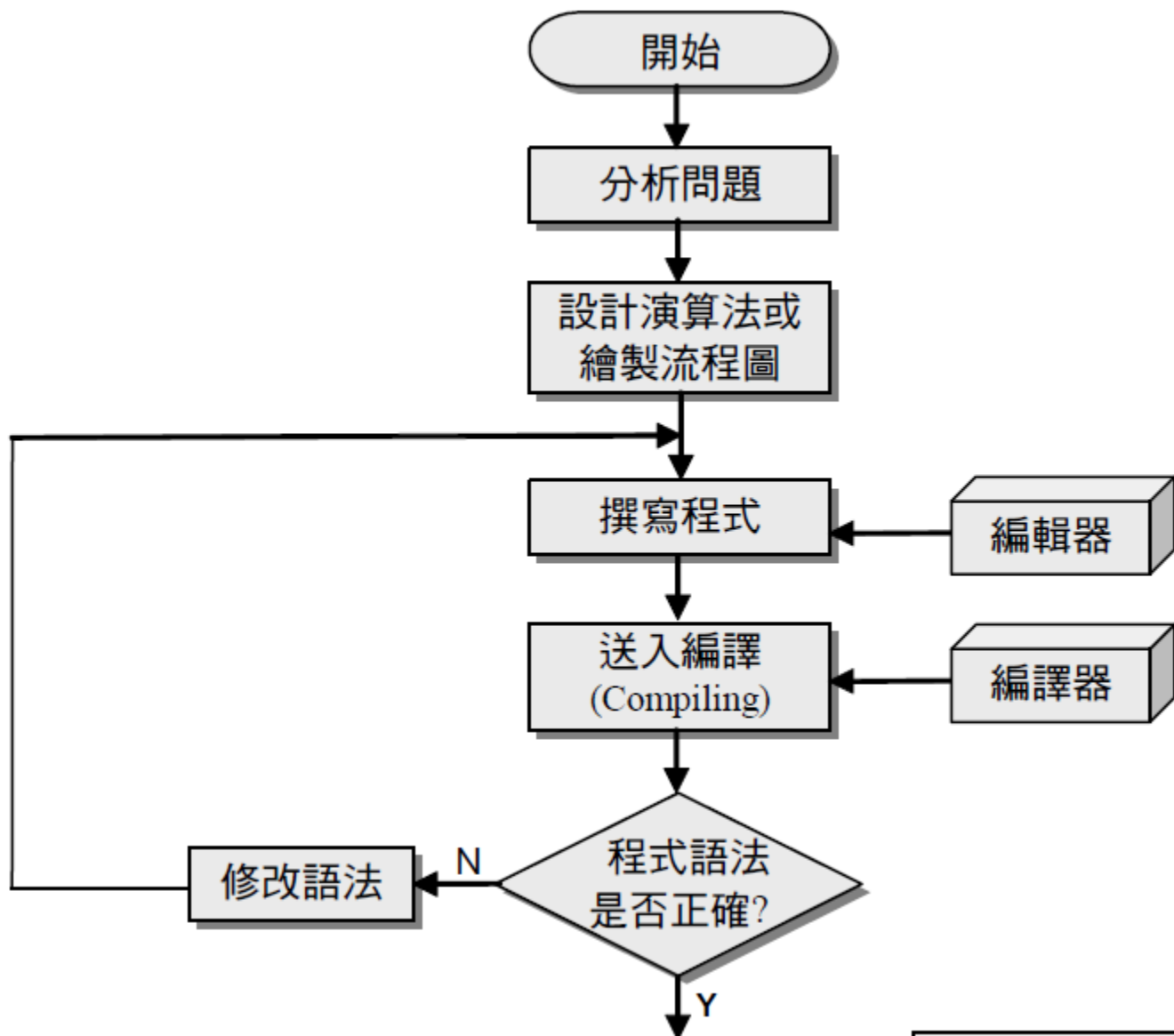
程式的測試與維護

(包含驗證、測試、除錯與維護)

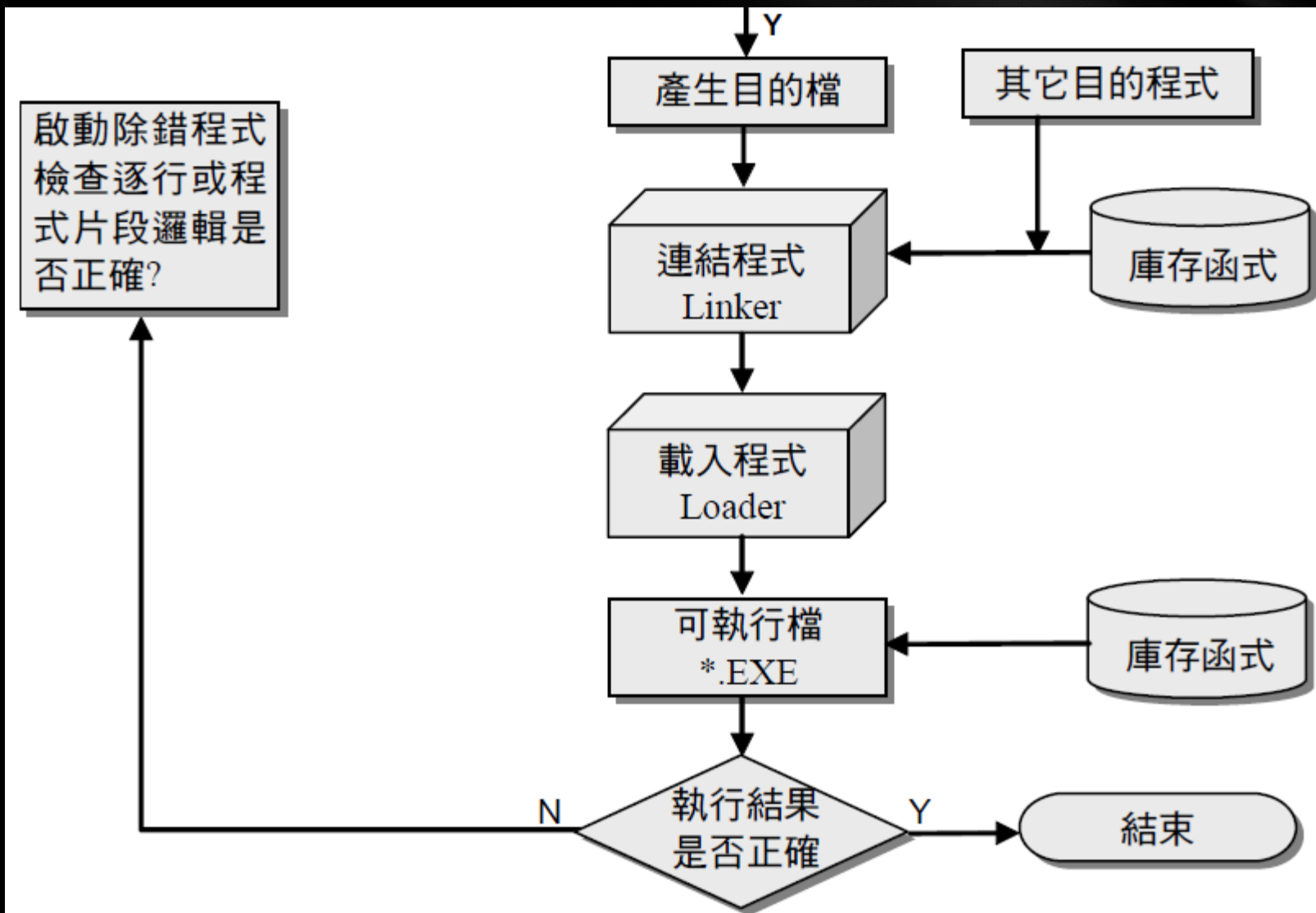
# 設計演算法

一個好的演算法應具備下列五大要件：

- ① 有限性：演算法必須在有限個步驟內解決問題。
- ② 明確性：演算法中的每個步驟都必須很清楚地表達出來。
- ③ 有效性：必須在有限的時間內完成。
- ④ 輸入資料：包含零個或一個以上的輸入資料。
- ⑤ 輸出資料：至少產生一個輸出結果。





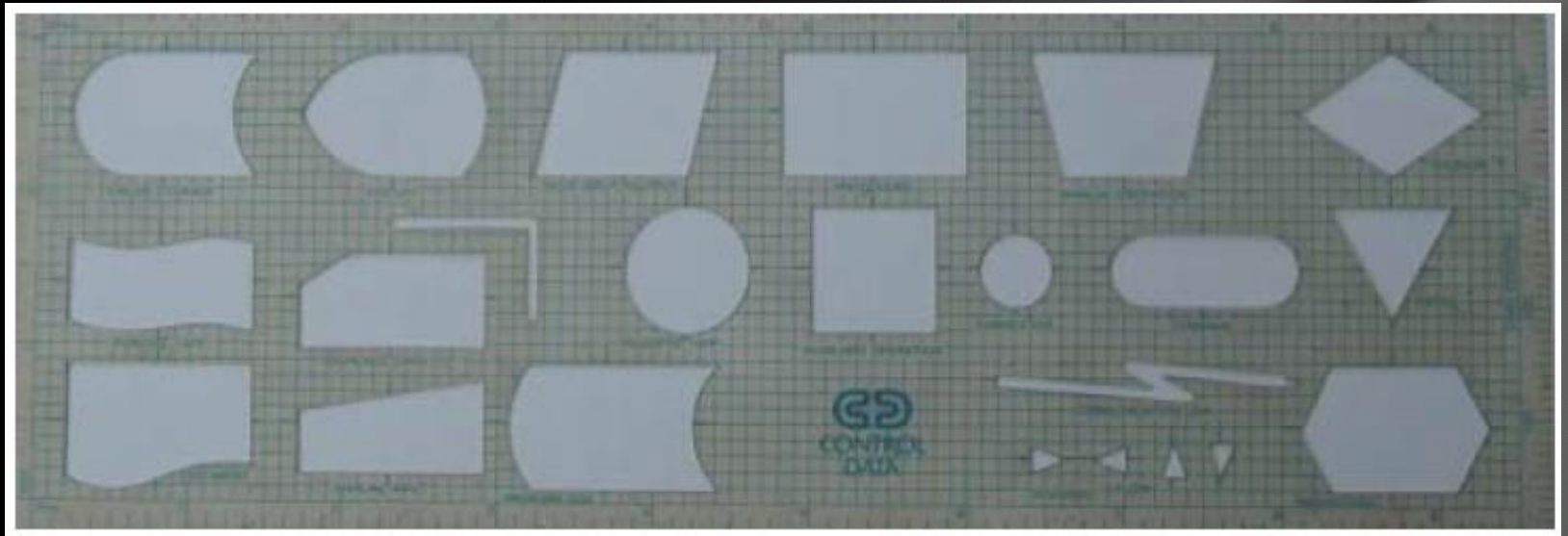


# 一個設計良好的程式所具備的條件


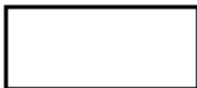





1. 程式具可讀性且程式中重要部分都有詳細註解
2. 程式執行結果符合預期且正確
3. 程式具模組化或結構化，以利程式修改或更新時更便捷
4. 程式架構有完整說明
5. 程式執行效率和相容性要高，不會因更換設備而造成錯誤或執行速度變慢

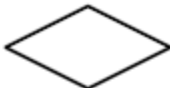


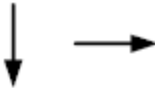


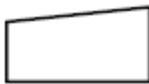
# 流程圖

所謂的「**流程圖**」即是使用各種不同的圖示符號來描述問題的解決步驟以及進行的順序。流程圖中所使用各種符號的繪製都已標準化。










符號	功能
	開始/結束
	程序處理
	輸入或輸出
	連結符號
	儲存資料
	準備作業
	內部儲存裝置

符號	功能
	判斷比較
	隔頁連結
	預設處理作業
	工作流向符號
	文件
	多重文件
	人工輸入

	卡片
	匯合連接點
	自動分頁
	抽選
	合併
	磁碟
	顯示

	打孔紙帶
	或
	排序
	延遲
	循序存取儲存裝置
	直接存取儲存裝置

由鍵盤輸入密碼，限輸入三次  
若在三內答對密碼，則顯示“密碼正確！”  
若連續答錯三次，則顯示“密碼錯誤！”  
其流程圖表示方式：

