

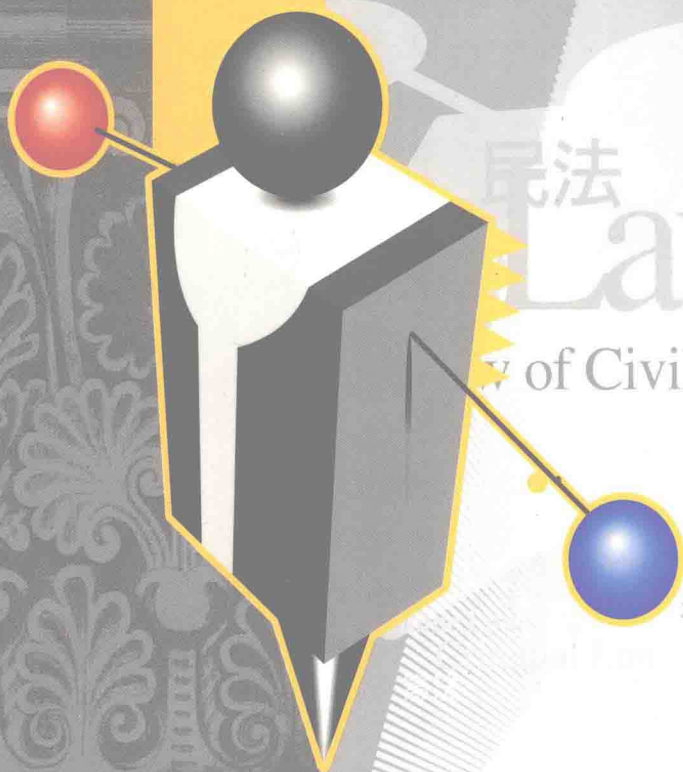
Programming Language

程式語言

重·點·整·理

檢察事務官必備用書

康仁德 編著



高
點

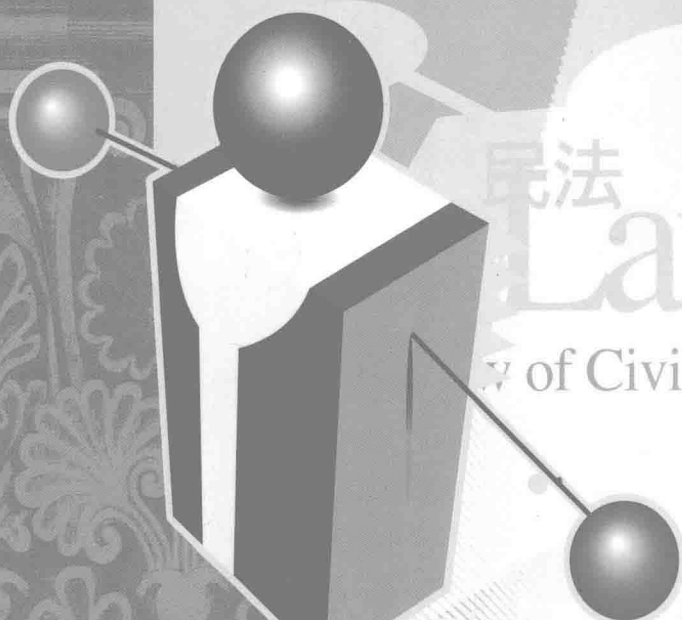
Programming Language

程式語言

重·點·整·理

檢察事務官必備用書

康仁德 編著



高
點

來勝 (License) 證照考試系列

程式語言

編著者：康仁德

出版者：高點文化事業有限公司

郵 撥：15834067高點文化事業有限公司

電 話：(02)2381-5766

傳 真：(02)2388-0876

網 址：www.get.com.tw

E-mail：publish@mail.get.com.tw

中華民國94年12月五版

行政院新聞局出版事業登記證局版臺業字第4833號

建議售價450 元

著作權所有・翻印必究

L350E

ISBN 957-814-359-1



檢察事務官應試須知

(表一)

類 科	應 試 資 格	應 試 科 目	
		普 通 科 目	專 業 科 目
偵查實務組 財經實務組 電子資訊組 營繕工程組 檢察事務官	1. 公立或立案之私立專科以上學校或經教育部承認之國外專科以上學校各所系科畢業得有證書者。	1. 中華民國憲法 2. 國文（作文、公文與測驗）	1. 行政法 2. 刑法 3. 智慧財產權法 4. 刑事訴訟法 5. 犯罪學 6. 英文
	2. 經普通考試或相當普通考試之特種考試及格滿三年者。		1. 刑法及刑事訴訟法 2. 證券交易法與商業會計法 3. 中級會計學 4. 銀行實務 5. 稅務法規 6. 審計學
	3. 經高等檢定考試及格者。		1. 刑法及刑事訴訟法 2. 系統分析 3. 資料結構 4. 計算機網路 5. 電子學與電路學 6. 程式語言
			1. 刑法及刑事訴訟法 2. 結構分析（包括材料力學與結構學） 3. 結構設計（包括鋼筋混凝土設計與鋼骨結構設計） 4. 施工法（包括土木、建築施工法與工程材料） 5. 營建法規 6. 政府採購法
考試日期	95年7月15日至17日		

【來勝證照考試中心提供】

檢察事務官錄取人數一覽表

(表二)

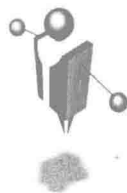
組別	年度	報名人數	到考人數	錄取人數	錄取率
偵查實務組	93	2447	1469	41	2.79%
	92	2379	1324	64	4.83%
	91	2477	1460	64	4.38%
	90	1694	1019	36	3.53%
財經實務組	93	735	340	8	2.35%
	92	807	378	11	2.91%
	91	761	344	11	3.20%
	90	687	333	10	3.00%
電子資訊組	93	370	168	6	3.57%
	92	434	194	7	3.61%
	91	446	251	7	3.26%
	90	290	138	7	5.07%
營繕工程組	93	435	164	6	3.66%
	92	546	206	7	3.40%
	91	655	292	7	2.40%
	90	739	307	5	1.63%

※實際報名數及錄取人數以考選部分公告為主。

【來勝證照考試中心提供】

高點律師、司法官系列用書

法學界的品質圖騰



二十一世紀資訊化時代來臨的同時，法律應用也將是時代的趨勢，無論是管理、行銷等商業行為，或是教育、生活的秩序建構，「完全法律人」的重要性與日俱增。

因此，高點開辦「律師、司法官班」的初衷，即啓始於未來世紀的宏觀視野。動員了聞名全國的高點研究室菁英體系，以全國首屈一指的氣勢，開創出令人震撼矚目的堅強輔考實力。這是高品質的圖騰象徵，在師資、設備、考情、出版……優勢整合的工作上，都是經得起挑剔的。

高點的出版亦是如此，這一套「律師、司法官考試專用叢書」，創新及實用的編輯新意，在法律權威專家的積極參與下，讀者將感受到前所未有的閱讀樂趣。舉凡簡明式綱要表解、重點提示、考情分析、內容說明等，研讀的便利性與有效性，自是不言而喻。同時這一系列叢書廣錄相關重要考試之歷屆試題，實戰演練、鑑往知來的效應當能事半功倍；全書系的索引功能，也是讀者可善加利用的特性之一。

「高點與高點賽跑」，向來是我們在考試出版領域自我鞭策淬礪的動力，堅持高品質的作法，以及對每一位讀者閱讀的尊重，亦是高點亟欲樹立典範的用心。我們常以爲：當您翻開這一本書時，一種由信任而產生的關係已經開始建立。歡迎您領略我們的用心與專業，期望您因此而成就高點！

著作權／不容侵犯

下列文字爲著作權法之部分條文，仁人君子敬請自重，凡侵犯著作權者，必依法究辦。

《著作權法》第六章 權利侵害之救濟

■第八十七條

有下列情形之一者，除本法另有規定外，視爲侵害著作權或製版權：

- 一 以侵害著作人名譽之方法利用其著作。
- 二 明知爲侵害製版權之物而散布或意圖散布而公開陳列或持有者。
- 三 輸入未經著作財產權人或製版權人授權重製之重製物或製版物者。
- 四 未經著作財產權人同意而輸入著作原件或其重製物者。
- 五 以侵害電腦程式著作財產權之重製物作爲營業之使用者。
- 六 明知爲侵害著作財產權之物而以移轉所有權或出租以外之方式散布者，或明知爲侵害著作財產權之物意圖散布而公開陳列或持有者。

《著作權法》第七章 罰 則

■第九十一條

擅自以重製之方法侵害他人之著作財產權者，處三年以下有期徒刑、拘役，或科或併科新臺幣七十五萬元以下罰金。

意圖銷售或出租而擅自以重製之方法侵害他人之著作財產權者，處六月以上五年以下有期徒刑，得併科新臺幣二十萬元以上二百萬元以下罰金。

以重製於光碟之方法犯前項之罪者，處六月以上五年以下有期徒刑，得併科新臺幣五十萬元以上五百萬元以下罰金。

著作僅供個人參考或合理使用者，不構成著作權侵害。

■第九十二條

擅自以公開口述、公開播送、公開上映、公開演出、公開傳輸、公開展示、改作、編輯、出租之方法侵害他人之著作財產權者，處三年以下有期徒刑、拘役，或科或併科新臺幣七十五萬元以下罰金。

本書精華要點導引

本書乃針對檢察事務官考試的考生所準備的一本程式語言考試用書。由於程式語言的相關書籍非常多，每一本書所涵蓋的範圍及強調的重點皆各有不同，但或多或少會在考試中用到。爲了能夠讓考生更有效率地準備這一科，筆者根據十幾年程式語言的教學經驗，綜合各參考書籍的精華重點，編寫完成此書，期能提供考生既完整又精要的準備資料。

全書共分十六章，第一章介紹程式語言的基本概念，第二章是語法和語意，第三至六章說明變數、運算式、控制結構及資料型態，第七、八章是副程式及其實作，第九至十一章探討資料抽象化，並行處理及例外處理等特性，第十二章是輸入和輸出，第十三至十六章分別介紹了LISP、Scheme、ML、Miranda、Prolog、Smalltalk、C++及Java等特殊的程式語言，附錄則蒐集近幾年檢察事務官之考題及解答。

本書的每一個章節，都是針對該章的重點做循序漸進且深入淺出的闡述，並附有眾多實例說明。在每一章最後的範題精選部分，涵括了最近五年內所有高考和特考的相關考題及解答，希望透過這些考古題的演練，能培養考生堅強的解題實力，並掌握未來考試的趨勢。

成功是不斷努力累積而成，善用工具書更是致勝的關鍵。

高點將繼續出版各類好書，幫助您勝利成功。

程式語言準備要領

一、基本要求

準備檢察事務官考試的程式語言，最好能先有程式設計的經驗，至少用過一或二種高階語言，如此對程式語言做進一步探討時，會有較清晰具體的概念。

二、相關學科

程式語言的語法、語意和「編譯程式」有直接的關係，而控制結構、資料型態則和「資料結構」有密切的關連，要得高分，最好亦能讀些相關書籍（請參閱高點出版「資料結構」，王致強著一書）。

三、先了解觀念再深入研討

程式語言考試的重點很多，在研習每一個重點時，都應了解其基本觀念，再深入探討各種理論及各種語言之做法。

四、從語言特性了解程式語言

由於高階語言有幾百種之多，要全部學會幾乎不可能，也沒必要。準備的要領是從語言的特性著手，對於每一種特性，應去探討有那些語言常被提出來討論，針對這些具有代表性的語言，比較它們在該特性上的差異，或分析其優缺點，藉此來了解這些程式語言。

五、從使用者觀點和系統觀點分別研討各種特性

使用者觀點是指如何利用這些語言特性來撰寫程式，發揮其效用；而系統觀點是指在系統內部如何實作這些特性，其使用的資料結構為何及進行何種操作。例如：各種資料型態的使用特徵及內部實作的描述子（descriptor）分別為何？或副程式的分類與特性及各種實作方法等等。

六、特殊語言的認識

有許多特殊類型的語言，如函數型語言（LISP、Scheme、ML……）、邏輯程式語言（PROLOG）、個體導向語言（Smalltalk）及網路程式語言（Java）等，常考程式了解的問題（即給一個程式，問你它有何功用？或針對某輸入資料可產生何種結果？）要回答這類問題，必須對該特殊語言有基本的認識，你不一定要具有撰寫完整程式的能力，但至少能了解其語法結構，能看得懂程式，並多演練該語言之程式的執行方式，如此便能應付自如。

七、個體導向程式設計（OOP）

近幾年的許多特考中，經常出現OO的程式設計問題。目前OOP所使用的語言主要仍以C++為主，但Java則是OO在網路程式設計的重要趨勢，因此考生除了應熟悉此種語言之外，最好亦能培養撰寫C++或Java程式的能力。

八、勤做考古題

欲鑑往知來，了解各種考試的出題方向及考題型態，考古題是最佳的途徑。因此，勤做考古題是培養基本應考實力的不二法門，各類考試中有關程式語言的題目，都值得蒐集並多加練習。

九、參考書目

以下所列為本書中的許多程式和圖表所參考之書籍：

- (一) R. W. Sebesta, *Concepts of Programming Languages*, Six Edition, 2004, 開發.
- (二) E. Horowitz, *Fundamentals of Programming Languages*, Second Edition, 1986.
- (三) T. W. Pratt, *Programming Languages: Design and Implementation*, Second Edition, 1984.
- (四) R. Sethi, *Programming Languages: Concepts & Constructs*, Second Edition, 1997, 台北圖書.
- (五) K. Slonneger, B. L. Kurtz, *Formal Syntax and Semantics of Programming Languages*, 1995, 新月圖書.

- (六)L. W. Friedman, *Comparative Programming Languages: Generalizing the Programming Function*, 1991, 新月圖書.
- (七)D.Appleby, J. J. VandeKopple, *Programming Languages: Paradigm and Practice*, 1997, 新月圖書.
- (八)R. Wilensky, *LISP Craft*, 1984, 開發.
- (九)W. F. Clocksin, C.S.Mellish, *Programming in Prolog*, 1981.

目 錄

本書精華要點導引

「程式語言」準備要領

第 一 章	程式語言導論	1-1
第 二 章	程式語言之語法和語意	2-1
第 三 章	變 數	3-1
第 四 章	運算式和指定敘述	4-1
第 五 章	控制結構	5-1
第 六 章	資料型態	6-1
第 七 章	副程式	7-1
第 八 章	實作副程式	8-1
第 九 章	資料抽象化	9-1
第 十 章	對稱與並行副程式	10-1
第十一章	例外處理	11-1
第十二章	輸入與輸出	12-1
第十三章	函數型程式語言	13-1
第十四章	邏輯程式語言	14-1
第十五章	個體導向程式語言	15-1
第十六章	網路程式語言－JAVA	16-1
附 錄	歷屆試題	A-1

Chapter 1

程式語言導論

1-1 程式語言基本概念

一、定義

由於程式 (program) = 演算法 (algorithm) + 資料結構 (data structure)，因此凡是能描述演算法和資料結構的系統化符號即稱為程式語言 (programming language)，它是可指揮電腦執行指定動作的一種語言。

二、程式語言的分類

程式語言的發展可分成好幾代，愈新一代的程式語言愈方便人們使用，也愈接近人類所講的語言。依程式語言的特性通常可區分為以下幾類：

(一)第一代程式語言——機器語言：機器語言 (machine language)

是由0與1組成的二進位碼所構成，不同類型的計算機擁有不同的指令群，其對應的二進位碼亦不相同，因此機器語言是一種與機器相關 (machine dependent) 的語言。機器語言可以直接被計算機接受並執行，但對程式的撰寫及維護都極為困難。除非有需要，否則很少直接利用它來寫程式。

(二)第二代程式語言——組合語言：由於機器語言的程式難以撰寫和

閱讀，於是將每一條機器語言指令用一個助記憶符號（*memonic symbol*）來取代，這些助記憶符號即稱為組合語言（*assembly language*），又稱為低階語言（*low-level language*）。例如：若 $4E_{(16)}$ 代表資料搬移的機器語言指令，在組合語言中可用 **MOVE** 這個較易記憶的符號來取代。顯然用組合語言撰寫程式要比機器語言方便得多，但計算機卻看不懂，因此必須透過一套系統程式來將組合語言程式轉換為機器語言，此套系統程式稱為組合格式（*assembler*）。組合語言指令與機器語言指令因具有一對一的對應關係，因此亦屬於與機器相關的語言。

(三) **第三代程式語言——高階語言**：由於組合語言仍為與機器相關的語言，須了解計算機之結構才能撰寫程式，而且所寫程式只能在特定機器上使用，不具有移植性，一般適用於電腦專業人員使用，不易普及。因此為了使電腦更廣泛地為人們所接受，專家們依據科學、商業或統計等需要而發展一些較接近人類日常語言的程式語言，即稱為高階語言（*high-level language*）。此種語言與計算機結構無關，具有移植性，可在不同的機器上使用，因此是一種與機器無關（*machine independent*）的語言。高階語言的程式必須透過編譯程式（*compiler*）或直譯程式（*interpreter*）轉換為機器語言後，才能在計算機上執行。此外，高階語言的每一句稱為一個敘述（*statement*）而不再是指令，它與指令為一對多的關係。程式的撰寫是利用這些敘述來描述如何解決問題的程序，因此高階語言又稱為程序導向語言（*procedure-oriented language*）。

(四) **第四代程式語言——超高階語言**：超高階語言（*very high level language*）通常稱為4GL（為 *fourth generation language* 之縮寫，而高階語言為3GL），是屬於問題導向語言（*problem-oriented language*）或稱非程序性語言（*non-procedure language*），因其程式中只需指示電腦去「做什麼」（*what to do*）即可，而不必告訴電腦該「如何做」（*how to do*）。

由於電腦硬體技術的突飛猛進，不僅在速度上有重大突破，且記憶體容量亦愈來愈大，因此程式設計師已不需再如過去那樣，要費盡心思精簡程式，以求更高的執行速度及降低記憶體需求。目前所關注的是如何快速地開發一套完整的應用程式，因此超高階語言便應運而生。

通常超高階語言都與解決某類特定問題有關，由一系列整合性模組軟體所構成，可供程式設計師快速地開發應用程式，屬於高生產力的程式語言（其生產力約為第三代語言的十倍）。

(四)第五代程式語言——自然語言：自然語言（natural language）是人類日常生活所用的語言，如中文、英文、日文……等均為自然語言。在計算機中，目前自然語言的處理技術尚不成熟，只能做有限度的運用。例如：用以存取知識庫的知識庫語言（knowledge-based language），即是以處理某特定領域的資訊為基礎的自然語言，但它對於較複雜的邏輯問題則仍無法處理。

→注意：高普特考程式語言的重點都是以第三代的高階語言為主，這也是本書的重點。

三、為何要學習程式語言

- (一)增進對所使用程式語言的了解。
- (二)可以改進程式的架構。
- (三)有助於選擇合適的程式語言。
- (四)比較容易學習新的程式語言。
- (五)較容易設計出一種新的程式語言。

四、電腦的應用領域及相關的程式語言

- (一)科學應用，如FORTRAN。
- (二)商業應用，如COBOL。

(三)人工智慧，如LISP、PROLOG。

(四)系統程式，如發展UNIX所用之C語言。另外有一些可應用在特定機器上開發系統軟體的機器導向高階語言（machine-oriented high-level languages），如：

IBM機器上之PL/S（PL/I之後代）

Digital機器上之BLISS

Burroughs機器上之Extended ALGOL

(五)超高階語言，如UNIX上面的各種腳本語言（scripting languages）：shell、ksh、awk、perl、……等；又如常用來執行資料庫操作的第四代語言：RAMIS（Mathematica公司）、FOCUS（IBM PC）、Informix（XENIX）。

(六)特殊目的語言（問題導向語言），如用於產生商業報表的RPG（Report Program Generator）、用於系統模擬的GPSS（General Purpose System Simulator）、用於結構工程的STRESS（Structure Engineering System Solver）……等。

1-2 高階程式語言

一、高階語言的優點

使用高階語言撰寫程式比起使用組合語言具有以下的優越性：

(一)學習容易，學習時間較短。

(二)生產力較高，對於人力及管理的需求較少。

(三)具有與機器無關的優越性。

(四)程式的除錯較容易。

(五)編寫較大的應用程式時，不至因瑣碎的程式細節而影響設計的進行。

(六)高階語言的結構本身就具有卓越的文件說明能力。

二、高階語言的實作

高階語言實作（implementation）的觀念，是指如何讓高階語言的程式在計算機中執行的作法。而高階語言實作的計算機階層圖如圖<1-1>：



圖<1-1>

高階語言實作的方法有三種：(一)編譯（compilation）——以編譯程式（compiler）來實作；(二)直譯（interpretation）——以直譯程式（interpreter）來實作；(三)不純的直譯（impure interpretation）——混合式實作法。

(一)編譯程式：

1. 定義：編譯程式是一種系統程式，用來將高階語言（如 FORTRAN、COBOL、PASCAL 等程式語言）所寫的原始程式（source program）轉換成能直接被機器接受的等效目的程式（object program）——亦即機器語言程式。