

計算機概論與原理

歷屆研究所及高、
普、特、留考試題精解

蕭培壩 編著

曉園出版社

計算機概論與原理

歷屆研究所及高、
普、特、留考試題精解

蕭培墉 編著

曉園出版社

版權所有・翻印必究

初 版

1986年元月第一次印刷發行
1986年5月第二次印刷發行

計算機概論與原理

定 價：新臺幣 180 元

原 著 者：蕭 培 墉

發 行 人：黃 旭 政

發 行 所：曉 園 出 版 社

臺 北 市 青 田 街 7 巷 5 號

電 話：(02) 394-9931 三 線

郵 撥：0 7 4 5 3 3 3-7 號

門市部(1)：臺 北 市 新 生 南 路 三 段 96 號 之 三

電 話：3917012・3947375

門市部(2)：臺 北 市 重 慶 南 路 一 段 61 號 地 下 樓

電 話：三 一 四 九 五 八 〇

門市部(3)：臺 北 縣 淡 水 鎮 英 專 路 71 號

電 話：六 二 一 七 八 四 〇

門市部(4)：臺 中 市 西 屯 區 文 華 路 113 號

電 話：(04)251-2759・254-6663

印 刷 所：遠 大 印 刷 廠

臺 北 市 武 成 街 36 巷 16 弄 15 號

出版登記：局版臺業字第 1244 號

著作執照：臺內著字第 號

自序

這是一本效率極高的書，是以極難找出簡捷確切的形容詞，傳達給諸君得知，所幸有二種方法可協助，讀者客觀評斷進而認同本書的價值。

第一就是買本回家精研。

第二就是以事實證明，並拿出最大的誠心。

關於後者，個人在此鄭重聲明，本書任何地方絕無故惠“留一手”，如有任何疑問，歡迎您提出您的質疑。

關於前者，個人已考上台大電機研究所及留歐公費留學，並曾考上電信，郵政特考。

寫一本讓讀者看來具有價值又效率高的書，所需投注的心力，非外人所能想像，除了個人的心血外，我必須誠摯的感謝何啓琳先生與蘇美子小姐，沒有他們的鼓勵與協助，本書則無從誕生，此外還要感謝台灣中區電信管理局的黃文生君與蔡有富組長，在百忙中仍給予我豐碩的經驗與實務的指導，最後，感謝曉園出版社的支持與編審人員的協助，始能順利發行。

本書承陳永孝君撥冗做最後編審校對，在此也感謝他的支持。

如果本書對您確有幫助，請不要忘記這是多人組合的貢獻，然而，儘管我感謝這許多人，但本書仍由我負責，謹祝諸君努力衝刺，更上層樓。

作者謹識

序

把蕭兄這本書看完後，我深受震撼，本書可說是百分之百考試取向，可讓你考試奪取高分，又能增進實力，蕭兄竟花如此大之心血完成這本幾近完美的書，此種精神令人感動，本書之惟一缺點是「出得太慢」，害我不能在考研究所前研讀之。

陳永孝 謹識于台大電機研究所

74 年 12 月

目 錄

問題 1	(a) Top-Down Design ? (73 中央)	
	(b) Bottom-Up Design ?	1
問題 2	Top-Down Design ? 優點 ? (73 中央)	1
問題 3	Structure Program 構成要件及優點 ?	
	(71 公費 · 72 普考)	2
問題 4	Structure Programming 基本結構 ?	
	(74 , 73 中央 · 72 交大)	3
問題 5	程式設計五大步驟 ? (74 中原 · 72 中科)	4
問題 6	Algorithm ?	5
問題 7	計算機架構幾種 ? (74 台大)	6
問題 8	PASCAL 資料型態 (data type) 分類。	
	五大基本資料型態為何 ? (70 清大)	8
問題 9	FORTRAN 77 與 FORTRAN IV 比較	9
問題 10	Concurrent ? Simultaneous ?	9
問題 11	Subroutine 、 Coroutine 、 Conroutine ? (69 電信)	
		10
問題 12	Subroutine 、 Macro ? (69 電信)	13
問題 13	Interrupt 、 Subroutine ?	14
問題 14	Maskable interrupt 、 Nonmaskable interrupt ?	
		14
問題 15	Interrupt 的種類 ? (73 清大)	15
問題 16	Isolated I/O ? Memory-Mapped I/O ? 並比較之。	

	(74 成大)	19
問題 17	三大 I/O Interfaces ?	
	Programmed I/O 、 Interrupt I/O 、 DMA ?	
	(72 中科 · 70 清大)	20
問題 18	(1) DMA Controller 詳述。(74 台大)	
	(2) DMA Transfer 與 Program Controlled I/O。	
	(71 公費)	21
問題 19	DMA 中的 Cycle-Stealing 如何完成 ?	
	DMA 何以比 Interrupt I/O 快 ? (72 台大)	29
問題 20	求二維 Array 的 Address Function。(73 中興)	
	一維、二維、三維、n 維 Array 定址方式。	
	Array 應用 (解三角形多元一次方程組) 。	30
問題 21	二維 Array 實例。(72 清大)	34
問題 22	二維 Array 典型題。	35
問題 23	Loader ? Linker ? (74 中原 · 72 中科 · 71 清大)	
		39
問題 24	System Program ? Application Program ?	
	(74 成大)	41
問題 25	Compiler ? Interpreter ? 並比較之。	
	(74 清大、成大、中原)	43
問題 26	Language 分那三類 ?	43
問題 27	Compiler 的五大 phases ? (72 中科)	44
問題 28	高階語言執行的步驟與功能。(73 普考)	46
問題 29	(1) Text Editor (2) Debugging Aids	
	(3) Preprocessor (4) Monitor (72 台大)	47
問題 30	(1) Source Program (2) Object Program	
	(3) Translator (4) Assembler (74 中原) ...	48

問題 31 (1) Pseudo-Instruction ? (73 台大)	
(2) Two-Pass Assembler ?	49
問題 32 組合語言之 Machine-Operation、Pseudo-Operation 及 Macro Call。	
Machine-Operation (Assembly) 指令分類。(71 公費)	51
問題 33 SPOOLing ? 功能、原理、操作原理。	
(74 成大 · 72 台大 · 70 清大)	54
問題 34 (1) Recursive subprograms	
(2) 比較 COBOL、FORTRAN、ALGOL、BASIC、 PL/1、RPG、APL、GPSS、LISP、SNOBOL、 PILOT、PASCAL、C 等。(71 高考 · 70 交大) ...	55
問題 35 Scratchpad Memory ? 優點 ?	
2-Port Memory (71 清大 · 70 清大)	60
問題 36 計算質數的演算法。	
演算法的重要性與學習建議。(69 年交大)	61
問題 37 執行排列 (Permutation) 的演算法。(74, 69 清大)	
執行組合 (Combination) 的程式。	63
問題 38 印出九九乘法表的流程圖。(74 中原)	66
問題 39 (1) Recursive Procedure ? (70 高考)	
(2) Recursive Procedure for $N!$ 。	
(72 高考 · 中科 · 71 特考)	67
問題 40 (1) Tower of Hanoi ?	
(2) Recursive algorithm for Hanoi Tower。	
(73 中興)	68
問題 41 以 Stack, Noncursive 方式取代 Recursive Algorithm。	
舉實例比較。(74 成大)	71

問題 42	(1) Preorder、Inorder、Postorder expression。	
	(2) Threaded Tree。(74 大同·73 中央·72 高考·71, 70, 69 清大).....	74
問題 43	(1) Tree ? Binary Tree ? 並比較之。	
	(2) Binary Tree 的用途。(74 清大)	
	Tree 的表示方法。.....	78
問題 44	(a) Tree 化成 Binary Tree。	
	(b) Preorder、Inorder、Postorder Expressions。	
	(73 台大)	82
問題 45	Tree 化成 Binary Tree。(74 中央)	83
問題 46	Quick Sort ? 詳述。實例印證。(74 清大·72 高考)	
		84
問題 47	Quick Sort 與 Insertion Sort 之比較。	
	為何稱之“ Quick ” Sort ? (72 台大)	
	Decision Tree	89
問題 48	Quick Sort 與 Heapsort 之比較。	
	如何加快 Quick Sort ? (73 台大)	
	Quick、Insertion、Heep、2-way Merge Sort 之比較。.....	92
問題 49	Searching 種類(七種)? (74 台大)	
	Hashing 功能? Hashing Search 優點。	
	(74 高考·72 高考)	
	Binary Search 實例追蹤(trace)、基本條件?	
	(72 高考)	
	Sequential、Binary、Hashing Search 比較。	
	(69 清大)	
	有關 Sequential、Binary Search 之問題。	

(70, 67, 65 交大)	93
問題 50 Sorting ? 用途 ? (74 清大)	
與 Searching、Hashing 比較。	102
問題 51 Hashing ? 用途、特點。(74 清大·74 中原)	103
問題 52 (1) Hashing (Hashing Functions) 有幾種(三種)?	
(72 高考)	
(2) Collision ? 如何解決?	
Perfect Hashing Function (PHF) ? Minimal	
PHF (MPHF)	103
問題 53 B-Tree ? 特性、用途、舉例。(74 清大、中原)	
Balanced Binary Tree ?	
m-way Tree ?	107
問題 54 計算 Fibonacci Number (Recursive、Nonrecur-	
sive)。(69 交大)	112
問題 55 IEEE-488, RS-232 C (70 資策·69 電信)	
(GPIO, MODEM, IEEE, EIA, 20 mA-current loop)	
(modulation, demodulation)	113
問題 56 RS-232 C 介面、電氣特性。DB-25 ? (73 清大) ...	115
問題 57 N! (Stack, Recursive, Nonrecursive)。(72 台大)	
.....	120
問題 58 Graph, Spanning Tree, Minimal Spanning Tree。	
(74 中原·73 中興)	123
問題 59 Relation 的四種表示方法(離散數學)。(73 中央)	
.....	124
問題 60 edge matrix, adjacency matrix, incidence	
matrix。	126
問題 61 Dynamic RAM, Static RAM,	

Mask ROM, EPROM, PLA。(71 清大·69 電信)...	127
問題 62 GCD (Recursive, Nonrecursive)。	
(71 高考·69 電信)	132
問題 63 全雙工、半雙工、單向單工。(70 清大)	
On-Line, Off-Line。	134
問題 64 Protocal ?	
非同步串列傳輸。	
Parity error, Framing error, Overrun error。	
LRC, CRC。	
並列與串列傳輸。(70 清大)	137
問題 65 Polling I/O, Interrupt driven I/O	142
問題 66 multiport memory, cache memory。	
Hit ratio, mapping process。(73 清大)	143
問題 67 Multiport memory, Crossbar switch,	
Single common bus, Dual-bus structure。	
Interprocessor communication bus arbitration,	
mutual exclusion。(72 台大)	145
問題 68 Bit-sliced microprocessor。(69 電信)	152
問題 69 Computer, Microprocessor, Microcomputer。	
(72 中科)	153
問題 70 Microcomputer。(72 交大)	155
問題 71 Memory module, External system bus,	
Interleaved Memory Bank。(72 台大)	156
問題 72 Addressing mode (absolute、immediate、	
index、indirect)。(73 台大·70, 68 清大)	159
問題 73 Addressing mode (5 種→11 種)。	
(70 高考、公費·68 清大)	163

- 問題 74 Microinstruction。(73 台大、中央·72 台大)
 Microprogramming。(70 清大)
 Macroinstruction。
 Control memory, Control word, Dynamic micro-
 programming, Hard-wired control, Microcode。
 165
- 問題 75 Hard-wired control unit & Microprogrammed
 control unit 比較。(70 交大) 168
- 問題 76 Microprogrammed controlled computer CPU
 架構 (Fetch, Execute, ADD)。(72 台大) 170
- 問題 77 Microinstruction
 Microprogramming & Control matrix (Fetch,
 Execute)。(73 中央)
 (Horizontal, Vertical Microinstruction)。
 Nanoinstruction。..... 177
- 問題 78 machine instruction → micro-routine → oper-
 ations。(74 中央)
 (Hardware-wired control unit & Micropr-
 ammed control unit) 。 180
- 問題 79 Handshaking (parallel I/O Interface), GPIB,
 IEEE 488。(71 清大)..... 186
- 問題 80 Inverted Sile (multilist)。(74 中原·70 交大)
 191
- 問題 81 Call by value、address (location、variable、
 reference)、name。
 Call by need。(73 資策·71 公費·68 清大)
 舉實例。(Formal parameter、Actual parameter)

.....	195
問題 82 類 81。(parameter transmission 的早晚順序)。 (69 高考)	202
問題 83 同上 (call-by-value-result)。(72 高考)	204
問題 84 同上 (比較, FORTRAN, ALGOL)。 (70 高考·69 電信)	207
問題 85 同上 (實例)。(72 高考)	209
問題 86 Software, Hardware 之分別。(74 高考) (Firmware, Courseware)	212
問題 87 二進位↔八進位↔十進位↔十六進位。 (74 清大、大同·73 交大·70 清大)	213
問題 88 1's, 2's complement (加、減、表示法)。(74 高考) (9's, 10's complement)	216
問題 89 十進位→二進位→演算→八進位。(73、72 普考)	221
問題 90 Floating point number (max., min., precision, bits)。 內碼 (Internal representation)。(69 電信)	223
問題 91 Boolean Function (Sum of minterm, Product of maxterm)。 Carnoical form, Standard form, Logic Diagram。 (74 清大·70 公費) Karnaugh Map, DeMorgan Theorem.	225
問題 92 類上題。Don't care conditions.	229
問題 93 Prime implicant set, Essential prime comple- ment set, Redundant group。(65, 63 交大)	232
問題 94 將 Boolean Function 繪成 Logic Diagram。 (72 台大·71 公費)	

Truth Table, 8×8 ROM (含 Address Decoder)	235
問題 95 將 Boolean Functions 繪成 Logic Diagram。 (74 中央)	240
問題 96 簡化 Logic Diagram。(73 台大) (DeMorgan Theorem, Karnaugh Map)	241
問題 97 NAND Gate 應用 (Ex-OR, Ex-NOR Operation)。 (73 台大)	243
DeMorgan Theorem, Duality Theorem。	248
問題 98 Half Adder、Full Adder、Serial Adder。 (74 大同、中原)	253
問題 99 兼具減法功能的全加器。(70 公費)	256
問題 100 Race Condition, Edge Triggering, Master-Slave Flip Flop。(73 中央)	259
附錄一 74 年各校資訊、電子、電機、相關研究所計算機概論 試題	267
附錄二 73 年各校研究所、高普考計算機概論相關試題	281
附錄三 72 年各校研究所、高普考計算機概論相關試題	295
附錄四 71 年各校研究所、高普考、公費留學計算機概論相關 試題	307
附錄五 70 年各校、研究所、高考計算機概論相關試題	339
附錄六 69 年各校、研究所、高考、特考計算機概論相關試題	371
附錄七 65、68 年各校研究所試題	377
附錄八 參考書目	381
索引	

《 問題 1 》

何謂：(a) **Top-down Design** ?

(程式設計時由上而下的設計方法)

(b) **Bottom-up Design** ?

((a) 73 年中央計研)

【 解 】

- (a) 從事結構化程式設計時，設計者若將整個問題的解 (The Whole Program) 依其邏輯性分解 (Functional Decomposition) 成數個次解 (Logical Subprograms)。每個次解又可往下細分為數個更小單元。依此類推，直到分解成程式語言很容易編寫的最小單元為止。如此，程式由最高層次逐次往較低層次撰寫的方式，稱為 *Top-down Design*。
- (b) 相反的，若設計者首先針對整個問題中，最容易以程式語言編寫的諸部分先編寫起來，然後逐級往上將各部分組合成完整的大程式。這種方式，稱為 *Bottom-up Design*。

《 問題 2 》

何以結構化程式設計採用 **Top-down Design** ? 有何優點 ?

(73 年中央計研)

【 解 】

Top-down Design 是代表結構化程式設計的最有力名詞，強調讀、寫有效的程式應由上而下著手。整個程式以 Main Routine 為首，接著下分為 Intermediate Routines，然後是 Minor Routines，…，逐次類推。程式架構有組織、有紀律、易於標準化，故廣為結構化程式設計所採用。

優點：

1. 由於整個程式的邏輯性是由上而下的，所以當欲了解整個程式邏輯時，不需跳讀於整個程式報表之間。維護容易，並可降低開發費用。
2. 減輕設計者的負擔。由於 Top-down Design 使得程式以 Modules 方式上下連接，設計者只要對各 Module（不管是 Main, Intermediate 或 Minor Routines）考慮設計便可。不需記憶整個程式的各部份細節。
3. 經 Top-down Design 完成的程式具有結構化圖形(Diagram)的特性。因此其本身便是一項重要的 External Documentation。



補充：

通常依據結構化原則設計的程式，其指令數量較非結構化者多。造成所需記憶體空間較大、執行時間較長之缺陷。然而，近年來計算機系統的執行速度越來越快，且記憶體越來越便宜。所以上述缺點並不重要。

《 問題 3 》

試述結構化程式 (**Structured Program**) 構成要件及優點。

(71 公費留考, 72 普考)

【解】 構成要件：

1. 結構化程式設計採用 Top-down Design (見問題 1, 2)。
2. 結構化程式設計將其程式模組化 (*Modularize* , 或稱單元化) , 使成各自獨立而可分解或結合的單元。
3. 結構化程式設計儘可能少用或不用 GO TO 指令 (*GO TO-less Programming*) 。

4. 結構化程式設計以下列三大邏輯結構為基礎

- (a) **Sequence**
- (b) **Selection** (含 Case)
- (c) **Iteration** (或 **Repetition**)

優點：

- 1. 程式易於閱讀、瞭解與維護。
- 2. 使冗長、複雜的問題得以簡化。
- 3. 程式由互相獨立的許多功能單元構成，使得一個大程式可由數人分工撰寫。降低程式開發費用，生產效率提高。
- 4. 每一功能單元 (Module) 可單獨測試 (Testing) 與除錯 (Debugging)，亦即簡化測試與除錯。



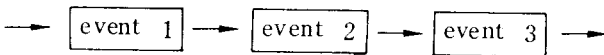
《問題 4》

結構化程式設計有那些基本結構 (**Basic Structures**)？試圖示之。
(73、74 年中央計研，72 年交大管研)

【解】

1. 順序結構 (**SEQUENCE**)：

由上而下逐一執行指令



2. 反覆執行 (**ITERATION** 或 **REPETITION**)

e.g. **DO-WHILE**, **FOR NEXT**, **REPEAT UNTIL**,
PERFORM UNTIL, **DO-UNTIL**.

