Visual Basic

CHAPTER 2

結構化程式設計與工程應用

(Programming vs. Applications)

張 榮 興 博士

豐映科技股份有限公司

E-mail: chang.ronhsin@msa.hinet.net

http://www.resi.com.tw/vb.htm

歷史上有三次震撼人心的大波動：第一波是「農業革命」、第二波是「工業革命」、第三波就是「資訊革命」

在科幻小說、影片及卡通中，時常將電子計算機（電腦）塑造成一種具有超人類智慧的「機器」，它們能流利的與人類溝通交談，能瞬間提供任何所需要的資訊、影像，更能獨立進行思考判斷、解決人類難以解決的複雜問題。這是人類的夢想，但也許是一個在我們有生之年就可以逐步看到它實現的夢想。雖然，到目前為止，電腦離具有高度人工智慧，仍然有相當遠的距離，但是它的高速、精確、可信賴的特性，卻給現代社會帶來相當大的衝擊。尤其近年來網際網路的急速發展，已經完全顛覆傳統人際溝通與資訊取得的管道。電子郵件、資訊共享、網路分工計算已成為主流趨勢，這種改變對工業及社會都將造成深遠的變革。

著名社會學家艾文托佛勒 (Alvin Toffler) 認為歷史上有三次震撼人心的大波動：第一波是「農業革命」、第二波是「工業革命」、第三波就是「資訊革命」。根據統計，已開發國家就業人口中約有百分之六十以上的人從事資訊相關的工作，亦即從事於「創造、處理或使用資訊來做事」的工作，其餘百分之三十從事於製造業，百分之十從事農業生產，可見資訊革命對現代社會影響是相當深遠的。最近幾年來，產業發展正面臨極重大的變革；傳統工業將快速凋零，資訊產業、電子工業及服務業快速崛起，電腦及軟體應用也就愈顯得重要。

第一節　工程應用

Visual Basic

現代社會中，電腦的應用正愈來愈廣泛，舉凡通訊、文書處理、即時資訊管理、輔助教學、農漁牧工商業的資料管理分析、醫療診斷、遊樂、運動各方面都可以看到電腦的蹤跡與貢獻。

由於電子計算機系統及微電腦控制設備的急速發展，亦使工業界受到相當大的衝擊，將電腦設備大量應用於工程分析、模擬、設計及程序控制已成為工程師的基本素養。利用電腦高速、精確、可信賴的特性，提高生產效率及產品品質，節省能源、原料及人力，降低成本，提高競爭力，已成為產業界努力的目標與基本策略。

要成為現代化的工程師，電腦應用的素養，尤其是軟體開發、設計及使用的能力是絕不可或缺的。本書的目的即在於提供電腦在工程上應用的最基本知識及工具，使讀者快速的具備使用電腦及設計程式解決問題的能力。本書編寫方式是希望以提出問題，藉著問題的解決方案，以系統化的方式讓讀者逐一了解各種數值方法極程式設計方法，並且能馬上利用書中所提供的完整程式，以極短的時間實際地去執行、去體驗

這種數值方法在解決工程問題上的用法。進而了解程式的修改、用法及特性，以提高學習效率。

作者一直認爲學習軟體應用應該就像學習駕駛汽車一般，除非必要，否則只要學會如何在一定的交通規則下，妥善的使用及駕駛汽車即可，而不需要爲了學習駕駛汽車，而先去學會如何打造一部汽車。同理，學習電腦應用軟體的方法，絕非讓自己花費相當長的時間焦頭爛額的坐在電腦前，爲設計一個基本程式而傷腦筋，而是應該了解與善用已有的軟體作適當修改，以相同或更短的時間，很快地精通各種數值方法程式的用法，才能收事半功倍之效。例如，我們在利用 Microsoft Office 時，根本不需要了解程式是如何寫出來的，更重要的是學會如何了解它的用法與特性，善用它爲我們解決問題、提供協助。

本書曾用於台大化工系做爲授課教材，是目前工程界計算機應用相關書籍中首度引用結構化程式設計 (Top-Down Design) 觀念的書籍，將數值計算方法以四種基本結構 (循序結構、IF-THEN-ELSE 條件結構、WHILE 重複結構及 REPEAT 重複結構) 編寫成邏輯清晰的程式設計，讓研習者能快速的瞭解程式設計的要領，進而在製程計算、電腦輔助設計及自動化控制計算方面，能得心應手的利用各種數值方法。

近年來 Microsoft Basic 已成爲最廣泛被使用的軟體語言之一，本書編寫時，爲了提高使用者的學習效率，決定將所有程式利用 Microsoft Visual Basic 編寫，並加入許多 Microsoft Excel 的應用。希望讓讀者能更有效率的學習及使用本書。以下針對各種工程應用，作簡要說明。

第二節　電腦輔助程序設計

Visual Basic

工程科學是一門隨著時代需要而演進的科學，因此，由於近代科學的發達，使得工程科學的涵蓋面漸趨廣泛，也使得工程師們需要接受更深更廣的訓練與經驗。近年來，由於能源及環保相關問題的產生，更加重了工程師們的責任，必須發展設計更好的工藝程序及更有效的操作方法。所幸由於電子計算機的高速發展與廣泛應用，使得工程師們繁複的計算與設計，能利用計算機作有效的處理而獲得圓滿的結果。

基本上，電腦輔助程序設計可分成兩大類，一爲複雜完整的程序模擬，通常需要強有力的計算工具及龐大的記憶體，其次爲經常性的質能平衡計算、設備設計及一般工程計算，如管線壓力降計算等，通常爲利用「程式型計算器」或「個人電腦」即可

完成。

完整的程序模擬程式能同時執行熱量及質量平衡計算，作初步的裝置設計，製作準確且詳細的流程圖 (Flow-Sheets)。但在計畫進行前所作的初步計算，利用完整的程序模擬並不恰當，此時，以採用簡單的質量平衡程式，快捷且經濟的製作初步的流程圖爲較可行的策略。

除了流程的質能平衡計算以外，工程師們也常需作經常性的簡單設計問題計算，或最適條件計算，例如選擇控制閥大小，計算流體在管線中的壓力降，或計算熱交換器的 LMTD 等。本書將在各章節的例題中提出許多這類的計算程式。

第三節　生產程序自動化

Visual Basic

就企業眼光而言，電腦除了在管理資訊自動化以外，在工業界最重要的應用可能是生產程序的自動化。近年來，由於世界性經濟持續不景氣，提高生產力，維持競爭優勢，已成爲企業界最關注的問題。尤其處在目前高價能源和高漲工資的年代，要想維持產品在市場上的高競爭力，就必須設法全力降低生產成本。推行自動化正是達成降低成本的必要途徑。

生產程序自動化的四個主要目的：

1. 提高產能及生產力

由於設備自動化，生產過程中可使用各級電腦作即時的靜態 (Static) 及動態 (Dynamic) 線上分析及控制。縮短製程時間，減少人爲操作錯誤，使不合格產品或廢料減至最低；使生產製程更爲順利，降低設備故障造成意外停機損失之機會。因此，可提高製程的產能及生產力。

2. 穩定及提高產品品質

在製造過程中及製造後，可利用電腦化設備自動分析、鑑定產品的各種化學成分及物理性質，適時調整製程之操作，使產品品質更爲標準化及高級化，降低剔退率，減少成本，並提高產品之市場形象，維持高優勢競爭力。

3. 降低成本、精簡人員

利用電腦系統管制原料的利用，可降低原料及半成品的安全庫存量，提高製程

的運轉效率，節約能源，並且能機動因應市場需要，安排較高利潤的產品組合，製造更多的利潤。此外，由於採用自動化作業，可取代相當數量的作業人力，達到精簡人員的目的，紓解工資高漲對企業經營所增加的負擔。

4. 確保人員及設備安全

利用電腦化自動化設備對現場作嚴密的監督及管制，減少人員操作錯誤的機會，因此，可改善員工工作環境的安全性，並增強設備運轉的安全性。

生產自動化的實現，最主要需要兩方面的密切配合，一是硬體結構，包括各級電腦、檢測設備及控制系統，其次是適當的軟體系統，用於驅動硬體結構，以達成使命。其中硬體方面由於近年來電腦設備的高速發展，已經不成問題，倒是軟體系統的規劃、設計，卻仍需工程界不斷的投入人力及物力。

第四節　程序自動化電腦系統

Visual Basic

人力成本的提高、產品品質穩定性的要求、及系統安全性的全面監督要求，使得工業建設逐步走上全面性的自動化。邏輯判斷、重複性工作、監督控制與系統調節，都借助於自動化電腦系統進行控制管理。工程界所使用的自動化電腦系統主要是由主電腦、程序控制電腦及大量的分散式控制系統所組成：

1. 主電腦系統

生產程序所使用的主電腦系統，通常是由一至數部大型電腦所組成。利用分布全工廠的終端機 (Terminal) 經數據機 (Modem) 傳遞資訊，來達成全廠的產銷系統、成本、會計、財務、人事、行政、設備維護管理、物料追蹤等的整體資訊管理系統。

2. 程序控制電腦系統

由各主要工廠的迷你型程序控制電腦或工業級電腦所組成。主要功能是依據整體資訊管理系統由數據機所傳輸進來的生產訂單 (或生產計劃)，安排及控制生產設備，生產所需產品，並將生產結果傳輸回整體資訊管理電腦系統，以完成資訊管理網路。其主要功能包括：

(1) 現場控制數據收集。

(2) 程序控制計算。
(3) 生產過程原料追蹤。
(4) 整場監督。
(5) 編印生產報告。
(6) 提供現場工程師操作指引。
(7) 數據傳輸。

3. 分散式控制系統：

由分散各工廠的微電腦控制設備組成的分散式控制系統，包含以微處理機 (microprocessor) 爲基礎的電子儀器控制設備，操作人員藉此直接控制現場設備的運轉，監視各種製程的變化。

要有效率地使用各級電腦系統，推動工業界的自動化，工程師們應對電腦運用作較深入的了解，培養設計及應用軟體的能力，以達成工廠全面自動化，降低生產成本及提高產品品質的目的。

第五節　電腦輔助流程計算

Visual Basic

程序設計流程計算程式，基本上可分爲穩定態程序模擬系統 (Steady State Process Simulator)，及動態程序 (Dynamic Process) 模擬系統兩大類，近年來已逐漸發展成爲一種瞭解與設計化工程序的有力工具。流程計算程式 (或稱爲化工程序設計程式) 的典型結構，如圖 2.1，主要包含：

1. 主執行程式

用於控制流程計算的執行及引導各副程式間的資訊聯絡。

2. 裝置性能副程式集

根據主程式所輸入數據資料，模擬裝置的性能，計算其輸出結果。

3. 物理性質數據資料庫

流程計算程式的完備性及其應用範圍大部分決定於物理性質資料庫是否完備。通常流程計算程式均含有數百種化合物的物理性質資料庫。

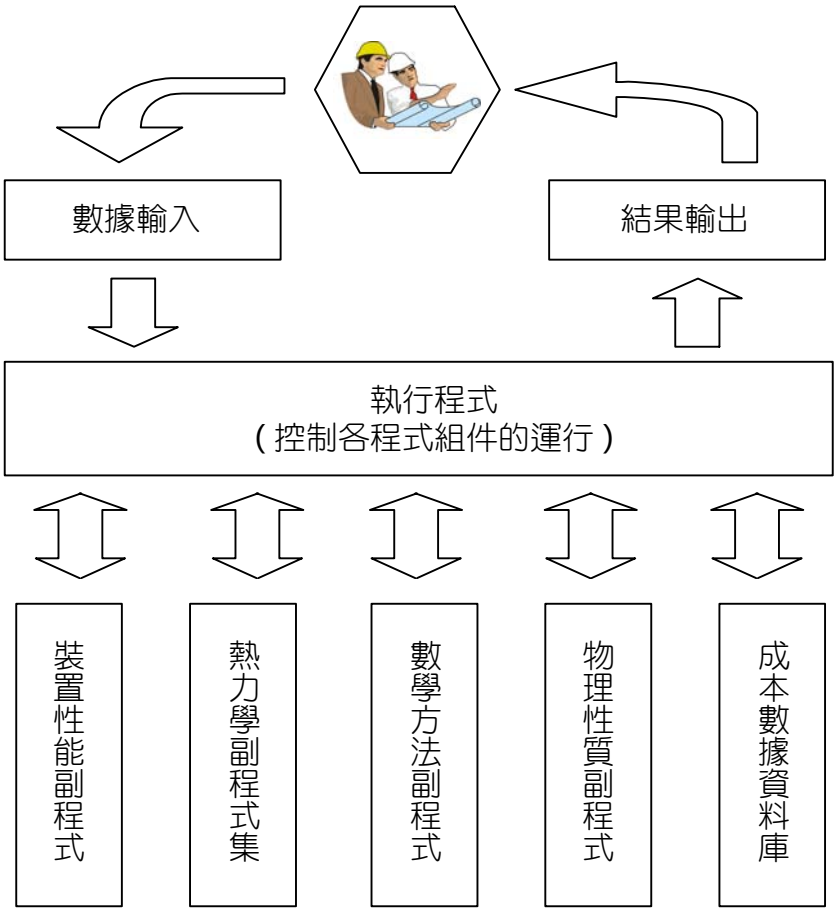


圖 2.1　典型的化工程序模擬程式系統

4. 熱力學副程式

用於執行蒸氣 — 液體間的相平衡計算，及流體流束內焓的計算等熱力學運算。

5. 成本評估副程式及資料庫

用於估計裝置的投資及操作成本。通常，完整的流程計算系統均可容許設計者考慮各種不同的製程方法，再利用成本評估程式作經濟效益比較，選擇最佳的製程方法。有些系統可能還含有最適化評估程式。

6. 數學方法副程式

用於提供及執行大部分的數學運算。包括物理性質的內插、質能平衡所得線性或非線性聯立方程式的求解、數據之微分、積分及迴歸分析、微分方程模式之求解等。這也是本書所介紹內容的主要範圍，流程計算程式的其他部分限於篇幅不作介紹。

第六節　程式規劃及設計

Visual Basic

以上各節簡略地說明了電子計算機在工業界的應用範疇，及泛用流程設計及模擬系統的概況。如圖 2.1 所示，這類程式系統的設計，通常均先作適當規劃，將整個系統分割成數個主要組件 (module)，再針對各組件所需擔任功能，細分爲更小的模組，進行程式設計及測試。

「程式設計是一種團隊精神的發揮」，有許多程式設計者對自己的程式作保護，不許使用者更改及瞭解，這種觀念並不是最正確的。其實，縱使一個程式是正確的，如果不易被人看懂，就不能算是一個好程式。

程式設計就像踢足球一樣，整個團隊間需有良好的合作及配合，才能創造出進步且完美的軟體。這個觀念對工程應用程式更是重要。程式設計過程可分成五個步驟：

1. 分析及確定問題

(1) 所要處理、解決的問題是什麼？
(2) 處理此問題的目的爲何？值不值得做？
(3) 是否可以利用電子計算機來處理？
(4) 需要何種數學方法的配合？
(5) 人機介面要處理到哪種程度？

2. 尋找及設計解決方法

這是程式設計過程中最重要的步驟，有妥善的程式結構規劃，才能使程式設計工作容易進行，使設計出的程式性能可靠又易於維護。

在這個步驟中，最重要的就是要找出最適當的解決方法及數學工具，然後利用「結構化程式設計」方法，將問題切割成幾個模組，使每一個模組的工作目標

簡單明瞭，以利設計出簡潔有彈性的程式，並利用「流程圖」釐清程式規劃的邏輯步驟，建立明確的程式架構 (結構化程式設計讀者可參閱專書或下一節說明)。

3. 編寫程式

選擇適當的程式語言，例如 BASIC、VISUAL BASIC、FORTRAN、PL/1、PASCAL、ADA、C、EXCEL 或 COBOL 等。然後，再依照已規劃好的程式邏輯及流程圖編寫程式。

4. 上機測試

修改語法錯誤。利用已知特性的模擬數據，作初步的實際執行，去除「邏輯上的錯誤」。

5. 撰寫說明文件

程式說明文件應包括設計目的、解決方法、流程圖、程式符號說明、程式列印、測試數據及結果。程式說明文件除了提供使用說明外，更應讓必須修改程式的使用者，能很快地看懂原有程式，而加以修正。

本書所有程式設計，均係依照上述步驟進行。近年來，由於個人電腦的風行，Visual Basic 語言已成爲最多人了解的語言之一，此外，Visual Basic 語言程式的可讀性高，易於使讀者馬上瞭解及修改程式，因此，本書程式均採用 Visual Basic 語言撰寫。

本書部分應用也介紹利用 Microsoft Excel 的使用方法。習慣使用 FORTRAN、C 語言或其他高階語言的讀者，由書中所附流程圖及程式結構，亦可輕易地將本書程式改寫成自己所需的程式。

第七節　結構化程式設計

Visual Basic

Visual Basic 基本上並非結構化的程式語言，但只要在程式設計時作適當處理，即可使程式結構變成簡潔、有條理、易於維護。在本書中所使用的流程圖稱爲結構化流程圖 (Top-Down Design)，基本上是由幾種含有基本指令的結構方塊堆疊而成，用以說明各個指令的執行順序。在撰寫程式時，可依據結構化流程圖很輕易的編寫程式碼。

傳統程式設計通常採用大量 GOTO 指令，非常容易造成邏輯混淆的困擾；因此，在本書所有程式實例中，爲了維護程式邏輯的簡單明確，我們完全捨棄 GOTO 指令，改成完全遵照以下所說明的五種基本控制結構來撰寫：循序控制結構、條件式選擇結構 (If-Then-Else)、多重選擇結構 (Select Case)、重複迴路結構 (For-Next) 及條件式重複結構 (Do-Loop While)。以下簡要說明各種基本控制結構及其基本語法。

循序控制結構

循序控制結構是最基本的程式組構元素，是由指令敘述區塊依序堆疊而成，其結構方塊圖如下圖所示。

循序控制結構的作用是先執行完指令敘述 1 後，接著執行指令敘述 2。其中指令敘述 1 及指令敘述 2 可以是任何基本敘述或指令。

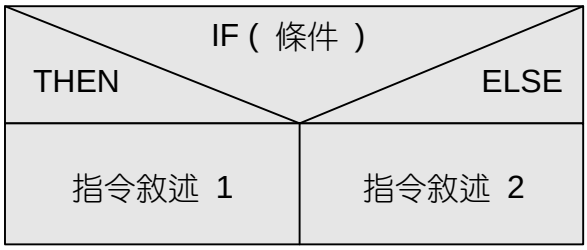
循序控制結構典型的例子如下：

範例 1　程式 201

```
Private Sub Form_Click()                    '按 Form 開始執行
Alpha = Val(InputBox("Alpha = "))           '利用 InputBox 輸入 Alpha 字串，再取其數值
Beta = Val(InputBox("Beta = "))          '利用 InputBox 輸入 Beta 字串，再取其數值
Debug.Print "Alpha = "; Alpha             '在即時運算視窗顯示
Debug.Print "Beta = "; Beta               '在即時運算視窗顯示
AB = Alpha + Beta                           'AB 等於 Alpha 加上 Beta
AD = Alpha – Beta                           'AD 等於 Alpha 減 Beta
Print "Alpha + Beta    = ";                 '列印; "…"
Print Format(AB, "    0.0000E+00")          '以 0; 0#; 的格式印出 AB 值
Print "Alpha - Beta = ";                    '列印; "…"
Print Format(AD, "    0.0000E+00");         '以 0; 0#; 的格式印出 AD 值
End Sub
```

條件式選擇結構 (IF-THEN-ELSE)

條件式選擇結構 (IF- THEN- ELSE) 的結構方塊圖如下，這種語法的作用是「如果條件成立，則執行指令敘述 1，否則就執行指令敘述 2。」



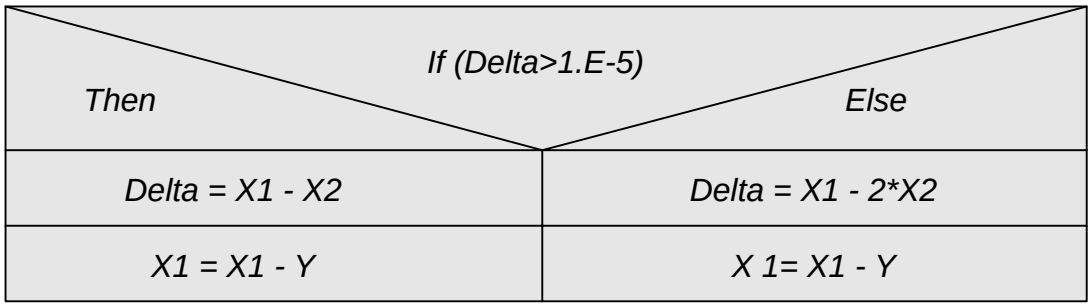

```
IF (Condition) THEN
      Statement Block 1
ELSE
      Statement Block 2
END IF
```

其中指令敘述 1 及指令敘述 2 可以是任何單一或多個基本敘述或指令。

範例 2 程式 202

```
Private Sub Form_Click()
X1 = Val(InputBox("X1 = "))          'X1 等於數入字串取數值
Y = Val(InputBox("Y ="))             'Y 等於輸入字串取數值
Delta = Abs((X1 - Y) / X1)           'Delta 等於 X1 與 Y 之相對差之絕對值
If (Delta >1.E-5) Then               '若 Delta 大於 1.E-5 則
      Delta = X1 - X2                'Delta 等於 X1 減去 X2
      X1 = X1 + Y                    'X1 等於 X1 加 Y
Else                                 '否則
      Delta = (X1 - 2 * X2)          'Delta 等於 X1 減去 2 倍 X2
      X1 = X1 – Y                    'X1 等於 X1 減去 Y
End If                               '結束 IF
Print "New X1 = ";
Print X1
End Sub
```

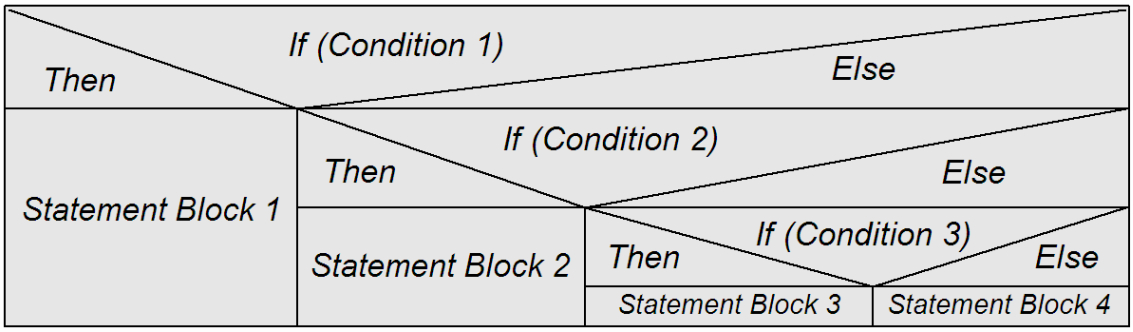
對應的 Top-Down Design 方塊圖爲



利用條件式選擇結構 (IF-THEN-ELSE)，可以進一步組合成多重條件式選擇結構。其基本語法如下：

```
IF (Condition 1) THEN
        (Statement Block 1)
ELSEIF (Condition 2) THEN
        (Statement Block 2)
ELSEIF (Condition 3) THEN
        (Statement Block 3)
ELSE
        (Statement Block 4)
ENDIF
```

對應的 Top-Down Design 方塊圖爲



多重選擇結構 (SELECT CASE)

多重選擇結構 (SELECT CASE) 的結構圖如下圖所示，其作用是「依待測敘述 (Statement A) 的值，決定應執行那一種狀態下的指令敘述。」

SELECT CASE (Statment A)				
Case a	Case b	Case c	.....	Case Else
指令敘述 1	指令敘述 2	指令敘述 3	.....	指令敘述 n

其中 a、b、c、…爲選擇狀態值；當待測敘述 Statement A 的值等於 a 時，執行指令敘述 1；當 Statement A 的值等於 b 時，執行指令敘述 2；……；若 Statement A 的值與各狀態均不同，則執行指令敘述 n。指令敘述 1、指令敘述 2、…、指令敘述 n 可爲任何一個或多個基本敘述或指令。

基本語法：

```
Select Case TestExpression
   Case value1, value2,...
        [statements]
   Case value3 to value4
        [statements]
   Case Else
        [statements]
End Select
```

應用實例如下所示。

範例 3 程式 203

```
Private Sub Form_Click()
Alpha = Val(InputBox("Alpha = "))          '利用 InputBox 輸入 Alpha 字串，再取其數值
Beta = Val(InputBox("Beta = "))          '利用 InputBox 輸入 Beta 字串，再取其數值
AB = Alpha + Beta                          'AB 等於 Alpha 加上 Beta
AB = Int(AB / 10)                          'AB 等於 AB/10 再取整數
Select Case AB                             '以 AB 的值作選擇
Case 1                                     '若 AB 的值等於 1
      AB = AB + 1                          'AB 等於 AB 加上 1
Case 2, 3                                  '若 AB 的值等於 2 或 3
      AB = AB + 2                          'AB 等於 AB 加上 2
```

```
Case 5                                  '若 AB 的值等於 5
      AB = AB / 2                       'AB 等於 AB 除以 2
Case Else                               '若 AB 等於其他值
      AB = 0                            'AB 等於 0
End Select                              '結束選擇作業
Print " AB = ";
Print Format(AB, " 0.0000E+00")
End Sub
```

在此例中，程式需先計算 AB 值，當 AB=1 時，執行 AB=A+1。
當 AB=2 或 3 時，執行 AB=AB+2。
當 AB=5 時，執行 AB=AB/2。
當 AB 等於其他值時，執行 AB=0。

重複迴路結構 (FOR – NEXT)

重複迴路結構的結構化流程圖如下圖所示，其作用是「如果滿足所定條件 For Count.Variable＝Initial.Value to Final.Value step Var，則重複執行指令敘述中的指令。」FOR-NEXT 重複迴路之基本語法如下所示，其作用是針對變異數 Count.Variable，由起始值 Initial.Value 開始，每次增量 (Step) 爲 Var，其中 Var 可以爲正值或負值；直到變異數 Count.Variable 達到終點值 Final.Value 爲止，重複執行 FOR-NEXT 間的指令。要中途脫離 FOR-NEXT 重複迴路，可以利用 Exit For 指令脫離。


```
FOR Count.Variable = Initial.Value TO Final.Value (STEP Var)
      Block Statement
      (Exit For)
      Block Statement
NEXT Count.Variable
```

FOR-NEXT 重複迴路典型的實例如下，爲利用 FOR-NEXT 迴路，設計一程式計算 1 到 10 的總和。

範例 4 程式 204

計算 1 至 10 的和

```
Private Sub Form_Click()
Sum = 0                                   '將總和 Sum 歸零
For I = 1 To 10 Step 1                    '將 I 值由 1 逐次增加 1 直到變成 10
      Sum = Sum + I                       'Sum 等於 Sum 加上 I 值
Next I                                    '下一個 I 值
Print "Sum of Integer from 1 to 10 = ";
Print Sum                                 '列印 Sum
End Sub
```

FOR-NEXT 重複迴路在矩陣及重複計算上，是一非常重要的指令。當一個以上的 FOR-NEXT 迴路組合在一起，可構成巢式迴路。典型的例子如下。

範例 5 程式 205

巢式迴路

```
Private Sub Form_Click()
For I = 1 To 10                           '將 I 值由 1 逐次增加直到變成 10
      Sum = 0                             '將總和 Sum 歸零
      For J = 1 To 100                    '將 J 值由 1 逐次增加到變成 100
            Sum = Sum + I * J             'Sum 等於 Sum 加上 I*J 值
      Next J                              '下一個 J 值
      Print "I = ";
      Print I;                            '列印 I 值
      Print "      Sum = ";
      Print Sum                           '列印 Sum 值
Next I                                    '下一個 I 值
End Sub
```

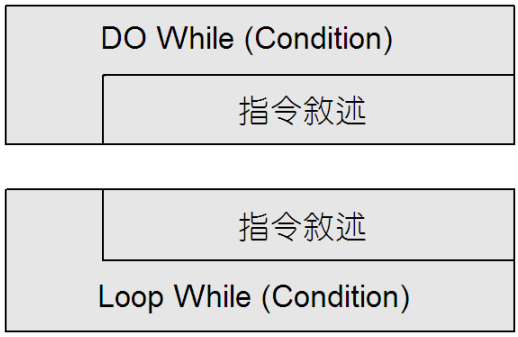
Visual Basic 語言由於可以接受長變數名稱，因此，為了提昇程式的可讀性，通常採用可對應實際意義的名稱撰寫程式。典型的實例如下：

```
intNumStudents = InputBox(prompt:="How many students?")
For intCounter = 1 To intNumStudents
      sglScore = CSng (InputBox (prompt:= "Enter score"))
      sglTotalScore = sglTotalScore + sglScore
Next intCounter
      sglAverage = sglTotalScore /intNumStudents
```

其中 int 表示整數 (integer)，sgl 表示單準實數 (single)。InputBox 為 Visual Basic 的標準輸入方塊，其中 Prompt:="How many students?" 會使對話方塊內顯示 How many students? 的提示。

條件式重複結構

條件式重複結構：若仍滿足所定條件，則繼續重複執行 Block 中的指令。其結構又可區分爲 Do-While-Loop 及 Do-Loop-While 兩種。其中 Do-While 條件式重複結構的基本語法爲：




```
DO {While (Condition)}
      Statements 1
      {Exit DO}
      Statements 2
LOOP
```

簡單實例如下：

```
Dim strPasswordTry As String
Const PASSWORD As String = "CHEER"
PasswordCount = 0
Do While strPasswordTry <> PASSWORD
    If Password Count = 0 then
      strPasswordTry = InputBox ("Enter password")
    Elseif PasswordCount >3 then
      Print "Password Input Error"
      Exit Do
    Else
      strPasswordTry = InputBox ("Password Error, Enter password")
    End If
    PasswordCount = PasswordCount +1
Loop
```

典型的實例可利用計算 1 到 10 的總和說明之。首先將總和 SUM 歸零，在 *I* <=10 條件下，讓總和 SUM 逐次累加 I 值，加總完後，讓 I 值增加一個增量 1；直到 *I* >10 不滿足重複結構之條件後，將總和 SUM 列印出。讀者可以仿照範例 4，將程式作如下之修改。

```
I=0
Sum = 0
Do While I <= 10
      Sum = Sum + I
      I = I + 1
LOOP
Print Sum
```

另一種條件式重複結構 DO-LOOP-WHILE，基本上語法與 DO-WHILE-LOOP 極為相似，作用也相同。其基本語法結構為：

```
DO
      Block 1
      {Exit DO}
      Block 2
LOOP {While (Condition)}
```

將以上計算 1 到 10 的總和實例，以 DO-LOOP-WHILE 語法改寫如下。

```
I=0
Sum = 0
Do
      Sum = Sum + I
      I = I + 1
LOOP While I <= 10
Print Sum
```

類似的條件重複結構：*若不能滿足所定條件，則繼續重複執行 Statement 中的指令*。

其結構又可以分為 Do-Until-Loop 及 Do-Loop-Until 兩種。其中 Do-Until-Loop 條件式重複結構的基本語法為：

```
Do
    [statements]
Loop Until [condition]
```

或

```
Do Until condition
    [statements]
Loop
```

其實例如：

```
Do Until EndOfFile = True
    [Read and process next line in file]
    [Set EndOfFile to True if end of file is encountered]
Loop
```

或

```
Dim intLoopCount As Integer

Do
    intLoopCount = intLoopCount + 1
Loop Until MsgBox("Loop?", vbYesNo) = vbNo
```

第八節　結構化程式設計與工具程式

Visual Basic

結構化流程圖不管程式多複雜，都是利用本章所說明之五種基本控制結構組合而成。構思程式結構時，建議先作整體上位觀念設計 (Top-Level Design)，將工作目的方塊先行畫出；再利用前述基本模組，逐漸將問題結構的細部勾勒出來。也就是先作分工的上位設計，再作細部規劃，以簡化程式設計邏輯。

編寫程式時，建議應先將要解決的問題及工具程式作結構化切割；將每一件重複性工作及獨立計算部分分割成小程式，以利程式編寫。編寫各模組程式時，建議應該在程式中加上詳細的說明及使用須知，以提昇程式之可讀性及操作性。一般而言，編寫結構化程式時，每一個程式模組通常以不超過 50 個指令爲原則，以提高程式之可讀性及維護性。以下所提供的工具程式模組，就是利用這種觀念進行設計的範例。

1. 數據輸入與存檔

```
' DATA ENTRY OR READ DATA FROM FILE
'
' PROGRAM DEVELOPED BY ENYA CHANG
' COPYRIGHT 2001      CHEER
'
Sub DataEntry(Xpos, Ypos)
      Dim Entry As String
      Dim X(100) As Single, Y(100) As Single, N As Integer
      Cls
'
'          DATA ENTRY MODE SELECTION
      Entry = InputBox("Data Entry From File <Y/N> ", "SELECT ENTRY MODE", "Y", Xpos, Ypos)
            If Entry = "Y" Then
            Call DataEntryFromFile(N, X, Y, Xpos, Ypos)
      Else
            Call DataKeyIn(N, X, Y, Xpos, Ypos)
      End If
'   CHEER 2001
End Sub


Sub DataKeyIn(N, X, Y, Xpos, Ypos)
      N = 0
      Do
      N = N + 1
      Do
            Debug.Print "X,Y OF THE #", N, " POINT";
            X(N) = Val(InputBox("Enter X value of the point", "X", , Xpos, Ypos))
            Debug.Print "    X ="; X(N);
            Y(N) = Val(InputBox("Enter Y value of the point", "Y", , Xpos, Ypos))
            Debug.Print "    Y ="; Y(N)
            YN$ = InputBox("Are the data input correct?", "YesNo", "Y", Xpos, Ypos)
      Loop While YN$ <> "Y" And YN$ <> "y"
      YN$ = InputBox("Input Next Data?", "YesNo", "Y", Xpos, Ypos)
      Loop While YN$ = "Y" Or YN$ = "y"
      Cls
      YN$ = InputBox("Save Data to File?", "YesNo", "Y", Xpos, Ypos)
      If YN$ = "Y" Or YN$ = "y" Then
      Call SaveDataToFile(N, X, Y, Xpos, Ypos)
      End If
End Sub


Sub SaveDataToFile(N, X, Y, Xpos, Ypos)
```

```
Dim FileNo As Long, FileName As String
    FileNo = FreeFile
    FileName = InputBox("Enter File Name ", "FILE NAME", "DataEntry.dat", Xpos, Ypos)
    Open FileName For Output As #FileNo
        Print " I", " X(I)", " Y(I)"
        For I = 1 To N
            Print #FileNo, X(I), Y(I)
            Print I, X(I), Y(I)
        Next I
    Close #FileNo
End Sub


Sub DataEntryFromFile(N, X, Y, Xpos, Ypos)
Dim FileNo As Long, FileName As String
FileNo = FreeFile
FileName = InputBox("Enter File Name for Data Input", "FILE NAME", "DataEntry.dat", Xpos, Ypos)

Open FileName For Input As #FileNo
    N = 0
    Print " I", " X(I)", " Y(I)"
    Do While Not EOF(FileNo)
        N = N + 1
        Input #FileNo, X(N), Y(N)
        Print N, X(N), Y(N)
    Loop
Close #FileNo
End Sub
```

2. 輸入字串轉換為數據

```
Sub DataStrIn(Xpos, Ypos)
Dim I As Long, keyin As String
Dim f(100) As Single, x(100) As Single

Do
    keyin1 = InputBox("輸入 x 值，以空白區隔。" & vbNewLine & "例：1 2 3", "x 值", keyin1)
    If Not SepStrToNumArray(keyin1, " ", 1, n, x) Then Call WrongValue(1)

    keyin2 = InputBox("輸入 f(x)，以空白區隔。" & vbNewLine & "例：10 20 30", "f(x)值", keyin2)
    If Not SepStrToNumArray(keyin2, " ", 1, m, f) Then Call WrongValue(1)

    If m <> n Then Call WrongValue(2)
Loop While m <> n

Cls
```

```
For I = 1 To n
    Print I, x(I), f(I)
Next
End Sub


Sub WrongValue(ID)
Select Case ID
Case 1
    MsgBox "所輸入的數值無效", vbExclamation, "警告"
Case 2
    MsgBox "所輸入的數值 m<>n 無效", vbExclamation, "警告"
End Select
End Sub


Function SepStrToNumArray(StrHandled, SepChr, BaseNumber, Number, RetrunData) As Boolean
Dim I As Long, J As Long, Temp As String

StrHandled = Trim(StrHandled)
Number = BaseNumber - 1

Do
    I = InStr(J + 1, StrHandled, SepChr, vbBinaryCompare)
    If I <> J + 1 Then
        Number = Number + 1
        If I <> 0 Or (I = 0 And J < Len(StrHandled)) Then
            If I <> 0 Then
                Temp = Mid(StrHandled, J + 1, I - J - 1)
            Else
                Temp = Mid(StrHandled, J + 1)
            End If
            If Not IsNumeric(Temp) Or InStr(Temp, ",") > 0 Then Exit Function
            RetrunData(Number) = Temp
        End If
    End If
    J = I
Loop Until I = 0

SepStrToNumArray = True
End Function


Sub DataKeyIn_Click()
Call DataStrIn(500, 4500)
End Sub
```

3. 計算機誤差估計

```
Sub MachineErrorCheck(Eps, U20, ReMin, RelErr)
'
'     Calculate Machine Round Off Error
'
Dim Eps, U20, ReMin, Rer, RelErr As Double
Eps = 1#
Do
      Eps = Eps / 2
      EpsP1 = Eps + 1
Loop While EpsP1 > 1
U20 = 20 * Eps
Rer = 2 * Eps + ReMin
'
'   Check for proper error tolerance
'
If RelErr < Rer Then
      RelErr = Rer
End If
End Sub
```

此程式用於計算電腦本身的機器捨去誤差 Eps，當一個數字小於 Eps 時，電腦基本上將它視爲零。

4. VB 函數

VB 函數分成內建函數及使用者自訂函數兩大類。所謂的內建函數，是 VB 程式語言系統針對特定運算目的事先編寫好的程式碼，供使用者直接呼叫使用。使用者自訂函數，就像本節所介紹的幾個副程式，是由使用者依所要執行的目的，自行規劃設計的程式碼。

VB 內建函數約有一百多個，以下依其使用功能整理列表，其使用細節，讀者可直接利用 VB 本身的線上協助 (Help) 功能查詢。

函數分類	函　　數
數值函數	Abs、Int、Fix、Sgn Exp、Log、Sqr、Rnd Atn、Cos、Sin、Tan Hex、Oct
字串函數	Chr、Asc、String Len、Left、Right、Mid Lcase、Ucase LTrim、Rtrim、Trim Str、Val、InStr、StrComp
時間函數	Now、Date、Time、Year、Month、Weekday Day、Hour、Minute、Second DateSerial、DateValue、TimeSerial、TimeValue Timer、DateAdd、DateDiff、DatePart
檔案函數	FileAttr、GetAttr、FilaDateTime、FileLen、FreeFile、CurDir、Seek、Input、Loc、LOF、EOF、Tab、Spc
資料型態	CCur、CDbl、CInt、CLng、CSng、Cstr、CVar、CVDat、VarType
資料庫函數	CreateDatabase、OpenDatabase
輸出入函數	RGB、InputBox、MsgBox、Format、LoadPicture、QBColor
選擇判斷函數	IIf、Choose、Switch、Isdate、Isempty、IsNall、IsNumeric
財務分析函數	Rate、MIRR、IRR、FV、IPmt、Nper、Pmt、PPmt、PV、SLN、DDB、SYD
系統函數	Shell、CurDir、Dir、Command、DoEvents、Environ
OLE 函數	CreateObject、GetObject
雜項函數	LBound、UBound、Partition、Error、Err、Erl

第九節　計算結果如何製成報表

Visual Basic

使用本書程式或修改本書程式以供特定目的使用時，若要列印報表，可以利用以下三種方式進行：

1. 直接執行程式，然後利用 Print Screen 按鍵複製畫面，再利用繪圖軟體剪貼所需要的部分，複製到所要撰寫的報告中。
2. 仿照前例中，副程式 **Sub SaveDataToFile (N, X, Y, Xpos, Ypos)** 的方式，開啓

輸出檔案，並將本書所附程式中的 Print 指令，更改爲 Print #FileNo, A, B, C 的形式，將結果輸出到使用者指定的檔案。再利用文書編輯程式，開啓該檔案進行編輯。

3. 將本所附程式中的 Print 指令，更改爲 Debug.Print，將結果輸出到 Visual Basic 的『即時運算視窗』中，再由使用者複製即時運算視窗內的資料，並利用文書編輯程式進行編輯。如圖 2.2 所示。

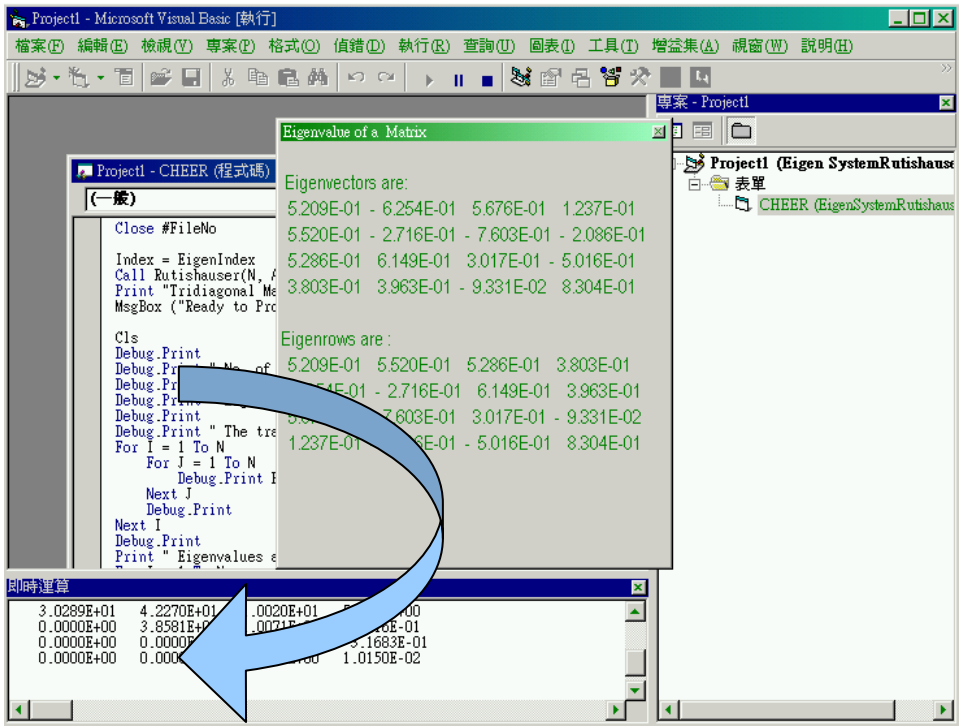


圖 2.2　利用『即時運算視窗』顯示運算資料

參考文獻

1. Merchant. Michael J.,“FORTRAN 77-Language and Style”, Wadsworth, 1981。
2. 張榮興，“電子計算機在化工上之應用”，儒林圖書公司，1986。
3. 張榮興，“流體在管線中的摩擦係數簡化計算法”，化工，No. 106, Jan, 2002。

習題

Visual Basic

1. 試撰寫一程式，列出 0 到 1000 間的所有質數，並求其總和。

2. 請在您所學習的課程中，列出三項必須利用計算機計算的問題，另找出三項可以編寫成計算機程式的問題。你暫時不需要編寫程式去解決這些問題，但請列出每一問題所需要用到的數值分析方法。

3. 在流體輸送問題計算上，常需計算壓力差、管徑、流量、流速等。但其中有一項重要參數 — 摩擦係數，工程師通常使用查圖估算。張榮興 (2002) 根據 Colebrook and White 的摩擦係數一般方程式，及 Manadili 的二項簡化式，提出一修正的簡化經驗式如下：

$$f = \frac{1}{\{-2\times\log[\dfrac{95.00/\mathrm{Re}^{0.983} - 96.82/\mathrm{Re}}{1+\dfrac{1}{6}\mathrm{Re}^{1/3}(\varepsilon/D)^{2/3}} + \dfrac{\varepsilon/D}{3.7}]\}^2}$$

其中 Re 爲雷諾數，ε/D 爲管線粗糙度。

(a) 試撰寫一副程式 Function FrictionFactor (Re, Roughness)，讓使用者呼叫時，只要輸入雷諾數 Re 及管線粗糙度 ε/D，即可計算摩擦係數 f。

(b) 試撰寫一副程式，建立如下表所示之 Re、ε/D 及 f 的關係。

雷諾數 Re	管線粗糙度 ε/D	摩擦係數 f
1,000	0.0001 0.0005 0.001 0.005 0.01	
2,000	0.0001 0.0005 0.001 0.005 0.01	
3,000 ⋮ 1,000,000	⋮	

4. Yaws, C. L. 將理想氣體的熱容量表示成溫度的多項式函數，並利用本書第七章所介紹的回歸分析法，建立各種理想氣體的參數。

$$C_p^0 = A + BT + CT^2 + DT^3$$

(a) 請編寫一副程式，可以將下表所列的熱容量參數數據建檔，檔案名爲 HeatCapa.Txt。

Index	Compound	A	$B\times10^3$	$C\times10^6$	$D\times10^9$
1	Sulfur Dioxide	5.85	15.4	–11.1	2.91
2	Nitrogen Dioxide	5.53	13.2	–7.96	1.71
3	Carbon Monoxide	6.92	–0.65	2.80	–1.14
4	Carbon Dioxide	5.14	15.4	–9.94	2.42
5	Hydrogen Chloride	7.24	–1.76	3.07	–1.00
6	Water	8.10	–0.72	3.63	–1.16
7	Nitrogen	7.07	–1.32	3.31	–1.26
8	Oxygen	6.22	2.71	–0.37	–0.22
9	Methane	5.04	9.32	8.87	–5.37
10	Ethane	2.46	36.1	–7.0	–0.46

(b) 請編寫一副程式 Sub CP (Index, A, B, C, D, T, CP)，讓使用者呼叫此副程式時，只要輸入氣體的編號 (Index) 及溫度 T，程式會自動由檔案中找出參數 A、B、C 及 D，並計算熱容量 C_p。

(c) 請擴充以上所編寫的副程式，使程式具有新增、修正及刪除資料的功能。

(d) 找出至少 50 種理想氣體的參數，並將熱容量參數數據建檔，檔案名爲 HeatCapa.Txt。請注意，你可以利用所編寫的程式建檔，也可以直接利用文書編輯器建檔。

(e) 請修改副程式 Sub CP (Index, Compound, A, B, C, D, T, C_p)，讓使用者呼叫此副程式時，只要擇一輸入氣體的編號 (Index) 或化合物名稱 (Compound) 及溫度 T，程式就會自動由檔案中找出參數 A、B、C 及 D，並計算熱容量 C_p。程式設計時，可以令 Index = 0 時，必須用化合物名稱搜尋資料。