

第三章 程式的資料類型與變數的宣告

本章將為您介紹程式語言的先頭結構。您必須於瞭解程式的先頭結構後，才能再對後續的程式元素「各個擊破」！本章要為您介紹的內容是：

●資料類型 ●變數宣告 ●函數與程序 ●變數的作用範圍與生命期

宣告



3.1 資料類型

任何一種電腦語言都會提供自己的資料類型，有了資料類型，電腦才能正確的資訊進行處理。以下就分二節來說明 Visual LISP 與 VBA 的資料類型。

3-1-1 Visual LISP 的資料類型

Visual LISP 將提供整數、實數、符號、字串、串列、選擇組、圖素名稱、內建函數、VLA 物件與檔案描述子等十種資料類型。表 3-1 就列出了一些 Visual LISP 可支援的資料類型：

表 3-1 Visual LISP 的資料類型

資料類型	範例
整數	12345

實數	1.2 , -0.32 , 2.2E-13
符號	() , ' ,
字串	"abc" , "123.5"
串列	(1 2 3 4)
選擇組	<Selection set: 2>
圖素名稱	<圖素名: 40031d58>
內建函數	#<USUBR @021ca140 XYSQUARE>
ActiveX 類型/VLA 物件	#<VLA-OBJECT IAcadModelSpace 020bcf64>
檔案描述子	#<file "d:\sample\ccen.lsp">

詳細說明：

● 原子 (Atom)

舉凡符號序列、整數、實數、符號與字串等類型資料均可稱為原子。

● 整數 (Integer)

整數由數位組成，不包含小數點。Visual LISP 的整數是 32 位元帶符號的數值，有效值範圍是 +2,147,483,647~-2,147,483,648。但在 Visual LISP 和 AutoCAD 之間的整數傳輸則被限制為 16 位元。例如，getint 函數只接受 16 位元的整數值，即 +32767~-32768。當您在 Visual LISP 運算式中直接使用整數時，該值就被稱為常數。像數字 8、-58 與 20,756 這樣的數值，都是有效的 Visual LISP 整數。

● 實數 (Real)

實數就是一含有小數點的數值，是以雙精度浮點格式來儲存的。它可以提供 14 位以上的位數（精度）。但 Visual LISP 並不顯示所有有效的位數。實數還可以使用科學記數法來表示，即用 e 或 E 與其後面的指數來表示資料。例如，0.00038 與 3.8e-4 是一樣的。

● 符號 (Symbol)

AutoLISP/VLISP 均透過符號來引用資料。符號並不區分大小寫，可以由字母、數字與標註符號的任何序列來組成，但表 3-2 所示的字元除外：

表 3-2 AutoLISP/VLISP 的符號定義

符號	名稱	作用
(左括弧	表示定義的開始
)	右括弧	表示定義的結束
.	句號	點對(Dot Pair)
'	單引號	QUOTE 函數的縮寫
"	雙引號	字串函數定界符號
;	分號	註解

● 字串 (String)

字串就是在雙引號中的一組字元。字串的最大長度為 132 個字元，如果超出 132 個字元，超出的字元將被截去。字串內若無字元 ("")，則表示長度為 0。反斜線 (\) 隨後跟隨著特定的字元將可以代表控制字元。表 3-3 將列出目前 AutoLISP/VLISP 可以識識的控制字元：

表 3-3 AutoLISP/VLISP 的控制字元

控制字元	意義	控制字元	意義
\\	即代表 "\" 字元	\"	即代表 " 字元
\e	跳出字元(Escape)	\n	換行(Newline)
\r	即表示 <Enter>	\t	跳位字元(Tab)
\nnn	八進位碼為 nnn 的字元		

● 串列 (List)

串列就是一種於 AutoLISP/VLISP 中的一種資料陣列結構。這部分我們將在本篇第七章中進行討論。

● 選擇組 (Selection Set)

選擇組是包含一個或多個物件（圖素）的組。使用者可以透過 AutoLISP/VLISP 指令來交互地將物件添加到選擇組中，或從選擇組中刪除物件。以下，我們就以 ssget 函數來將選擇的物件加入選擇組 ss 中（請直接於控制台中執行）：

```
_$ (setq ss (ssget))
<Selection set: 4>
```

● 圖素名稱 (Entity Name)

圖素名稱就是圖形在其圖形資料庫中所對應的數字標籤。它是一個用以指向 AutoCAD 圖素物件的指標。經由這個指標，就可以搜尋到資料庫記錄以及其向量。AutoLISP/VLISP 都可以經由呼叫此標籤來存取並處理該圖形物件。下示範例就是一個使用 entlast 函數來擷取新畫圖形物件圖素名稱的語法：

```
_$ (setq se(entlast))  
<圖素名：27f0540>
```

● 內建函數 (Subrs)

Visual LISP 也提供大量的內建函數。例如，在主控台中輸入下示函數，將得到：

```
_$ abs  
#<SUBR @01e1ee10 ABS>
```

表 3-4 將列出 Visual LISP 可支援的內建函數：

表 3-4 Visual LISP 可支援的內建函數

函數名稱	範例	說明
應用程式處理函數	(vl-vbaload "firstvba.dvb")	載入 Visual Basic 專案
數學函數	(abs -2)	傳回參數的絕對值
and 條件函數	(and 0 1)	傳回運算式的邏輯運算結果
錯誤處理函數	(*error* "error operater")	使用者定義的錯誤處理函數
函數處理函數	(defun first () princ())	定義一個函數
串列操作函數	(car vlist)	傳回串列 vlist 的第一個元素
字串處理函數	(strlen "firstvba.dvb")	傳回代表字串長度
符號處理函數	(type "dvb")	傳回指定專案的類型

此外，使用者還能根據自己的需要來添加內建函數，如表 3-4 中「函數處理函數」的範例就定義了一個函數(first)。

● ActiveX 資料類型

圖形中的物件可以用圖素名稱來表示，以可以被表示成一 VLISP ActiveX (VLA 物件)，這是 Visual LISP 的一種資料類型。在使用 ActiveX 函數時，您必須引用 VLA 物件。如 2-3 節中 Visual LISP 例子裡的第 (9) 條程式：

```
(setq mSpace (vla-get-ModelSpace acadDocument))
```

就是欲取得一個 VLA 物件 mSpace 來指向模型空間，以配合隨後使用的 ActiveX 函數 vla-addcircle (該範例的第 (13) 條程式)。

在 Visual LISP 中，凡是對 VLA 物件進行操作的函數一般都以 vla-開頭，這些函數運算子的資料類型均屬 ActiveX 的資料類型。我們將在第十一章中再做詳細介紹。

● 檔案描述子

檔案描述子就是經由 Visual LISP open 函數來打開指定檔案的一個指標。open 函數將傳返回一文字標籤，而其他 Visual LISP 函數就可以依此檔案描述子來讀寫該檔案。例如，我們希望以唯讀方式來打開檔案 d:\sample\oilcup.txt：

```
_$ (setq f (open "d:\\sample\\oilcup.txt" "r"))
```

那麼 open 函數將傳回該檔案描述子：

```
#<file " d:\\sample\\oilcup.txt ">
```

注意：在 AutoLISP/VLISP 裡，爲了避免系統誤認，所以規定以控制字元 “\\” 來表示 “\”。

3-1-2 VBA 的資料類型

在 VBA 方面，Visual Basic 提供了諸如 Boolean、Byte、Currency、Date、Decimal、Double、Integer、Long、Object、Single、String、使用者定義、Variant.. 等資料類型。表 3-5 將列出 VBA 可支援的資料類型：

表 3-5 VBA 可支援的資料類型

資料類型	範例
Boolean	True , False
Byte	25
Currency	12.2354
Date	#January 1
Decimal	2.35
Double	234563.5 , 1.79769313486232E308
Integer	125
Long	5342345
Object	用 Set 語法來宣告 Object 的變數值
Single	4325.5 , 1.401298E-45
String	"teststring"
使用者定義	myType , 詳見後文
Variant	未被明確宣告的資料類型

詳細說明：

● **Boolean**（布林型）

Boolean 資料類型的儲存長度 16 位元（即 2 個位元組），值為 True 或 False。如果將之轉換為整數則為 1 和 0。

● **Byte**（位元組型）

Byte 資料類型的儲存長度為 8 位元（即 1 個位元組），範圍在 0~255 之間的數字。

● **Currency**（貨幣型）

Currency 資料類型的儲存長度為 64 位元（即 8 個位元組），儲存類型為整數除以 10,000 後所得到的一個常數。其小數點左邊可有 15 位數字，右邊可有 4 位數字。Currency 資料類型所表示的數值範圍可以從 -922,337,203,685,477.5808~922,337,203,685,477.5807。

● **Date**（日期型）

Date 資料類型的儲存長度為 64 位元（即 8 個位元組），儲存類型為浮點數，其表示的日期範圍可從 100 年 1 月 1 日~9999 年 12 月 31 日，時間

可以從 0:00:00~23:59:59。

● Decimal (十進位型)

Decimal 資料類型的儲存長度為 112 位元（14 個位元組），將儲存無符號的整數型式，並除以一個 10 的冪數。沒有小數點時的最大值為：

+/-79,228,162,514,264,337,593,543,950,335 ,

而當小數點右邊有 28 位數時爲：

$\pm 7.9228162514264337593543950335$;

最小的非零值爲：

[illegible]

● Double (雙精度浮點型)

Double 資料類型的儲存長度為 64 位元（8 個位元組）浮點數值的型式，有效的負值表示範圍為：-1.79769313486232E308～-4.94065645841247E-324，而正值範圍為：4.94065645841247E-324～1.79769313486232E308。

● Integer (整數型)

Integer 資料類型將用來儲存 16 位元（2 個位元組）的數值形式，其有效數值範圍將從 -32,768~32,767。

● Long (長型)

Long 資料類型的儲存長度為 32 位元(4 個位元組)有符號的數值型式，其有效數值範圍將從 -2,147,483,648~2,147,483,647。

● Object（物件型）

Object 資料類型的儲存長度為 32 位元（4 個位元組）的位址型式，它將透過 Set 語法來使用。

● Single（單精確度浮點型）

Single 資料類型的儲存長度為 32 位元（4 個位元組）的浮點數型式。此值的負值有效範圍是：-3.402823E38~-1.401298E-45，而正值有效範圍是從 1.401298E-45~3.402823E38。Single 類型的宣告字元為 “!”。

● String（字串型）

String 的字元碼範圍是 0 到 255。其中字串有兩種：變長與定長的字串。變長字串可包含 (2^{31}) 個字元，而定長字串可包含 $1 \sim (2^{16})$ 個字元。

● 使用者定義型

可以是任何使用 Type 語法所定義的資料類型。使用者自定義類型可包含一個或多個某種資料類型的資料元素、陣列或一個先前已定義的使用者自定義類型。使用者定義型類似 C 語言中的結構。例如，我們可以使用以下語法來定義一個 Student 類型：

```
Type Student
    Name As String      '定義 String 類型來儲存學生姓名
    Age As Integer       '定義 Integer 類型來儲存學生年齡
    Sex As Byte          '定義 Byte 類型來儲存性別
End Type
```

● Variant（變式型）

Variant 資料類型是所有沒被明確宣告為其他類型變數的資料類型。Variant 是一種特殊的資料類型，它可以包含除了定長 String 資料以及使用者定義類型外的任何種類資料。在 AutoCAD VBA 中，Variant 類型經常用於點的變數宣告。

● VBA 新增資料類型

對應內含 VBA 的應用程式，VBA 還提供許多應用程式的資料類型。如，AutoCAD 的 VBA 中將提供：線變數 (AcadLine)、圓變數 (AcadCircle) … 等這類的資料類型；而在 Word 和 Excel 的 VBA 則提供了：行變數

(Rows)、列變數 (Columns) ..等這類的資料類型。

3-2 變數宣告

接下來在本節中要談的就是變數宣告。在正式的程式語法中，您一定要先宣告程式中所使用到的變數，如此，程式才能正確的被執行。一樣的，以下我們將分二小節來談 VLISP 與 VBA 的變數宣告。

3-2-1 Visual LISP 的變數宣告

Visual LISP 變數的資料類型在賦予數值時，必須指定它的資料類型。所以在為一變數指定新值前，該變數值將一直保留原來的值與及資料類型。以下，就是在 Visual LISP 裡使用 setq 函數來做變數宣告的範例：

```
_$ (setq int 8)
8
_$ int
8
_$ (dbl 3.1415)
3.1415
_$ dbl
3.1415
_$ (setq str "teststr")
"teststr"
_$ str
"teststr"
_$ (setq str 1e-5)
1.0e-005
_$ str
1.0e-005
```

從上面的範例可以看出：賦予一變數值，同時也將決定這個變數的資料類型。其中，對變數 str 先賦值 String 資料類型，再賦予浮點資料類型，正說明同一個變數可以透過直接賦予值來改變其資料類型，但在 VBA 中，一個變數資料類型的轉變則需透過類型轉變函數來處理。

3-2-2 VBA 的變數宣告

VBA 中有兩種宣告變數的型式：「明確宣告」與「隱藏宣告」。以下分三小節來闡述。

3-2-2-1 明確宣告

明確宣告將使用 Dim 來表示宣告變數。如：

Dim int2 As Integer	'宣告一個整數型變數
Dim dbl As Double	'宣告一個雙精度變數
Dim str As String	'宣告一個字串變數
Dim circ As AcadCircle	'宣告一個 AutoCAD 圓變數
Dim centerpt As Variant	'宣告一個變式變數，通常在 AutoCAD 中表示點
Dim pt(0 To 2) As Double	'宣告一個一維雙精度浮點陣列，在 AutoCAD 中通常表示點

在同一個模組中，變數宣告後，後續的程式就會依宣告的類型來使用該變數，如果在其中又賦予新值給其他類型變數，那麼就需經過資料類型的轉換。在 VLISP 中，這種轉換將自動進行。例如，輸入如下的程式：

```
(1) 'Variabe Type Change Program-----change.dvb
(2) 'function:change variable type
(3) Public Sub change()
(4) Dim int2 As Integer, dbl As Double
(5) dbl = 6.1
(6) int2 = dbl
(7) Debug.Print "int2="; int2
(8) Debug.Print "dbl="; dbl
(9) End Sub
```

分析：我們在第(7)行將浮點數轉換為整數，結果就發生了類型轉變。因此，第(8)行與第(9)行的輸出結果就不一樣。如果不指定資料類型或物件類型，則變數的預設資料類型將為 Variant。如下示範例中：

```
Dim dbl
Dim str, str2 As String
```

其中，dbl 和 str 均為 Variant 變數，str2 為字串型變數。而第 2 條語法中的 str 則為 Variant 變數，如要將 str 和 str2 均宣告為字串型變數，則應將之寫為：

```
Dim str As String, str2 As String。
```

3-2-2-2 隱藏宣告

如果在變數後再加上某一個類型宣告字元，這樣的變數宣告型式就稱為「隱藏宣告」。例如，Currency 的類型宣告字元就是“@”；所以若在變數後加上 @，如 Cur@=352478，就表示 Cur 為一個貨幣型變數。所有的類型宣告字元如下表所示：

表 3-6 VBA 的隱藏宣告字元

變數類型	類型宣告字元	變數類型	類型宣告字元
Integer	%	Long	&
Single	!	Double	#
String	\$	Currency	@

以下，我們就以下面這個簡單的範例來示範隱藏宣告變數（輸出 var= 3.14，int2= 5，dbl= 4.2）：

- (1) 'Hidden Announce Program-----HideAnnounce.dvb
- (2) 'Hidden Announce Variable Demo
- (3)
- (4) Public Sub HideAnnounce()
- (5) int2% = 5 '隱藏宣告一個 Integer 資料類型
- (6) dbl# = 4.2 '隱藏宣告一個 Double 資料類型
- (7) var = 3.14 '預設隱藏宣告一個 Variant 資料類型
- (8) Debug.Print "int2="; int2
- (9) Debug.Print "dbl="; dbl
- (10) Debug.Print "var="; var
- (11) End Sub

3-2-2-3 Option Explicit 語法

隱藏宣告變數雖然可以讓程式簡潔，但並不直接，所以在程式閱讀上為我們帶來了困難。此外，有時在使用變數時，如果忘記了宣告就直接使用，就會相當於隱藏宣告了 Variant 類型，如此，不但不利於程式偵錯並且還將佔用大量的系統資源。在這樣的情況下，使用 Option Explicit 語法就可以強制對模組中的所有變數進行明確宣告，否則在程式編寫執行過程中，就會出現一錯誤訊息來警示我們。例如，如果您將 3-2-2-2 節的範例寫成如下語法，那麼就會在執行過程中出現一提醒您的錯誤訊息框：

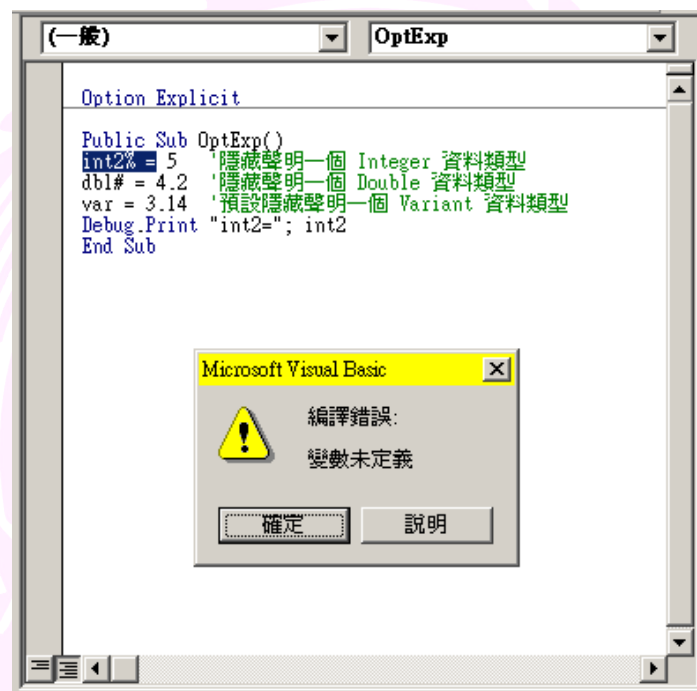


圖 3-1 出錯的隱藏宣告

如此，您就可以藉此修改程式語法。

3-3 函數

瞭解過資料類型與變數宣告後，接下來要談的就是重要的函數定義了。

3-3-1 Visual LISP 的函數

AutoLISP/VLISP 的函數定義都是使用 defun 函數，defun 函數的標準語法如下：

```
(defun func_name ( arguments / local_variables )  
  [expressions...]  
)
```

其中，func_name 是函數名稱，這是固定不可少的。函數名稱後的括弧內為：arguments(參數)與 local_variables(局部變數)所組成的參數，它們之間將以 “/” 來隔開。參數是可有可無的。expression 即運算式，絕不能省略。

在 AutoCAD 中有下示三種主要類型的函數：

- 第一類是經由 Defun 函數所定義的指令。其特色是使用 “c:” 來執行定義的函數。例如：

```
(defun c:func()  
  (princ)  
)
```

如此，只要在指令提示區鍵入：func 或 (c:func)，就可以執行這個程式。

- 第二類是在建立程式時不加 “c:” 。這類函數屬副程式，它可以被其他函數所呼叫。如果要在指令提示區中執行這種副程式，那麼要將此函數名稱放入一對括弧中。例如：

```
(defun func( )  
  (princ)  
)
```

如此，只要在指令提示區鍵入：(func)，就可以呼叫這個副程式來執行。

- 第三類的函數類型為 S::STARTUP。這類函數可放在 Acad2008doc.lsp(或 Acad2000doc.lsp)中，每當您新開啓一圖檔時，S::STARTUP 函數中的內容就被載入一次。如此，在支援多重文件檔 (MDE) 的 AutoCAD 2000 中，利用 S::STARTUP 函數將是提高繪圖效率的好方法。例如：S::STARTUP 函數可用在圖框的設定中：

```
(defun s::startup ()  
  (command "rectang" "width" 0.3 "0,0" "100,100")  
)
```

例如，如下程式語法將可讓您在指令提示區中鍵入 test 後，就輸出 3：

```
(1) 'Function Call Program-----callfunc.lsp  
(2) 'function: Call a function  
(3)  
(4) (defun func(a b / c)  
(5) (setq c (+ a b))  
(6) )  
(7) (defun c:test()  
(8) (func 1 2)  
(9) )
```

3-3-2 VBA 的函數與程序

就如同其他的高階語言，VBA 與 Visual LISP 也內建了許多函數，如數學函數（Abs 等）、資料轉換函數（CDBl 等）、字串函數（Strlen 等）。這些函數都可以直接用於運算式中。而與 Visual LISP 相似的使用者自定義函數在 VBA 中，確切地說，應該是一個「函數程序」的概念，意即這類函數是一種程序。在 VBA 中，程序分三類：Sub 程序，函數程序與屬性程序，其中 Sub 程序也稱為「副程式」或「巨集」。在 VBA 編輯器中，選擇「插入」下拉式功能表裡的「程序(P)...」選項後所出現的「新增程序」視窗中，在其內的「型態」框中就有這三種程序：

● Sub 程序

Sub 程序是由一系列的 Sub 和 End Sub 語法所包含起來的 VB 程式集所組成的。其標準語法為：

```
Sub test()  
  Expression  
End Sub
```

在 Sub 之前還可以使用關鍵字。例如，在前文中提到的 Sub 程序範例中

的 Public（公有）與 Private（私有）宣告。這是用於指定該程序可以被其他程序所呼叫的限制範圍。

Public 關鍵字將使該程序可以被其他程式模組的程序呼叫，Private 關鍵字則可以讓該程序僅可在該程式模組中被呼叫。如果 Sub 前不加關鍵字，則預設值為 Public。

如果您在程式中經常需要多次使用同一運算式或同一群運算式時，您可以將此一或這群運算式編寫成 Sub 程序（副程式）。

● 函數程序

函數程序將傳回一個值，如下範例所示：

```
Function distance(sp As Variant, ep As Variant) As Double
Dim x As Double
Dim y As Double
Dim z As Double
x = sp(0) - ep(0)
y = sp(1) - ep(1)
distance = Sqr((x ^ 2) + (y ^ 2))
End Function
```

這個 distance 函數程序將傳回兩個點之間的距離。在您經常需要多次使用同一公式時，就可以考慮編寫這類的函數程序。

● 屬性程序

我們直接以下述範例來說明屬性程序的應用（在視窗中列印出：PenColor is Red）：

- (1)'Get Property Program-----propget.vba
- (2)'function: Get property
- (3)
- (4)Dim CurrentColor As Integer '定義筆顏色
- (5)'將筆顏色轉用常數表示
- (6)Const BLACK = 0, RED = 1, GREEN = 2, BLUE = 3
- (7)'以字串類型來傳回筆的顏色
- (8)Property Get PenColor() As String

```
(9)    Select Case CurrentColor
(10)   Case RED
(11)       PenColor = "Red"
(12)   Case GREEN
(13)       PenColor = "Green"
(14)   Case BLUE
(15)       PenColor = "Blue"
(19)'呼叫 Property get 程式來取得筆的顏色
(20)
(21)Public Sub GetDemo()
(22)CurrentColor = 1
(23)ColorName = PenColor
(24)Debug.Print "PenColor is "; ColorName
(25)End Sub
```

分析：這個範例我們已經在註解列中說明的很清楚。此例的 Property 程序將以傳回屬性值（顏色字串）來達到目的。

3-4 變數的作用範圍與生命期

現在，我們要在這一節討論的是有關變數更進一步的內涵以及它們在程序中的屬性。我們一樣分 Visual LISP 與 VBA 二方面來談。

3-4-1 Visual LISP 的局部變數與整體變數

Visual LISP 的變數可分為：局部變數與整體變數。它們都可以在函數或運算式中定義。從上文可瞭解到：在函數定義的參數中，位於“/”右邊的均為局部變數。事實上，只要沒有特地標示為局部變數的變數均屬整體變數。

整體變數的作用範圍就是目前的 AutoCAD 程式本文檔，當本文檔關閉後，該變數群即失效，在整個程式本文檔工作期間，運算式或函數都可改變它的值。而局部變數作用範圍就只有在使用它的函數中，當函數結束後即失效。因此，不同名稱的函數可以有同名的局部變數。

整體變數將作用於整個程式中，所以在程式的任一函數中都可以存取並修改它的值。整體變數的壞處是：您可能不經意地修改了它的值，而導致程式的失誤。如此，程式愈大，偵錯就會愈困難。使用局部變數的優點在於：在函數中的局部變數不會受到其他外部應用程式的影響，並且在呼叫函數的任務結束後，該變數

就會失效且釋放其所佔的記憶體空間。

在一般的情況下，程式中的多數變數多屬局部變數，我們也建議您儘可能的少用整體變數，因此，在變數的使用上就要有規劃。標記全程使用的變數是一個好習慣，我們也建議您在變數名稱的首尾再添加兩個星號來表示整體變數。例如，*FileSaved*。

以下就是一個學習 Visual LISP 的局部變數與整體變數的範例。這個範例將用來計算一個數學函數的值：

$$f(x) = x^2 - 5x + 10$$

以及在 $x=3.5$ 處的斜率。而斜率將使用以下公式近似計算：

$$f'(x) = (f(x+h) - f(x-h)) / 2h$$

其中， h 是一個小的增量。

```
(1);;;Local Variable Program-----lovar.lsp
(2);;;function: Calculate an math equation
(3)
(4)(defun f(x / temp)
(5)  (setq temp (+ (* x x) (* -5 x) 10))
(6)
(7)(defun slope(x / f1 f2 incrim temp )
(8)  (setq incrim (* 0.01 x))
(9)  (setq f1 (f (+ x incrim)))
(10) (setq f2 (f (- x incrim)))
(11) (setq temp (/ (- f1 f2) 2 incrim))
(12)
(13)(defun c:lovar(/ x)
(14) (setq x 3.5)
(15) (princ "f(")(princ x)(princ ")=")(princ (f x))(princ "\n")
(16) (princ "f'(")(princ x)(princ ")=")(princ (slope x))(princ
(17))
```

執行方式：載入後，在主控台中依下執行：

```
_$(c:lovar)
f(3.5)=4.75
```

$f(3.5)=2.0$

分析：從第(4)行起將定義一個數學函數 f 計算公式。其中，temp 為局部變數，x 雖然沒有定義為局部變數，但它只是作為傳遞參數的作用，在程式中並沒有使用 setq 來賦予它值，因此，x 也不是整體變數。

3-4-2 VBA 的模組級變數與程序級變數

在 VBA 的變數方面，它分為「模組級變數」與「程序級變數」。所謂「模組級變數」就是在模組中所使用的，其範圍又分為「公共模組級變數」與「私有模組級變數」：

- 公共模組級變數的宣告是在模組程序以外的，您只要在變數前加上關鍵字 Public 即可，如：

```
Public a As Integer
```

公共模組級變數在專案中的所有模組都能使用，當專案結束時，這類變數就算失效。

- 私有模組級變數的宣告也是在模組程序以外，您只要在變數前加上關鍵字 Private，或直接用 Dim。例如以下兩句作用完全一樣的語法：

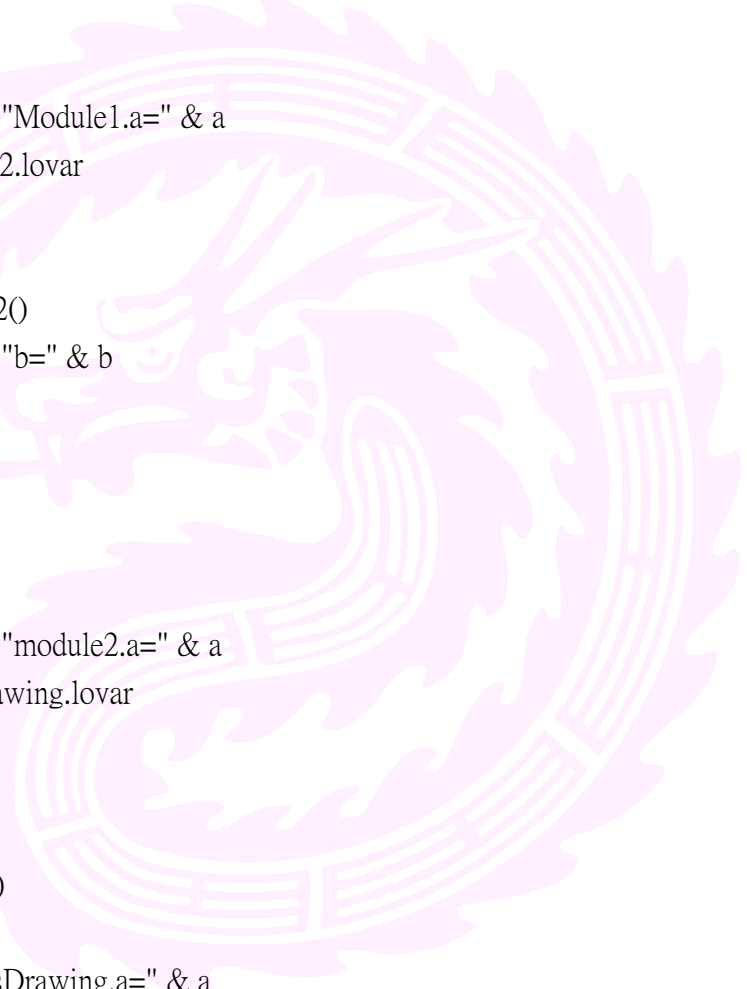
```
Private a As Integer  
Dim a As Integer
```

私有模組級變數在同一模組中的所有程序都能使用。當該模組結束時，這類變數就會失效。

程序級局部變數宣告將在模組內進行，一般以 Dim 開頭來宣告。例如：

```
Sub test()  
Dim a As Integer  
    Expression...  
End Sub
```

程序級變數只能在宣告該變數的程序中使用。當程序結束時，該變數即失效。



```
(1)'Local Variable Program-----lovar.vba
(2)'function: Calculate an math equation
(3)
(4)'module1
(5)Option Explicit
(6)Public a As Integer
(7)Dim b As Integer
(8)Public Sub lovar()
(9)    a = 5
(10)   b = 6
(11)   Debug.Print "Module1.a=" & a
(12)   Call Module2.lovar
(13)   Call lovar2
(14)End Sub
(15)Private Sub lovar2()
(16)   Debug.Print "b=" & b
(17)End Sub
(18)'module2
(19)Option Explicit
(20)Sub lovar()
(21)   a = 9
(22)   Debug.Print "module2.a=" & a
(23)   Call ThisDrawing.lovar
(24)End Sub
(25)'thisdrawing
(26)Option Explicit
(27)Public Sub lovar()
(28)a = 10
(29)Debug.Print "ThisDrawing.a=" & a
(30)End Sub
```

執行方式：執行巨集 Module1.lovar 後，在視窗中列印出：

```
Module1.a=5
Module2.a=9
ThisDrawing.a=10
Module1.b=6
```

分析：在程式的第(6)行，我們定義 a 為公共模組級變數，第(7)行定義 b 為私有模組級變數。變數 a 和 b 將於模組 Module1 中被使用。在程式的第 (12) 行裡，我們令其呼叫另一模組 Module2 中的 lovar 程序，第 (21)，(22) 行則呼叫了公共模組級變數 a。第(29)行在 Thisdrawing 中也呼叫了公共模組級變數 a。

3-4-3 VBA 的靜態變數

在前面的範例中，雖然在程序結束時，程序級的局部變數就失效。不過，VBA 還可允許您將局部變數宣告為靜態變數（即在變數之前加上 Static 關鍵字）。如：

```
Static ID As Integer      '宣告 ID 為靜態變數
```

您也可在程序定義之前加 Static，將程序中的所有局部變數均宣告為靜態變數。例如：

```
Public Static Function Circumference(Radius)
```

'在函數叫用之間所有的本地變數值都將保留。

```
Half = 3.14 * Radius      'Half 為靜態變數
```

```
Circumference = 2 * Half 'Circumference 為靜態變數
```

```
End Function
```

靜態變數是一種 VBA 中特殊的局部變數，它不但在程序呼叫之間保留了局部變數的值，同時當程序結束時，靜態變數仍可保留了它的值。這是因為編譯器可支援這種語言特性，將靜態變數儲存在不同的記憶體位置上。這些記憶體位置的內容可在程式執行過程中一直保留的緣故。您可以在不同的程序中使用相同名稱的靜態變數，這種重複並不會造成編譯器的混淆，因為它一直記錄著哪個程序擁有哪些靜態變數。下面我們就再舉一個例子來說明靜態變數的宣告型式：

(1)'Static Local Variable Program-----lovar.vba

(2)'function:Use Static Local Variable for Calculating

(3)

(4)Option Explicit


```
(5)Public Function Mean(x As Double)
(6)    Static sum As Double
(7)    Static sumx As Double
(8)    sum = sum + 1
(9)    sumx = sumx + x
(10)   Mean = sumx / sum
(11)End Function
(12)Public Sub stvar()
(13)   Debug.Print "mean=" & Mean(1)
(14)   Debug.Print "mean=" & Mean(2)
(15)   Debug.Print "mean=" & Mean(4)
(16)   Debug.Print "mean=" & Mean(10)
(17)   Debug.Print "mean=" & Mean(12)
(18)End Sub
```

執行方式：在視窗中列印出

```
mean=1
mean=1.5
mean=2.33333333333333
mean=4.25
mean=5.8
```

分析：本例中宣告了含有靜態變數的函數程序 Mean。第(6)、(7)行宣告了靜態變數 sum 與 sumx。第(8)行語法將 sum 加 1，第(9)行則將變數 sumx 加上參數 x，第(10)行將傳回更新後的平均值。由於程序 stvar 中呼叫了函數程序 Mean，所以第 (13)~(17) 行將顯示更新後的平均值。如果不用靜態變數，那就必須使用模組級變數。

請注意：多次執行此程式後，在視窗中所得到的結果是不一樣的。這是因為靜態局部變數還保持原來的值。您必須在「執行(R)」下拉式功能表中選取「重設(R)」選項，才能使其失效。

啓發性習題

一.選擇題(單複選混合)

1.() 以下何者是 Visual LISP 的有效整數值？

- (a) 1,234,567,890~-1,234,567,890
- (b) 2,147,483,647~-2,147,483,648
- (c) 3,147,483,647~-3,147,483,648
- (d) 以上皆非

2.() 以下何者是 AutoLISP/VLISP 裡的有效符號：

- (a) (
- (b) \$
- (c) &
- (d) ;

3.() 以下哪一個控制字元代表「換行(Newline)」？

- (a) \"
- (b) \n
- (c) \t
- (d) 以上皆非

4.() VBA 可以支援的「Currency (貨幣型)」資料類型的數值範圍是：

- (a) 3,147,483,647~-3,147,483,648
- (b) -322,337,203,685,477.5808~322,337,203,685,477.5807
- (c) -922,337,203,685,477.5808~922,337,203,685,477.5807
- (d) 以上皆非

5.() 以下有關「Variant (變式型)」資料類型的敘述，何者為非？

- (a) 凡是所有沒被明確宣告為其他類型變數的資料類型就是 Variant
- (b) 它可以包含除了定長 String 資料以及使用者定義類型外的任何種類資料。
- (c) 它是一種代表數學公式的資料類型
- (d) 以上皆非

6.() 以下有關「明確宣告」與「隱藏宣告」的敘述，何者為真？

- (a) 如果忘記了宣告就直接使用，就會相當於隱藏宣告了 Variant 類型，如此，不但不利於程式偵錯並且還將佔用大量的系統資源
- (b) 隱藏宣告變數雖然可以讓程式簡潔，但並不直接，在程式閱讀上會較困難
- (c) 只有 VLISP 需要宣告，VBA 就不需要
- (d) 以上皆非

7.() 在 VBA 中，程序分哪三類？

- (a) 程序
- (b) 函數程序
- (c) 屬性程序
- (d) 以上皆是

8.() 以下有關變數的敘述，何者為非？

- (a) 變數可分為：局部變數與整體變數
- (b) 宣告局部變數會比較佔記憶體
- (c) 整體變數應儘量放在 acad.lsp 檔中
- (d) 以上皆非

9.() 以下何者是在 AutoLISP/VLISP 中宣告局部變數的語法？

- (a) (defun c: xxxx(/ a b c d)
- (b) (defun c: xxxx(\ a b c d)
- (c) (defun c: xxxx(! a b c d)
- (d) 以上皆非

二.實作問答題

1. 請試述 AutoLISP/VLISP 的有效資料類型。
2. 請試述 VBA 的有效資料類型種類。
3. 請試述何謂靜態變數。