

# 3. 變數與常數

\*\*\*\*\*

## 3.1 進制轉換

## 3.2 變數

## 3.3 常數

## 3.1 進制轉換

我們從小慣用的進制為 10 進制，數字的組合由 0~9 來構成，而二進制的數字組合只有 1 和 0，十進制超過 9 則左邊的位數便加 1，而二進制也是如此，唯一的差異在二進制是超過 1 就在左邊的位數加 1。

為什麼要使用二進制呢~原因出在於電腦只認得 0 與 1 的信號，而我們所看見的美麗的圖案~文字，其實也都是 0 和 1，只要透過軟體的幫助，可以讓我們輕易的判別電腦上的資訊。

那十六進制是做什麼用的勒~如果有一長串的二進制，可以將其換成十六進制，較容易辨視，且二進制轉換成十六進制是十分便利的。

下表列出 0~15 的二進制、十進制與十六進制的對照：

二進制	十進制	十六進制
0000	0	0
0001	1	1
0010	2	2
0011	3	3
0100	4	4
0101	5	5
0110	6	6
0111	7	7
1000	8	8
1001	9	9
1010	10	A
1011	11	B
1100	12	C
1101	13	D
1110	14	E
1111	15	F

至於我們該如何知道一個 2 進制的數相當於 10 進制的多少勒？其實，很簡單！只要用很基本的計算方法就可以換算出來，而且在後面的章節，我們也將導入一個進制轉換的程式來加以說明。

### 3.1.1 進制轉換

#### 3.1.1.1 二進制轉十進制

由上表中可知在 2 進制中 1001 為 10 進制的 9，接下來就是數學計算的問題：

$$\begin{array}{ccc} \text{左} & \xrightarrow{\hspace{10em}} & \text{右} \\ 1001 = & \mathbf{1} \times 2^3 + \mathbf{0} \times 2^2 + \mathbf{0} \times 2^1 + \mathbf{1} \times 2^0 & = 8 + 0 + 0 + 1 = 9 \end{array}$$

#### 3.1.1.2 十進制轉二進制

若是想由 10 進制轉 2 進制則必須用到短除法，計算方式如下，結果為 1001

$$\begin{array}{r|l} 2 & 9 \\ \hline & 4 \quad \dots 1 \\ 2 & 2 \quad \dots 0 \\ 2 & 1 \quad \dots 0 \\ & 0 \quad \dots 1 \end{array} \begin{array}{l} \uparrow \text{左} \\ \downarrow \text{右} \end{array}$$

#### 3.1.1.3 二進制轉十六進制

由上表中可知在 2 進制中 101 1111 為 16 進制的 5F，要轉換之首要動作是，先把它分成 4 個位元一組(從右邊開始數)，然後把每一組裡的二進制轉成十六進制，它有  $2^4$  種可能：0000~1111<sub>(2)</sub>=0~9、A、B、C、D、E、F<sub>(16)</sub>

$$\begin{array}{ccc} \text{左} & \xrightarrow{\hspace{10em}} & \text{右} \\ 0101\ 1111 & = & (\mathbf{0} \times 2^3 + \mathbf{1} \times 2^2 + \mathbf{0} \times 2^1 + \mathbf{1} \times 2^0)_2 \\ & & (\mathbf{1} \times 2^3 + \mathbf{1} \times 2^2 + \mathbf{1} \times 2^1 + \mathbf{1} \times 2^0)_2 \\ & & = (0 + 4 + 0 + 1)(8 + 4 + 2 + 1) = (5F)_{16} \end{array}$$

#### 3.1.1.4 十六進制轉二進制

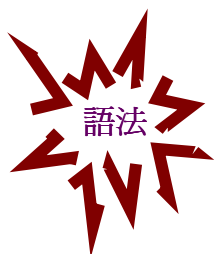
若是想由 16 進制轉 2 進制，可以用查表的方式，或是將十六進制大於十的部份都換成十進制，再用短除法計算出來。

$$5F_{(16)} = 0101\ 1111_{(2)}$$

### 3.1.2 進制轉換程式

在介紹進制轉換程式之前，我們先來學一下待會兒會使用到的敘述唄～

**Do While ... Loop 迴圈**：如果條件判斷式成立，一直重複做下列敘述或算式，直到條件判斷式不成立，才停止迴圈。

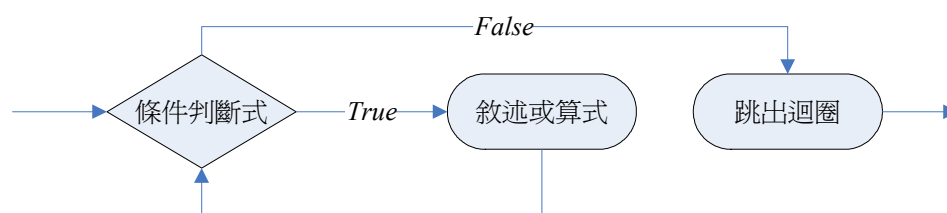


Do While 條件判斷式

...  
要執行的敘述或算式的放置區塊  
...

Loop

Do While ... Loop 流程圖：



**If ... then ... 敘述**：如果條件判斷式成立，則做下列敘述或算式；否則跳過，視同沒看到此敘述。



If 條件判斷式 then 要執行的敘述或算式

或

If 條件判斷式 then

...  
要執行的敘述或算式的放置區塊  
...

End If

或

If 條件判斷式 then

...

要執行的敘述或算式的放置區塊

...

Else

...

非 IF 的情況才執行的敘述或運算

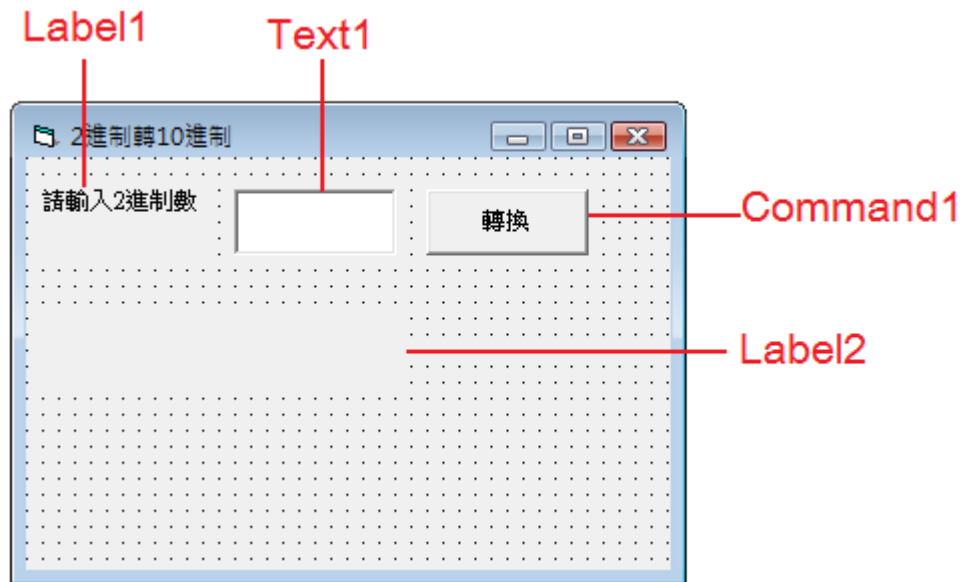
...

End If

現在就先將上述幾個語法熟練，到了第四章我們會加以介紹其他的寫法，程式的流程不外乎由上而下、分岐決策、迴圈，有了上述的語法，已可足夠撰寫大多數的程式。

### 3.1.2.1 進制轉十進制

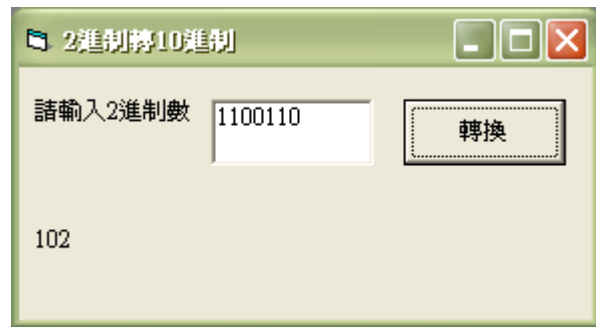
表單和程式碼



```
Private Sub Command1_Click()  
    Dim Ten As Integer '用來存放轉換後的十進位  
    Dim Binary As String '存放輸入的二進位  
    Dim Length As Integer '所須處理二進位的長度  
    Dim Temp As String '暫用用的字串變數  
    Binary = Text1.Text '從Text1取得輸入的值存入Binary  
    Length = Len(Binary) 'len是取得長度的內建函數  
    Do While Length > 0  
        Temp = Right(Binary, 1) '取出最右邊的一個位元 ex:1010→temp = 0  
        Binary = Left(Binary, Length - 1) '去掉最右邊已處理過的位元  
        Ten = Ten + Val(Temp) * 2 ^ Counter '將已轉換成十進位的值先存進Ten  
        Counter = Counter + 1 '下一個位二代表的值是現在的2倍  
        Length = Length - 1 '所需處理的長度減少一個位元  
    Loop  
    Label2.Caption = Ten '將結果輸出  
    '附註  
    ' val() 是將字串轉換成數值的內建函數  
    ' 對Visual Basic 而言, 所有的輸入預設的型態都是字串  
End Sub
```

執行後，在文字方塊打上你想轉換的二進制，

按下按鈕後，就會出現十進制了



### 3.1.2.2 十進制轉二進制

表單和程式碼：



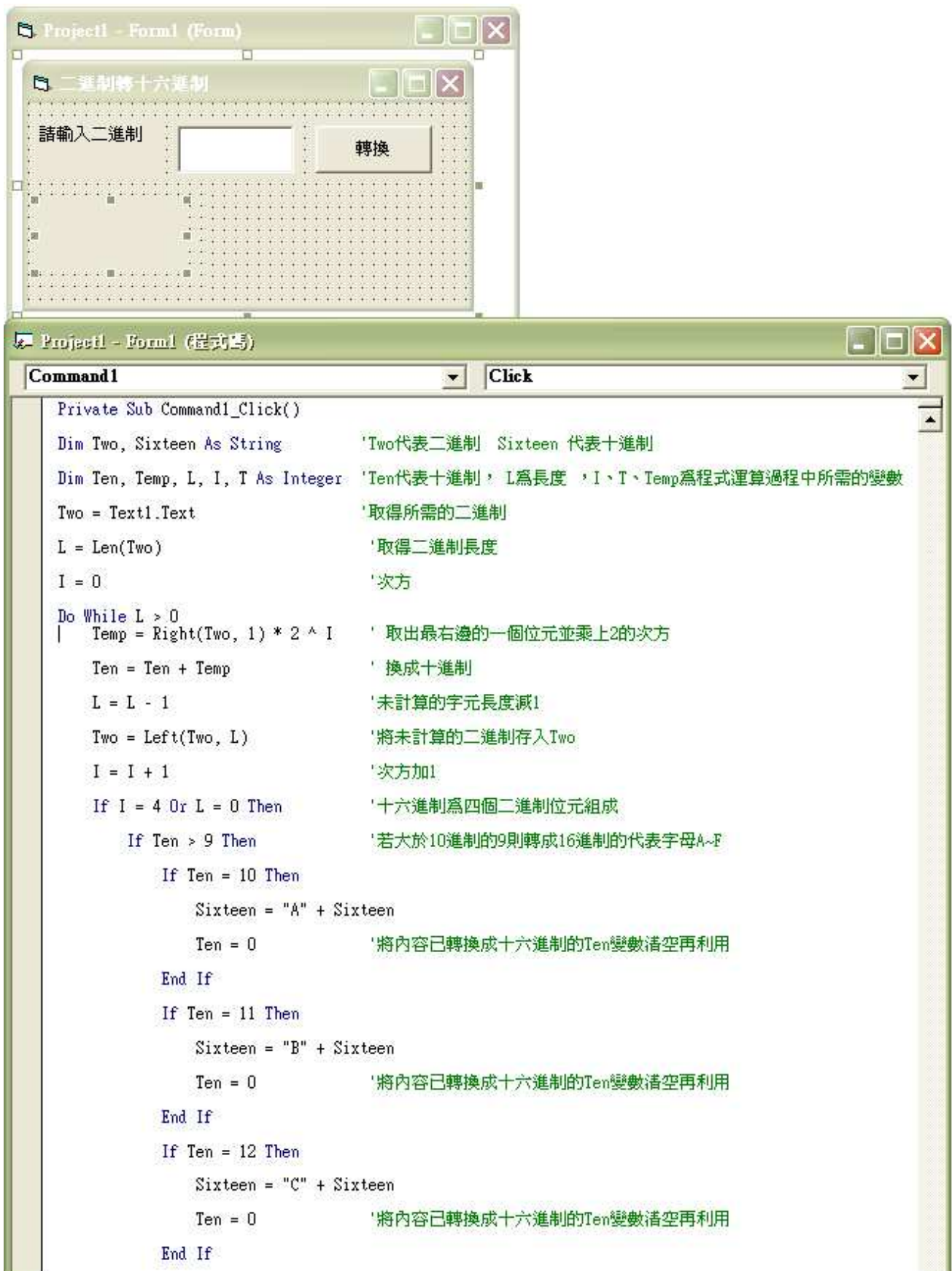
執行後，在文字方塊打上你想轉換的十進制按下按鈕後，就會出現二進制了





### 3.1.2.3 二進制轉十六進制

表單和程式碼



The image shows a Visual Basic IDE with two windows. The top window, titled 'Project1 - Form1 (Form)', displays a user interface with a text box labeled '請輸入二進制' and a button labeled '轉換'. The bottom window, titled 'Project1 - Form1 (程式碼)', shows the code for the 'Click' event of 'Command1'.

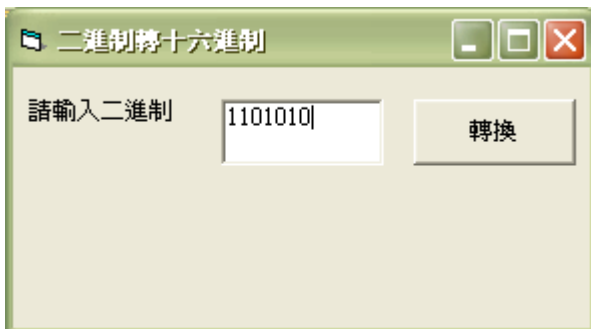
```
Private Sub Command1_Click()  
    Dim Two, Sixteen As String      'Two代表二進制 Sixteen 代表十進制  
    Dim Ten, Temp, L, I, T As Integer 'Ten代表十進制, L為長度, I、T、Temp為程式運算過程中所需的變數  
    Two = Text1.Text                '取得所需的二進制  
    L = Len(Two)                    '取得二進制長度  
    I = 0                            '次方  
    Do While L > 0  
        Temp = Right(Two, 1) * 2 ^ I '取出最右邊的一個位元並乘上2的次方  
        Ten = Ten + Temp              '換成十進制  
        L = L - 1                    '未計算的字元長度減1  
        Two = Left(Two, L)           '將未計算的二進制存入Two  
        I = I + 1                    '次方加1  
        If I = 4 Or L = 0 Then        '十六進制為四個二進制位元組成  
            If Ten > 9 Then           '若大於10進制的9則轉成16進制的代表字母A-F  
                If Ten = 10 Then  
                    Sixteen = "A" + Sixteen  
                    Ten = 0           '將內容已轉換成十六進制的Ten變數清空再利用  
                End If  
                If Ten = 11 Then  
                    Sixteen = "B" + Sixteen  
                    Ten = 0           '將內容已轉換成十六進制的Ten變數清空再利用  
                End If  
                If Ten = 12 Then  
                    Sixteen = "C" + Sixteen  
                    Ten = 0           '將內容已轉換成十六進制的Ten變數清空再利用  
                End If  
            End If  
        End If  
    End Do  
End Sub
```

```
    If Ten = 13 Then
        Sixteen = "D" + Sixteen
        Ten = 0          '將內容已轉換成十六進制的Ten變數清空再利用
    End If
    If Ten = 14 Then
        Sixteen = "E" + Sixteen
        Ten = 0          '將內容已轉換成十六進制的Ten變數清空再利用
    End If
    If Ten = 15 Then
        Sixteen = "F" + Sixteen
        Ten = 0          '將內容已轉換成十六進制的Ten變數清空再利用
    End If
Else
    Sixteen = Str(Ten) + Sixteen
    Ten = 0              '將內容已轉換成十六進制的Ten變數清空再利用
End If
I = 0                   '次方歸零
End If
Loop
Label2.Caption = Sixteen '輸出十六進制
End Sub
```

雖然這個程式碼看起來很長，可是其實它並不多，它有一半以上在寫類似的指令，這個程式和下一個十六進制轉二進制的程式都比前兩個程式還要難。

執行後，在文字方塊打上你想轉換的二進制

按下按鈕後，就會出現十進制了



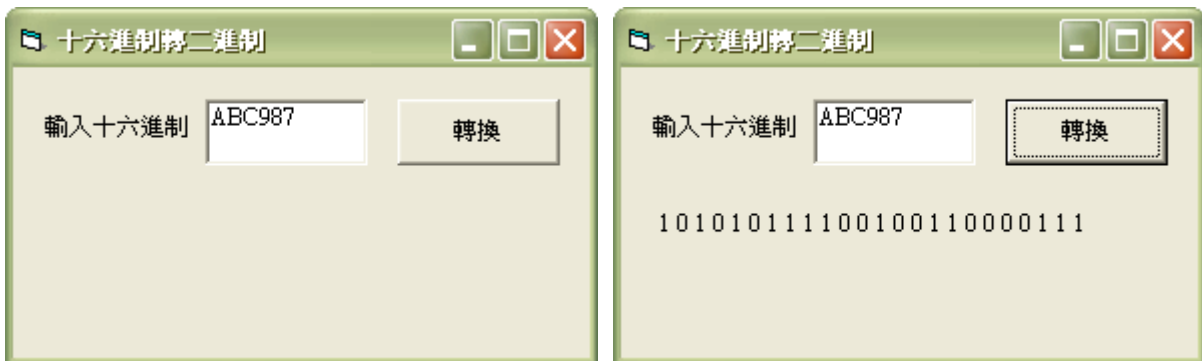
### 3.1.2.4 十六進制轉二進制

#### 表單和程式碼



這個程式碼也很長，它跟二進制轉十六進制的程式類似，你不妨比較看看兩者不同處。

執行後，在文字方塊打上你想轉換的十六進制          按下按鈕後，就會出現二進制了



## 3.2 變數

什麼是變數呢？變數就是記憶體中可以存放資料的地方，而其所存放的內容可以在應用程式執行時被改變。

例：你的生命值：當你被對方用千年殺攻擊，你的生命值就少了 a H P，當你被對方用奪命剪刀腳攻擊，你的生命值就會少了 b H P，但是當你用急救技能，你的生命值就回復了 c H P；其中的 a b c 都是變數，是依你的等級、防禦與對方的等級、攻擊力而定，這些都是由你——程式設計師來設定的。

Dim 是 Dimension 的縮寫，我們都是用 Dim 來宣告變數名稱和變數的資料型態，以開啓適當的記憶體空間來存放該變數的資料，並在應用程式需要用到此變數資料的時候，從變數中取出來使用，用完再把記憶體釋放，真的是很環保。變數的宣告語法如下：



Dim 變數名稱 As 變數的資料型態

### 3.2.1 變數名稱

爲了要分辨每一個記憶體空間，我們就給它們取個名字，而你所宣告的變數名稱，就是此記憶體空間的名字。就像是新開一個資料夾來存我的照片，我可能會想用“My Photo”來當這個資料夾的名稱。但是變數名稱可不是你想取什麼就取什麼的哦~是有些忌諱的：

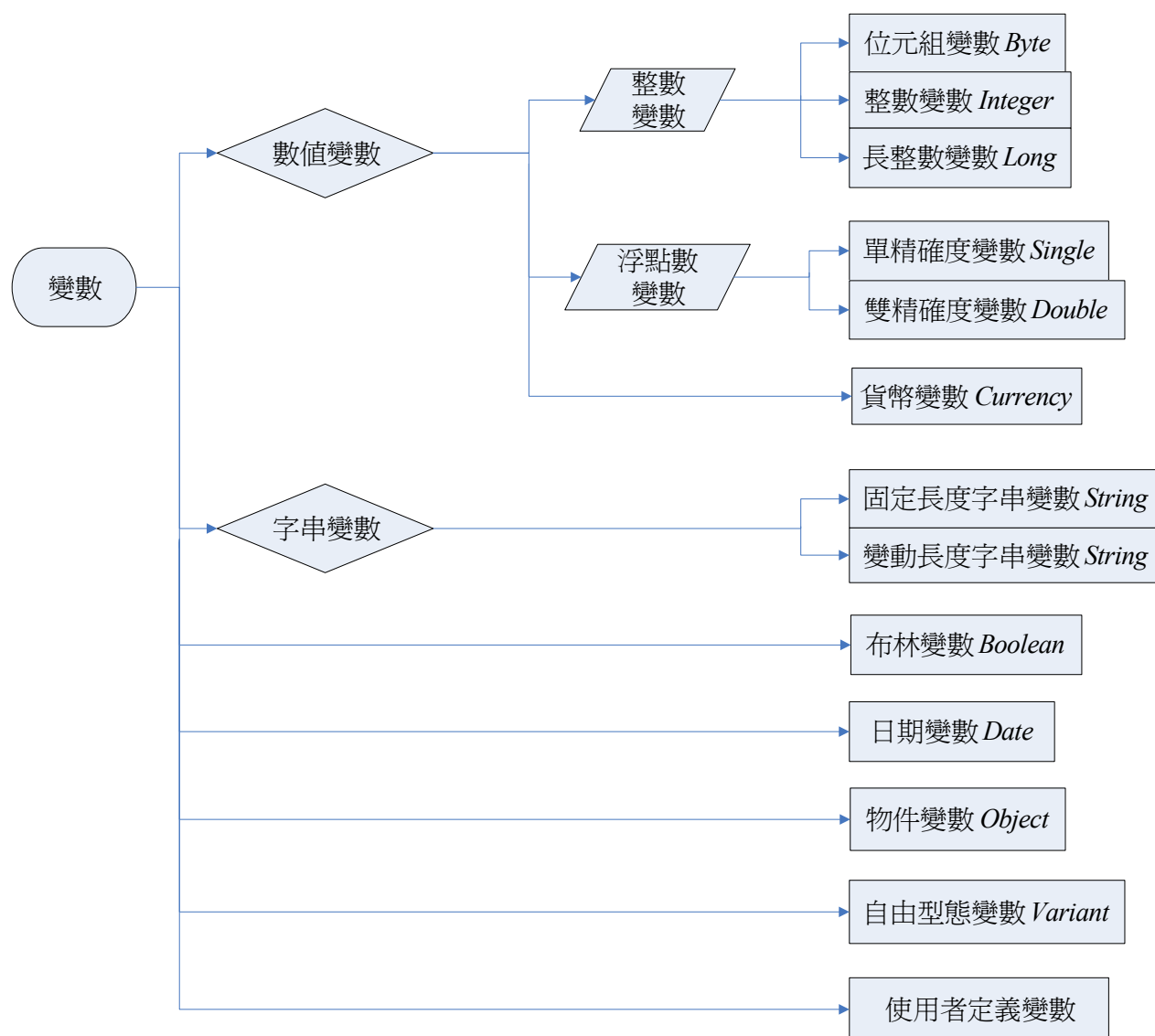
1. 名稱長度不可超過 255 個字元。
2. 不能使用 Visual Basic 的保留字。例：if、end、for...。
3. 第一個字不能使用底線『\_』or 數字。
4. 第一個字可以是大小寫英文字母 or 中文。
5. 第二個字之後可以是大小寫英文字母 or 中文 or 阿拉伯數字 or 底線。
6. 盡量使用跟變數本身的意義相同的英文名稱，以提高可讀性，讓程式維護的設計師能夠更快了解該變數的實際意義。例：總和可叫做 sum。
7. 英文的大小寫會被視爲相同的。例：SUM、Sum、sum 是一樣的。
8. 盡量不要使用中文。

註：如果沒有設定初始值，數值變數的預設值爲「0」，字串變數的爲「空字串」。

註：變數最好加個註解，以免健忘症發作時的捶胸頓足。

### 3.2.2 變數的資料型態

變數有很多種，它可以是數值，可以是字串，也可以是日期．．．等等。然而，要宣告一個變數，首先要認定你宣告它是要做什麼用的，估計它的大小，然後再選擇一個最適合它的資料型態，Visual Basic 的資料型態有很多種，如下表所示：



$$1\text{KB} = 2^{10} \text{ Bytes} = 1024 \text{ Bytes} \approx 10^3$$

$$1\text{MB} = 2^{20} \text{ Bytes} = 1,048,576 \text{ Bytes} \approx 10^6$$

$$1\text{GB} = 2^{30} \text{ Bytes} = 1,073,741,824 \text{ Bytes} \approx 10^9$$

$$1\text{TB} = 2^{40} \text{ Bytes} = 1,099,511,627,776 \text{ Bytes} \approx 10^{12}$$

下表為各資料型態的記憶體空間和有效範圍:

資料型態	佔用記憶體空間	有效範圍
Byte(位元組)	1Byte	0~255
Integer(整數)	2Bytes	-32,768~+32767
Long(長整數)	4Bytes	-2,147,483,648~+2,147,483,647
Single(單精度浮點數)	4Bytes	-3.402823E+38~-1.401298E-45 +1.401298E-45~+3.402823E+38
Double(雙精度浮點數 or 倍精度浮點數)	8Bytes	-1.79769313486231E+308~ -4.94065645841247E-324 +4.94065645841247E-324~ +1.7969313486231E+308
Currency(貨幣)	8Bytes	-922,337,203,685,477.5808~ +922,337,203,685,477.5807
String(固定長度字串)	每一個字元佔 1Byte	0~65,535 個字元
String(變動長度字串)	10ytes+字串長度	可變長度字串最多 2 <sup>31</sup> 個字元
Boolean(布林)	2Bytes	True(成立)或 False(不成立)
Date(日期)	8Bytes	January 1, 100 ~ December 31, 9999
Object(物件)	4Bytes	可引用任何一個 Object
Variant(自由型態)	數值:16 Bytes 字串: 22 Bytes+字串長度	任何數值、字串、物件、陣列、 Null、Error 等
Decimal	14 Bytes	Variant 之副型態，有效位數為 28 位，變數不能宣告為 Decimal 的型態，則須使用 Cdec 函數來 建立

在介紹範例之前，我們要先認識一種格式化輸出—Print 方法

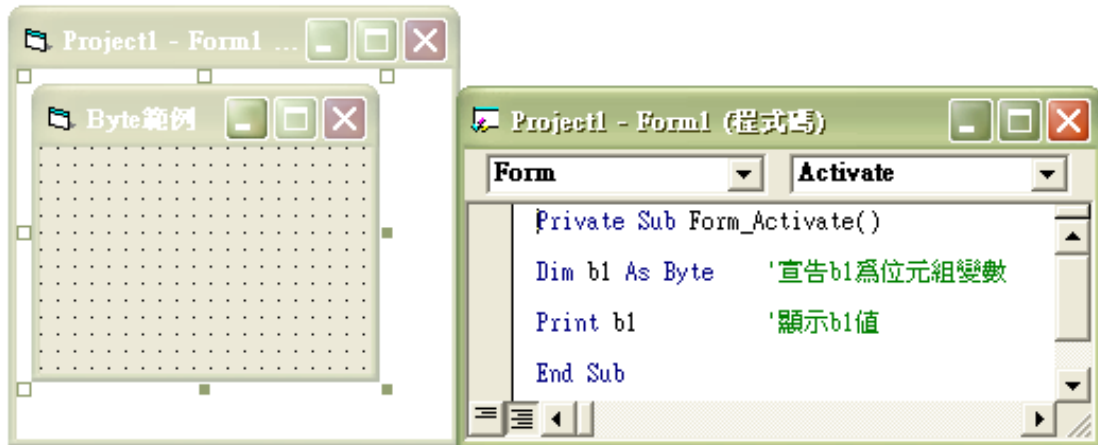


## Print 你想輸出的內容

你想輸出的內容可以是數值、字串、變數內容…等等。Print 方法是很常用到的，下面我們將直接舉例 Print 方法可以怎樣使用，並用其來比較看看，用什麼樣的資料型態會有什麼樣的輸出結果。



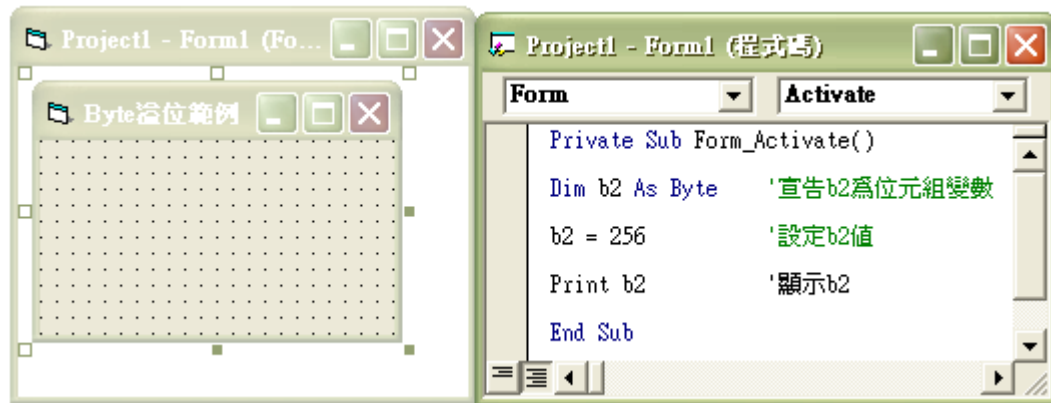
## Byte 範例 \* \* \* \* \*



執行結果：因為沒設定 b1 值，所以 Print 出來的是預設值 0。

因為是數值，所以 0 的前面還會留有一小格，如果是字串就不會有。所以你不想要那一小格，就把它轉成字串吧~用 str() 函數，以後會教到的。

## Byte 溢位範例 \* \* \* \* \*

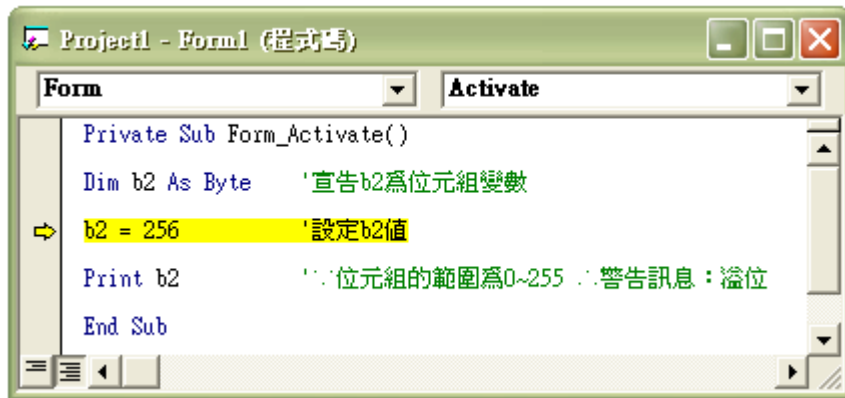


執行結果：因為初始值為 256，不在 0~255 之間，所以溢位(overflow)。



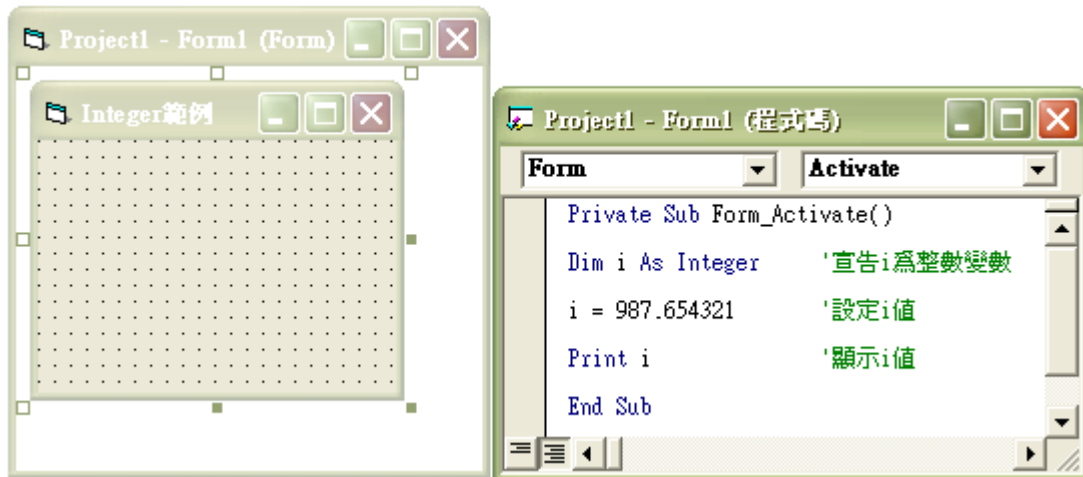


按下“偵錯”之後：



```
Project1 - Form1 (程式碼)
Form Activate
Private Sub Form_Activate()
Dim b2 As Byte '宣告b2為位元組變數
b2 = 256 '設定b2值
Print b2 '位元組的範圍為0~255 :警告訊息:溢位
End Sub
```

### Integer 範例 \* \* \* \* \*

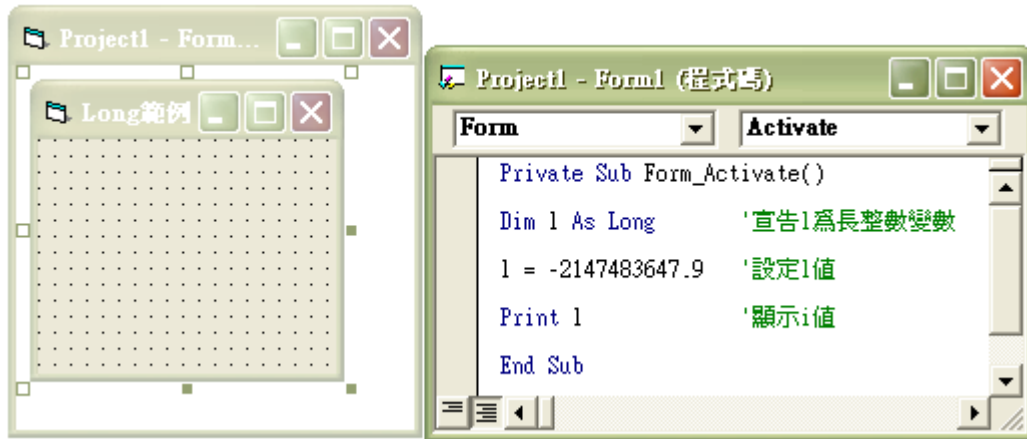


```
Project1 - Form1 (Form)
Integer範例
Project1 - Form1 (程式碼)
Form Activate
Private Sub Form_Activate()
Dim i As Integer '宣告i為整數變數
i = 987.654321 '設定i值
Print i '顯示i值
End Sub
```

執行結果：因為宣告為整數變數，所以小數點後面不但不會出現，而且還會四捨五入耶~



## Long 範例\*\*\*\*\*

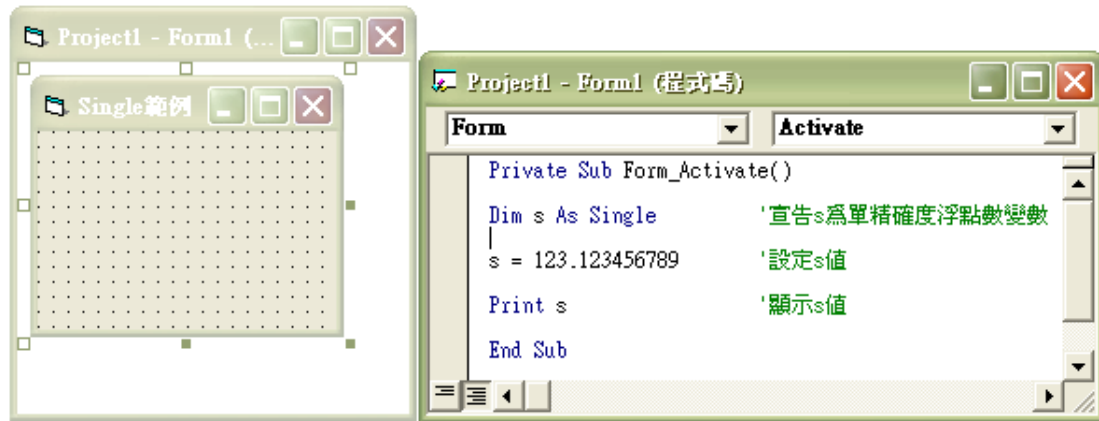


執行結果：



長整數變數的範圍為-2,147,483,648~+2,147,483,647，變數 l 四捨五入之後，剛好在邊緣處～

## Single 範例\*\*\*\*\*

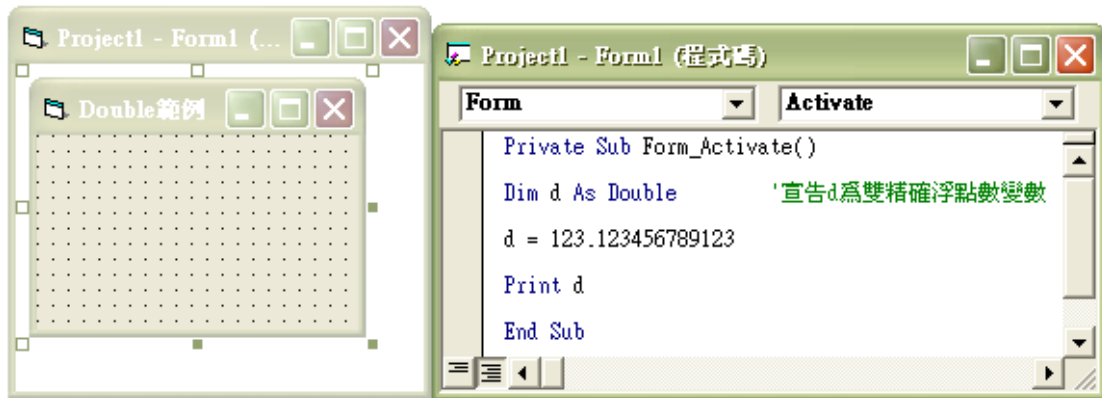


執行結果：



單精確度浮點數變數只到小數點後第 4 位，不過它也會四捨五入唷～

## Double 範例 \* \* \* \* \*

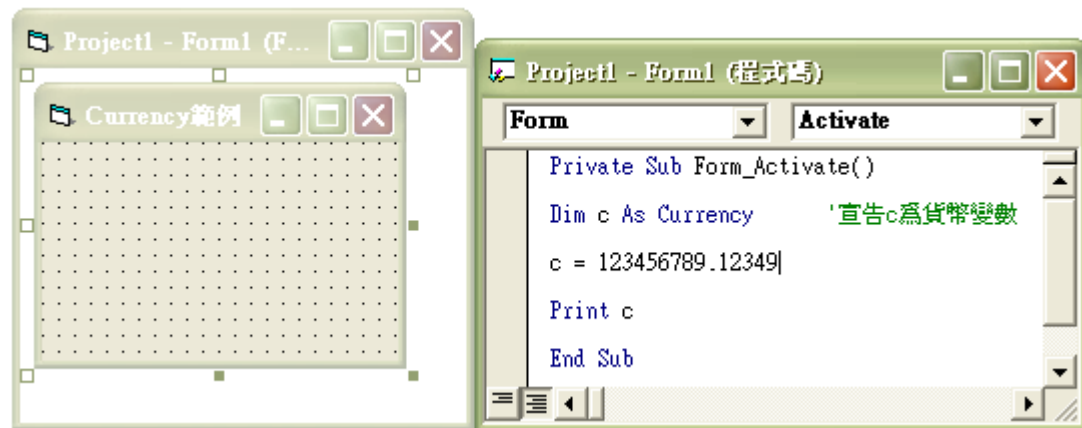


執行結果：



雙精確浮點數變數到小數點後第 12 位，後面的無條件捨去，你可以試試看小數點後再多打幾個數字，會有什麼結果。

## Currency 範例 \* \* \* \* \*



執行結果：

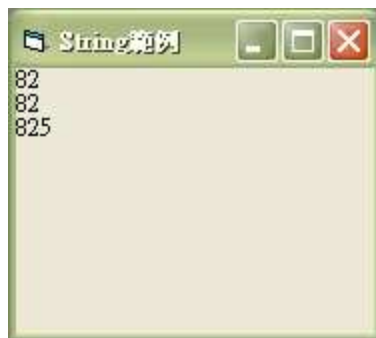


貨幣變數只可處理小數點左邊 15 位，小數點右邊 4 位的實數而已。

## String 變動長度字串變數範例 \* \* \* \* \*

```
Project1 - Form1 (程式碼)
Form Activate
Private Sub Form_Activate()
    Dim str1 As String * 4      '宣告str1為固定長度字串變數，並設定字串長度為4。註：字串長度>0
    Dim str2 As String        '宣告str2為變動長度字串變數
    str1 = "我是野原新之助"   '設定str1值，如果沒有打"雙引號"，字就不會出現哦~
    str2 = "，今年五歲，是個害羞的小男孩"
    Print str1 & str2
    str1 = "我是野原" & "新之助" '重設，這種設定跟前面那種設定是一樣的
    str2 = "，今年五歲，" & "是個害羞的小男孩" '重設
    Print str1 & str2
End Sub
```

執行結果：

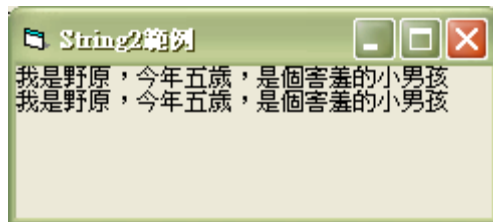


因為是字串，所以前面不會留一小格

## String 固定長度字串變數範例 \* \* \* \* \*

```
Project1 - Form1 (程式碼)
Form Activate
Private Sub Form_Activate()
    Dim str1 As String * 4      '宣告str1為固定長度字串變數，並設定字串長度為4。註：字串長度>0
    Dim str2 As String         '宣告str2為變動長度字串變數
    str1 = "我是野原新之助"    '設定str1值，如果沒有打"雙引號"，字就不會出現哦~
    str2 = "，今年五歲，是個害羞的小男孩"
    Print str1 & str2
    str1 = "我是野原" & "新之助"    '重設，這種設定跟前面那種設定是一樣的
    str2 = "，今年五歲，" & "是個害羞的小男孩"    '重設
    Print str1 & str2
End Sub
```

執行結果：



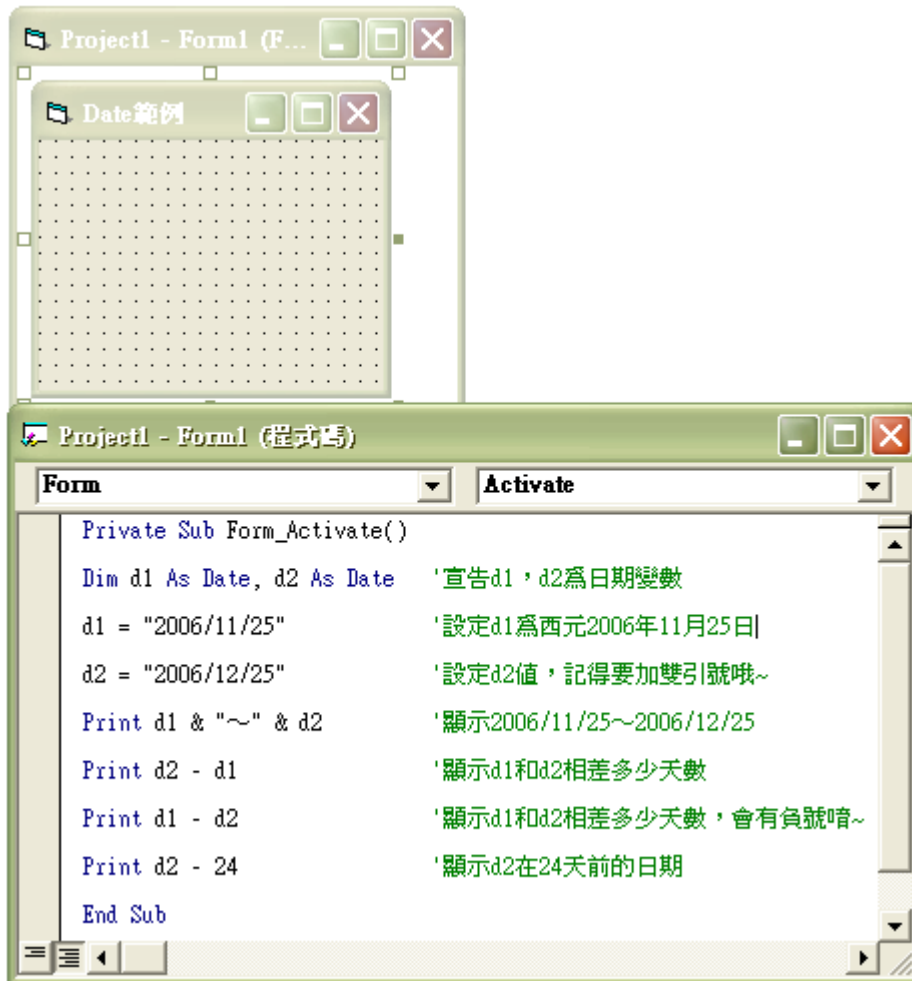
## Boolean 範例 \* \* \* \* \*

```
Command1 Click
Dim A As Boolean
Private Sub Command1_Click()
    A = Not A      '將A的值反相
    If A = False Then    '假如A的值是False(未開始播放)則
        Command1.Caption = "播放"    '將按鈕上顯示『播放』
    Else    '否則
        Command1.Caption = "暫停"    '將按鍵顯示『暫停』
    End If
End Sub
Private Sub Form_Load()
    A = False    '預設A的值是False(未播放)
    Command1.Caption = "播放"    '按鍵上顯示『播放』
End Sub
```



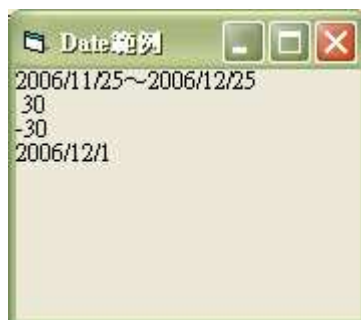
布林值只會有二種狀態

## Date 範例\*\*\*\*\*



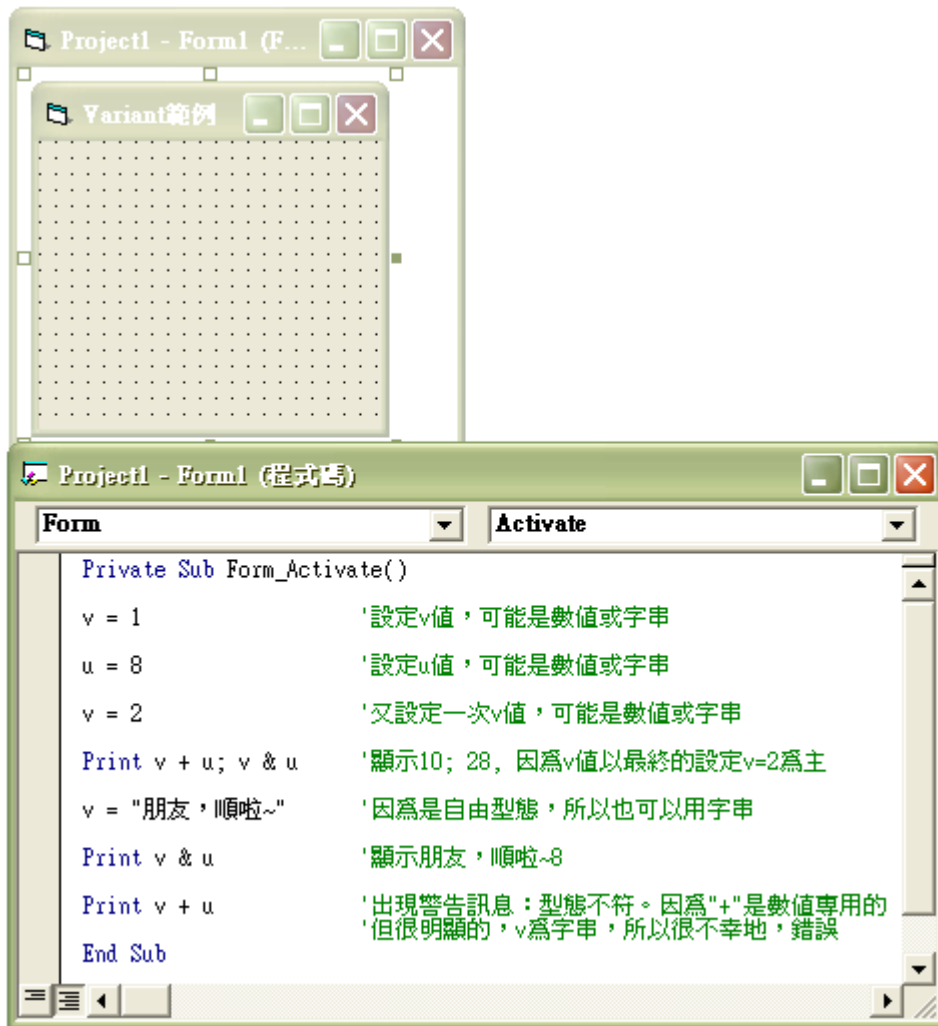
```
Private Sub Form_Activate()  
    Dim d1 As Date, d2 As Date '宣告d1, d2為日期變數  
    d1 = "2006/11/25" '設定d1為西元2006年11月25日  
    d2 = "2006/12/25" '設定d2值, 記得要加雙引號哦~  
    Print d1 & "~" & d2 '顯示2006/11/25~2006/12/25  
    Print d2 - d1 '顯示d1和d2相差多少天數  
    Print d1 - d2 '顯示d1和d2相差多少天數, 會有負號喲~  
    Print d2 - 24 '顯示d2在24天前的日期  
End Sub
```

執行結果：



```
2006/11/25~2006/12/25  
30  
-30  
2006/12/1
```

## Variant 範例 \* \* \* \* \*



執行結果：



因為程式的判斷是由上而下，所以前面沒有錯誤的也會 Print 出來，然後遇到後面的錯誤再出現警告訊息。

Print 方法不只這樣，它在這只能算是冰山一角，正所謂學海無涯，以後我們還會陸續學到偉大的 Print，敬請期待。

資料型態的認知是很重要的。如果用了不適當的資料型態，就會造成溢位或浪費記憶體空間或其他。

何為浪費記憶體空間呢？這還需要解釋嗎？好吧！給你來個誇飾法：過年期間，你把車票全都買下來，別人不就沒得買了。你花那麼多錢買那麼多座位，結果車上客人只有你一個，空著其他一大堆空位，別人都沒得坐，你說，別人會不會罵你！

何為溢位？簡單來說，就是記憶體空間不夠用，資料就滿出來了啦~如是你很堅持的，不管旁人的勸阻，硬是要執行，它就會嚴重地警告你，不要再讓它溢位了。

例：設定位元組變數的值為負數或大於 255，溢位，因為不在它的範圍。

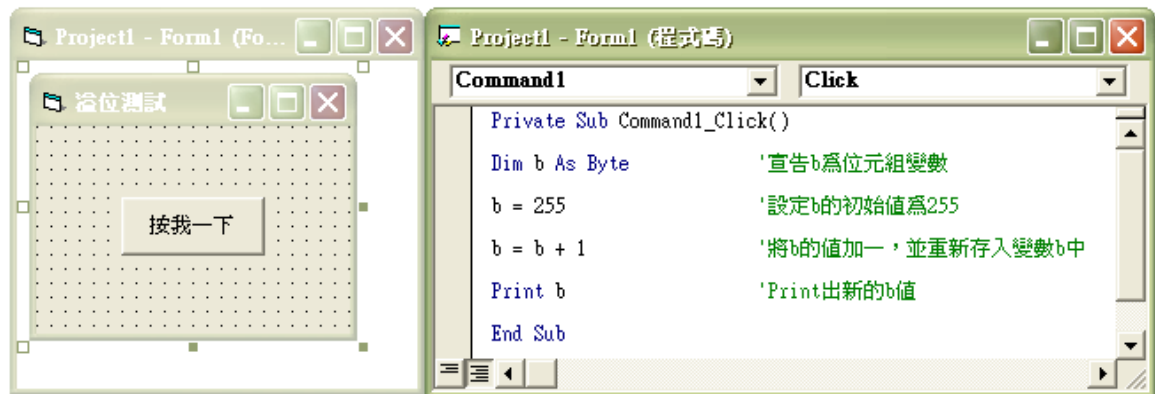
例：如果你只是要做個九九乘法表的程式，卻宣告成自由型態變數，只需要數字 1~81，卻用了超大的記憶體空間，還讓程式浪費時間和精神去判斷它們的變數值是什麼，過份!

例：如果你把字串宣告成數值，它會出現警告訊息：型態不符。



正所謂～動腦不如動手，就讓我們來看一看什麼叫做溢位吧！

下面有一個很 easy 的測試：



執行結果：

按下“按我一下”之後：



按下“偵錯”之後：



因為加一之後就變成 256 了，超過位元組變數的範圍 0~255 了，所以很可惜的，它溢位(overflow)了。

經過我的介紹，相信你們都對溢位有進一步的認識了，接下來大家就可以自己動手 DIY，用自己的方法讓程式溢位。

### 3.2.3 無宣告的後果

在 VB 中，即使我們沒有宣告資料型態，變數仍然可以使用，因為系統會幫你預設了資料型態(Variant)，它可儲存各種型態的變數。

另外，VB 還有一個特性：在使用一個變數之前，**可以不用宣告就直接使用**。例如直接打 num=1 而不宣告，VB 會自動幫你宣告一個資料型態為 Variant 的變數 num。

這特性雖然很方便，但其實它有個**致命的缺點**：如果我們將 num 不小心打錯成 nun，系統就會判斷這是不同的變數名稱，而幫我們再去宣告一個資料型態為 Variant 的變數 nun，如此一來就造成程式的錯誤，而且執行時也許不會出現警告訊息，你就不能馬上知道你哪裡錯誤；**但是**如果你有宣告 num，那麼執行時就會出現警告訊息，你就可以很輕易的知道錯誤在哪，好處當然不只如此，因為 **Variant 所佔用記憶體空間很大，而且還要判斷其變數值的資料型態，就會使效率變差**，所以如果你宣告時用了適當的資料型態，就不會浪費記憶體了。

### 3.2.4 全域、區域變數

★**全域變數**：在程式碼最上面就宣告的一群變數。

**優點** 可被任何副程式共用，甚至做不同應用，因而減少記憶體的佔用。

**缺點** 雖然佔用的記憶體空間較小，但要等到整個程式執行完後才被釋放。除錯(debug)時，你會想哭，因為大家都長的一樣，你很難找出是誰拉的屎(引用自沈雄生老師之名句)。

★**區域變數**：放在副程式中，只可被該副程式利用，若其他副程式也想要享用，那只好在自己的地盤再宣告一次或讓此變數成為全域變數。

**優點** 變數所在的副程式結束後，記憶體空間就可以釋放了，甬等到程式結束。

除錯時，比較容易找出問題在哪。

區域變數和全域變數重複時，區域變數會被優先使用，全域變數的內容則會被隱藏起來，等到副程式結束時，區域變數被釋放，所有因重複而被隱藏的全域變數的內容會被恢復

**缺點** 我還真想不到有啥缺點可說

### 3.3 常數

什麼是常數呢？變數與常數最大的差異為：在程式運作中，其值是否可以被改變。變數可以；常數則是以不變應萬變，當然在執行過程中，佔用記憶體的大小亦不會改變。例：重力加速度  $g$ 、圓周率  $\pi$ 、光速……等等。常數包括：字元常數、字串常數、整數常數、浮點常數。

**Const** 是 Constant 的縮寫，我們都是用 Const 敘述 來宣告常數名稱



**Const 常數名稱 As 資料型態 = 常數值**

**或 Const 常數名稱 = 常數值**

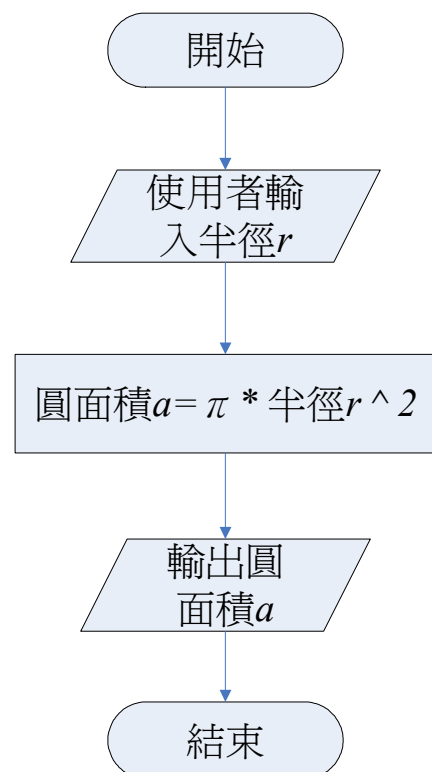
#### Const 範例：

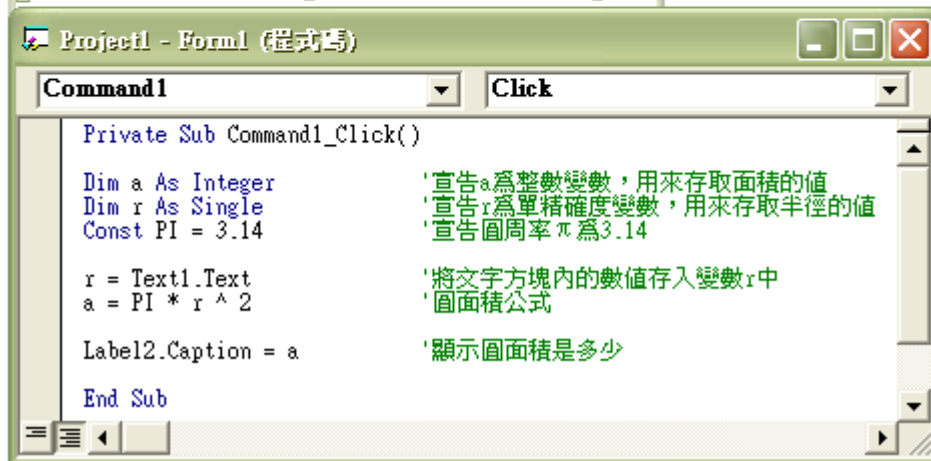
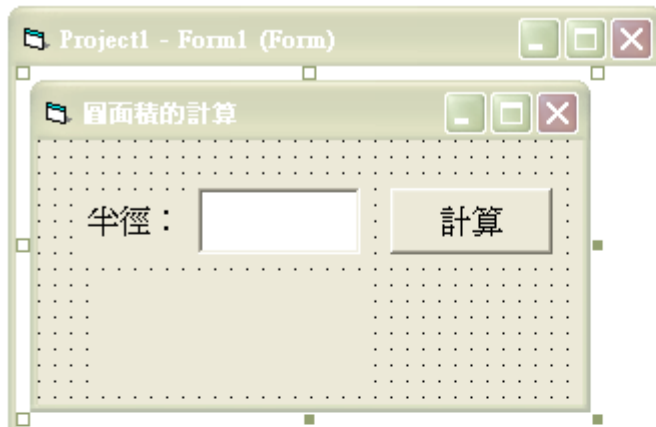
<pre>Const G = 9.8 Const TEST = ""Hi~baby!""</pre>	宣告 G 為浮點數常數 宣告 TEST 為字串常數(如果你想要 print 出雙引號你就要再打兩對雙引號；原本的一對+後來的兩對=三對雙引號)
<pre>Print "地心引力常數 G = " &amp; G Print "花輪最喜歡講" &amp; TEST</pre>	顯示地心引力常數 G = 9.8 顯示花輪最喜歡講"Hi~baby!"

接下來，有個求圓面積的例子給你當個參考，圓周率  $\pi$  是常數 3.14，你想計算的半徑  $r$  是變數（在此我設定  $r$  是可以有小數點的），計算出來的面積  $a$  也是變數（在此我設定  $a$  是整數）。

右邊為此程式的流程圖：

下面為表單及程式碼：





執行後，在文字方塊打上你想計算的半徑

按下按鈕後，就會出現圓面積了

