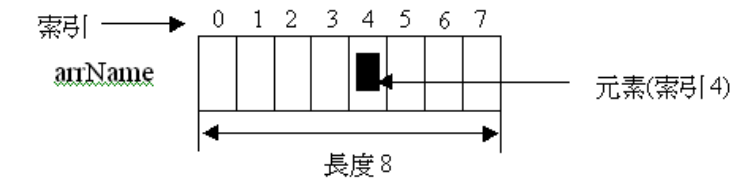
字串與陣列

資訊科技系
林偉川

一維陣列的處理

- 「陣列」（Array）是一種基本的資料結構，它是將相同資料型別的變數集合起來，使用一個名稱代表，然後使用索引值存取變數的值，如下圖所示：



2

宣告一維陣列-宣告

- VB.NET陣列同樣使用【Dim】指令宣告，我們可以在宣告時同時指定陣列的尺寸，一維陣列的宣告語法，如下所示：

  Dim 陣列名稱(最大索引) As 資料型別

- 上述語法宣告名為「陣列名稱」的陣列，元素個數為括號的最大索引數加一，「資料型別」是VB.NET基本資料型別、String資料型別、結構和類別物件。陣列宣告的範例，如下所示：

  Dim arrScore(4) As Integer

  Dim arrName(4) As String

3

宣告一維陣列-初值

- 在宣告陣列時，不指定陣列最大索引，而是直接指定陣列各元素的初值，如下所示：

  Dim arrScore() As Integer = {60, 89, 75, 68, 90}

- 上述宣告的陣列大小是初值的個數，陣列索引的最大值是初值個數減一。當然我們也可以使用指定敘述指定陣列值，如下所示：

  arrScore(0) = 60

  arrScore(1) = 89

  arrScore(2) = 75

  arrScore(3) = 68

  arrScore(4) = 90

4

宣告一維陣列-存取

- 取出陣列值的程式碼，如下所示：

```
Dim score As Integer
score = arrScore(3)
```

- 上述程式碼取得陣列索引3的值，因為索引值從0開始，也就是陣列的第4個元素值68。

5

For Each迴圈存取陣列元素與邊界函數-語法

- VB.NET的For Each迴圈就可以輕鬆走訪整個陣列，其語法如下所示：

```
For Each 變數 In 陣列
  程式區塊
Next
```

- 上述「變數」取得陣列的一個元素，變數需要和陣列屬於相同的資料型別，此迴圈自動從索引0開始，每執行一次迴圈取得一個元素值並且自動移至下一個。

6

For Each迴圈存取陣列元素與邊界函數-範例

- 計算arrScore陣列各元素的總和，如下所示：

```
For Each element In arrScore
    total += element
Next
```

7

For Each迴圈存取陣列元素與邊界函數-函數

- VB.NET提供函數可以取得陣列的邊界，傳入的參數是陣列變數，如下表所示：

函數	說明
LBound(Array)	傳回整數值的陣列最小索引值，因爲 VB.NET 預設的索引是從 0 開始，其傳回值是 0
UBound(Array)	傳回整數值的陣列最大索引值

8

字串的基礎

- VB.NET的字串是String資料型別的變數或字串值，字串是0或多個依序的Unicode字元使用ASCII字碼的雙引號所括起的文字內容，如下所示：

```
Dim str As String = "VB.NET程式設計範例教本"
Dim str1 As String
str1 = "ASP.NET網頁製作徹底研究"
```

9

字串長度與大小寫轉換

函數	說明
Len(Stmt)	傳回整數的字串長度，擁有多少個字元或中文字
UCase(Stmt)	將參數字串或字元的英文字母轉換成大寫
LCase(Stmt)	將參數字串或字元的英文字母轉換成小寫
LTrim(Stmt)	刪除字串開頭的空白字元
RTrim(Stmt)	刪除字串結尾的空白字元
Trim(Stmt)	刪除頭尾兩端的空白字元
Space(num)	傳回參數 num 個空白字元的字串
Asc(Stmt)	傳回參數 Stmt 字串第 1 個字元的 ASCII 碼，例如：Asc("A")爲 65

10

取出子字串與字串反轉

函數	說明
Mid(Stmt, start[, length])	從參數 Stmt 字串的 start 位置（從 1 開始）取出長 length 的子字串，如果沒有 length 參數，就是從 start 位置到字串結尾的所有字元
Left(Stmt, length)	從參數 Stmt 字串從開頭起算共 length 長度的字元，如果爲 0 傳回空字串，如果大於字串長度，傳回整個字串
Right(Stmt, length)	從參數 Stmt 字串從右邊回頭起算共 length 長度的字元，如果爲 0 傳回空字串，如果大於字串長度，傳回整個字串
StrReverse(Stmt)	將參數的 Stmt 字串反轉，例如：VB.NET 成爲 TEN.BV

11

子字串的搜尋與取代

- VB.NET並沒有字串取代函數，如下所示：

```
out = txtInput.Text
strLeft = Mid(out, 1, pos)
strRight = Mid(out, pos + length + 1)
txtInput.Text = strLeft & txtReplace.Text & strRight
```

函數	說明
InStr(start, Stmt1, Stmt2[, Type])	在參數 Stmt1 字串的 start 位置（從 1 開始）開始找尋 Stmt2 字串，如果找到傳回找到的位置，沒有找到傳回 0，Type 是 Option Compare 的 Text 或 Binary 的字串比較方式，如果沒有指定，使用 Option Compare 的設定

12

多維陣列的處理

- 「二維陣列」（Two-dimensional Array）或多維陣列都是一維陣列的擴充，如果將一維陣列想像成一度空間的線，二維陣列就是一個二度空間的平面，三維陣列即空間。

Scores(0,0)=54	Scores(0,1)=68	Scores(0,2)=93
Scores(1,0)=67	Scores(1,1)=78	Scores(1,2)=89

13

多維陣列的處理-範例

- 學生成績的二維陣列，如下所示：

  Dim Scores(,) As Integer={{ 54, 68, 93 }, { 67, 78 , 89} }

- 上述程式碼宣告二維陣列Scores，並且指定元素值，在第一維共有2個元素，每一個元素是一位學生成績的一維陣列{ 54, 68, 93 }和{ 67, 78, 89 }，各擁有3個元素，分別是國文、數學和英文成績。

14

多維陣列的處理-走訪

- 在設定好陣列值後，可以使用巢狀迴圈存取二維陣列，如下所示：

```
For i = 0 To 1
   For j = 0 To 2
    Sums(i) += Scores(i, j)
   Next
Next
```

15

動態陣列與參數傳遞

- VB.NET陣列可以動態調整其大小，換句話說，如果當初宣告的陣列不夠用時，可以擴充陣列大小，如果太大時，可以縮小尺寸，幫助您有效的進行記憶體管理。
- 因為字串與陣列都屬於參考型別，就算使用ByVal傳值方式傳遞到函數，函數仍然會更改變數值。

16

動態陣列

- VB.NET的陣列可以使用【ReDim】指令在程式執行時重新調整大小，不過ReDim指令並不能宣告陣列，只能更改已經宣告的陣列尺寸，如下所示：

```
Dim arrSize(4) As Integer
  ……
ReDim arrSize(6)
```

- 上述程式碼將原來的arrSize陣列大小從4改為6。
- Preserve關鍵字保留原陣列的內容，如下所示：

```
ReDim Preserve arrSize(6)
```

17

傳遞字串與陣列參數

資料型別	ByVal 方式	ByRef 方式
數值型別	無法變更變數和成員	可以變更變數和成員
參考型別	程序無法變更變數，但可以變更成員	可以變更變數和成員

18

陣列的排序與搜尋

- 「排序」(Sorting)和「搜尋」(Searching)在計算機科學屬於資料結構與演算法的範疇，事實上，電腦有相當多的執行時間都是在處理資料的排序和搜尋，排序和搜尋實際應用在資料庫系統、編譯程式和作業系統之中。

19

陣列的排序

- 排序的工作是將一些資料依照特定的原則排列成遞增或遞減的順序。例如：整數陣列Data的內容，如下所示：

  Data(0)=89 Date(1)=34 Date(2)=78 Date(3)=45
- 上述陣列以整數大小將陣列內容依遞增的順序排序，排序的結果如下所示：

  Data(0)=34 Date(1)=45 Date(2)=78 Date(3)=89
- 上述陣列Data已經排序，其大小順序如下所示：Data(0) < Data(1) < Data(2) < Data(3)

20

陣列的排序-泡沫排序法

- 泡沫排序法使用交換方式進行排序，將較小的元素逐漸搬移到陣列的開始，將較大的元素慢慢的浮往陣列的最後，如同水缸中的泡沫，慢慢往上浮，故稱為泡沫排序法。
- 原來的陣列內容，如下所示：

  Data(0)=89 Date(1)=34 Date(2)=78 Date(3)=45
- 交換陣列元素，如下所示：

  Data(0)=89 > Date(1)=34 => Data(0)=34 Date(1)=89 交換

  Data(1)=89 > Date(2)=78 => Data(1)=78 Date(2)=89 交換

  Data(2)=89 > Date(3)=45 => Data(2)=45 Date(3)=89 交換

21

陣列的搜尋

- 搜尋是在資料中找出是否存在與鍵值相同的資料，如果資料存在，就進行後續的資料處理。
- 搜尋方法依照搜尋的資料分為兩種，如下所示：
  - **沒有排序的資料**：針對沒有排序的資料執行搜尋，我們需要從資料內的第1個元素開始比較，從頭到尾以確認資料是否存在。
  - **已經排序的資料**：搜尋就不需要從頭開始一個個的比較。例如：在電話簿找電話，相信沒有人是從電話簿的第一頁開始找，而是直接從姓名出現的頁數開始找，這是因為電話簿已經依照姓名排序好了。

22

陣列的搜尋-線性搜尋法

- 線性搜尋法是從陣列的第1個元素開始走訪整個陣列，然後一個一個比較是否擁有搜尋的鍵值，因為需要走訪整個陣列，所以陣列資料是否排序都無所謂。

23

陣列的搜尋-二元搜尋法

- 二元搜尋法屬於一種分割資料的搜尋方法，搜尋的資料需要是已經排序的資料，二元搜尋法先檢查排序資料的中間元素，如果等於鍵值就是找到，如果小於鍵值，表示資料是在前半段，否則在後半段。
- 例如：Data陣列索引的上下範圍分別是low和high，中間元素mid是(low + high)\2。在執行二元搜尋時可以分成三種情況，如下所示：
  - **搜尋鍵值小於陣列的中間元素：**鍵值在資料陣列的前半部。
  - **搜尋鍵值大於陣列的中間元素：**鍵值在資料陣列的後半部。
  - **搜尋鍵值等於陣列的中間元素：**找到搜尋的鍵值。

24

作業

- 以陣列顯示出一個數的各個數字切割後的結果

25