Indexing: Just More Of The Same

Giving a unique name to things complicates referring to them. Indexing simplifies reference by allowing a single base name to be “specialized” by a number.

Indexing, The Basic Idea

- Motivation: When there is a large number of similar things that must be referenced and manipulated, it can be inconvenient to think up a unique name for each, and to refer to them by the name.
  
  + For example: Each day in May could have been given a different name, but these would be hard to remember and inconvenient to use.

- Indexing names the items by associating a base name and a number -- the index -- with each one.

- Computer notation: May(5)

  Streets and Avenues
  Superbowls
  Congresses
Indexing Particulars

- In everyday indexing, it is common to begin the indexing with 1, e.g. May 1, SuperBowl I, etc.
- The number at which indexing begins is known as the origin
- Many computer languages use 1 as the origin, but many others, including Visual Basic 6.0, use 0 as the index origin

Arrays

- A collection of like things named using indexing is called an array
- Arrays are used for representing collections of data values, e.g. integers, strings, etc.
  
  For example:
  ```
  colorList(0) = "red"
  colorList(1) = "white"
  colorList(2) = "blue"
  ```
- The elements of an array must all be of the same type
- The index of an array element is also known as a subscript
Arrays In VB6.0

- Arrays are declared like any other variable using a Dim statement

\[
\text{Dim colorList(2) As String}
\]

- Notice
  - The syntax is just like a normal declaration except for the parenthesis pair
  - In the parentheses is the largest desired index
  - The total number of elements of the array will be one more than the largest index, since the origin is 0
  - The type applies to all of the elements

Indexing Arrays

- To refer to different elements of the array, it is necessary only to change the index ...

\[
\begin{align*}
&\text{Form1.Line (100, 100) – (500, 200), QBColorByName(colorList(0)), B} \\
&\text{Form1.Line (100, 200) – (500, 300), QBColorByName(colorList(1)), B} \\
&\text{Form1.Line (100, 300) – (500, 400), QBColorByName(colorList(2)), B}
\end{align*}
\]

- The index value must be an integer constant (1), a variable (\textit{myNdex}) or expression (\textit{myNdex}+1)

\[
\begin{align*}
&\text{colorList(0) = “red”} \\
&\text{colorList(1) = “white”} \\
&\text{colorList(2) = “blue”}
\end{align*}
\]
Combining Indexing, Arrays, Loops

- To sweep through the elements of an array, use a loop

\[
\text{For } i = 0 \text{ To } 2 \\
\quad \text{Form1.Line (100, i*100+200)-(500, i*100+300),} \\
\quad \text{QBColorByName(colorList(i)),B} \\
\text{Next } i
\]

- A common error is to index beyond the end of the array ...