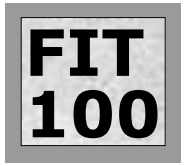
Indexing -- Just More Of The Same



A common way to refer to many instances of the same thing is to give them a single name and index them. So we have Super Bowl XX, Pope John 23, Taco Bell Franchise 229, etc. Indexing is handy in programming

FIT 100

Indexing, The Basic Idea

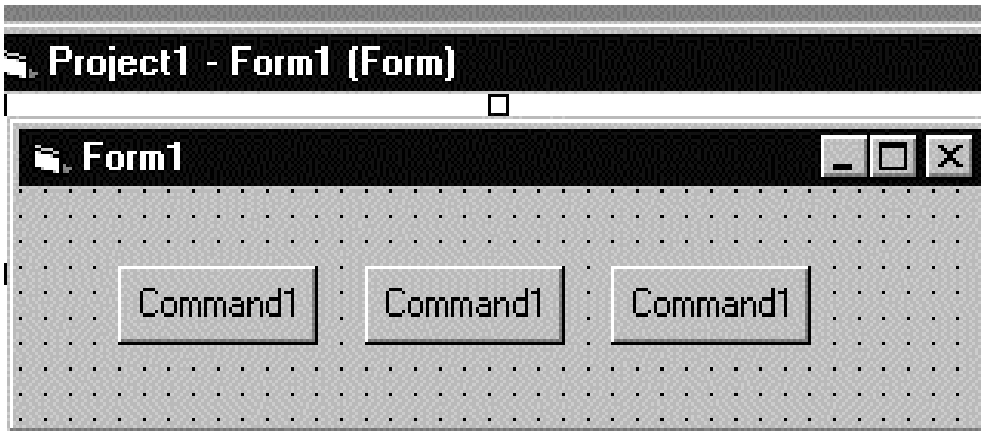
- Motivation: When there is a large number of similar things that must be referenced and manipulated, it can be inconvenient to think up a unique name for each, and to refer to them by the name
  - For example: Each of the Seven Dwarfs has a name, but who can remember them? Also, it is difficult to refer to them in a loop since there is no way to enumerate them

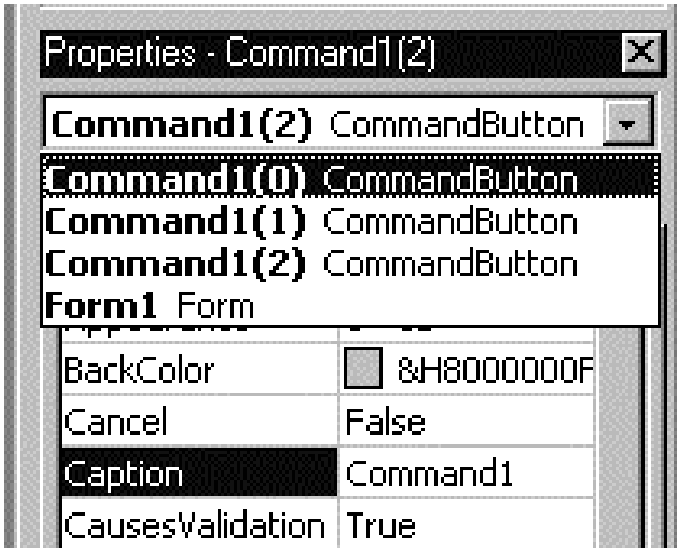
7 Dwarfs
Sneezy
Dopey
Sleepy
Grumpy
Happy
Doc
Bashful

- Indexing names the items by associating a base name and a number -- the index -- with each one
- Computer notation: Dwarf(5) $\Leftrightarrow$ Happy

FIT 100 Indexing Particulars

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





FIT 100 Arrays

- When a variable is indexed it is called an *array*
- Arrays are used for representing collections of data values, e.g. integers, strings, etc.

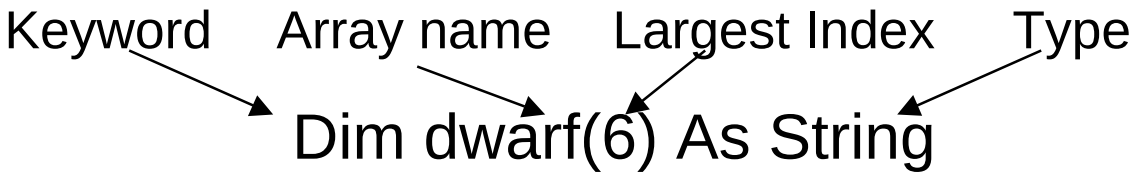
  For example:
  ```
  dwarf(0) = "Sneezy"
  dwarf(1) = "Dopey"
  dwarf(2) = "Grumpy"
  ...
  ```
- Elements of an array must all be of the same type
- The index of an array element is also known as a *subscript*

Notice x0 and x1 are variable names, while x(0) and x(1) are different elements of array x

FIT 100

Arrays In VB6.0

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

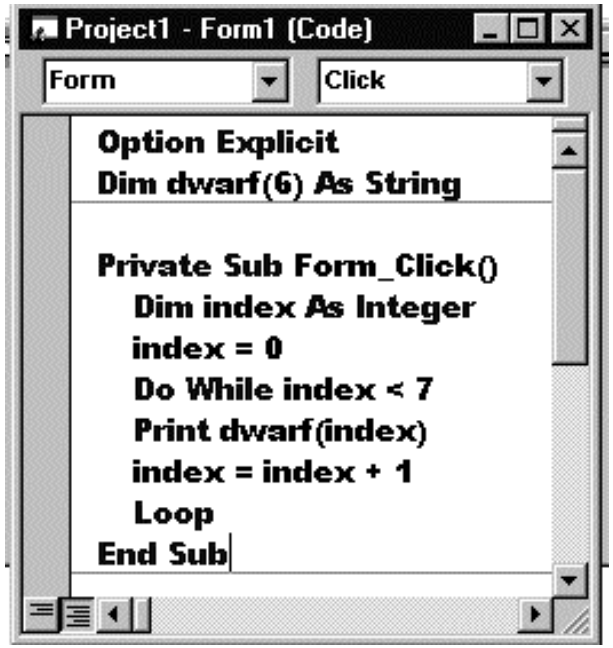


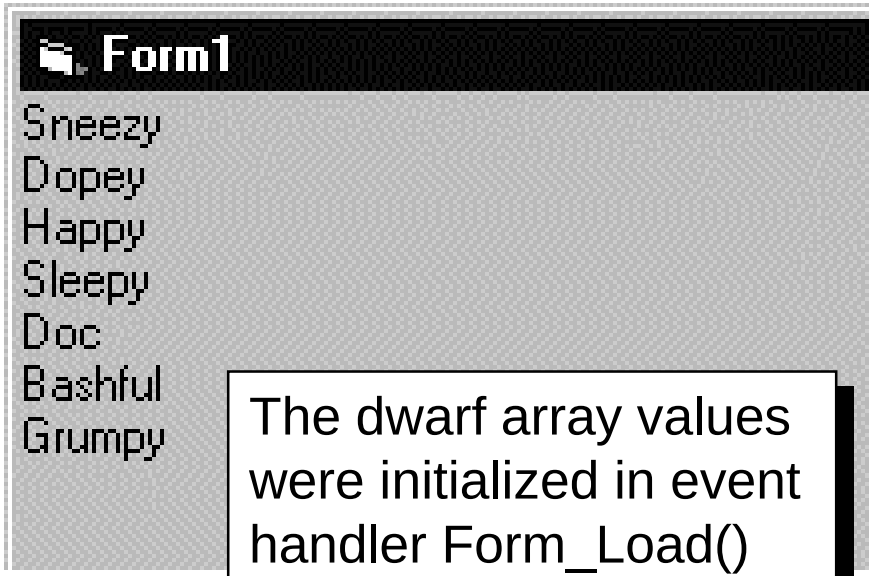
- Notice
  - The syntax is just like a normal declaration except for the parenthesis pair
  - In the parentheses is the largest desired index
  - The total number of elements of the array will be one more than the largest index, since the origin is 0
  - The type applies to all of the elements

FIT 100

Indexing Arrays

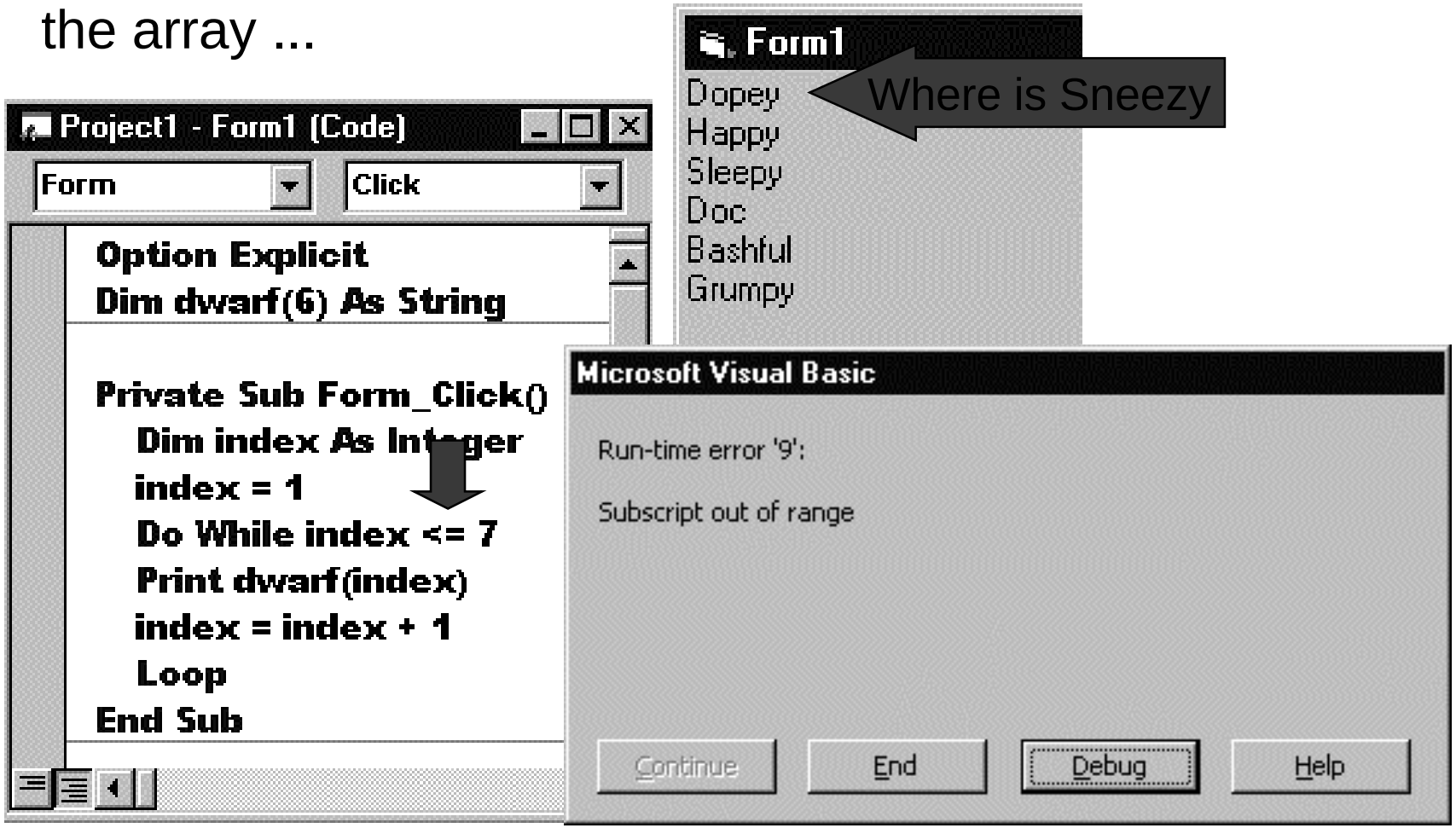
- To refer to different elements of the array, it is necessary only to change the index …
- The index value must be an integer constant (**1**), a variable (**myNdex**) or expression (**myNdex+1**)
- A loop can sweep through all elements





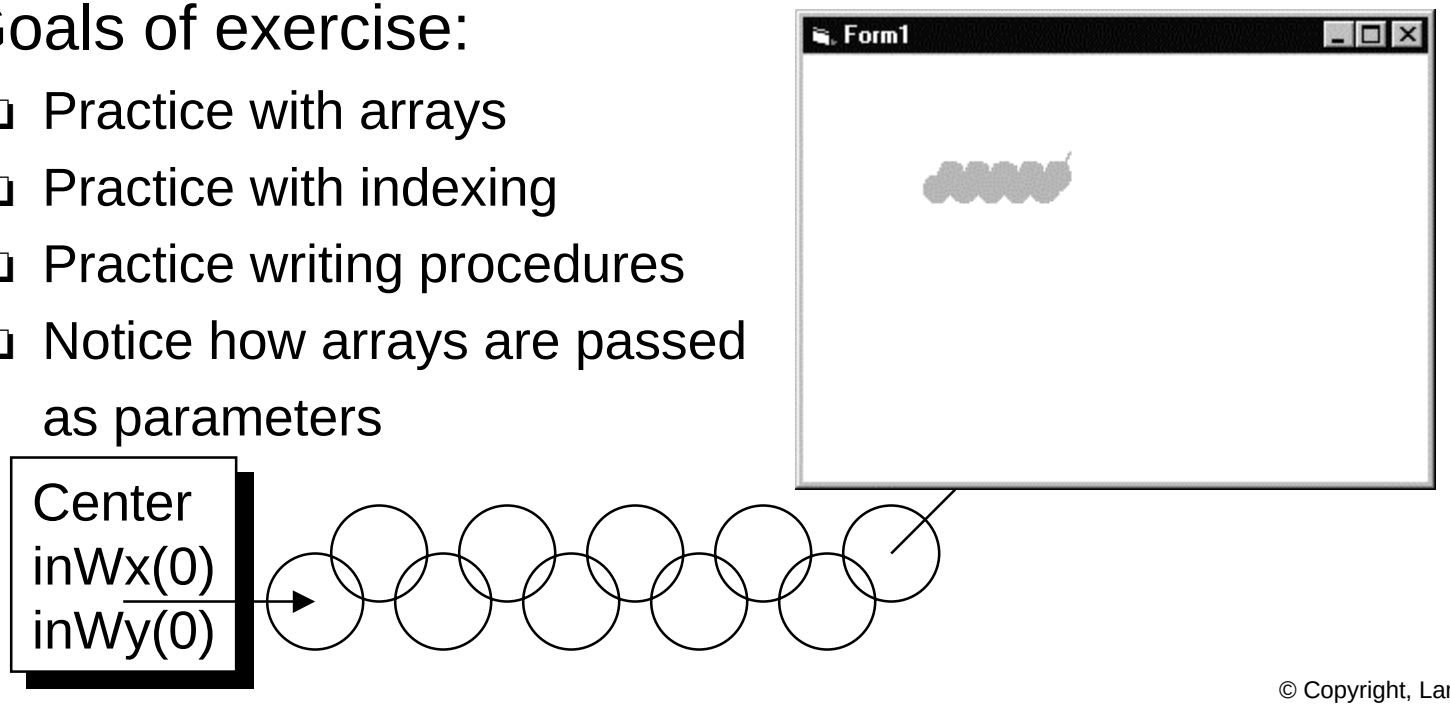
FIT 100 Combining Indexing, Arrays, Loops

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



FIT 100 Practice Using Arrays

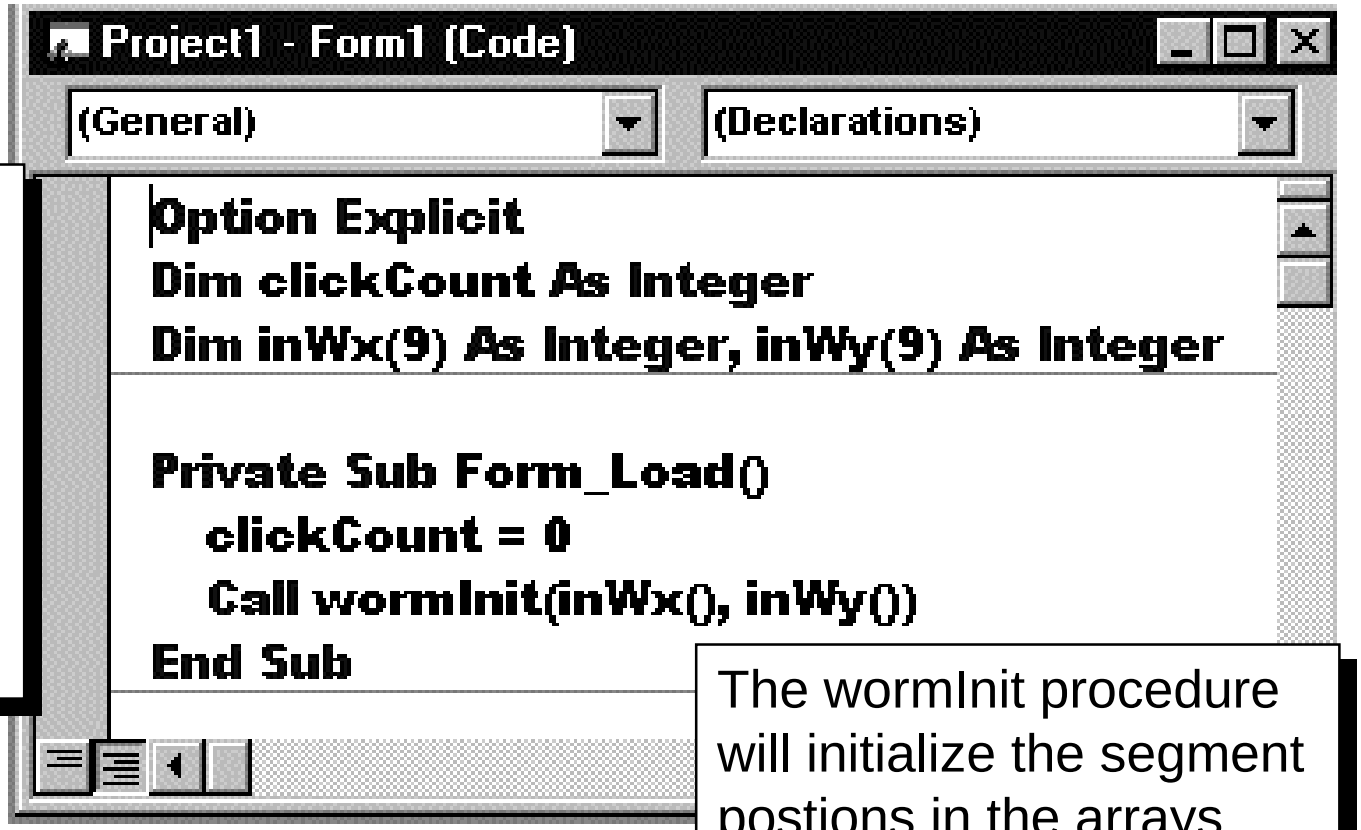
- Draw a 10-segment “inch worm” on the screen and move it forward
- Use arrays to keep the positions of the segments
- Write procedures to initialize worm and draw it
- Goals of exercise:
  - Practice with arrays
  - Practice with indexing
  - Practice writing procedures
  - Notice how arrays are passed as parameters



FIT 100

Programming Drawing -- Step 1

- The first step is to declare the arrays and variables
- Will use the form click event handler to control the operation

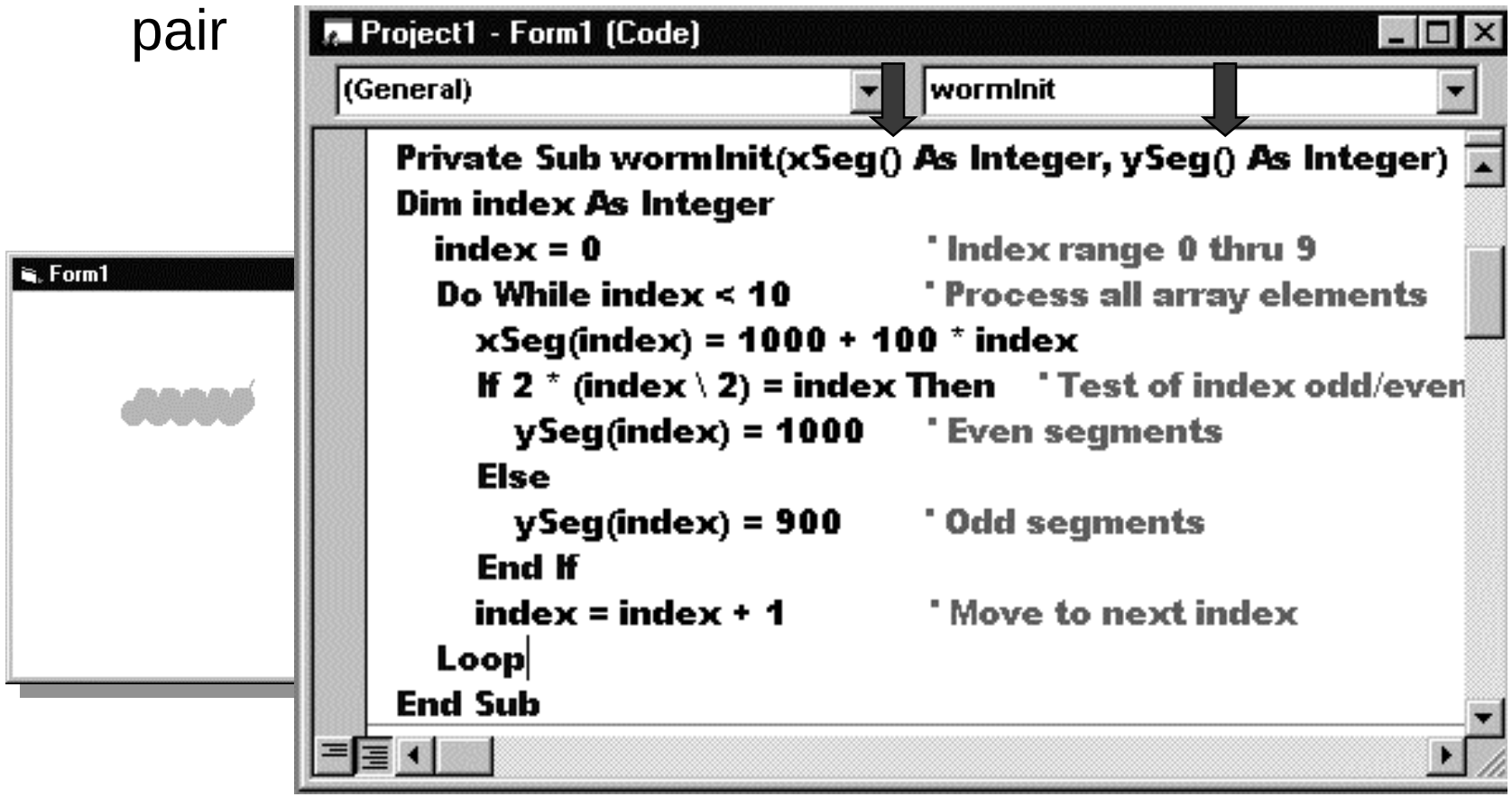


The array inWx will store the x-coordinates for the segments, and inWy will store the y values There are 10 segments

The wormInit procedure will initialize the segment postions in the arrays

FIT 100 Step 2, Initialize The Coordinates

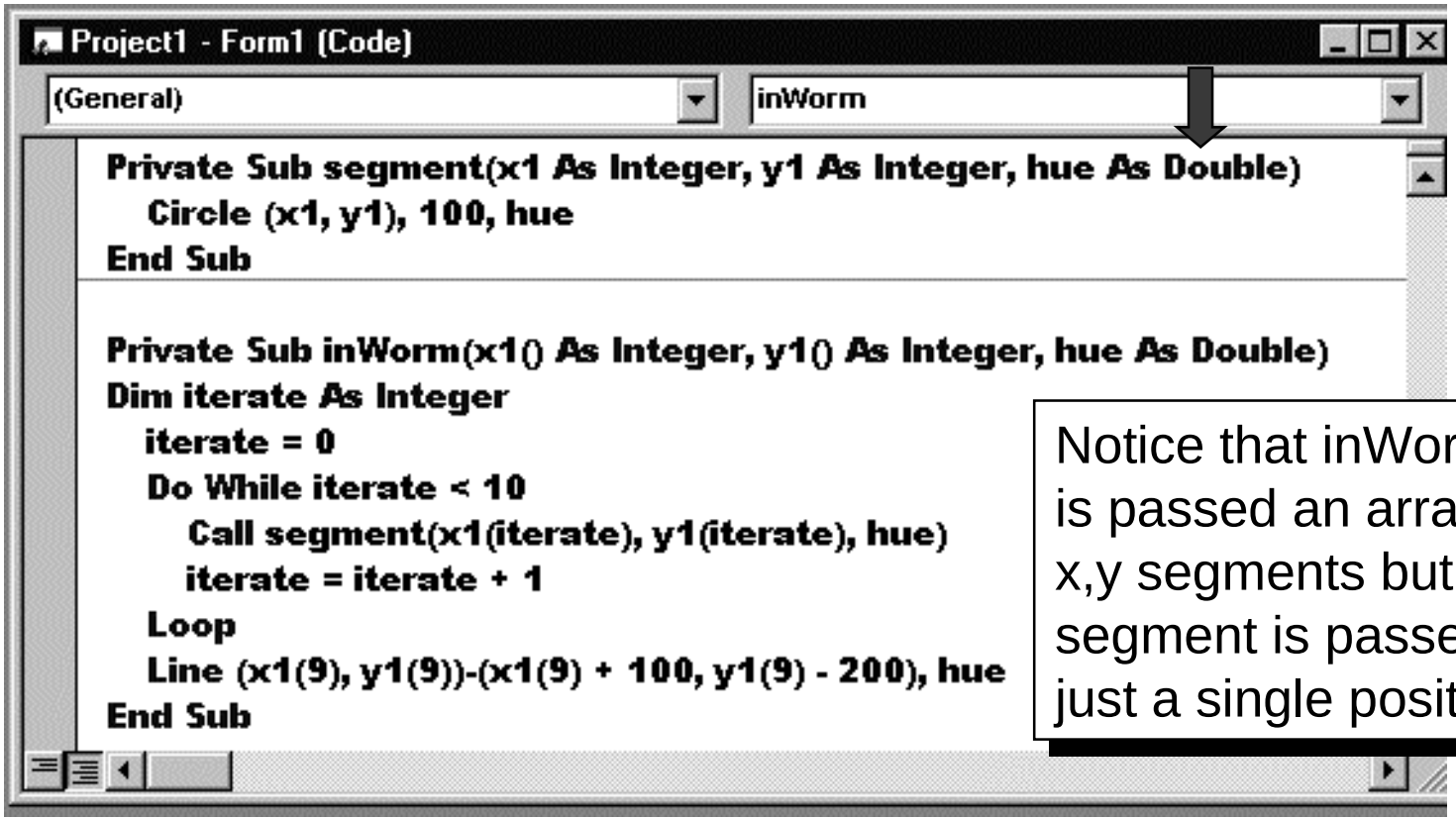
- When arrays are passed to procedures, the formal parameter must show this with an empty parenthesis pair



FIT 100

Step 3 -- Draw The Figure

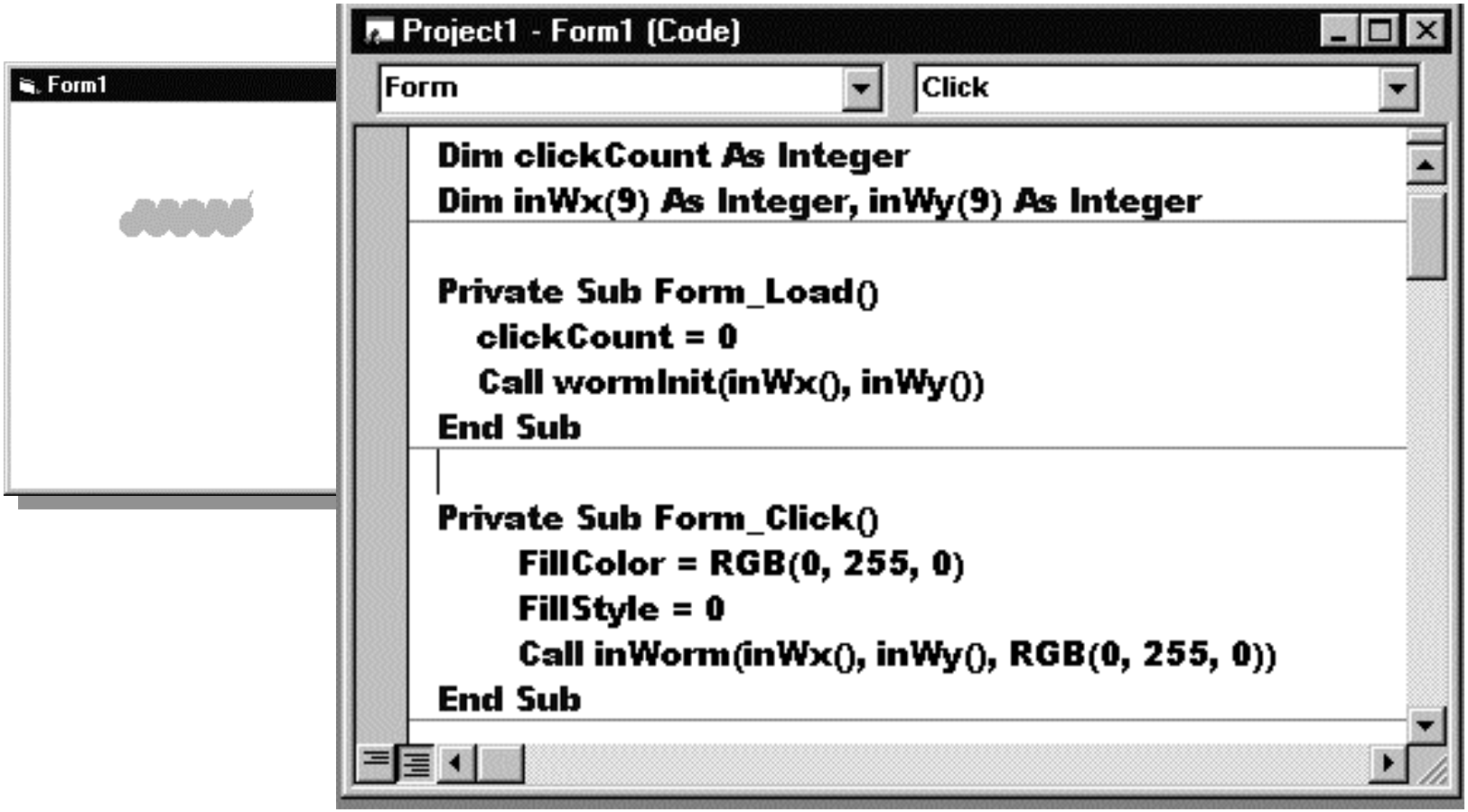
- It will be necessary to draw the inch worm in different colors, so the color becomes a parameter -- Double



Notice that inWorm is passed an array of x,y segments but segment is passed just a single position

FIT 100

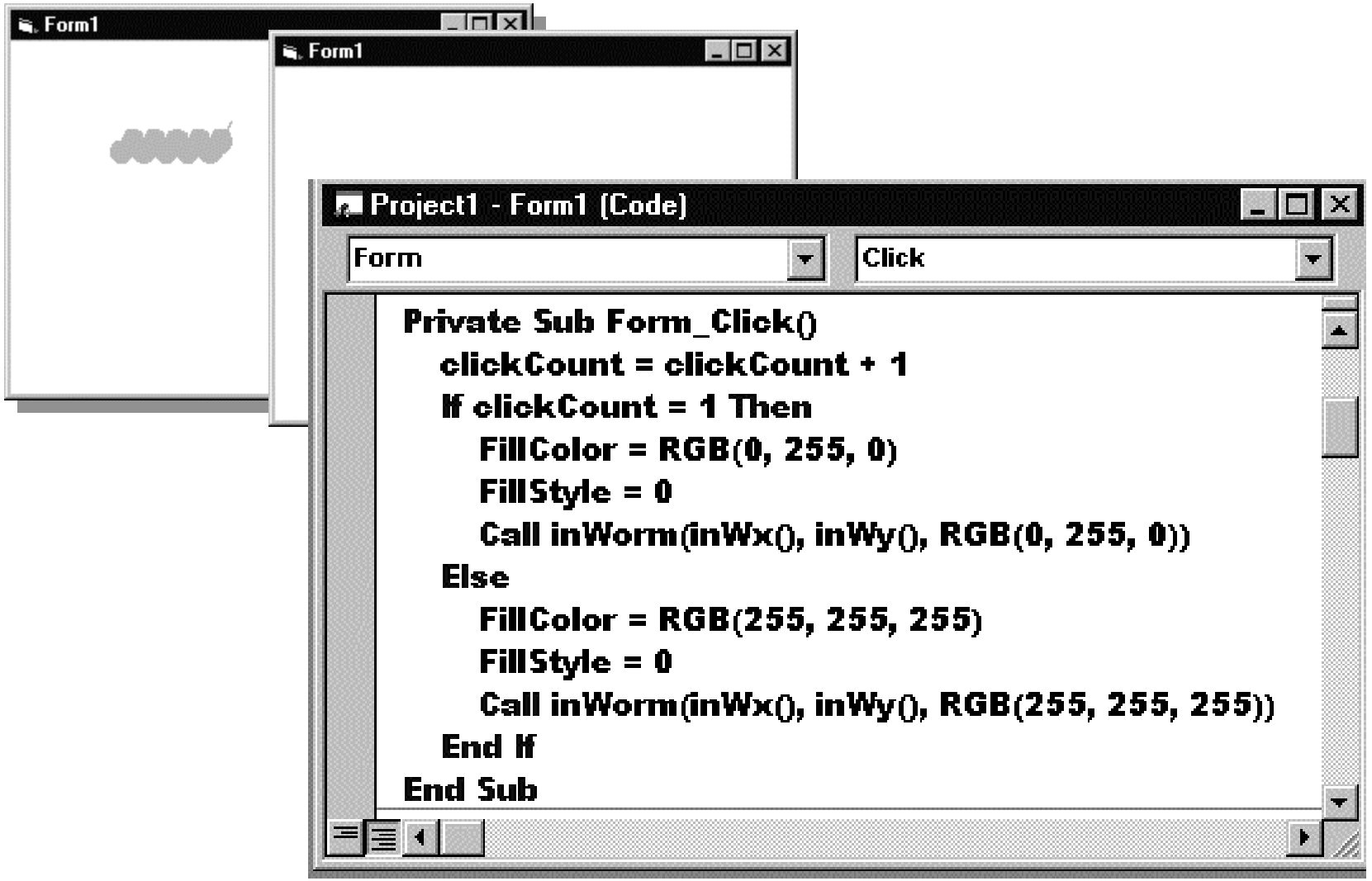
Draw On Click



- To make the worm move, it must be erased/redrawn

FIT 100

Click To Draw, Click To Erase



FIT 100

Move Worm With Repeated Drawing