A. Arrays

1D and 2D Arrays

Creating an Array Control and Constant

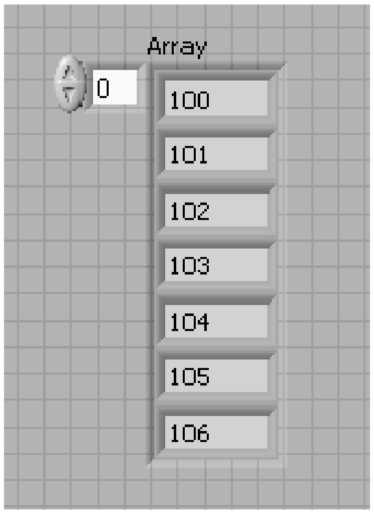
Initializing Arrays

Arrays

An array:

- Is a collection of data elements that are of same type.
- Has one or more dimensions.
- Contains up to $(2^{31})–1$ elements per dimension, memory permitting.
- Accesses elements by its index.

  Note: The first element is index 0.



1 ni.com/training

Arrays are lists of elements of the same data type. They are analogous to arrays in traditional languages.

Arrays can have one or more dimensions.

Arrays can have up to 2^31 elements per dimension. Actual array sizes that students can create is limited by memory.

Elements are accessed by an index. The index ranges from 0 to N-1 (N = number of elements in the array).

Arrays are zero-indexed (first element is zero) in each dimension.

A 2D array is analogous to a spreadsheet or table.

Example: If your data contains temperature readings and time stamps, one column is time values and the other column is readings.

Be careful to specify the element you really want. Example: The first element in an array is array (0), not array(1).

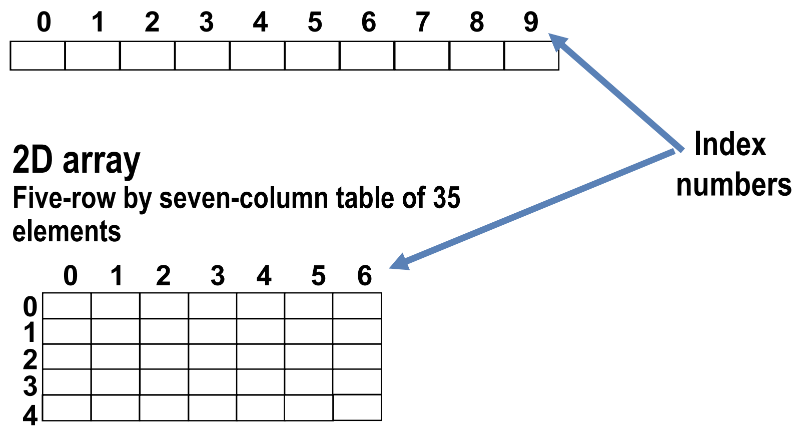
Arrays – 1D and 2D Examples

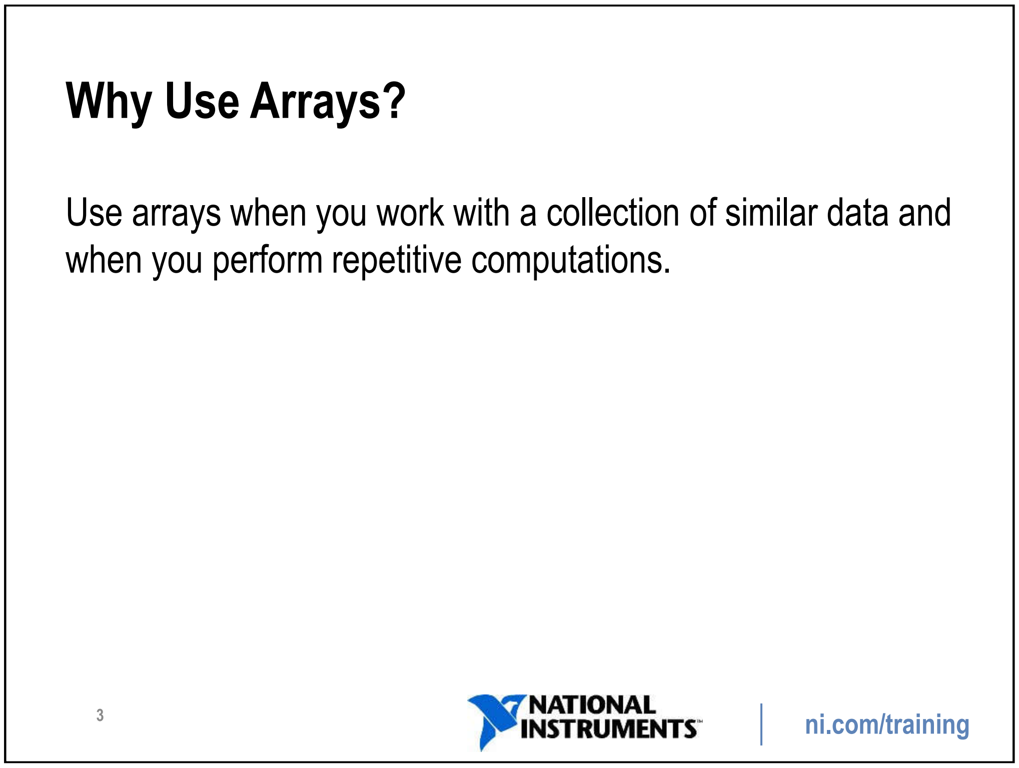
ID array

One row of 10-elements

2D array

Five-row by seven-column table of 35 elements





Why Use Arrays?

Use arrays when you work with a collection of similar data and when you perform repetitive computations.

3

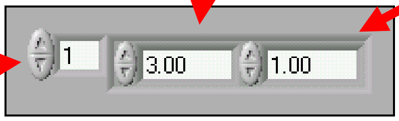
ni.com/training

Viewing Arrays on the Front Panel

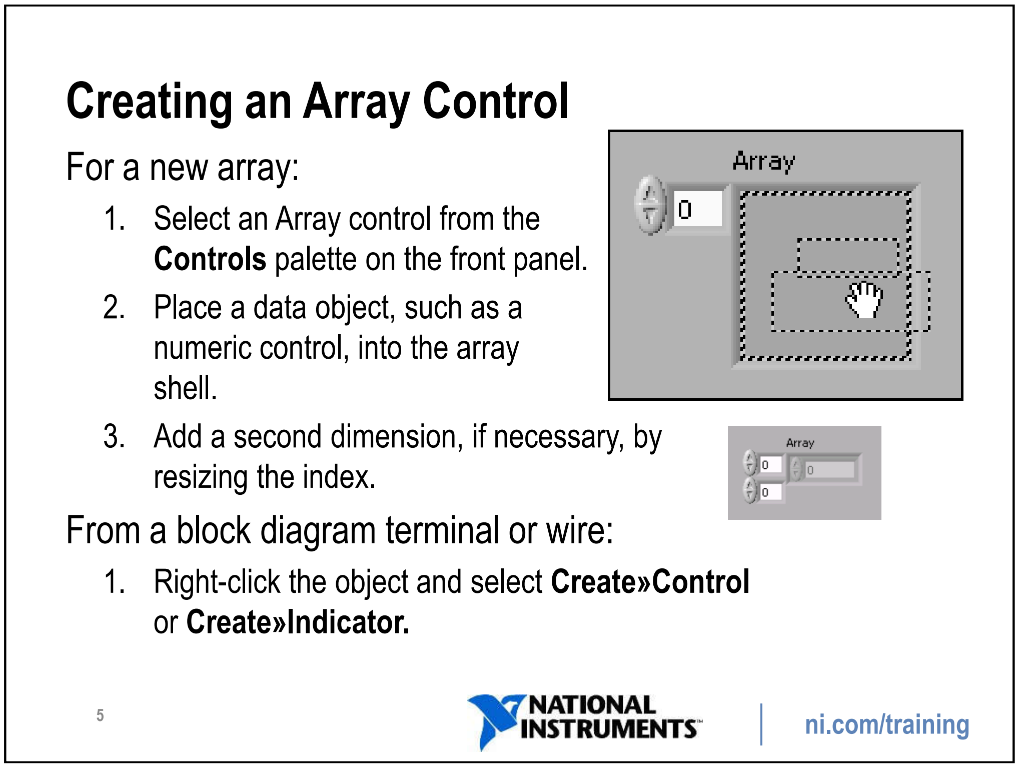
The elements at index 0 are not shown because element 1 is selected in the index display.

First element at index 1

Second element at index 2



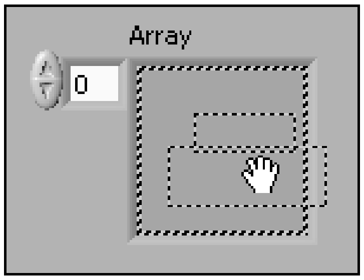
The element selected in the index display always refers to the element shown in the upper-left corner of the element display.



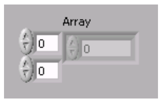
Creating an Array Control

For a new array:

1. Select an Array control from the **Controls** palette on the front panel.
2. Place a data object, such as a numeric control, into the array shell.



3. Add a second dimension, if necessary, by resizing the index.



From a block diagram terminal or wire:

1. Right-click the object and select **Create»Control** or **Create»Indicator.**

5 NATIONAL INSTRUMENTS | ni.com/training

Creating array controls in LabVIEW:

•You can place any data type in an array shell except an array.

• You cannot have an array of arrays; use a 2D array instead.

•Emphasize that this is a two-step process. Students often place only empty array shells on the front panel. Remind them they

• must place a data type inside the array shell.

Demonstrate the following on your computer:

Create a numeric array.

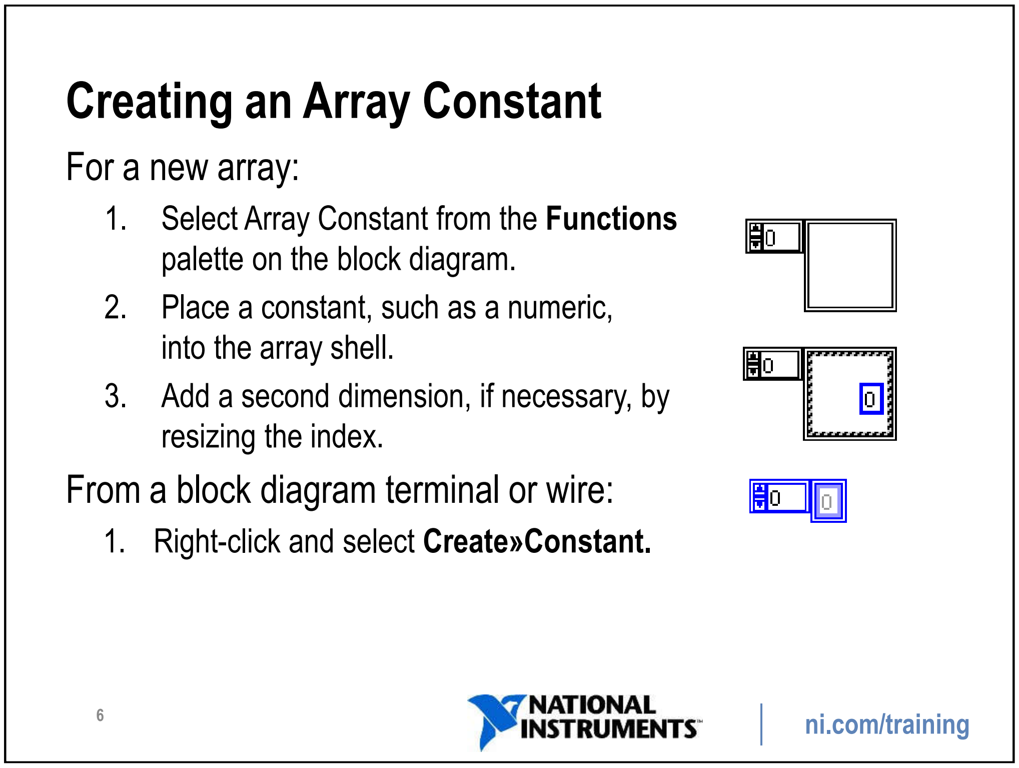
Point out index and data object components.

Show how to create a 2D array.

Show how to display multiple array elements.

Show that index elements always reference the upper-leftmost object in the array display.

Show how elements in an array are initially grayed out, indicating that a portion of the array has not been defined.



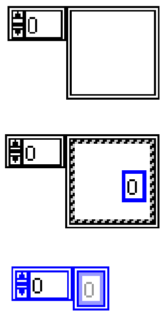
Creating an Array Constant

For a new array:

1. Select Array Constant from the **Functions** palette on the block diagram.
2. Place a constant, such as a numeric, into the array shell.
3. Add a second dimension, if necessary, by resizing the index.

From a block diagram terminal or wire:

1. Right-click and select **Create»Constant.**



6

ni.com/training

Creating array constants uses a two-step process.

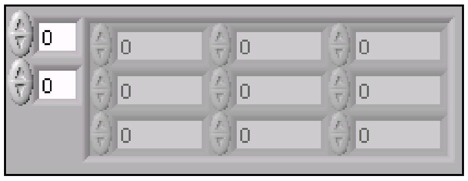
1. Select Array Constant from the **Functions** palette.
2. Place a constant data type inside the empty array shell.

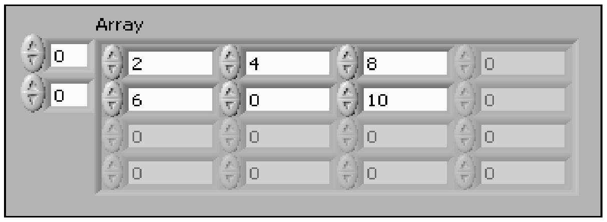
Note again that the data types are grayed-out and must be manually defined if the user wants to store values in the array

constant.

Initializing Arrays

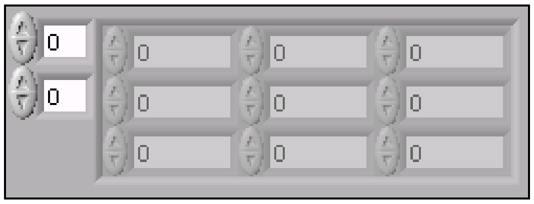
- You can initialize an array or leave it uninitialized.
- For initialized arrays, you define the number of elements in each dimension and the contents of each element.
- Uninitialized arrays have dimension but no elements.





ni.com/training

2D Arrays



- 2D arrays:
  - Store elements in a grid.
  - Require a column index and a row index to locate an element, both of which are zero-based.
- Create a multidimensional array on the front panel by right-clicking the index display and selecting **Add Dimension** from the shortcut menu.
- Resize the index display until you have as many dimensions as you want.

NATIONAL INSTRUMENTS™ | ni.com/training

B. Common Array Functions

Array Size

Initialize Array

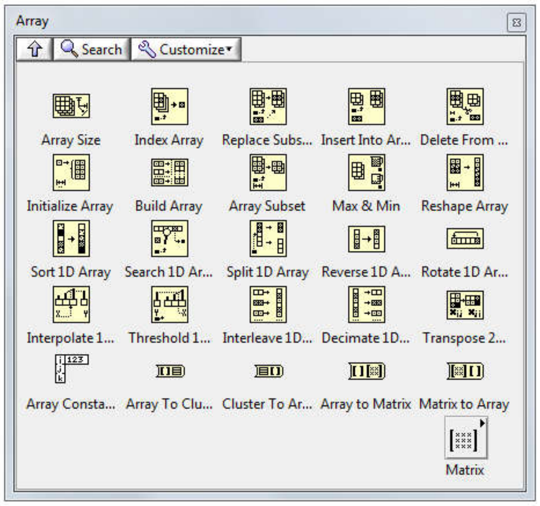
Array Subset

Build Array

Index Array

Common Array Functions

- Array Size
- Initialize Array
- Array Subset
- Build Array
- Index Array



NATIONAL INSTRUMENTS | ni.com/training

LabVIEW has many functions to manipulate arrays. These functions are located on the **Array** subpalette of the **Functions** palette.

Instructor:

Choose one of the following approaches to help students understand how to use these common array functions.

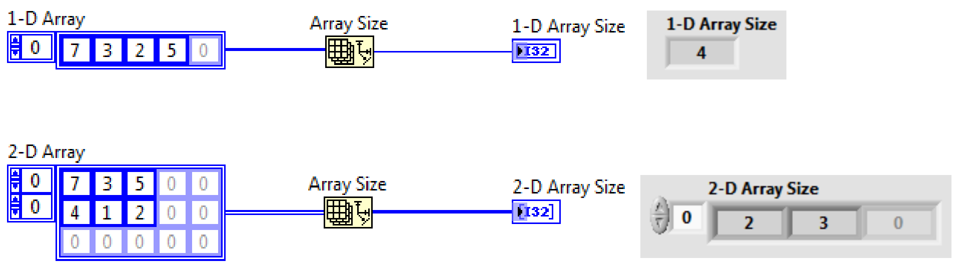
-Walk through the following slides.

-Open the associated demo VIs in the <Exercises>\Demonstrations\Arrays and Polymorphism\ folder.

-Create demos on the fly.

-Show related shipping examples. For example, Build Array.vi has more combinations than shown in the following slides.

Array Size

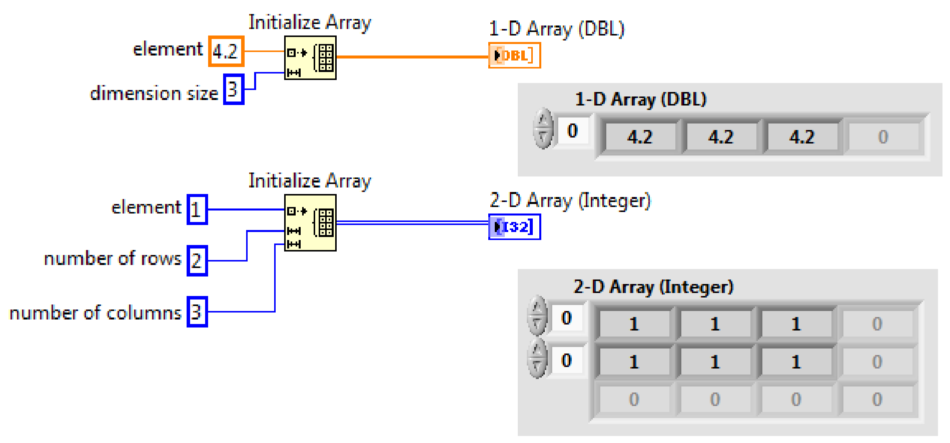


The Array Size function returns the number of elements in the array.

1D array:
Output is a numeric.

Multidimensional array:
Array output has elements signifying size of each dimension. For example: A 2 x 4 array will output an array of two elements. The first element = 2 and second element = 4.

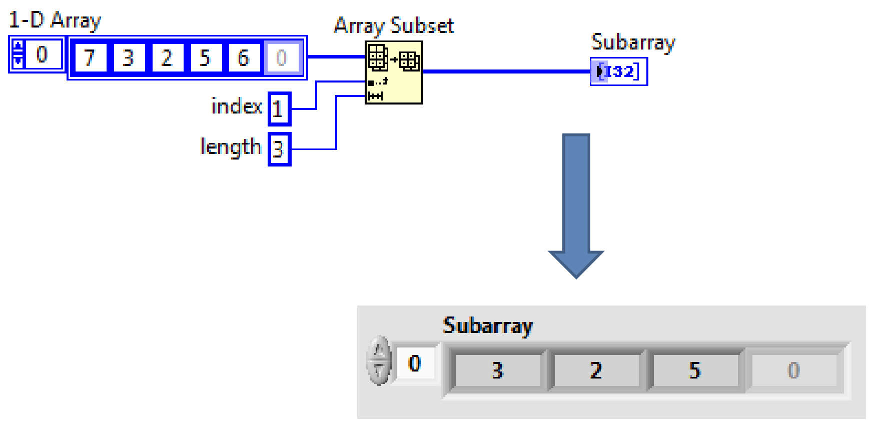
Initialize Array



The Initialize Array function creates an n-dimensional array in which every element is initialized to the value of **element.**

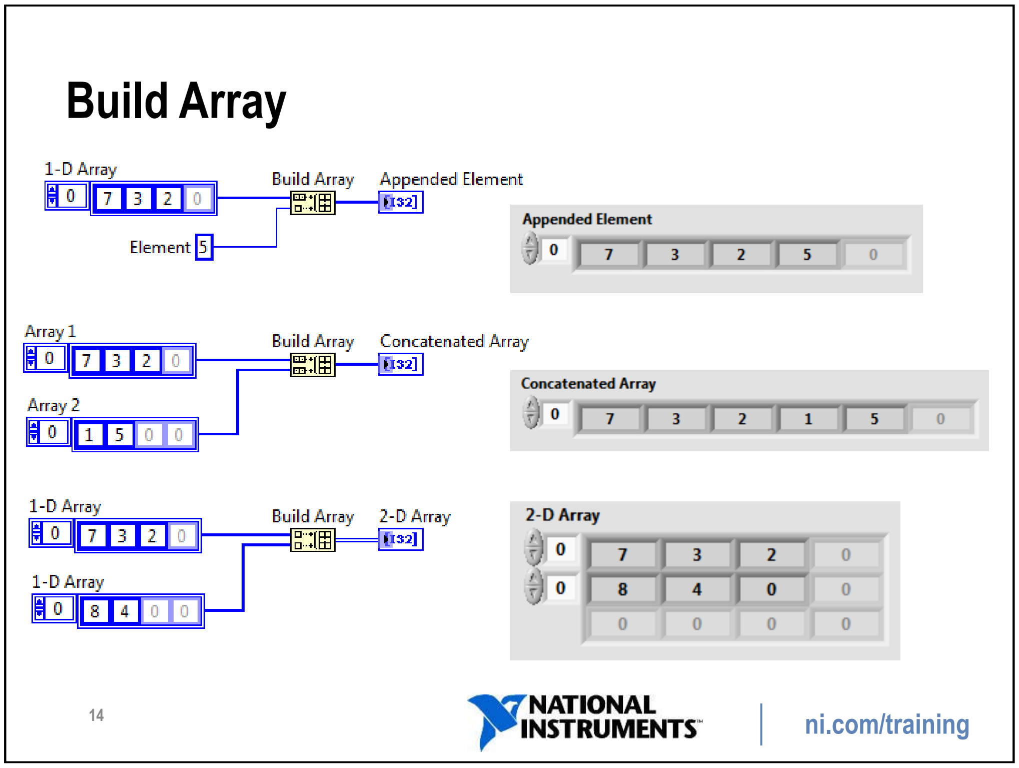
Initialize array is a resizable node. Use the Positioning tool to resize the function and increase the number of dimensions (element, row, column, page, and so on) of the output array. You must have one **dimension size** input terminal for each dimension in the array.

Array Subset

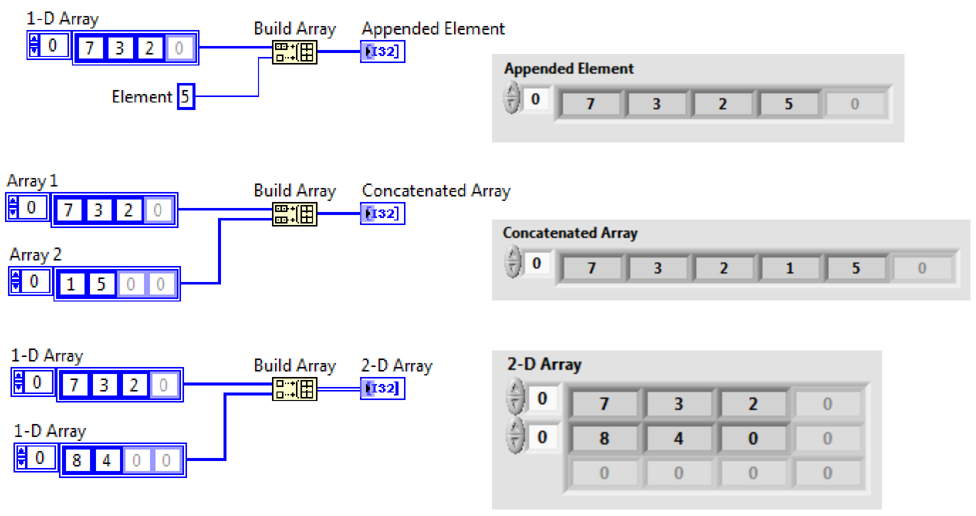


NATIONAL INSTRUMENTS | ni.com/training

The Array Subset function returns a portion of an array starting at index and containing **length** number of elements.



Build Array



14

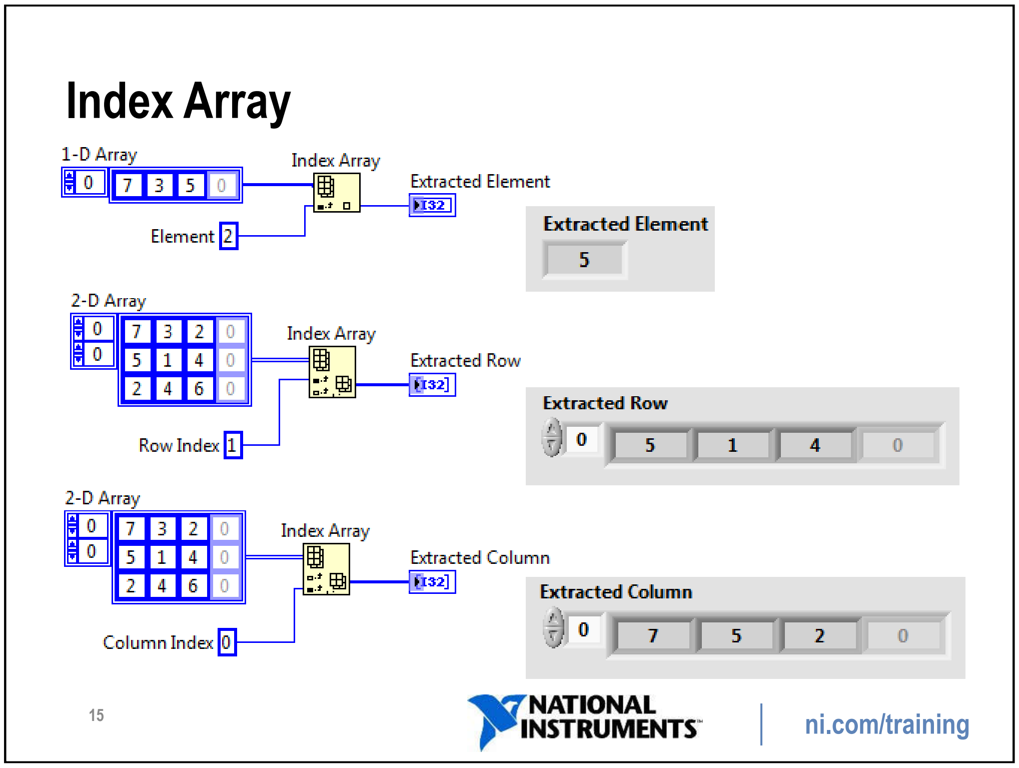
NATIONAL INSTRUMENTS | ni.com/training

The Build Array function concatenates elements together into one array or concatenates multiple arrays together into arrays of higher dimension.

Wiring a scalar element into the Build Array function automatically configures the Build Array function to concatenate elements.

When wiring in two or more arrays, right click on the output of the Build Array function to enable or disable concatenated inputs.

The Build Array function is a resizable node and can be resized to add additional input terminals.



Index Array

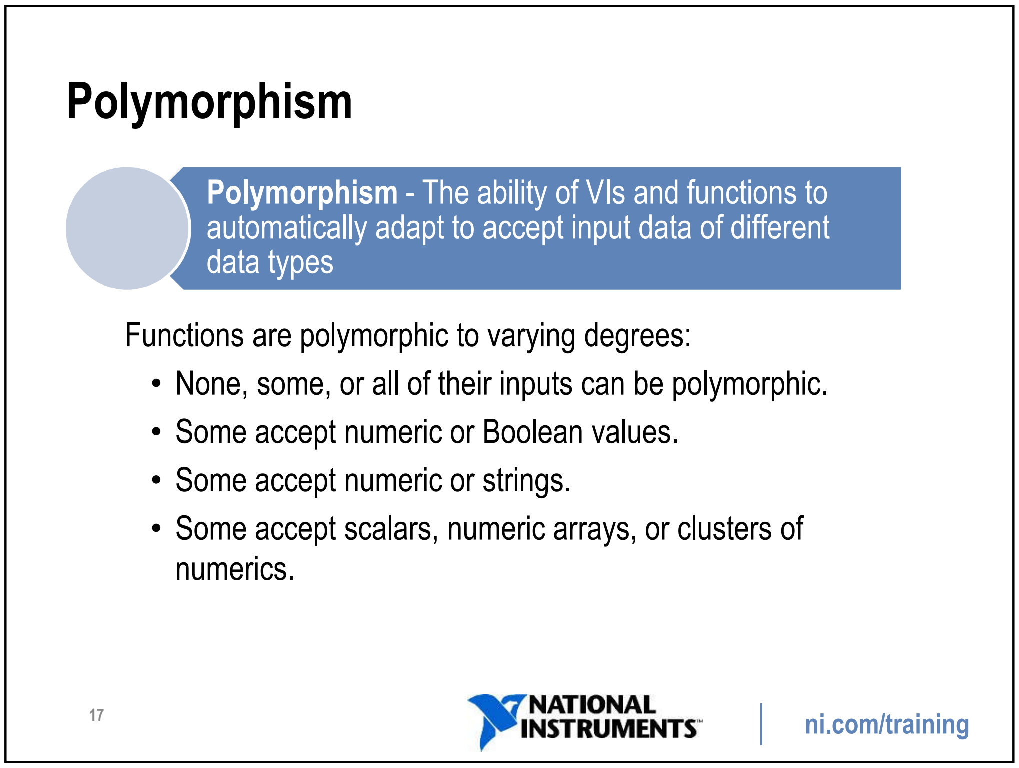
The Index Array function : Returns the **element or subarray** of **n-dimension array** at **index**. When you wire an array to this function, the function resizes automatically to display **index** inputs for each dimension in the array you wire to n-dimension array.

The slide shows examples of the function accessing a single element of a one-dimensional array, a row of a two-dimensional array, and a column of a two-dimensional array.

Index Array does not remove the element from the array.

Remind students that arrays start their indices at zero. Index 2 in the example actually accesses the third element in the array.

C. Polymorphism



Polymorphism

Polymorphism - The ability of VIs and functions to automatically adapt to accept input data of different data types

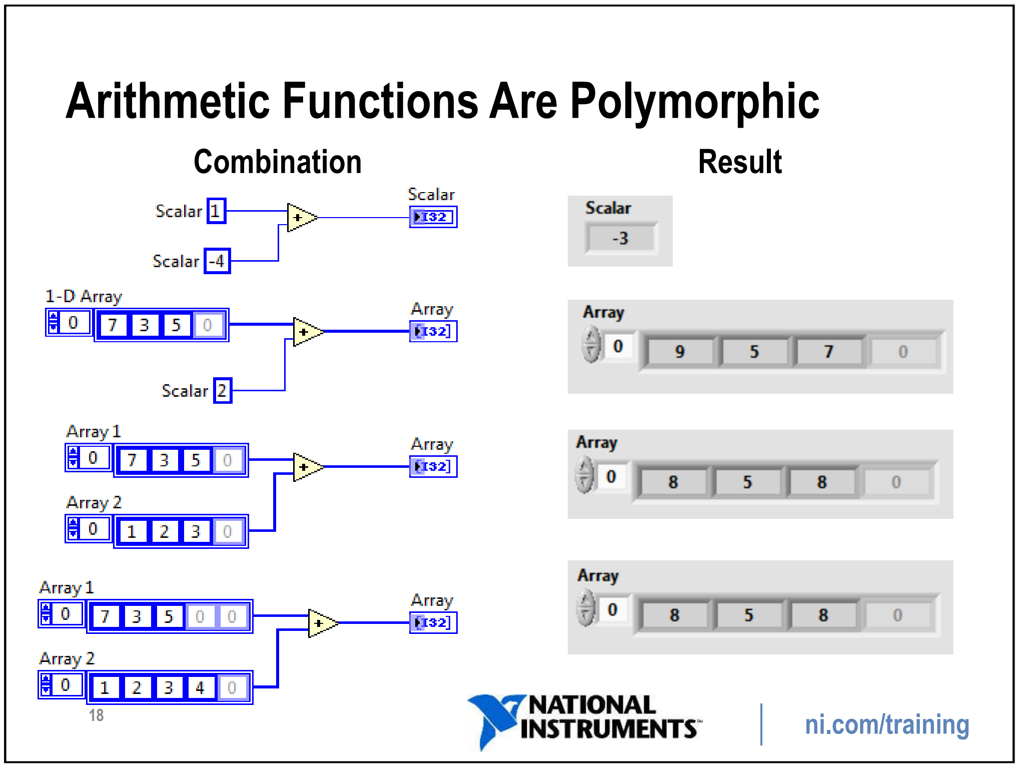
Functions are polymorphic to varying degrees:

- None, some, or all of their inputs can be polymorphic.
- Some accept numeric or Boolean values.
- Some accept numeric or strings.
- Some accept scalars, numeric arrays, or clusters of numerics.

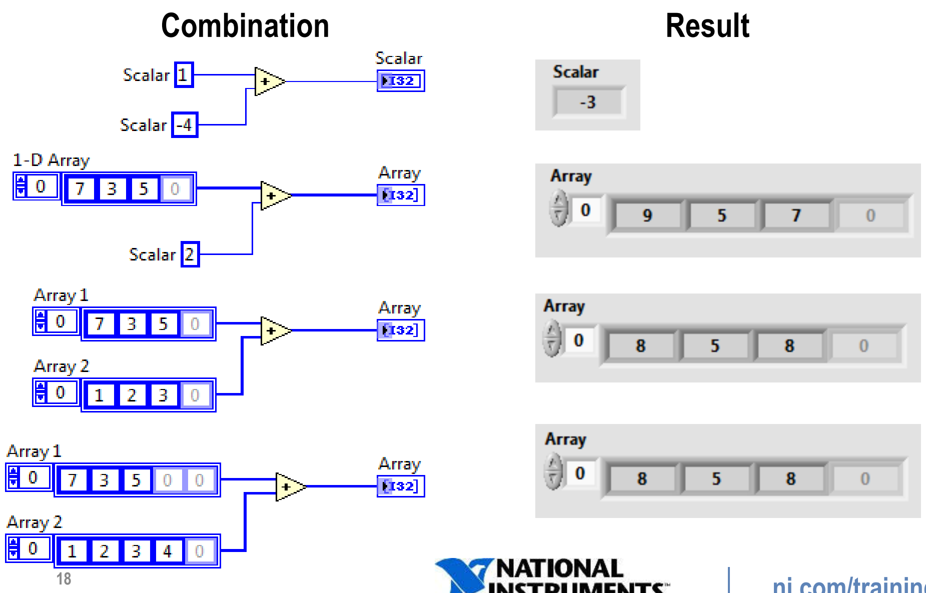
17

NATIONAL INSTRUMENTS | ni.com/training

Functions are polymorphic to varying degrees—none, some, or all of their inputs can be polymorphic. Some function inputs accept numeric values or Boolean values. Some accept numeric values or strings. Some accept not only scalar numeric values but also arrays of numeric values, clusters of numeric values, arrays of clusters of numeric values, and so on. Some accept only one-dimensional arrays although the array elements can be of any type. Some functions accept all types of data, including complex numeric values.



Arithmetic Functions Are Polymorphic



NATIONAL INSTRUMENTS™ | ni.com/training

LabVIEW arithmetic functions are polymorphic.

- Inputs to arithmetic functions can be of different data types.
- The function automatically performs the appropriate operation on unlike data types.
- LabVIEW arithmetic functions greatly simplify array arithmetic.

Examples of polymorphism on this slide:

Scalar+Scalar: Scalar addition.
Scalar+Array: The scalar is added to each element of array.
Array+Array: Each element of one array is added to the corresponding element of other array.

In case students ask, polymorphism does not perform matrix arithmetic when inputs are 2D arrays. (For example, multiplying two 2D array inputs with the Multiply function performs an element by element multiplication, not matrix multiplication).

LabVIEW ships with other examples of arithmetic functions showing polymorphic combinations of inputs.

D. Auto-Indexing

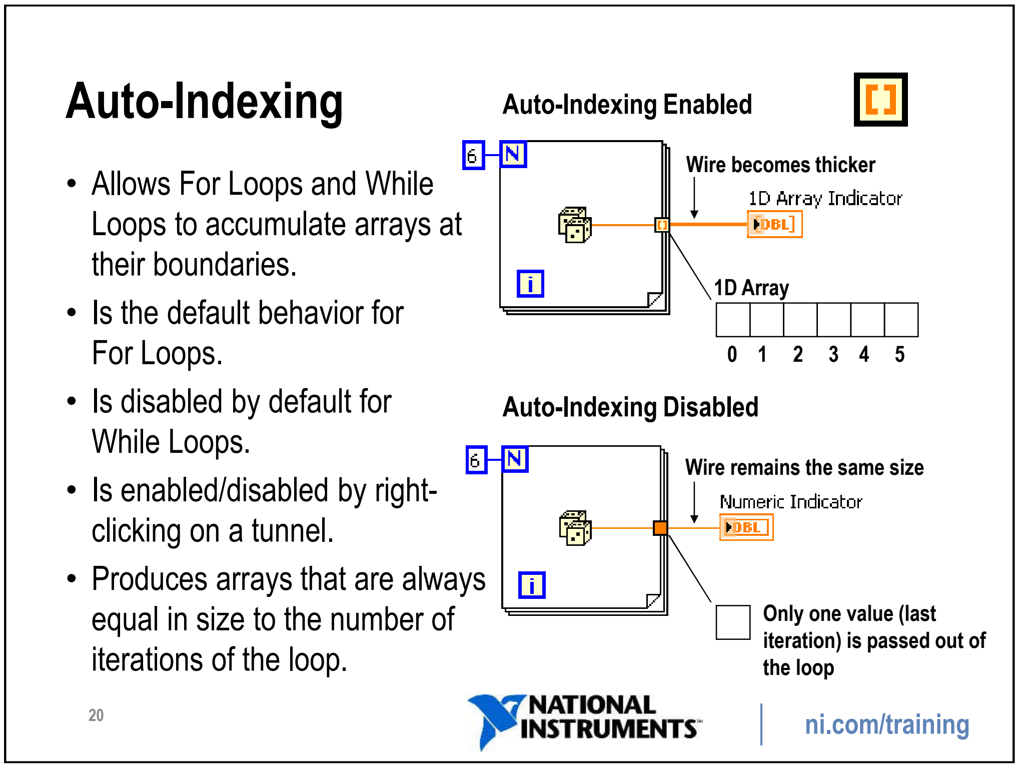
Use in For Loops and While Loops

Waveform Graphs

Auto-Indexing with a Conditional Terminal

Creating 2D Arrays

Auto-Indexing Input to a Loop



Auto-Indexing

- Allows For Loops and While Loops to accumulate arrays at their boundaries.
- Is the default behavior for For Loops.
- Is disabled by default for While Loops.
- Is enabled/disabled by right-clicking on a tunnel.
- Produces arrays that are always equal in size to the number of iterations of the loop.

20

NATIONAL INSTRUMENTS | ni.com/training

For Loops and While Loops can index and accumulate arrays at their boundaries. This is known as auto-indexing.

The indexing point on the boundary is called a tunnel.

The For Loop default is auto-indexing enabled.

The While Loop default is auto-indexing disabled. Only one value (the last iteration) is passed out of the While Loop by default.

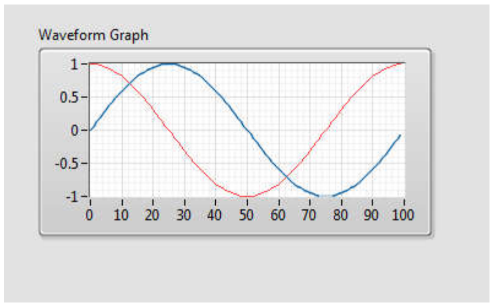
Examples:

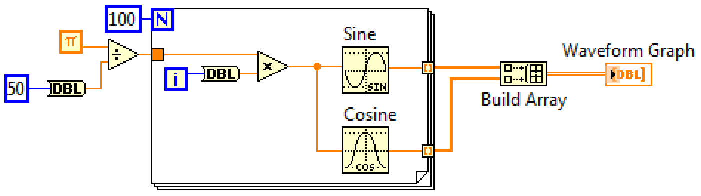
Enable auto-indexing to collect values within the loop and build the array. All values are placed in the array upon exiting the loop.

Disable auto-indexing if you are interested only in the final value.

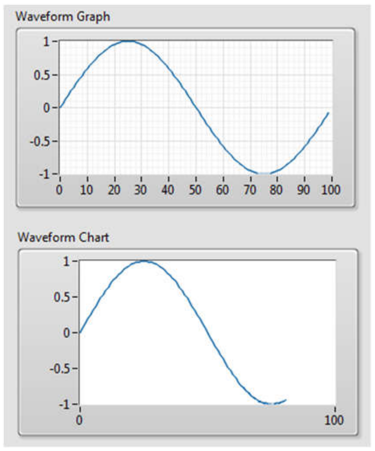
Waveform Graph

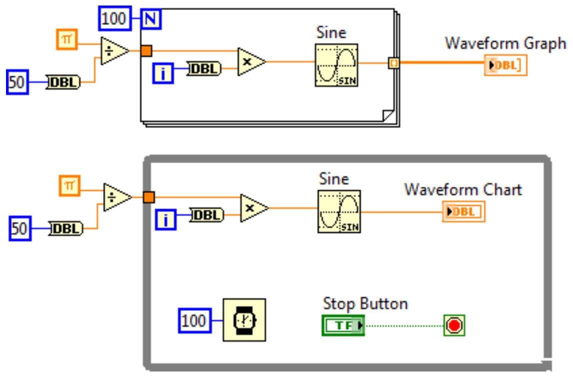
- Is a graphical display of data.
- Displays one or more plots of evenly sampled measurements.
- Is used to plot pre-generated arrays of data.
- Can display plots with any number of data points.





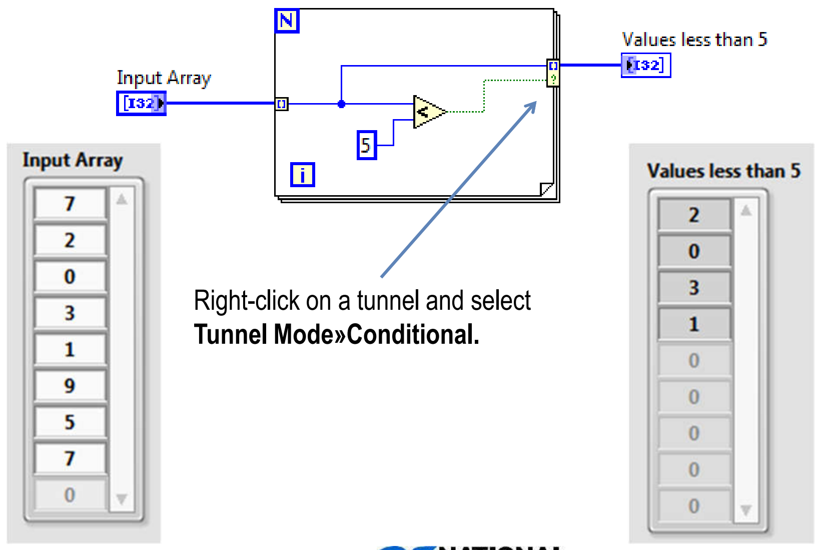
Charts vs. Graphs – Single-Plot





NATIONAL INSTRUMENTS | ni.com/training

Auto-Indexing with a Conditional Tunnel



Right-click on a tunnel and select **Tunnel Mode»Conditional.**

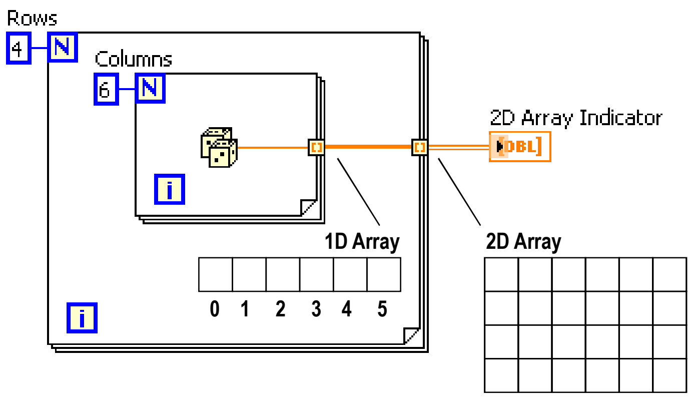
23

ni.com/training

You can determine what values LabVIEW writes to the loop output tunnel based on a condition you specify by right-clicking the loop output tunnel and selecting **Tunnel Mode»Conditional** from the shortcut menu.

Because of the conditional tunnel, the **Values less than 5** array contains only the elements 2, 0, 3, and 1 after this loop completes all iterations.

Creating 2D Arrays



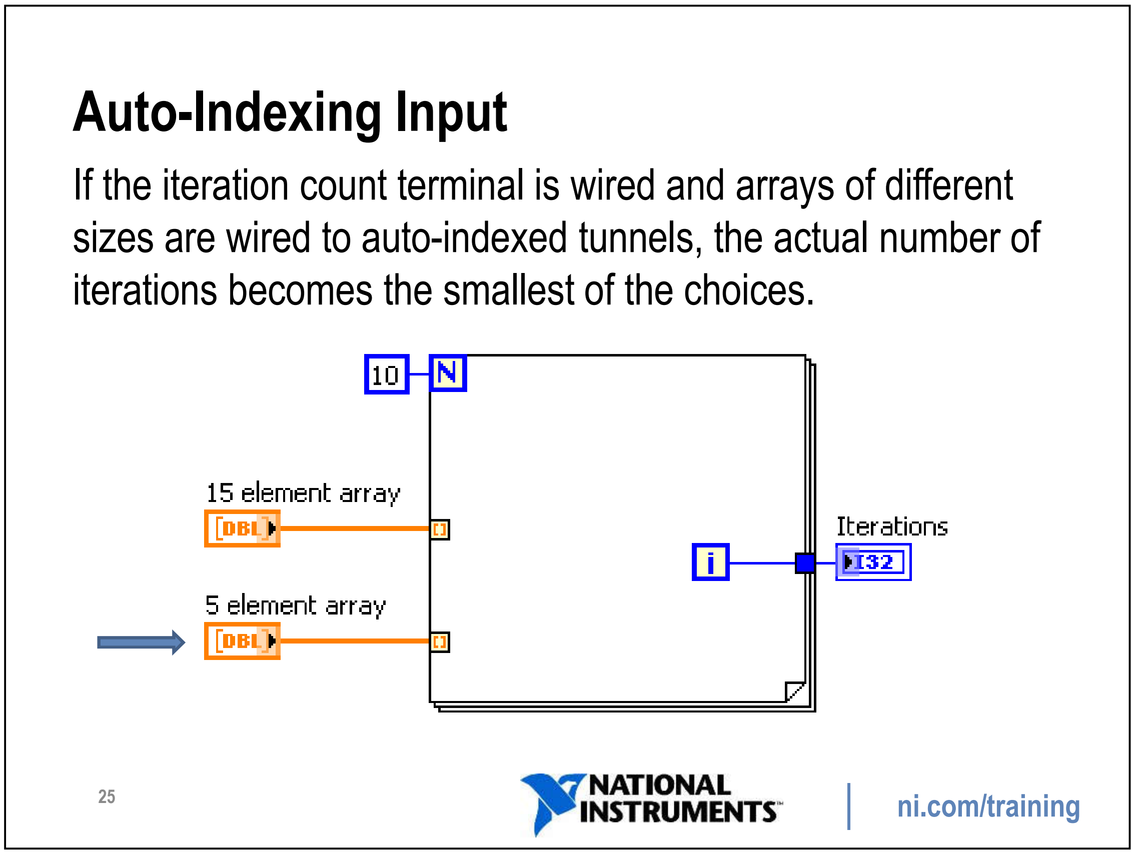
- Inner loop creates column elements.
- Outer loop stacks column elements into rows.

You can use two nested For Loops to create a 2D array. Auto-indexing must be enabled for both loops.

Explain the different line thicknesses in the wire connecting the Random Number function to the 2D Array Indicator.

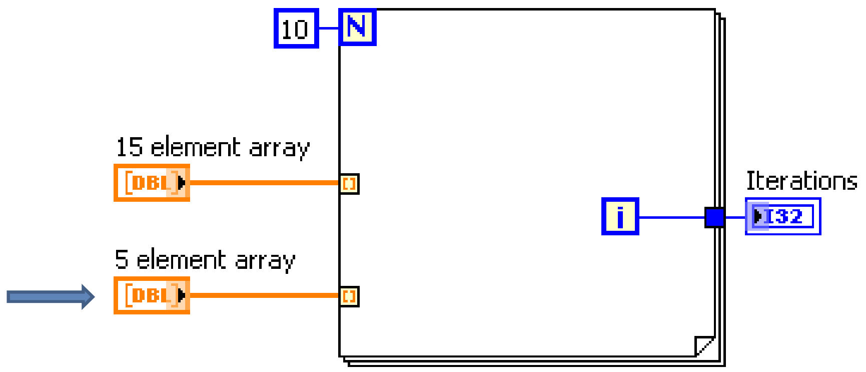
Demonstrate on your computer:

- How to change indexing and line thickness.
- When you see **Enable indexing** in the shortcut menu it means that indexing is currently disabled. The menu choice is the opposite of the current indexing mode. Students get confused about this feature.



Auto-Indexing Input

If the iteration count terminal is wired and arrays of different sizes are wired to auto-indexed tunnels, the actual number of iterations becomes the smallest of the choices.



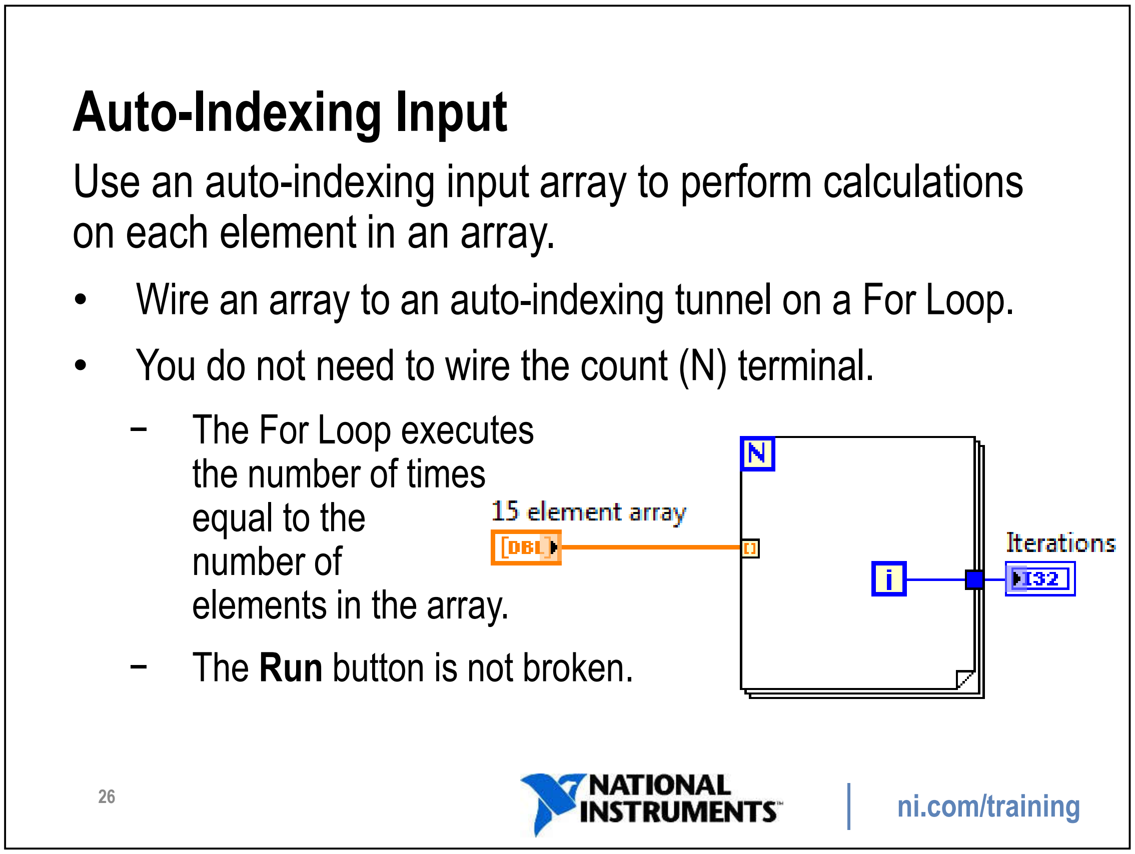
25

NATIONAL INSTRUMENTS | ni.com/training

LabVIEW does not exceed an array size. This helps to protect against programming errors.

If you wire a 10-element array and a 5-element array to a For loop, the loop iterates five times.

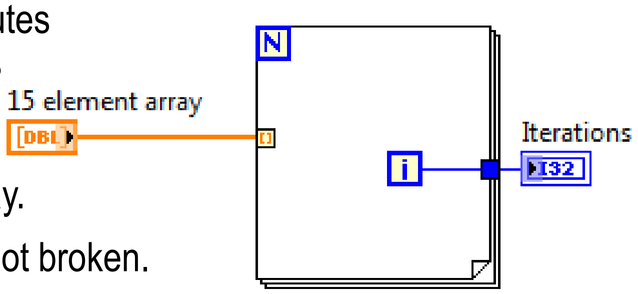
Note: Although the For loop iterates 5 times, the number of iterations is zero-based. Therefore, the final value of the **Iterations** indicator is 4.



Auto-Indexing Input

Use an auto-indexing input array to perform calculations on each element in an array.

- Wire an array to an auto-indexing tunnel on a For Loop.
- You do not need to wire the count (N) terminal.
  - The For Loop executes the number of times equal to the number of elements in the array.
  - The **Run** button is not broken.



26

NATIONAL INSTRUMENTS | ni.com/training

If you enable auto-indexing on an array wired to a For Loop input terminal, LabVIEW sets the count terminal to the array size so you do not need to wire the count terminal. Normally, if the count terminal of the For Loop is not wired, the **Run** arrow is broken.

In this slide, the For Loop executes a number of times equal to the number of elements in the array.

Use auto-indexing on input arrays for calculations to be performed on each element of array.

To pass the entire array into a loop, disable auto-indexing.