

Array Functions

There are a great many array functions in LabVIEW. We will only discuss some of the more basic and prevalent functions.

Array Max & Min

Array to Cluster

Index Array

Reshape Array

Rotate 1D Array

Sort 1D Array

Threshold 1D Array

Interleave 1D Arrays

Cluster to Array

Array size

Build Array

Initialize Array

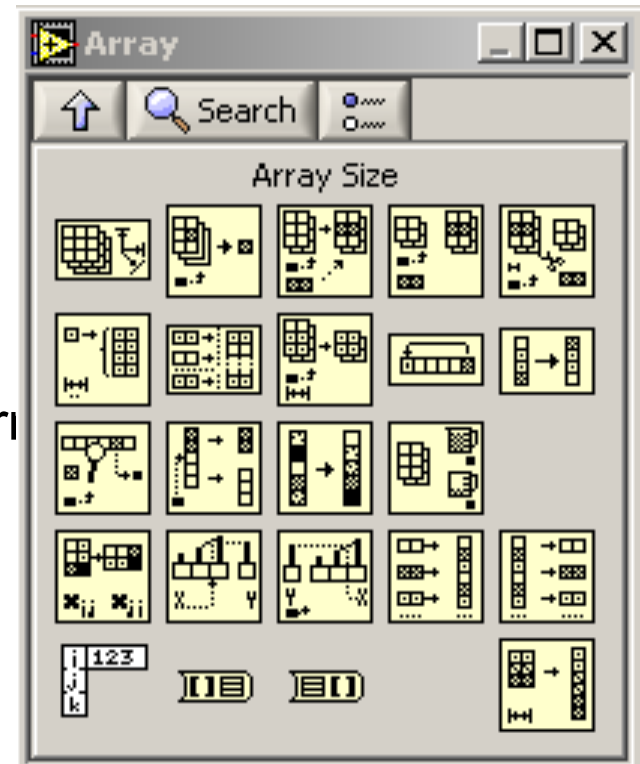
Reverse 1D Array

Search 1D Array

Split 1D Array

Transpose 2D Array

Array Subset



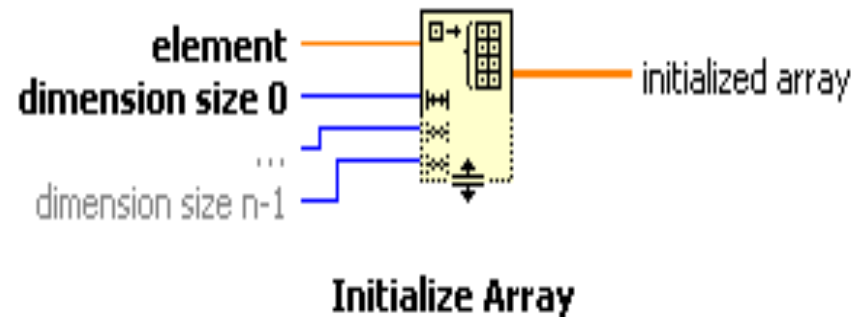
Initialize Array Function

Creates an n-dimensional array in which every element is initialized to the value of **element**.

element determines the type of the array and the initial value of each array element. Element can be any data type except an array.

dimension size. You must add a dimension size terminal for each dimension of the initialized array.

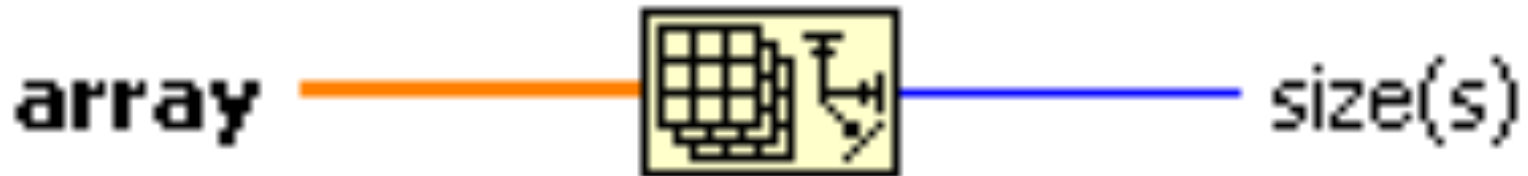
The function coerces dimension size to a 32-bit integer if you wire another representation to it. The function creates an empty array if any dimension size is 0.



Creates an n-dimensional array in which every element is initialized to the value of **element**.

Array Size Function

Input can be an n-dimensional array of any type.

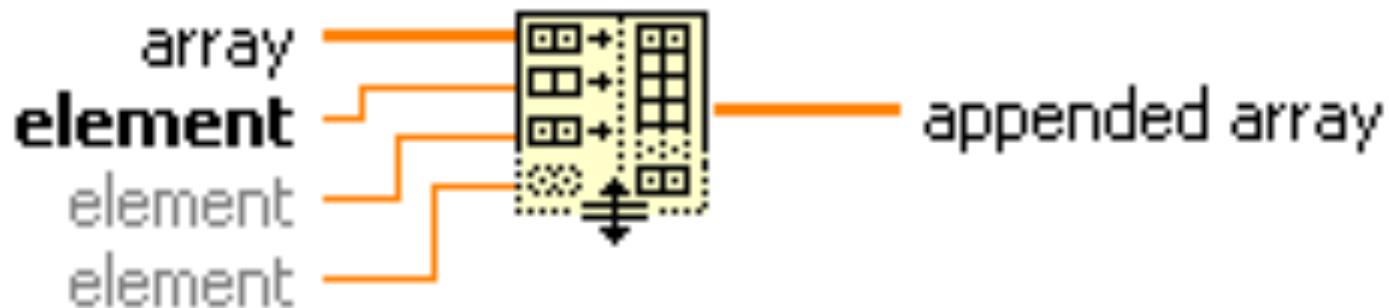


Array Size

Returns the number of elements in each dimension of **array**.

Build Array Function

Appends any number of array or element inputs in top-to-bottom order to create array with appended element



Build Array

Concatenates multiple arrays or appends elements to an n-dimensional array.

Build Array Function

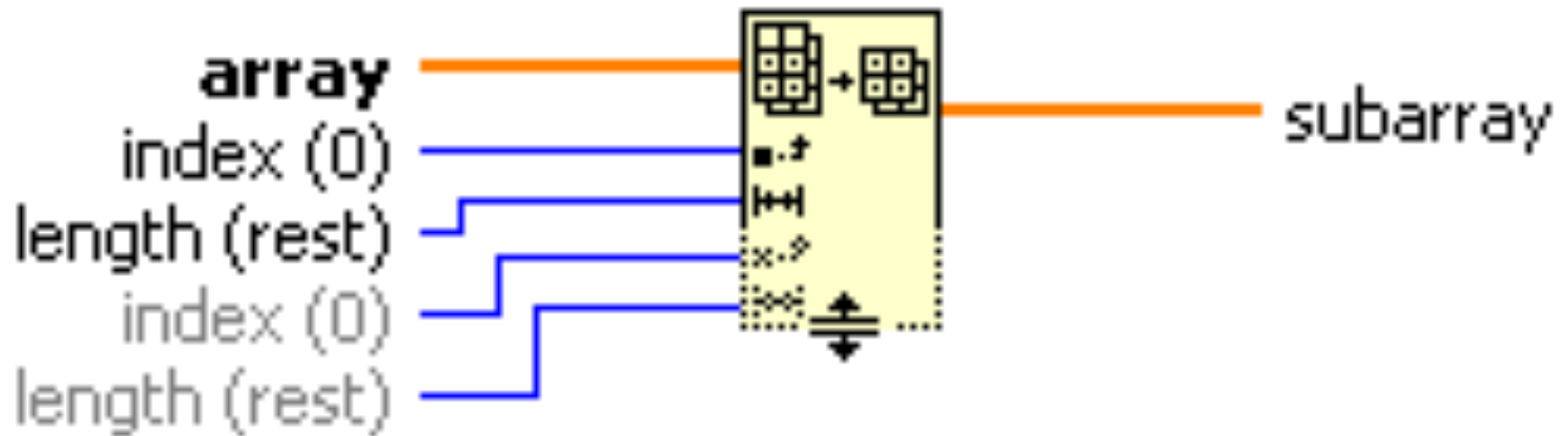
Initially, this function has two element inputs.

To change an element input to an array input, pop-up on the input and select Change to Array.

To create a 1D array, connect scalar values to the element inputs and 1D arrays to the array outputs .

To build a 2D array, connect 1D arrays to element inputs and 2D arrays to the array outputs.

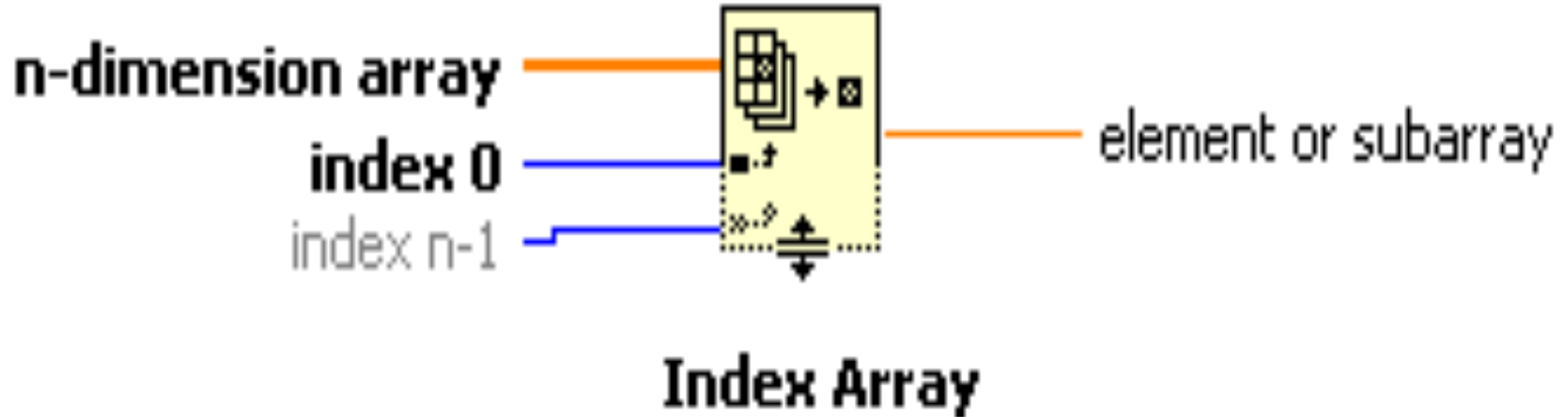
Array Subset Function



Array Subset

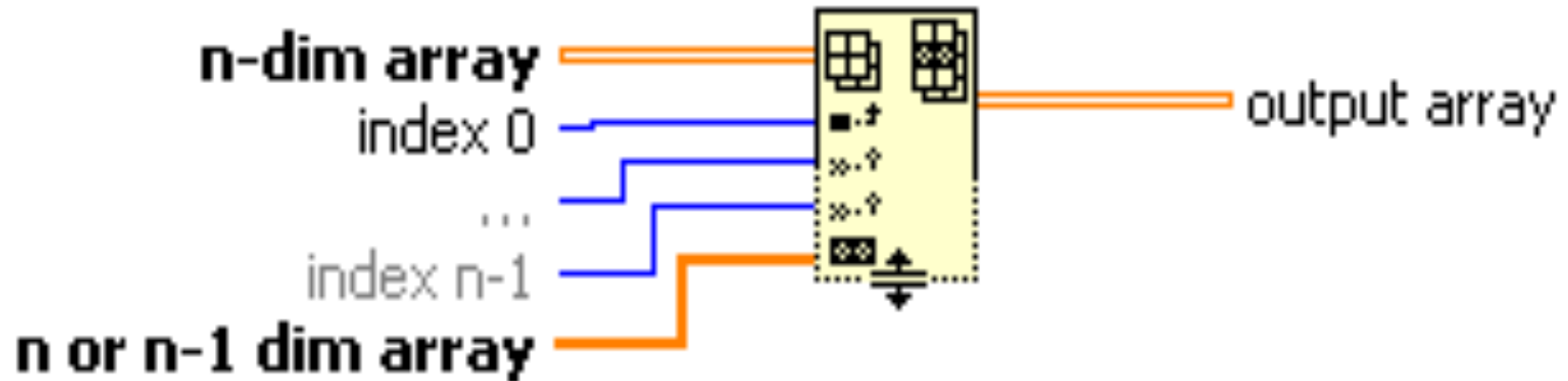
Returns a portion of **array** starting at **index** and containing **length** elements.

Index Array Function



Returns the **element or sub-array** of **n-dimension array** at **index**.

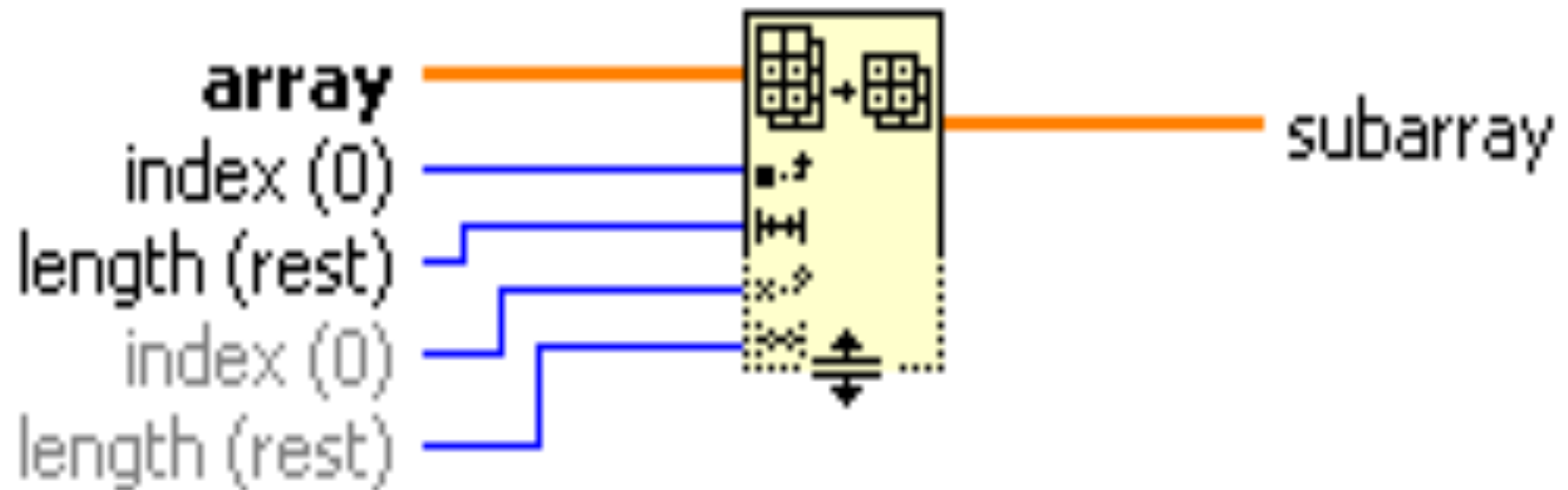
Insert into Array



Insert Into Array

Inserts an element or subarray into **n-dim array** at the point you specify in **index**.

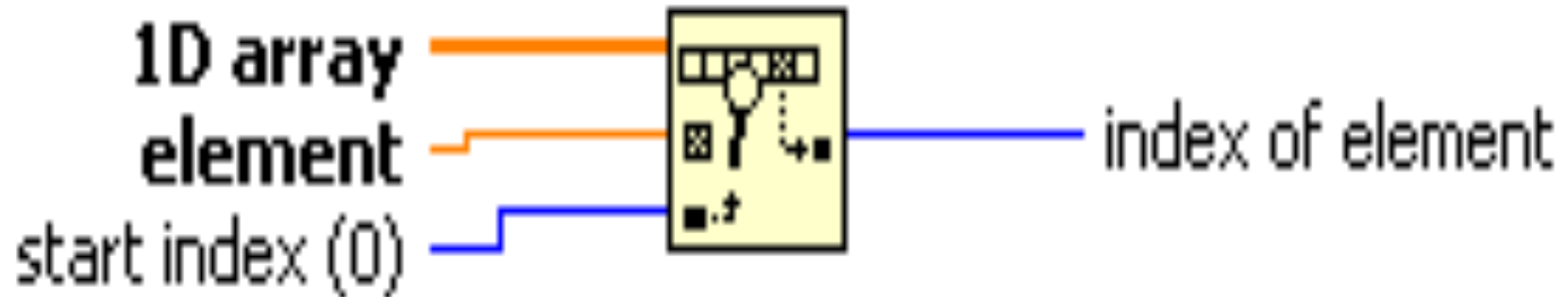
Array Segments



Array Subset

Returns a portion of **array** starting at **index** and containing **length** elements.

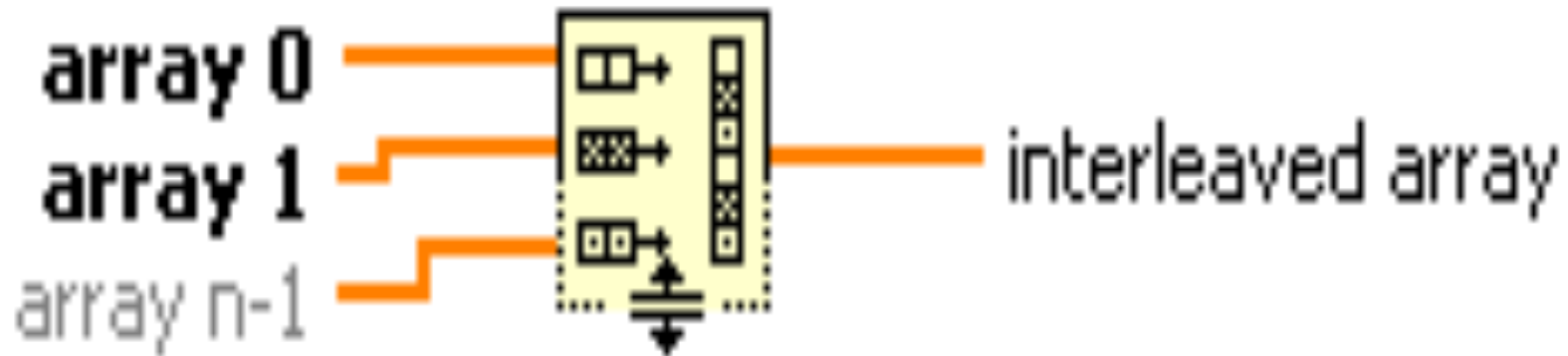
Search 1D Array



Search 1D Array

Searches for an **element** in a **1D array** starting at **start index**. You do not need to sort the array before calling this function, because the search is linear.

Interleave Arrays

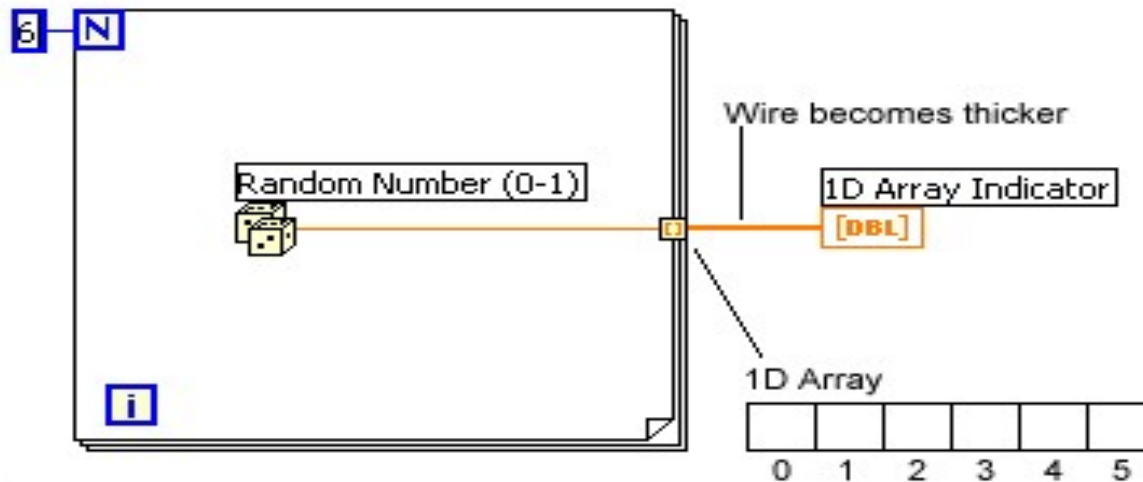


Interleave 1D Arrays

Interleaves corresponding elements from the input arrays into a single output array.

For Loop: Auto-Indexing

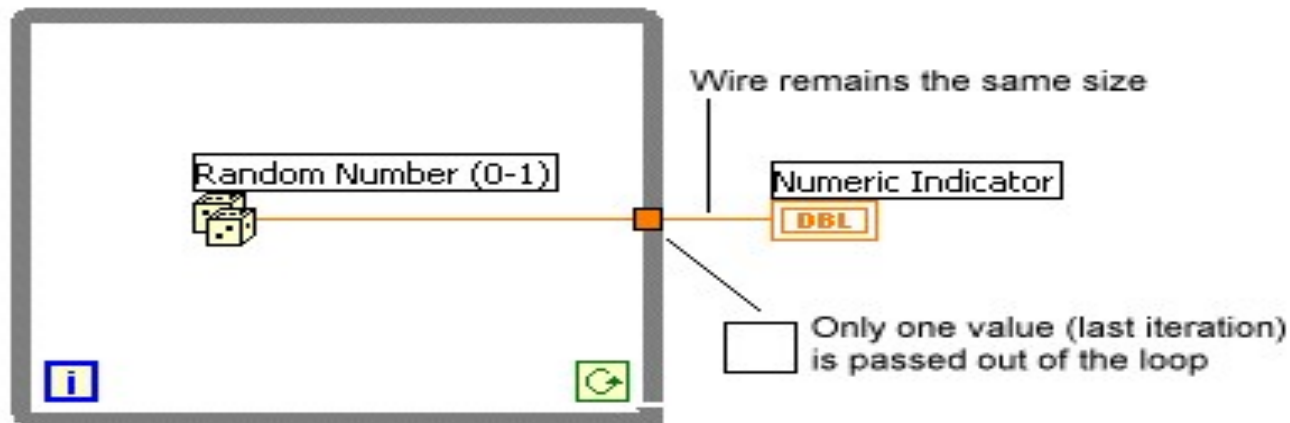
Auto-Indexing Enabled (Default of For Loops)



If you wire an array to a For Loop or While Loop input tunnel, you can read and process every element in that array by enabling auto-indexing. When you auto-index an array output tunnel, the output array receives a new element from every iteration of the loop. The wire from the output tunnel to the array indicator becomes thicker as it changes to an array at the loop border and the output tunnel contains square brackets representing an array, as shown in the example above.

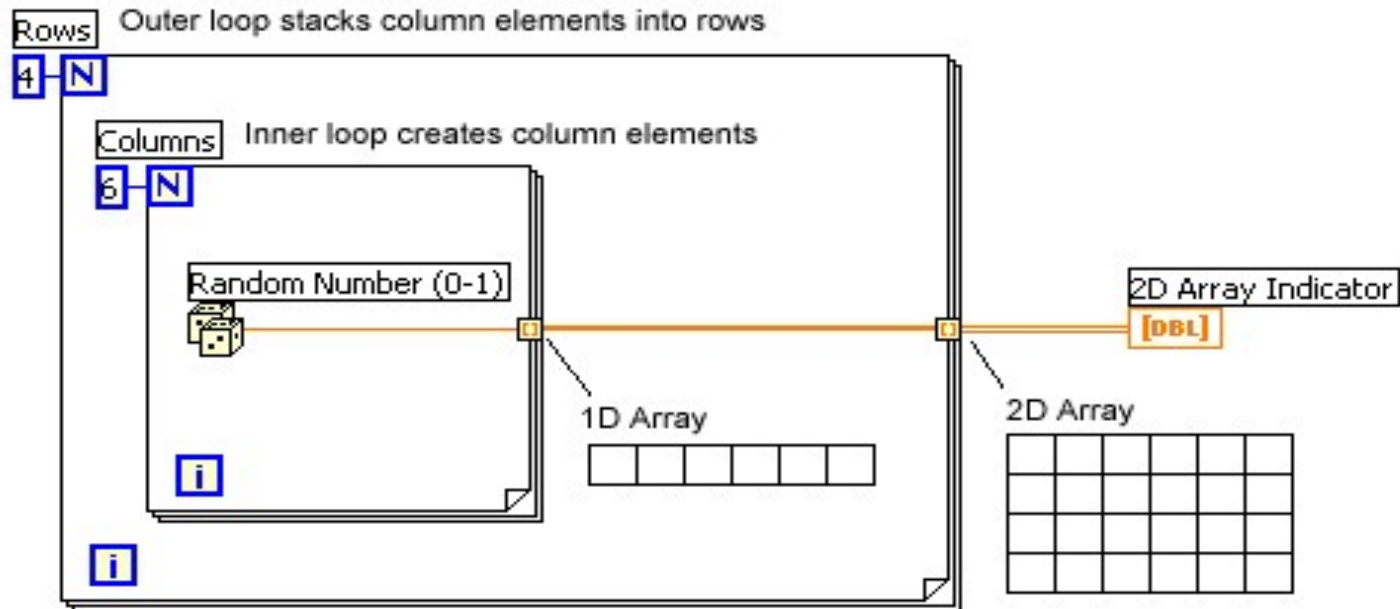
While Loop

Auto-Indexing Disabled (Default of While Loops)



LabVIEW disables auto-indexing for While Loops by default. Only the value of the last iteration is passed through a While Loop tunnel as shown in the example above.

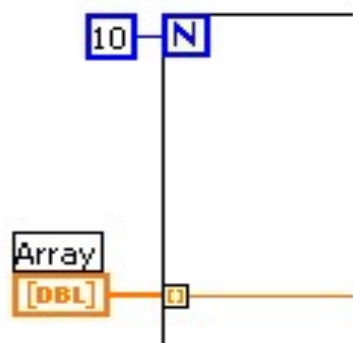
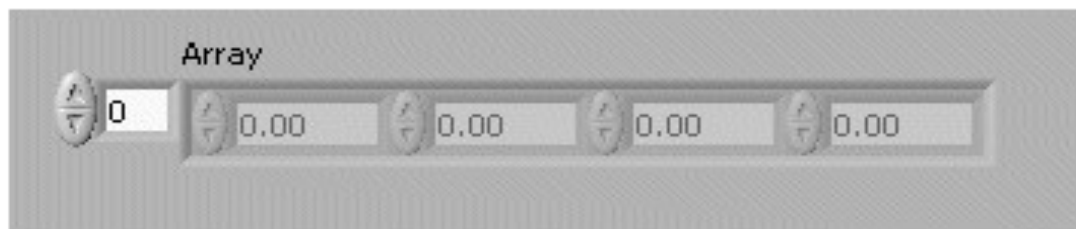
2D Arrays



Creating Two-Dimensional Arrays

You can nest a For Loop inside another For Loop to create a 2D array. The outer For Loop creates the row elements and the inner For Loop creates the column elements as shown on this slide.

For Loop



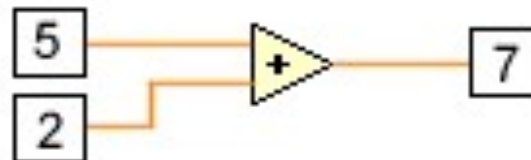
If you enable auto-indexing on an array wired to a For Loop input terminal, LabVIEW sets the count terminal to the array size, so you do not need to wire the count terminal. If you enable auto-indexing for more than one tunnel, or if you set the count terminal, the count becomes the smaller of the two choices. For example, the array wired to the input tunnel on the For Loop shown on this slide has 4 elements. The count terminal is set to 10. The loop will execute 4 times.

Polymorphism

Combination

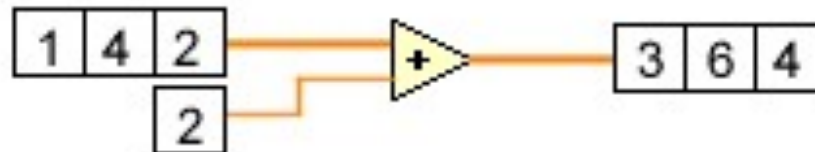
Result

Scalar + Scalar



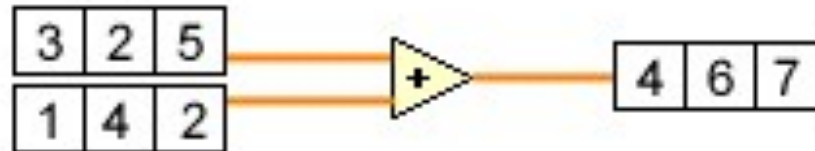
Scalar

Array + Scalar



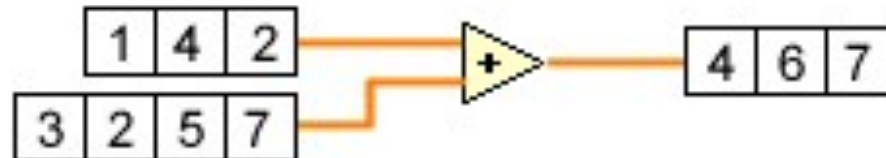
Array

Array + Array



Array

Array + Array



Array

Add Array Elements Function

With this function we can easily add the elements of an input numeric array.



Add Array Elements

Returns the sum of all the elements in **numeric array**.

Example: Building Arrays Front Panel

Building Arrays.vi Front Panel

File Edit Operate Tools Browse Window Help

13pt Application Font

build Array

Press the run button to execute this VI. The array indicators will show two methods of building arrays by using auto-indexing. Press <ctrl-h> for instructions.

A.- Random Numbers

| | | | | |
|---|------|------|------|------|
| 2 | 0.82 | 0.41 | 0.15 | 0.12 |
| 3 | 0.01 | 0.75 | 0.65 | 0.85 |
| | 0.89 | 0.25 | 0.06 | 0.49 |

Row i: 9
Column j: 9

B.- Random Numbers

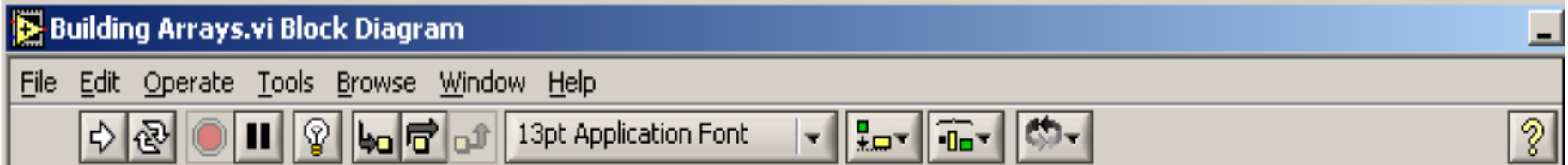
| | | | | |
|---|------|------|------|------|
| 1 | 0.95 | 0.10 | 0.76 | 0.88 |
|---|------|------|------|------|

Element i: 100

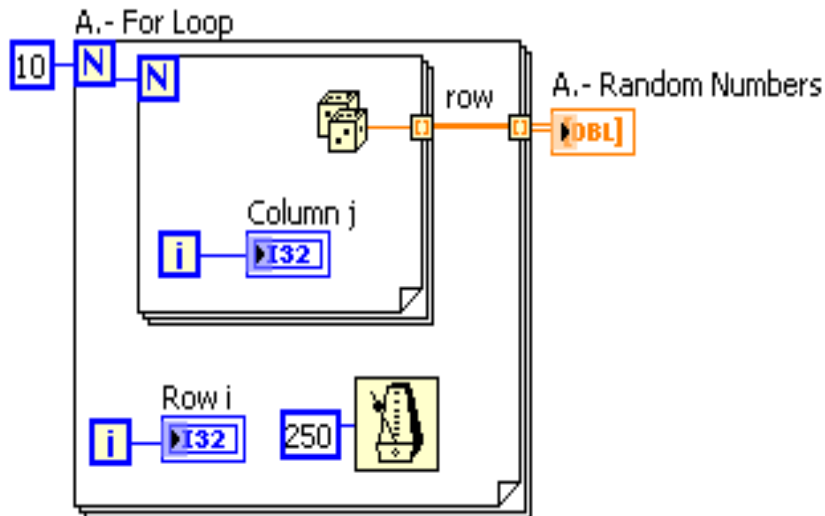
The button stops building array B.

STOP

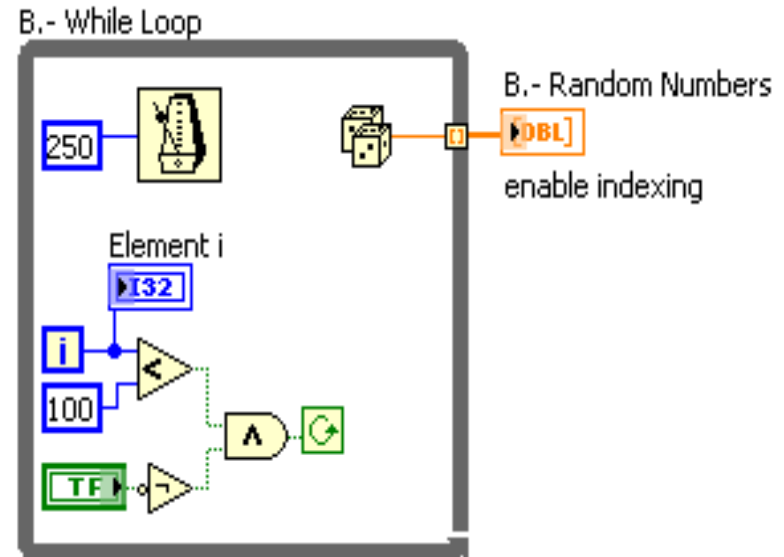
Example: Building Arrays Block Diagram



A.- The For Loop uses auto-indexing as its default, which is the best method when the number of values is known. Wire the variable inside the inner loop directly to an array terminal. Since one For Loop is placed inside another For Loop, a 2D array will be produced.



B.- The While Loop is the best method when the number of values is unknown so the user or program determines the size of the array. Wire the variable inside the loop directly to an array terminal. Then right-click on the tunnel and select "enable indexing".



Example: Building Arrays

A.- The For Loop uses auto-indexing as the default, it is the best method when the number of values is known. Wire directly the variable inside the loop subdiagram to an array terminal. To create a 2D array, include one loop inside another one.

B.-The While Loop uses disable indexing as the default. It is the best method when the number of values is unknown so the user or a program variable determines the size of the array. Wire directly the variable inside the loop subdiagram to an array terminal, use "enable indexing from the tunnel (black box) pop up menu.

Example: Separate Array Values Front Panel

Separate Array Values.vi Front Panel

File Edit Operate Tools Browse Window Help

13pt Application Font

Sepr array values

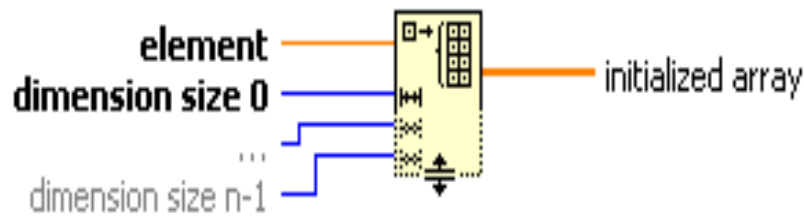
Press the run button. This example will sort the Input Array into two separate 1D arrays containing positive values and negative values.

| Input Array | Positive Array | Negative Array |
|-------------|----------------|----------------|
| 6.00 | 6.00 | -5.80 |
| 3.90 | 3.90 | -2.60 |
| -5.80 | 0.00 | -1.00 |
| 0.00 | 2.20 | -4.30 |
| -2.60 | | |
| -1.00 | | |
| 2.20 | | |
| -4.30 | | |

Example: Separate Array Values Block Diagram

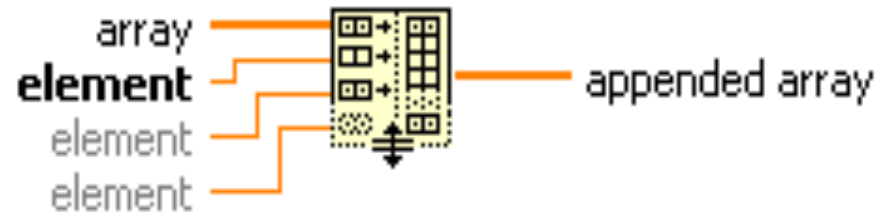
Data type of wire

- CI** (1-D array of)
- DBL** (double [64-bit real (~15 digit precision)])



Initialize Array

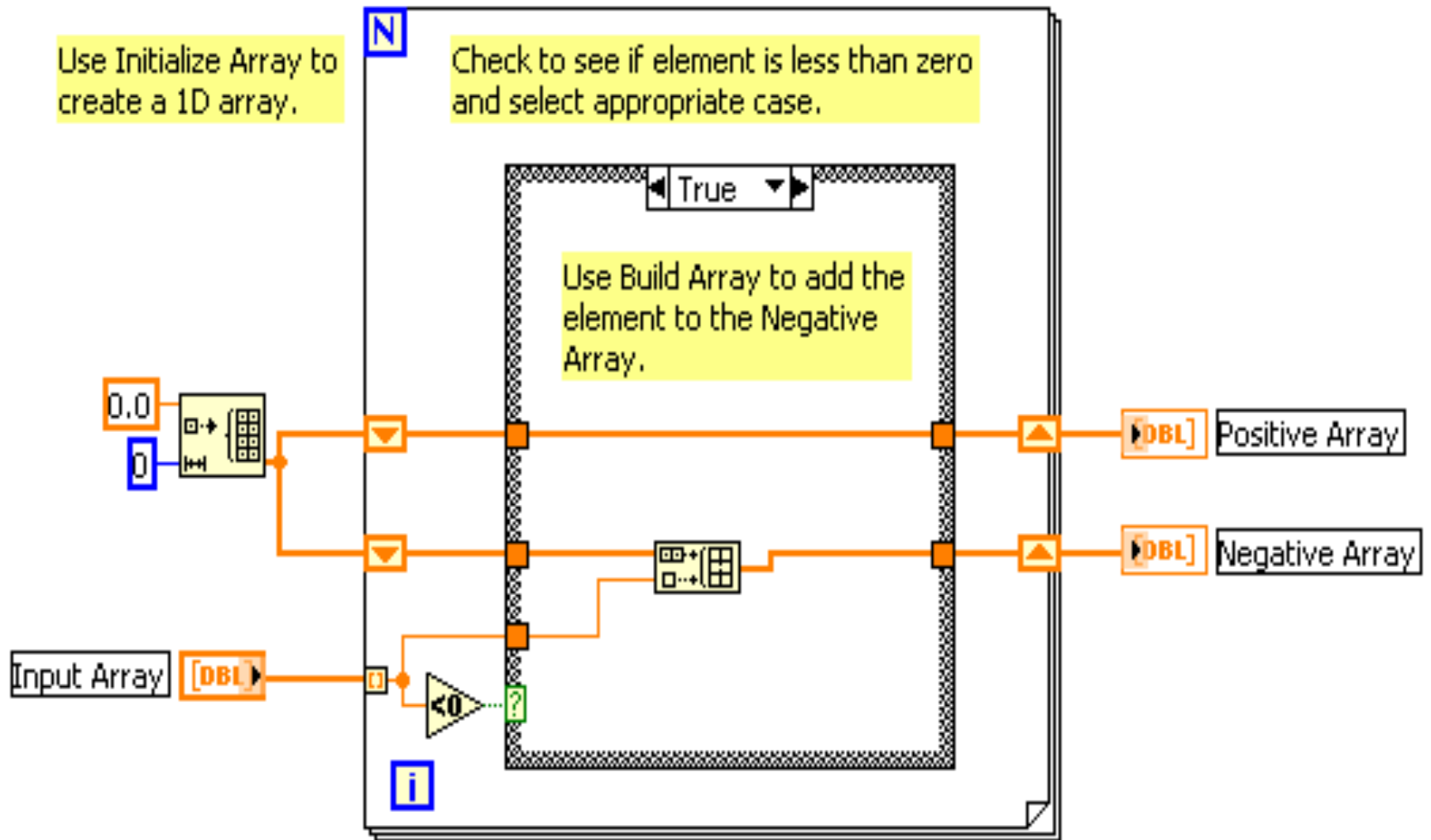
Creates an n-dimensional array in which every element is initialized to the value of **element**.



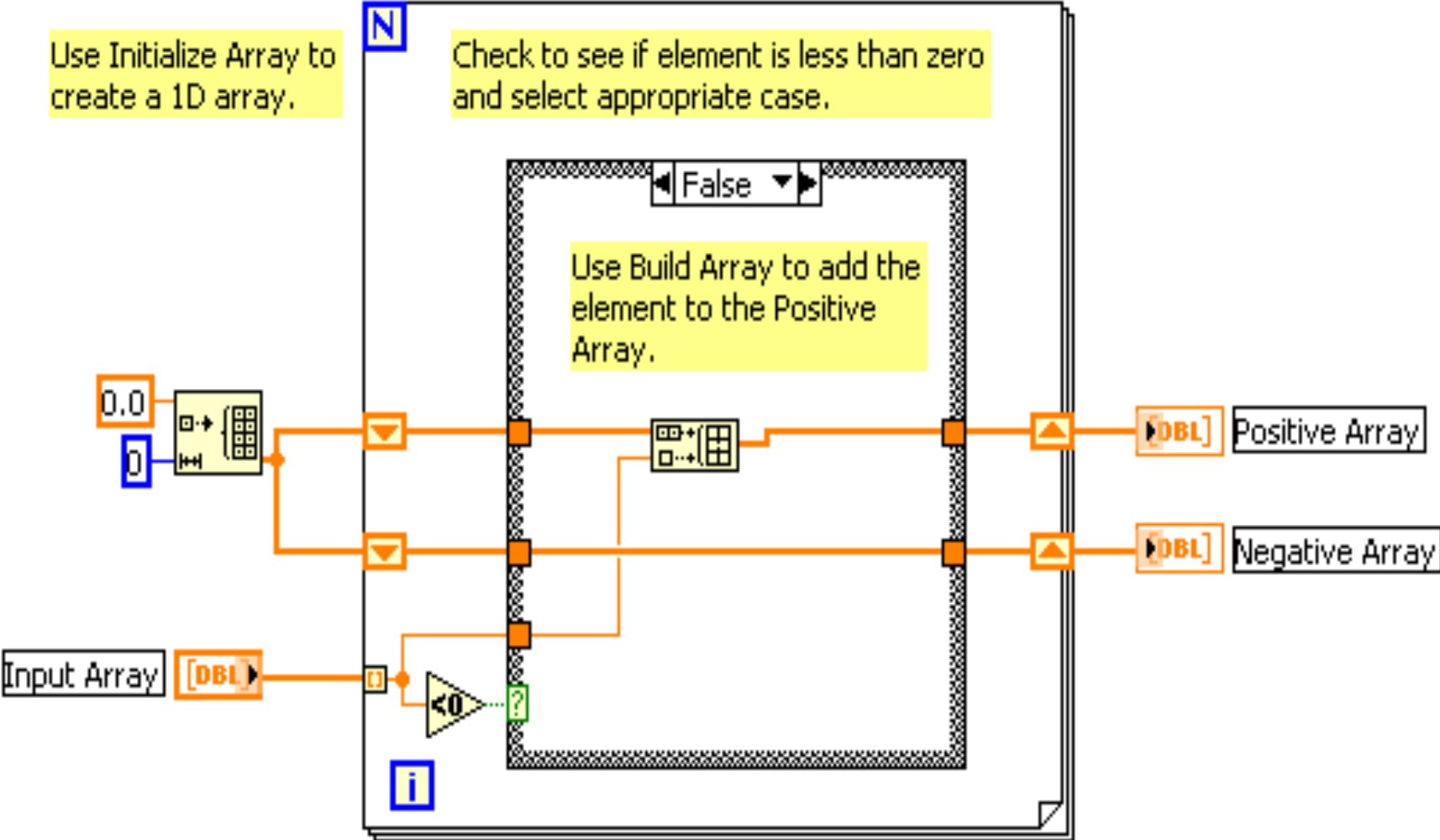
Build Array

Concatenates multiple arrays or appends elements to an n-dimensional array.

Example: Separate Array Values Block Diagram



Example: Separate Array Values Block Diagram



Clusters

- A cluster is a data structure that groups data of different types. A cluster is analogous to a record in Pascal or a struct in C. A cluster may be thought of as a bundle of wires, much like a telephone cable. Each wire in the cable represents a different element in the cluster.
- Clusters are an extremely useful construct which can greatly simplify the wiring of a VI block diagram. While the subject of clusters is a more advanced topic, we will look at some rudimentary issues associated with them.

Primary Cluster Formation Rule

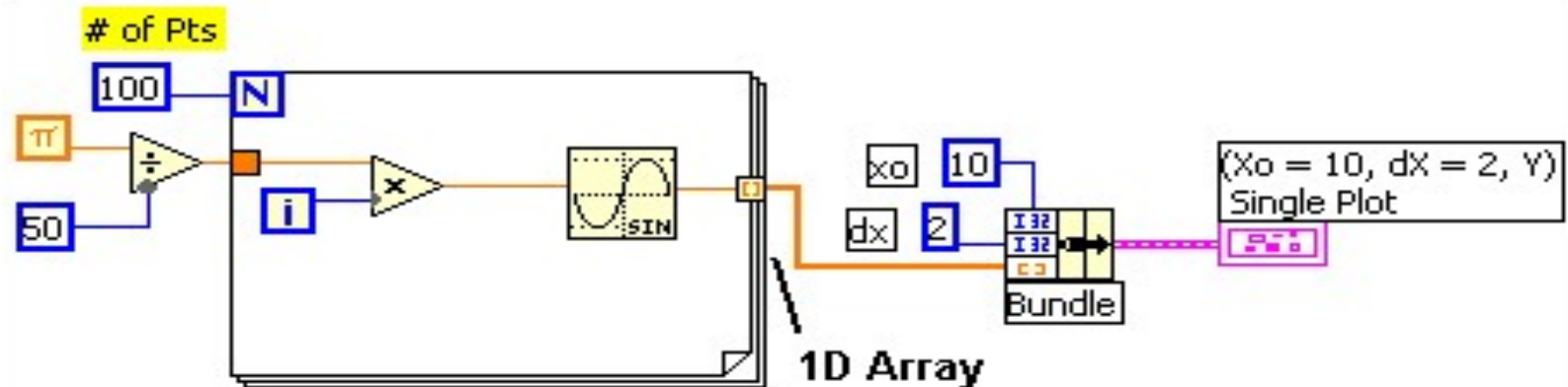
Objects inside a cluster must be all controls or all indicators!

You cannot combine both controls and indicators inside the same cluster.

Controls palette>>Array & Cluster>>Cluster

Functions palette>>Cluster

Bundle Function



Use the Bundle function to assemble individual input elements into a single cluster or to change the values of individual elements in an existing cluster.

In the Waveform and XY Graph lesson, we used the Bundle function to assemble output from arrays and to increment the x-axis values to create single plots (as shown on this slide) or multiple plots.

The following slides show you how to use the Bundle function to change the values of individual elements of a cluster.