LabVIEW Introduction

Digital Wireless Communication Lab:
Physical Layer Exploration using the NI USRP

Useful Tutorial & Course Online

- Tutorial:

https://learn.ni.com/learn/article/labview-tutorial

- Online Training Course:

  - ➢ Core 1:

https://learn.ni.com/learn/learning-path/labview-core-1

  - ➢ Core 2:

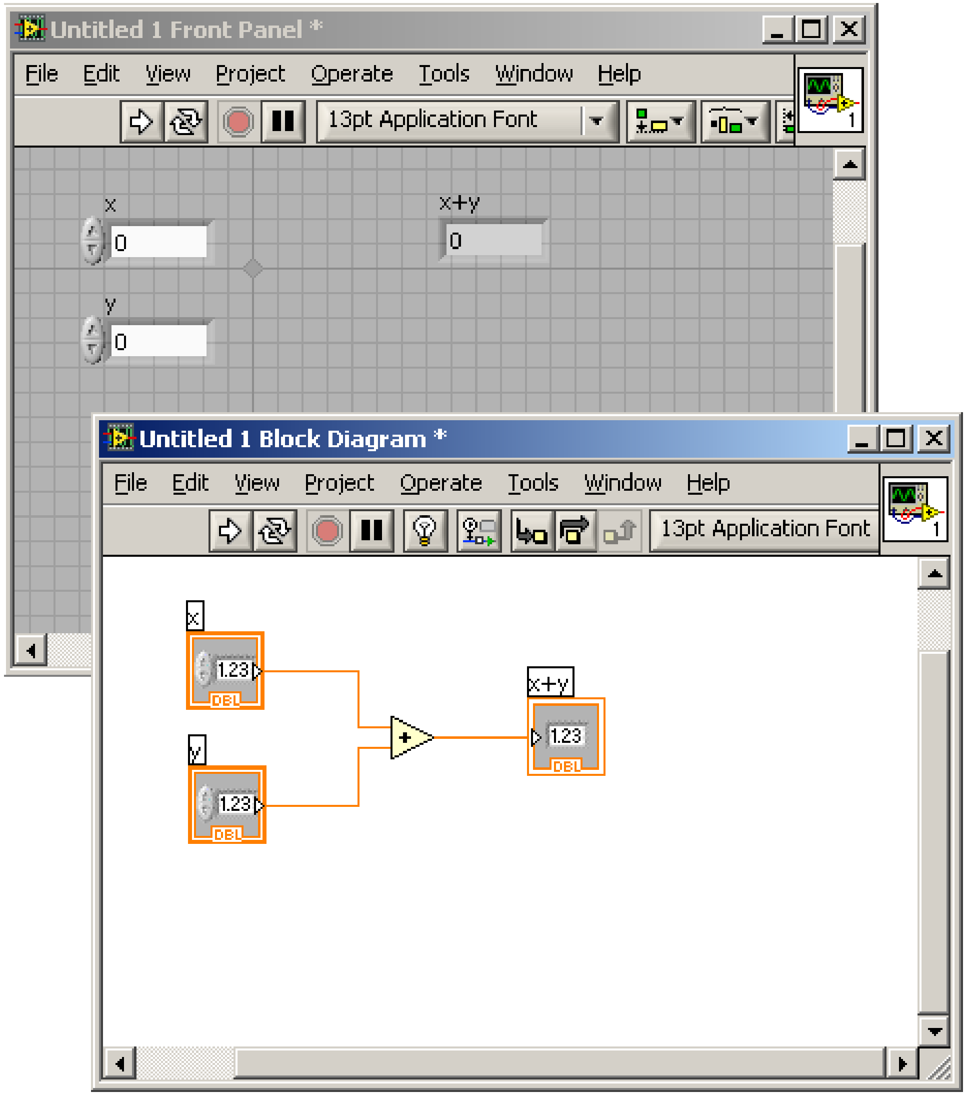
https://learn.ni.com/learn/learning-path/labview-core-2-course

Outline

- Basic Introduction to LabVIEW
  - Intro to LabVIEW – Front Panel
  - Intro to LabVIEW – Block Diagram
- Example – Add2_Nos.vi
- Programming Structures
  - If/Else
  - For
  - While
- Data Structures
  - Arrays
  - Clusters
- Documentation

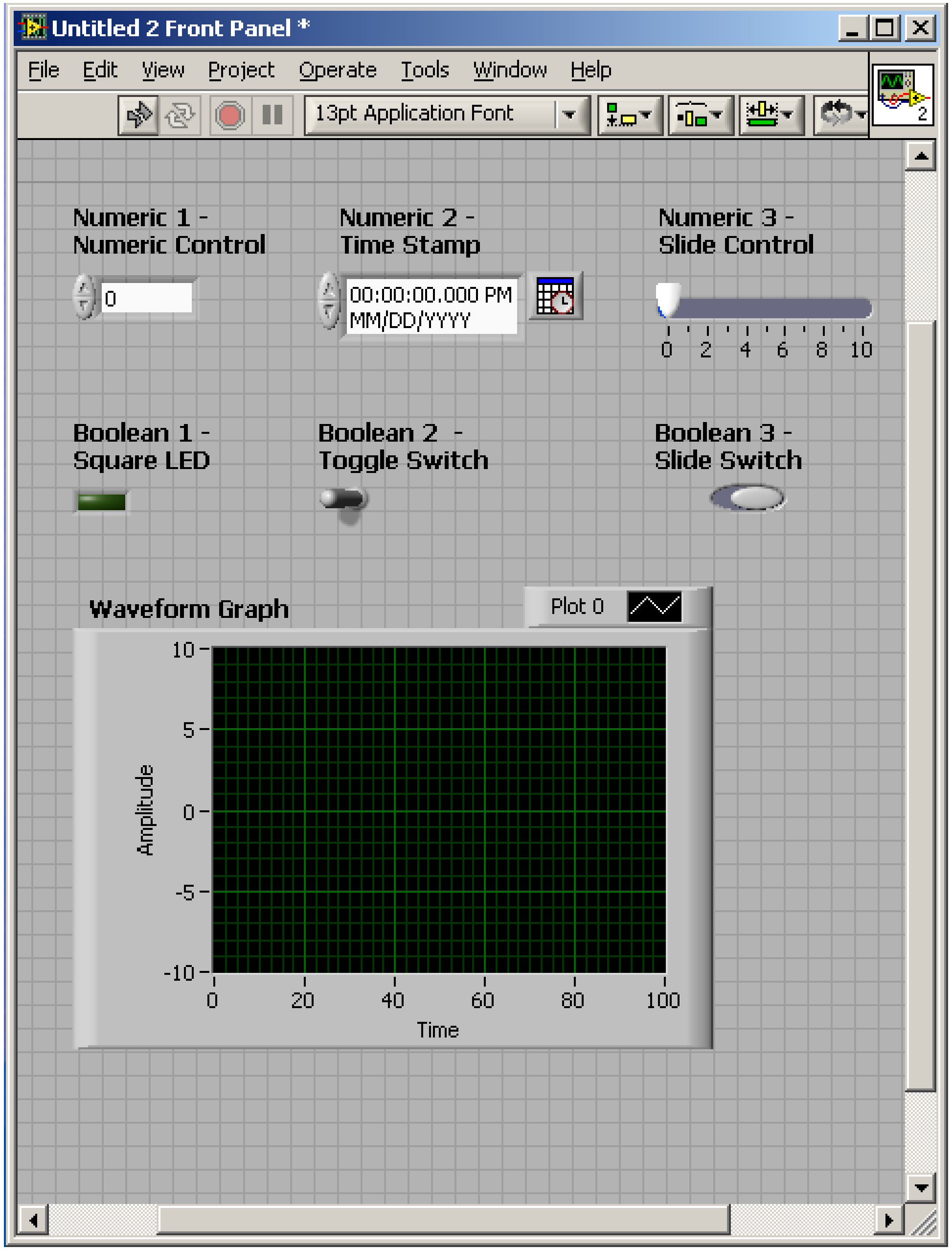
Basic Introduction to LabVIEW

- LabVIEW is a graphical language for programming math and signal processing applications
- ‘Code’ in LabVIEW is in the form of a Virtual Instrument (VI)
- A VI consists of
  - Front Panel: Top-level (user) interface
  - Block Diagram: Actual structure of the code
- Terms: sub-vi, terminal, etc.


Untitled 1 Front Panel *
File Edit View Project Operate Tools Window Help
13pt Application Font
x
0
x+y
0
y
0
Untitled 1 Block Diagram *
File Edit View Project Operate Tools Window Help
13pt Application Font
x
1.23
DBL
y
1.23
DBL
+
x+y
1.23
DBL

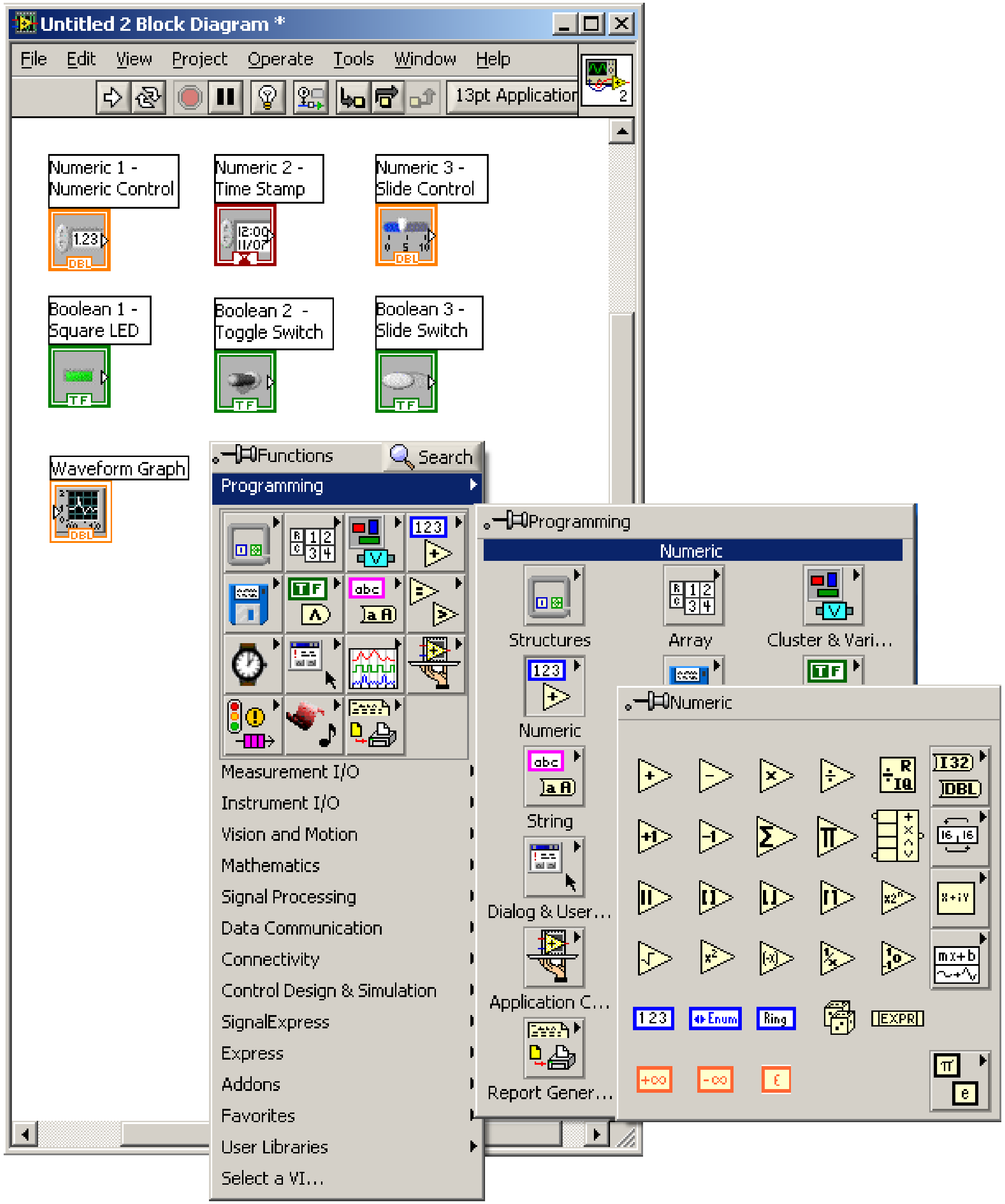
Intro to LabVIEW – Front Panel

- User interface to the code (in block diagram)
  - Controls (Input): Boolean controls, Numerical controls, etc.
  - Indicators (Output): Graphs, Charts, Numerical indicators, etc.
  - Other
    - You can have many other items, such as tab controls, boxes, etc. for organizing your GUI



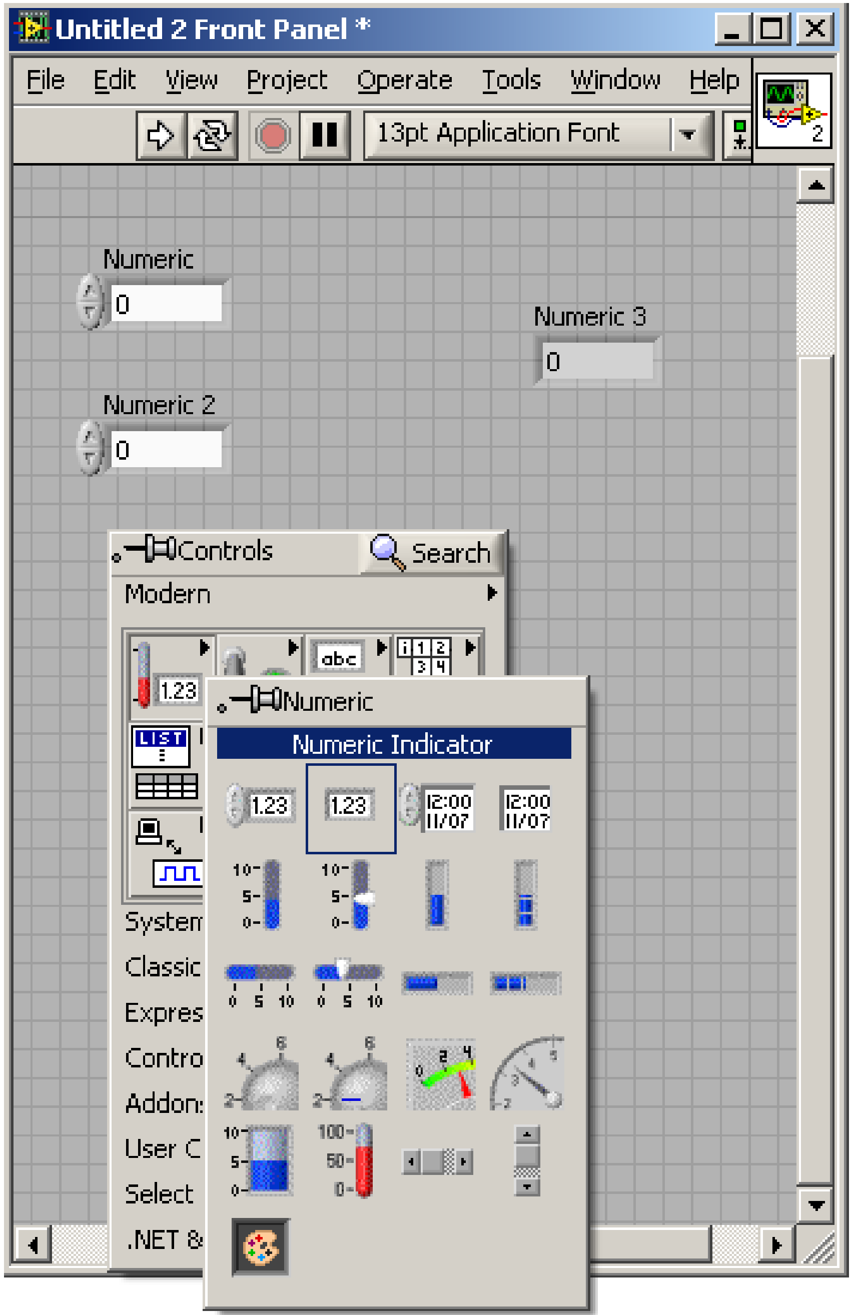
Intro to LabVIEW – Block Diagram

- Structure of code constructed
  - Inputs/Outputs: controls (indicators) from Front Panel are set as inputs (outputs) on the Block Diagram
  - Function Pallete: Math, Signal processing, Loop structures, Arrays, etc



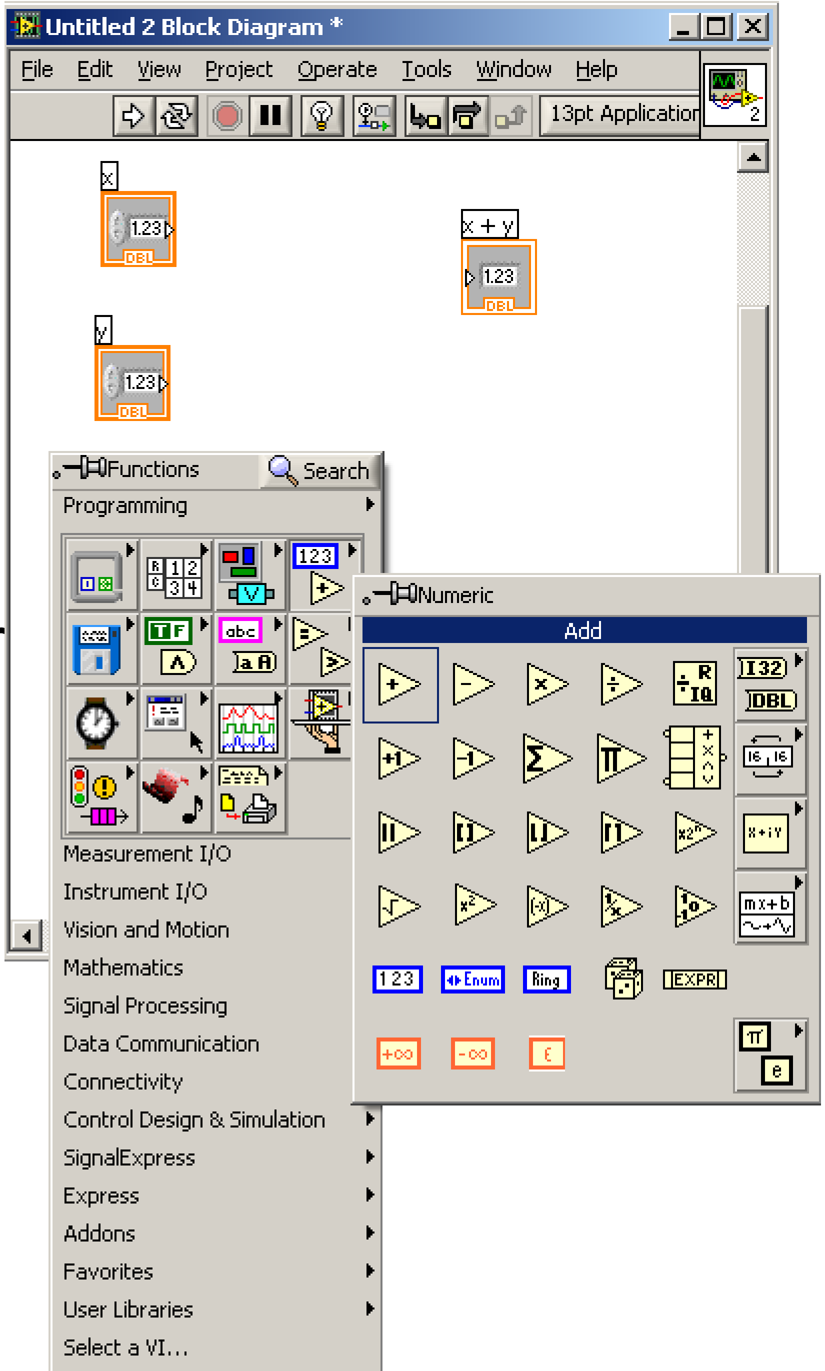
Example – *Add2_Nos.vi*

- Construct a VI to add two nos.
- Step-wise procedure
  - Step 1: Start >> LabVIEW >> Blank VI (from ‘Getting Started’ screen
  - Step 2: On the Front Panel, add two ‘Numeric Controls’ (inputs) and one ‘Numeric Indicator’ (output)
  - Step 3: Name them ‘x’, ‘y’ and ‘x+y’



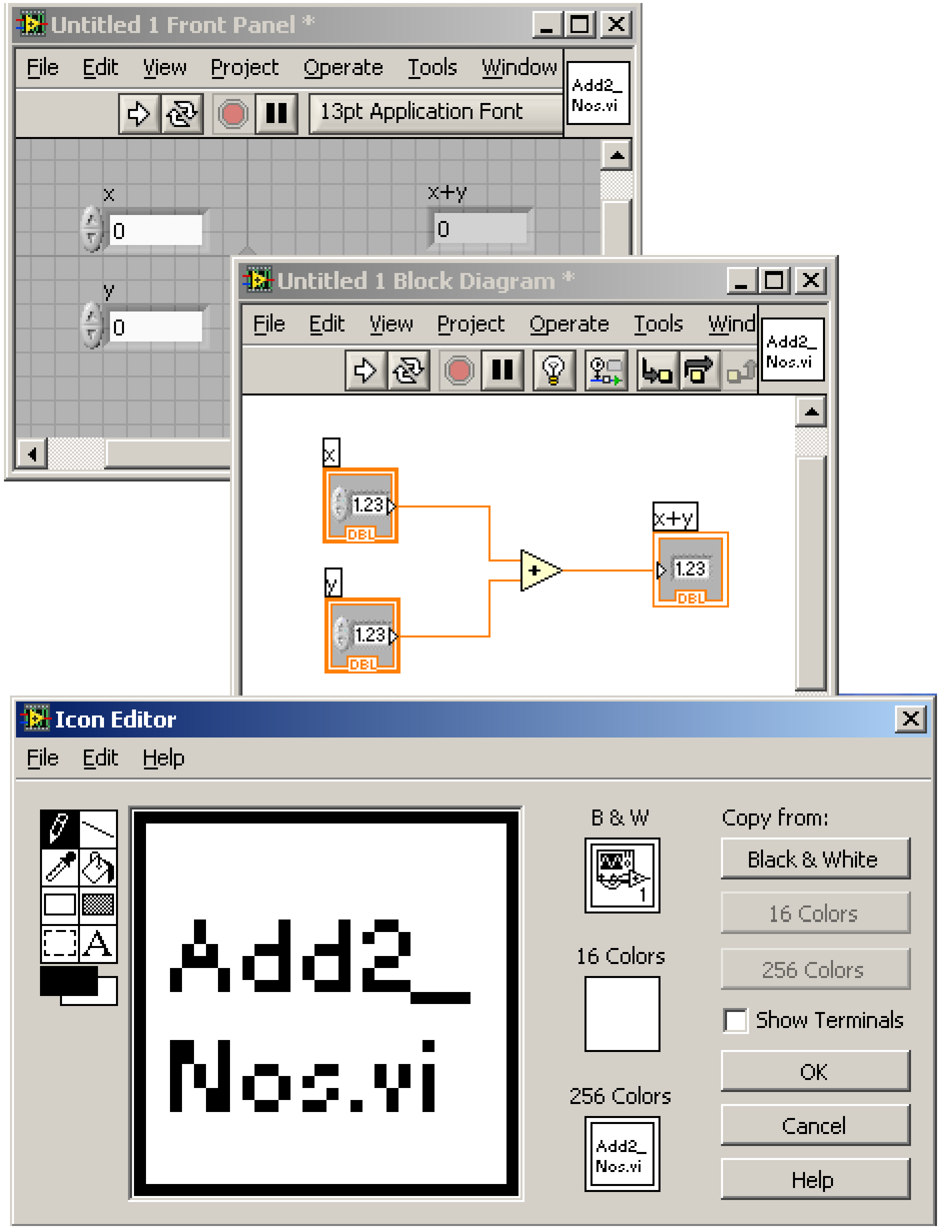
Example – *Add2_Nos.vi*

- Step-wise Procedure (contd.)
  - Step 4: Switch to the Block Diagram – note the inputs and outputs (same names as Front Panel)
  - Step 5: Right-click on the Block Diagram and choose ‘Add’ under Numeric Palette – drag and drop on the Block Diagram
  - Step 6: Connect the inputs and outputs using the mouse for wiring



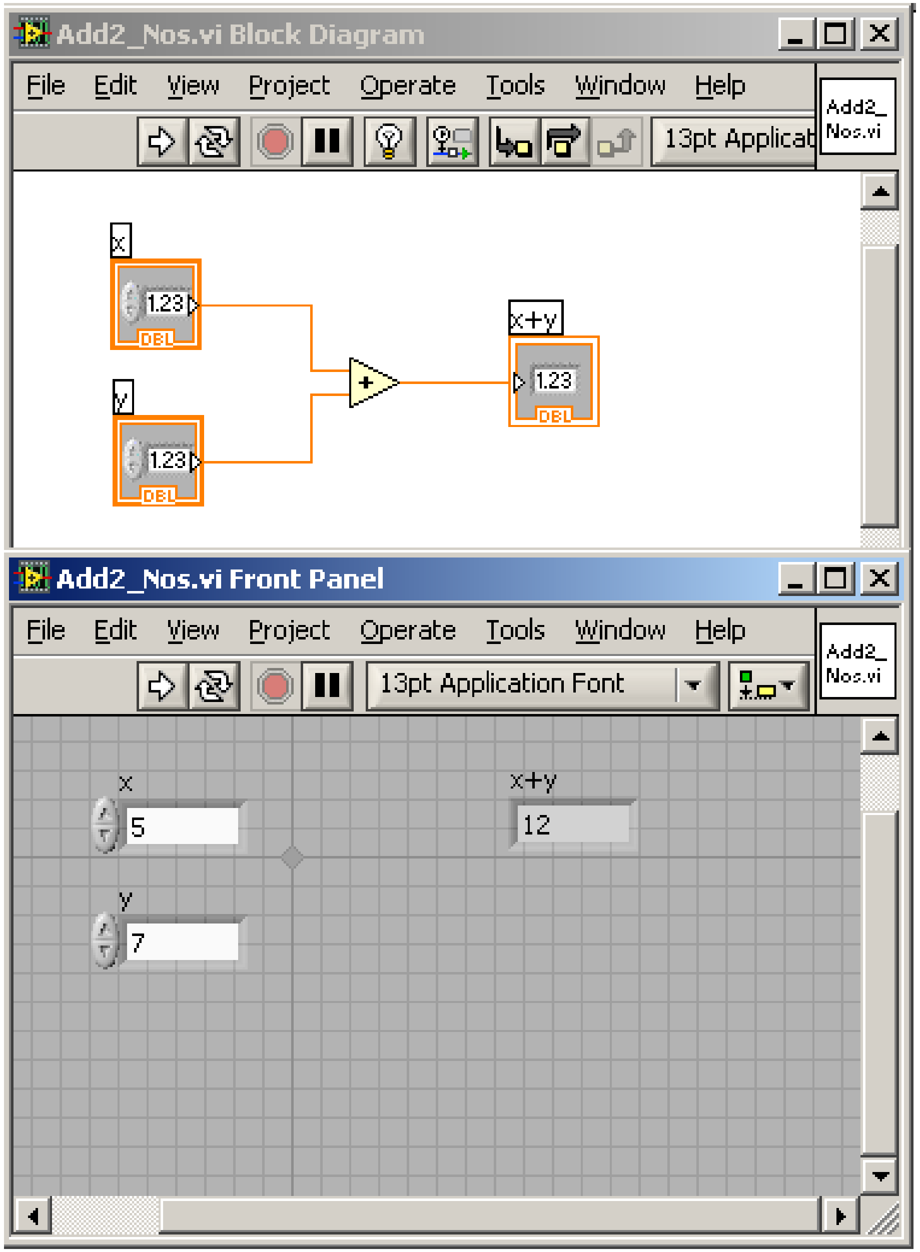
Example – *Add2_Nos.vi*

- Step-wise Procedure (contd.)
  - Step 7: Save the VI as *Add2_Nos.vi*
  - Step 8: Modify the icon (top right corner) by right clicking on the icon and choosing ‘Edit icon’
  - Step 9: After editing the icon, right click on the icon again and choose ‘Show connector’ and connect the inputs and output appropriately.



Example – *Add2_Nos.vi*

- The final VI should look like this
- To run the VI
  - enter x and y values in the Front Panel
  - Click the ⇨ icon in the top left corner of the Front Panel


Add2_Nos.vi Block Diagram
File Edit View Project Operate Tools Window Help
13pt Applicat
Add2_ Nos.vi
x
1.23
DBL
y
1.23
DBL
+
x+y
1.23
DBL
Add2_Nos.vi Front Panel
File Edit View Project Operate Tools Window Help
13pt Application Font
Add2_ Nos.vi

x: 5

y: 7

x+y: 12

Programming Structures

- LabVIEW (like C or Matlab) supports the following structures:
  - If / Else
  - For loop
  - While loop
- These can be found in ‘Structures’ in the Programming Palette, by right clicking on the Block Diagram

Programming Structures – If/Else

- LabVIEW is graphical, so code of the form below is written graphically

```
if{condition = true}
        {Program for true condition}
else (if{condition = false})
        {Program for false condition}
```

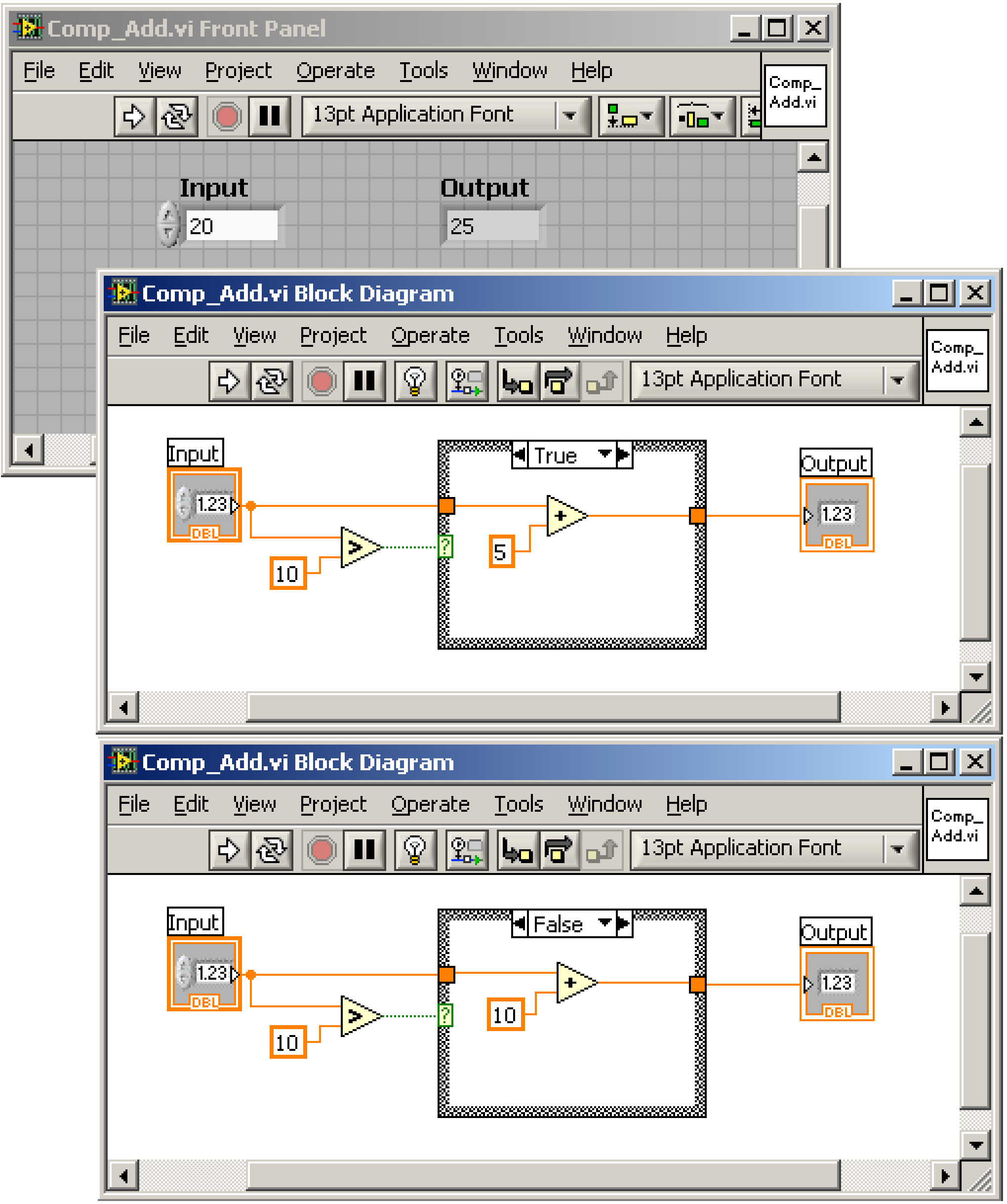
- Basic Steps
  - Use a comparison to produce a true or false, then wire that result to the Case Structure
  - Place code in both the True and False parts of the case structure selected at the top of the structure
  - When the code runs, a true boolean value will run the True case, while a false will run the False case

Programming Structures – If/Else

- Example: Create a VI to add ‘5’ to a number if it is greater than ’10’, else add ’10’.
- Step-wise Procedure
  - Step 1: Open a blank VI
  - Step 2: On the Front Panel, insert a ‘numeric control’ to obtain the input number and a ‘numeric indicator’ for the ouput
  - Step 3: In the Block Diagram, go to ‘Structures’ in the Programming Palette (right click for the Palette)
  - Step 4: Click on ‘Case structure’, drag and drop it onto the Block Diagram. Adjust the size of the structure as needed
  - Step 5: Insert a ‘*Greater?.vi*’ from ‘Comparison’ in the Programming Palette
  - Step 6: Compare the input to a ‘constant’ (Numeric << Programming Palette) set to 10.

Programming Structures – If/Else

- Step-wise Procedure (contd.)
  - Step 7: If the condition (number > 10) = True, set the case structure to ‘True’ and ‘Add’ 5 to the number
  - Step 8: If the condition (number > 10) = False, set the case structure to ‘False’ and ‘Add’ 5 to the number
  - Step 9: Edit the icon and the connector, after saving the VI

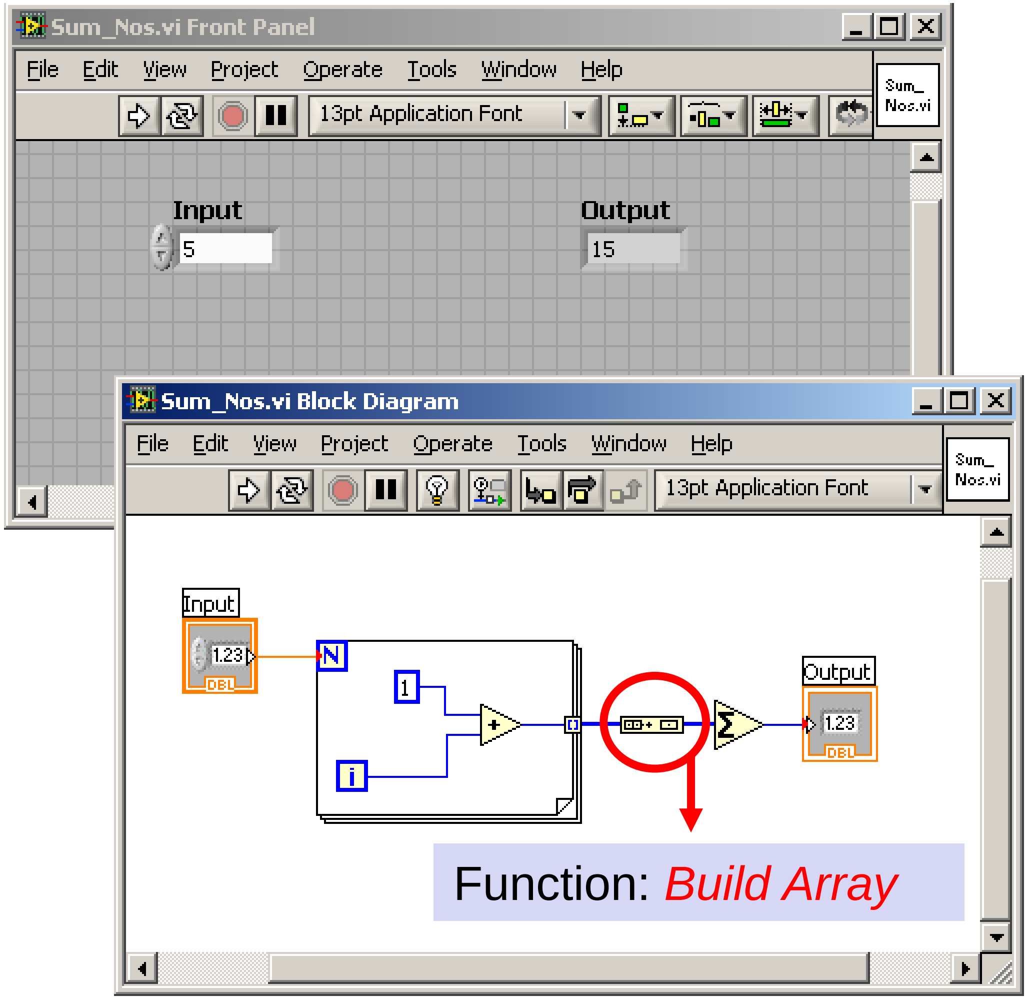


Programming Structures – For Loop

- ‘For loop’ in LabVIEW is the same as in other languages
  - Each for loop has a ‘Loop Count’ input to the loop and a ‘Loop Iteration’ output inside the loop
  - Simply wire up the number of loops you require to Loop Count and put appropriate code inside loop
- Example: Create a VI to output the sum of numbers from 1 to the number input.
- Step-wise procedure
  - Step 1: Open a Blank VI
  - Step 2: In the Front Panel, insert a ‘numeric control’ and name it ‘input’ and a ‘numeric indicator’ and name it ‘output’

Programming Structures – For Loop

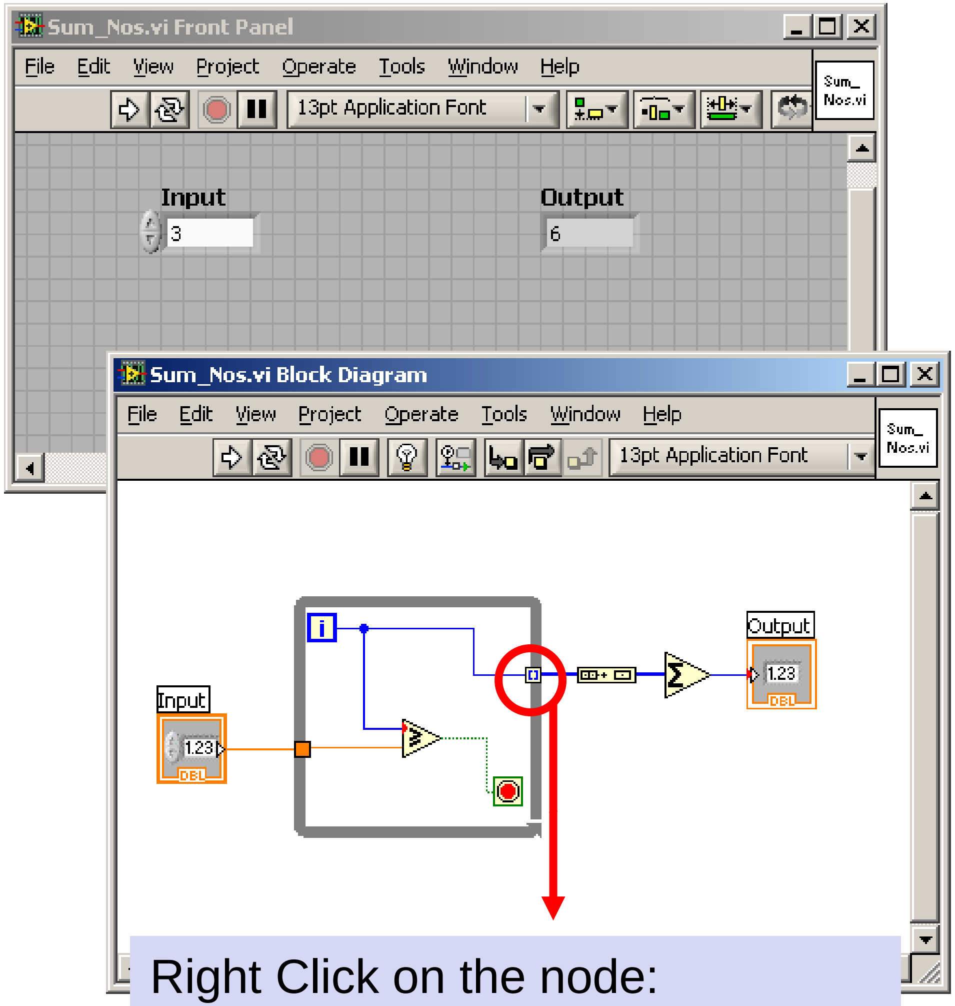
- Step-wise procedure
  - Step 3: On the Block Diagram, drag and drop a ‘For Loop’ from Structures in the Programming Palette
  - Step 4: Connect the ‘Loop Count’ to ‘input’ and the ‘Loop Iteration’ to an ‘*Add.vi*’ (the loop iteration starts from 0 to (N-1)) and build an array with the results.
  - Step 6: Sum the array elements.
  - Step 7: Edit the icon and connector.



Function: *Build Array*

Programming Structures – While Loop

- The While Loop in LabVIEW is similar to ‘For Loop’
  - Loop runs until the stop condition is met
  - Repeat the previous exercise with a While Loop that stops when iteration count is ≥ input number
- Hints:
  - Right-click on stop button – can be continued if true
  - Can right-click on For Loop and replace with While



Right Click on the node:
→ Change *Tunnel Mode* between *Last Value* and *Indexing*.
See what changes at output.

Data Structures

- Arrays
  - Multidimensional collections of like data
  - Vectors, matrices, array of booleans, etc.
- Clusters
  - Collections of unlike data used for conveniently transporting the data from one place to another
  - Similar to the idea of a Struct in C or Matlab

Data Structures - Arrays

- You have already been introduced to arrays – when talking about the ‘For’ and ‘While’ loops
- Can have arrays of virtually anything
  - Controls, indicators, numerics, booleans, etc.
- Can specify many dimensions
- Make an array – new VI
  - Right-click on the Front Panel and choose Array under “Array, Matrix...”
  - This is a shell array, place numeric control inside
  - Add values inside array to use in Block Diagram

Data Structures - Arrays

- Useful Array functions
  - Array Size – gives you an integer of the array size
  - Build Array – allows you to concatenate arrays and other data together into one array
  - Max & Min – gives you value and index of max/min
  - Array Subset – allows you to resize an array given new dimensions and starting index
  - Many more...

Data Structures - Clusters

- You can mix various types of data into a single cluster – mostly for passing to/from subVIs
- Let's examine a cluster
  - Place the control “Error in 3D.ctl” found in “Array, Matrix...” on the Front Panel
  - This control is heavily used and consists of 3 types
    - Boolean status – true for no error, false for error
    - Code – allows for a particular numeric code to be passed
    - Source – a string describing the source of the error
- In the Block Diagram, use Bundle and Unbundle to access elements of the cluster

Documentation

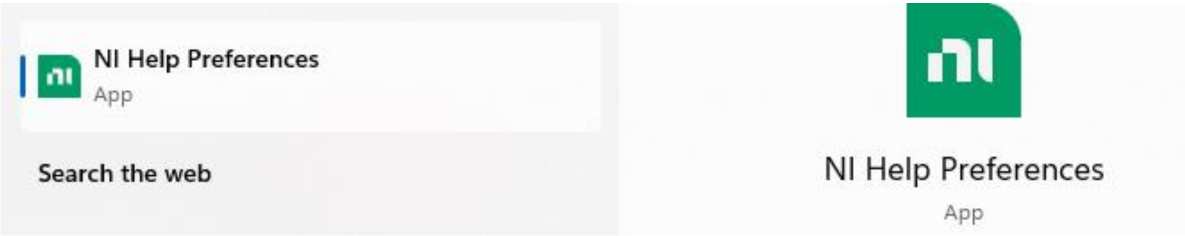
- Always document your code
  - Your lab grade depends on it!
- Many methods of documentation
  - Uniquely label all controls/indicators
    - Self-explanatory and very easy
  - Free labels
    - Easiest documentation type possible
    - Double click on block diagram and start typing
    - Use this to explain code when anything more complicated than simple math is being done
  - Icon and connector editing
  - Edit the Context Help (will not affect your grade)

Documentation

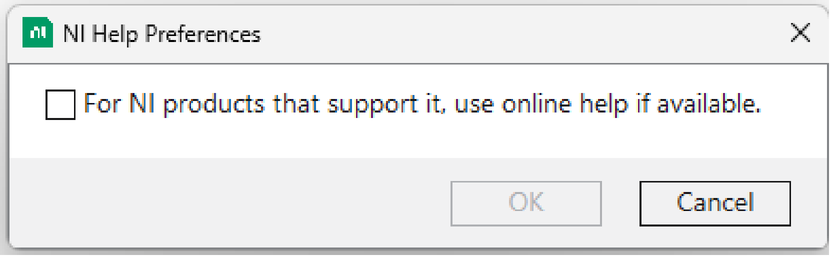
- Icon Editing
  - many resources – see links below
  - easy way
    - copy an icon from your favorite subVI
    - make it your own
      - change the color, text etc. to meet your needs
- Links for Icon Editing
  - http://www.ni.com/devzone/idnet/library/icon_art_glossary.htm
  - http://zone.ni.com/reference/en-XX/help/371361B-01/lvconcepts/creating_subvis/

How to use NI offline Help on lab computers

- Search ***NI help preference*** in windows

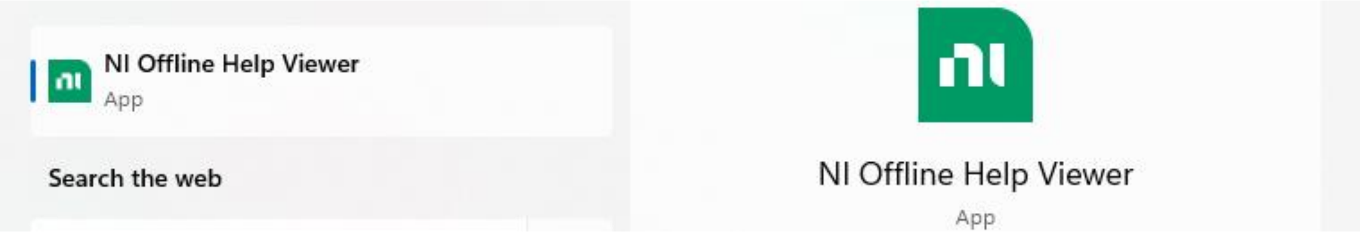


- Disable use onlline help as below



□ For NI products that support it, use online help if available.

- Search and open ***NI Offline Help View***. You may search functions in it now.



Help in LabVIEW

- Many, many, many resources available
- Getting Started screen when starting LabVIEW
  - LabVIEW Help – all functions with descriptions
  - Getting Started with LabVIEW document
- Web Resources (www.ni.com)
  - Support Portal
  - Discussion Forums
- Examples!

Summary

- You now have the basics needed to start programming in LabVIEW
- You will be introduced to more concepts and examples in Lab 1.1
- Good luck!