



# LabVIEW Academy

---

LabVIEW Certification (CLAD Test) - Answers



## Table of Contents

<b>Problem Solving.....</b>	<b>2- 15</b>
<b>Navigating LabVIEW.....</b>	<b>16- 39</b>
<b>Troubleshooting and Debugging VIs.....</b>	<b>40- 62</b>
<b>Implementing a VI .....</b>	<b>63- 86</b>
<b>Relating Data .....</b>	<b>87- 106</b>
<b>Storing Measurement Data.....</b>	<b>107- 126</b>
<b>Developing Modular Applications .....</b>	<b>127- 144</b>
<b>Acquiring Data .....</b>	<b>145- 166</b>
<b>Instrument Control .....</b>	<b>167- 188</b>
<b>Common Design techniques and Patterns.....</b>	<b>189- 221</b>
<b>Communicating Among Multiple Loops .....</b>	<b>222- 245</b>
<b>Event Programming .....</b>	<b>246- 262</b>
<b>Improving an Existing VI .....</b>	<b>263- 281</b>
<b>Controlling the User Interface .....</b>	<b>282- 300</b>
<b>Advanced File I/O Techniques .....</b>	<b>301- 324</b>
<b>Creating and Distributing Applications .....</b>	<b>325-341</b>

Complete the following sentence.

**Maintenance** is the ongoing process of resolving programming errors.

*Maintenance is defined as continuing to resolve issues as they arise.*

Complete the following sentence.

Make sure to **test** your implementation with data that is both logical and illogical for the solution you created.

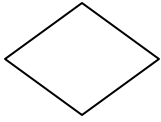
*Testing logical data verifies that the inputs produce the expected result. By testing illogical data, you can test to see if the code has effective error handling.*

**Problem solving** skills are essential to creating solutions in LabVIEW.

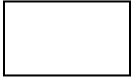
*Making a program is usually based on solving a problem. If the programmer has sub-par problem solving skills, he/she will have a difficult time creating efficient code.*

Which flow chart block is associated with...

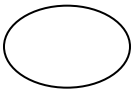
1. A decision



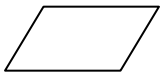
2. An action or process



3. Start and end



4. Data



A State Transition Diagram is a specific type of

- a. Algorithm
- b. Flowchart**
- c. Spreadsheet
- d. Table

*A State Transition Diagram is a specific type of flowchart that is commonly used when creating LabVIEW State Machines.*

Which of the following is **NOT** a step in the Design Process?

- a. **Define the problem**
- b. Identify the inputs
- c. Identify the outputs
- d. Identify any additional requirements
- e. Design the algorithm/flowchart/state diagram

*Define the problem is listed as part of the Software Development Method, not the Design Process.*

*All other options are defined in the manual as being parts of the Design Process.*



A karaoke user interface is developed that lets a user add their name and the song they want to sing to a queue. The interface displays the singer and song that is next in line.

Identify the inputs.

*The inputs to this system are the name of the user and their song.*

Identify the outputs.

*The output of this system is the next singer and their song.*

The Design step is an important part of the Software Development Method. Fill in the blanks in the following paragraph.

The **inputs** indicate the raw data that you want to process during the problem solving process.

The **outputs** represent the result of the calculation, processing or other condition that the problem solving process implements.

In the **implementation** stage, you create code for your algorithm. Displaying an algorithm as a **flowchart** is a good way of translating an algorithm into code.

*During implementation you translate an algorithm into code. Because LabVIEW is a graphical programming language, working from a flowchart is a good way to translate an algorithm into LabVIEW code.*

The implementation of a state transition diagram requires which of the following?

- a. Converting states and transitions to code**
- b. Breaking down each state into its elements**

c. A full test and verify scheme

*Test & verify come after implementation*

d. An output from each state

*Not all states need to give an output*

An alarm system measures wind speed and direction every 30 seconds and gives a signal indicating when the wind speed from the north is above 30 miles per hour.

a. What are the inputs to this alarm system?

*The inputs are the wind speed, direction, and the alarm limit (30 mph and northerly).*

b. What are the outputs to this system?

*The expected output is the alarm signal. Other potential answers may include the wind speed and direction, if additional processing of the wind speed and direction elsewhere.*

c. What is the relationship between the inputs and outputs of this system?

*The signal output will be true when the wind speed and direction are above 30 mph and northerly, respectively. This signal will be processed every 30 seconds.*

From the given scenario, choose the option which identifies only the elements problem.

Assume you must cure a material at a certain temperature for a set amount of time in a furnace. The exact temperature will be recorded along with the exact amount of time the metal remains in the furnace.

- a. You must recognize the cure time, cure temperature as well as some other variables such as time of day.
- b. You must remember that along with the cure time and temperature, the data is also being recorded.
- c. The entire scenario is the problem.
- d. The elements of the problem are only the cure time and cure temperature.**

*Choice a is incorrect because the time of day has no bearing on the variables we are interested in.*

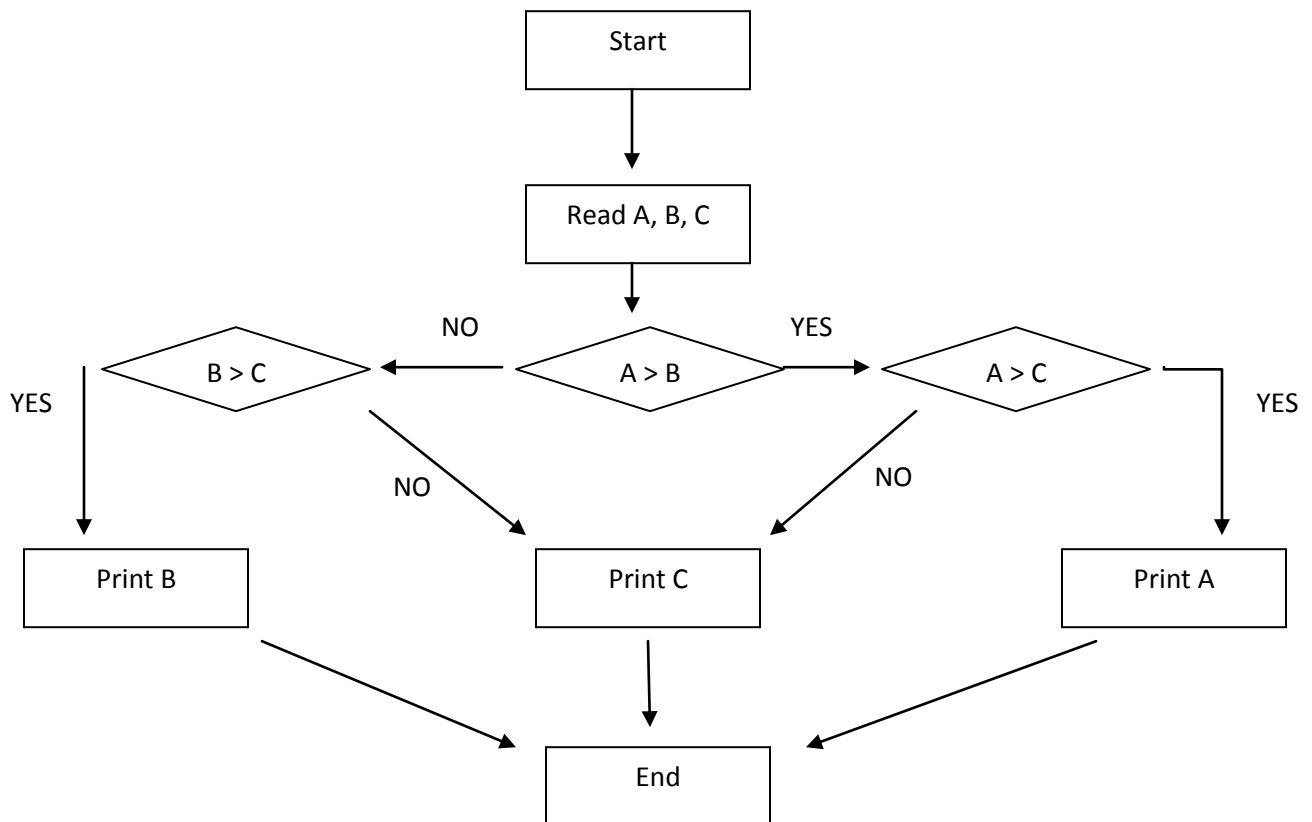
*Choice b is incorrect because the statement says that the data will be recorded. It does not say that you need to identify how to record it or interact with the recording at all.*

*Choice c is incorrect because the problem needs to be isolated, and therefore cannot be the whole scenario.*

*Choice d is correct because the two variables that are specifically identified in the scenario are the cure time and cure temperature.*

Design a flowchart to find the largest of three numbers A, B, and C.

*Example Solution:*



Match the following steps in the Software Development Method to their purpose.

1. Defining the problem...
2. Designing the algorithm...
3. Implementing the design...
4. Testing and verifying the implementation...
5. Maintaining and updating the implementation...

1. c Allows you to remove extraneous factors and focus on the core problem.
2. b Sets the inputs, outputs, and additional requirements for the problem so a solution can be developed.
3. e Is where you create and build the code for your algorithm or flowchart.
4. d Allows you to see if your algorithm gives the expected results for valid and invalid data.
5. a Is an ongoing process of resolving errors and adding changes to the original solution.



What does LabVIEW stand for?

*Laboratory Virtual Instrumentation Engineering Workbench*

What is a “VI” and what does it mean with respect to LabVIEW?

*VI stands for Virtual Instrumentation. Computers can simulate the functionality of instruments virtually, via software, using LabVIEW.*

**Front Panel**, **Block Diagram**, and **Icon/Connector Pane** are the three parts of a LabVIEW VI.

On a connector pane, what is assigned to the terminals?

*Good programming practice dictates that controls are assigned to the left side and indicators on the right side. They should try to avoid using the top and bottom terminals but if it is necessary, they are usually used for controls or error wiring.*

What are the main differences between a Modern controls & indicators, Classic controls & indicators, and System controls & indicators on a front panel?

*System controls and indicators take on their appearance from the operating system where as Modern and Classic are set via the LabVIEW programmer. The difference between Modern and Classic controls and indicators are that Classic controls maintain their backwards compatibility to older versions of LabVIEW and display correctly on low color displays*

What is a subVI and what are the requirements to use a subVI?

*A subVI is a VI used in another VI. In order for a VI to be used as a subVI, it must be saved and its connector pane must be setup properly.*

In LabVIEW terminology, a **target** is any device that can run a VI.

*This term is referred to in the Project Explorer introduction.*

True or False: Boolean controls and indicators represent only a True or False status.

***False:*** Booleans can only represent Boolean values.



True or False: All LabVIEW VIs must be created within a LabVIEW project or it will be impossible to edit them after the first save.

**False:** *A VI can be edited as frequently as wanted without being part of a Project. A project is only needed in some situations, such as when using certain types of variables and creating an EXE or installer.*

Which of the following can be associated with the block diagram? Choose all that apply.

- a. **Run button**
- b. **Stop button**
- c. **While loop**
- d. **Numeric control**
- e. **Highlight execution**

Without the use of property nodes, how can the Front Panel Toolbar be made invisible for a particular VI while it is running?

- 1) Click **File >> VI Properties**
- 2) Select **Window Appearance** from the **Category** drop down menu
- 3) Click **Customize**
- 4) Uncheck **Show toolbar when running**

Which of the following can be associated with the front panel?

- a. **Run button**
- b. **Stop button**
- c. While loop
- d. **Numeric control**
- e. Highlight execution

True or False: Under the Files page in your Project Explorer window, project operations reflect and update the files on the local machine's disk.

**True:** *When viewing the project with the Files page option, all changes affect the files on disk. When viewing the project in Items page mode, the user is just viewing the project tree.*

True or False: You have to add dependencies and build specification items to your project in order to see them as family options in the Project Explorer window.

**False:** *The Dependencies and Build Specifications family items are visible by default in the Project Explorer window. The user doesn't have to add a dependency or build specification in order to see the family item in their Project Explorer window.*

Which of the following does **NOT** appear on the front panel of a LabVIEW VI? Choose all that apply.

- a. **subVI**    *has no indicator on the front panel of the VI it is used in.*
- b. waveform chart
- c. file path control
- d. text ring
- e. numeric indicator
- f. label
- g. gauge

Highlight execution allows the user to:

- a. follow and track the flow and processing of data on their block diagram.
- b. debug the source of delays and other errors in their code.
- c. to see which VIs are executed first in parallel processes.
- d. all of the above
- e. A only
- f. both A & B**

*Though not clearly stated in the manual, both of these features are inherent on how highlight execution can help the user.*



You can add items (i.e. files and folders) to a project through what methods?

- a. Right-click on My Computer and select Add>>File.
- b. Right-click the target and select New>>VI.
- c. Drag and drop the VI icon in the upper right corner of a front panel or block diagram.
- d. Drag and drop a folder or file from the file system of the machine onto the project's target.
- e. **All of the above.**

*All of these are listed in the section.*

Decide if each of the VIs listed below are used for acquiring data or analyzing data.

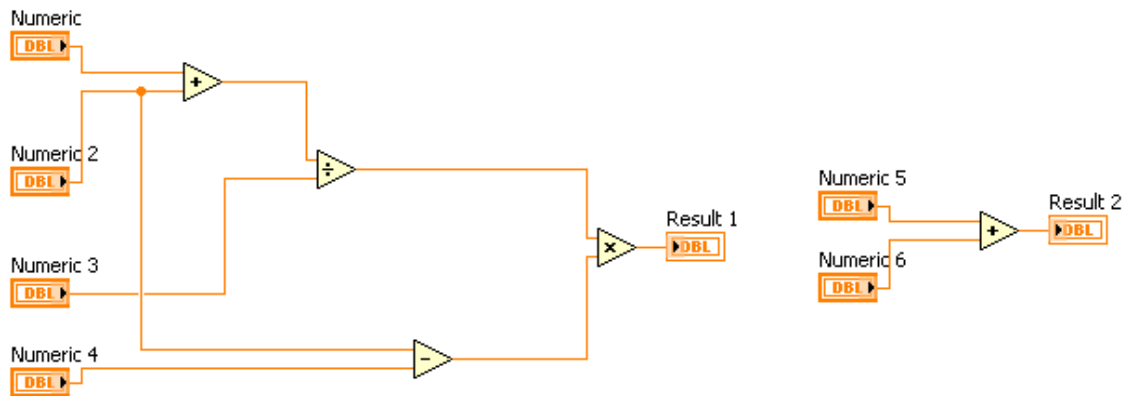
1. Acquire Data
  2. Analyze Data
- 
- a. DAQ Assistant
  - b. Instrument I/O Assistant
  - c. Simulate Signal
  - d. Amplitude and Level Measurements
  - e. Statistics
  - f. Spectral Measurements
  - g. Read From Measurement File
  - h. Tone Measurements
  - i. Filter

*Acquire Data* applies to the DAQ Assistant, Instrument I/O Assistant, Simulate Signal and Read From Measurement File. These VIs are meant to acquire data only and do not manipulate the data.

*Analyze Data* applies to Amplitude and Level Measurements, Statistics, Spectral Measurements, Tone Measurements, and Filter. These VIs perform some action on the data.

True or False: Result 1 will be displayed before Result 2.

Result 1 will be displayed before Result 2.




**False:** *There is no way to determine which operation will occur first. Placement on the block diagram does NOT specify execution order*



Complete the following sentences.

The above icons are examples of **Boolean** controls and indicators. The **Vertical Toggle** and **Stop** are controls and the **Round LED** is an indicator. This data type has **2** parts.

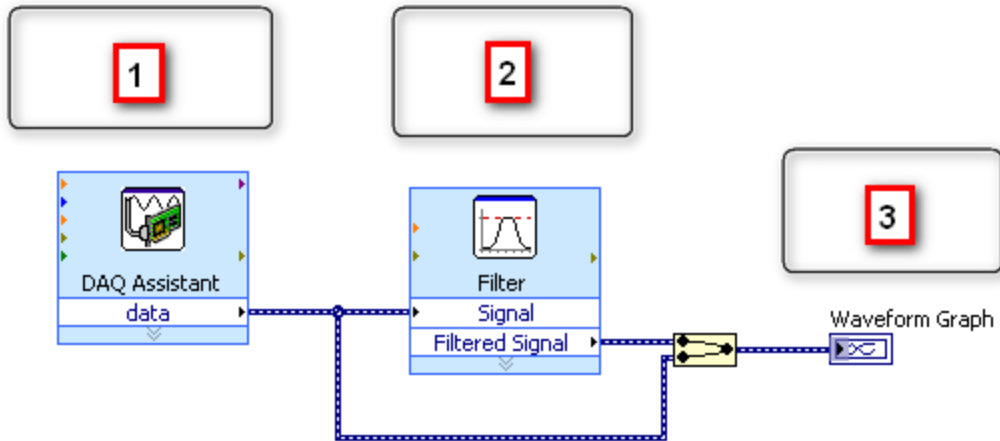
*The Vertical Toggle and Stop require manipulation by the user making them controls. The Round LED displays output and therefore is an indicator. Boolean data has two parts such as true/false or on/off.*

A user is trying to select a control in the block diagram to move it around; however the cursor is stuck as a spool.  How can the user fix this?

*The problem is that the user has turned off Automatic Tool Selection and has the probe tool selected. In order to fix this, the user should select View » Tools Palette from the Menu bar and select the Automatic Tool Selection button:*

*Automatic Tool Selection →*





Most LabVIEW VIs have three main tasks. Match each number with the phrase that should appear in each number's respective box.

1. **c**

2. **a**

3. **b**

a. Analyzing the acquired data

b. Presenting the result

c. Acquiring some sort of data

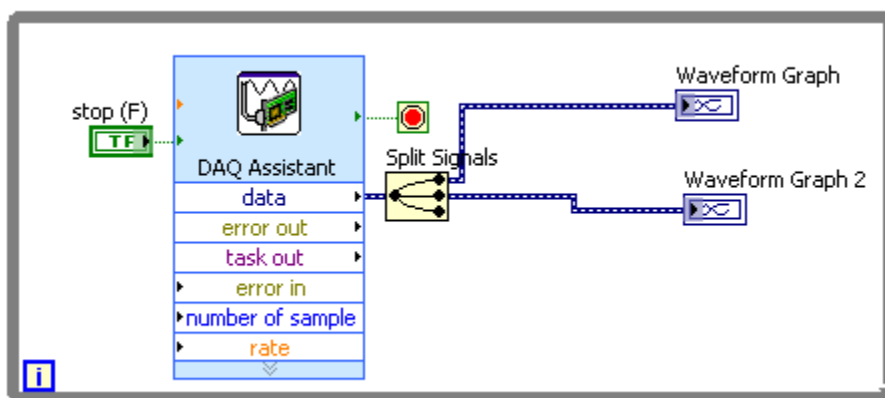
*The three main tasks of most VIs are acquiring data, analyzing data, and presenting the result. The DAQ Assistant acquires the data. The Filter analyzes the data and the waveform graph displays the result.*

On the line below each picture, label if the picture is a front panel or block diagram.



**Front Panel** – *Displays inputs and outputs to the user*

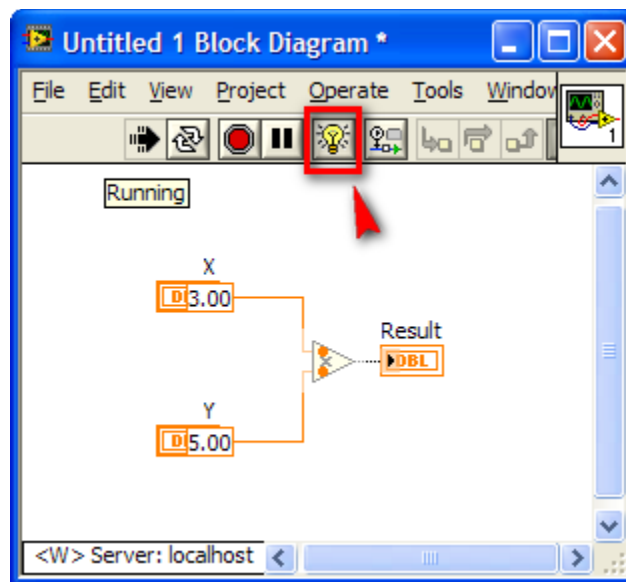
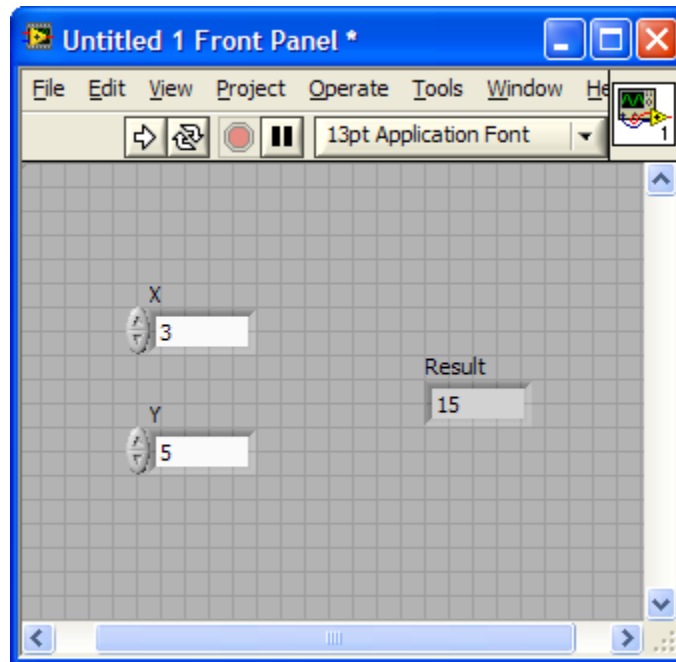
---



**Block Diagram** – *Shows the code that controls the front panel*

---

The following code simply takes 2 numbers and multiplies them together:



The code appears to be working fine; however, it runs slowly. Determine what is happening and how it can be fixed.

*The code runs slowly because highlight execution is turned on. Simply turn click the light bulb button and the code will run faster.*



What kind of problems can highlight execution cause when debugging data acquisition or time dependent applications?

*Student's answer should indicate that highlight execution significantly reduces the speed at which a VI executes and therefore can lead to buffer overflows and errors for applications that are time dependent.*

What does the acronym NaN mean? What is significant about this value in a VI?

*Not a Number. When this value is present, it invalidates all subsequent operations that are dependent on that data.*

How does the VI Hierarchy help when debugging?

*The VI Hierarchy helps find unwired subVIs. Unlike wired functions, unwired VIs do not always generate errors unless you configure an input to be required. If you mistakenly place an unwired subVI on the block diagram, it executes when the block diagram does. Consequently, the VI might perform extra actions.*

**Highlight Execution** allows you to watch the data move through the block diagram.

*Execution highlighting displays the data flow as the VI runs and will show where errors originate.*

Complete the following sentences with the appropriate troubleshooting tool.

Use a **probe** to check intermediate values on a wire as a VI runs.

Place a **breakpoint** on the block diagram to pause execution at that location.

*Use the probe tool to click any wire. The Probe window appears. LabVIEW increments the numbers in the Probe window automatically and displays the corresponding number in an icon on the wire you clicked.*

*Use the breakpoint tool to click nodes or wires. Place a breakpoint on the block diagram to pause execution after all nodes on the block diagram execute.*

True or False: The information in error wires are organized in a cluster?

**True:** *You can unbundle the error wire cluster to get different pieces of information such as error status, source, and error code.*

True or False: Error wires could be used to establish ordered execution?

**True:** *Ordered execution in LabVIEW is done via the data flow model. Once a node has current data at its terminal, the node will execute. You can also use sequence structures to establish ordered execution.*

What will cause a broken wire?

- a. Connecting a control to an indicator.
- b. Connecting a control to another control**
- c. Connecting a double control to an integer indicator
- d. Connecting an array output into a cluster bundle input

*Choice a will not produce a broken wire and is a good way to debug controls.*

*Choice c will produce a coercion dot, not a broken wire.*

*Choice d will not produce a broken wire because this is an accept operation.*



Where can LabVIEW users commonly and easily find free, fully functional programs that can be used as found or modified to suit particular needs? (Choose all that apply.)

**a. Example Finder**

**b. Developer's Zone**

c. LabVIEW help

*LabVIEW Help will teach you how to use certain VIs, but there is no written code included.*

d. LabVIEW forums

*The online discussion forums will occasionally have code posted, but this is not tested and only rarely found.*

**[www.ni.com/support](http://www.ni.com/support)**

There are several resources included in LabVIEW that provide assistance for programming effectively. List two of these resources and briefly explain the functionality of each resource.

- a. **Context Help** – the context help window displays basic information about LabVIEW objects when you move the cursor over each object.
- b. **Example Finder** – Contains examples that demonstrate how to use LabVIEW to perform a wide variety of test, measurement, control, and design tasks.

*In order to become good programmers, students need to be able to solve their own problems. This question highlights two methods that allow students to work through programming issues more independently.*

What information does the Context Help window provide?

*The Context Help window shows the icon for subVI's, functions, constants, controls, and indicators. It also shows the labels for terminals (required, recommended, or optional). It also provides the title and a brief description of the item.*

Please select all methods to reach the LabVIEW Help Window.

- a. **Control + shift + ?**
- b. **Help Menu > Search the LabVIEW Help**
- c. **Right-click a VI in the block diagram > select Help**
- d. **Navigate to Programs > National Instruments > LabVIEW x.x> LabVIEW Help**

*All of these methods take you to LabVIEW Help.*

The context help box contains which of the following when the mouse is hovering over a subVI?

- a) Inputs and outputs for the subVI
- b) Name of subVI
- c) Full description of subVI's function
  - *False, there is a brief description and a link to the full description, but the full description is not displayed.*
- d) Link to LabVIEW help
- e) a, b and d**
- f) All of the above

What does a broken arrow on the run button mean?

- a) The error cluster wire must be wired
  - *Wiring the error cluster through the code is highly recommended, but not required for a VI to run. However, an error wire that is partially wired may cause a broken arrow.*
- b) Front panel elements are missing
  - *Elements on the front panel are directly linked to icons on the block diagram. Therefore, if an element is not shown on the front panel, it is not present in the block diagram either. If a control is changed to a constant, then the front panel object disappears because it is not needed on the front panel. Missing, or seemingly missing, front panel objects will not cause a broken arrow.*
- c) **There are errors in the block diagram**
  - *Some errors will prevent the code in the block diagram from being able to run, and a broken arrow will appear. Click on the broken arrow to view the errors.*
- d) **The VI cannot run as is**
  - *There is some error present that keeps the code from being able to execute.*

True or False: Examples found in the Example Finder can be modified.

**TRUE:** *Examples can be modified. Be sure to save the modified example under a different name. If an example is saved over by the modified example, the original example is no longer available on that computer. In other words, the modification is permanent. (The original can be found online at ni.com in the NI Developer Zone)*

List at least three utilities provided with LabVIEW that allow users debug issues with their code.

1. *Highlight Execution*
2. *Probe Tool*
3. *Single-Stepping*
4. *Breakpoints*
5. *Suspending Execution*
6. *Context Help*
7. *LabVIEW Help*
8. *Error Handling/Checking*
9. *Explain Error feature in the Help Menu*

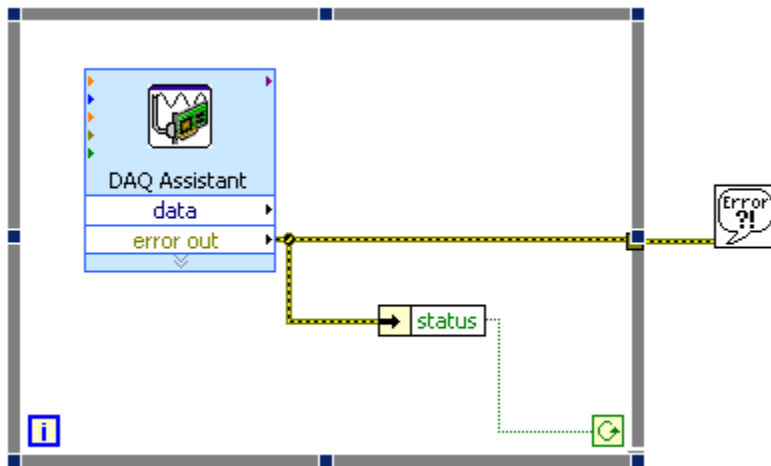


Match the step buttons with their description.

1.   a   Step Out
2.   b   Step Into
3.   c   Step Over








How many times will this loop execute? Explain your reasoning.

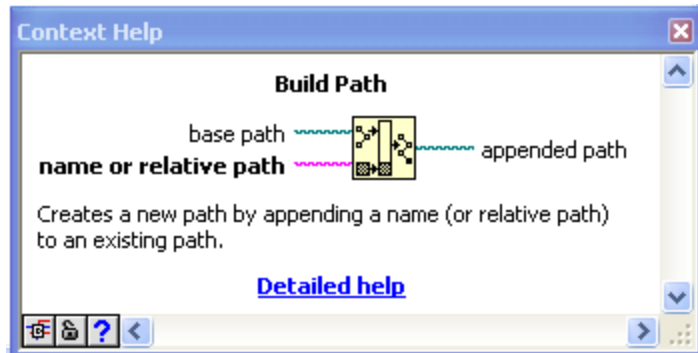


*If the DAQ Assistant executes without error then the loop will execute once because the while loop's condition terminal is set to "Continue if True" and the error status is false (right click to change its state). If the DAQ Assistant executes with error then the loop will execute until the abort button is pressed (infinite loop because the status of the error is true) or until the error clears (status changes to false).*

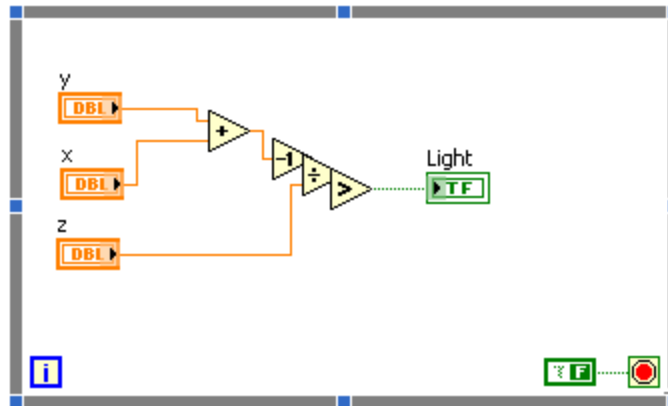
Match the following buttons with their function:

1.  **Show/Hide Context Help**
2.  **Highlight Execution On/Off**
3.  **Step Into**
4.  **Step Out**
5.  **Step Over**

When using context help to learn more about a VI, some inputs are bold while others are not. Why are these inputs bold?



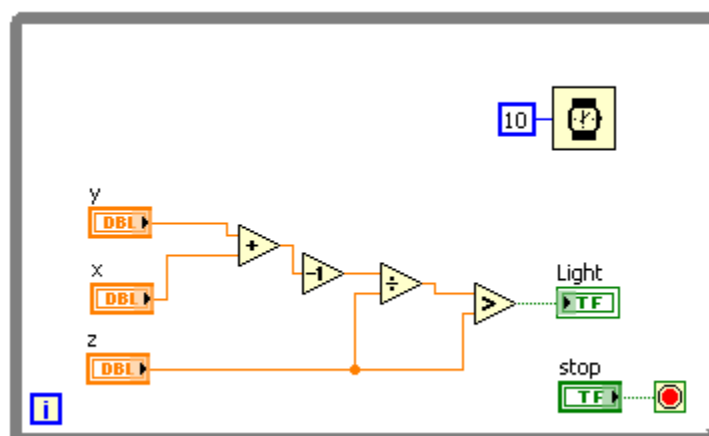
The bold inputs are required while the others are optional.

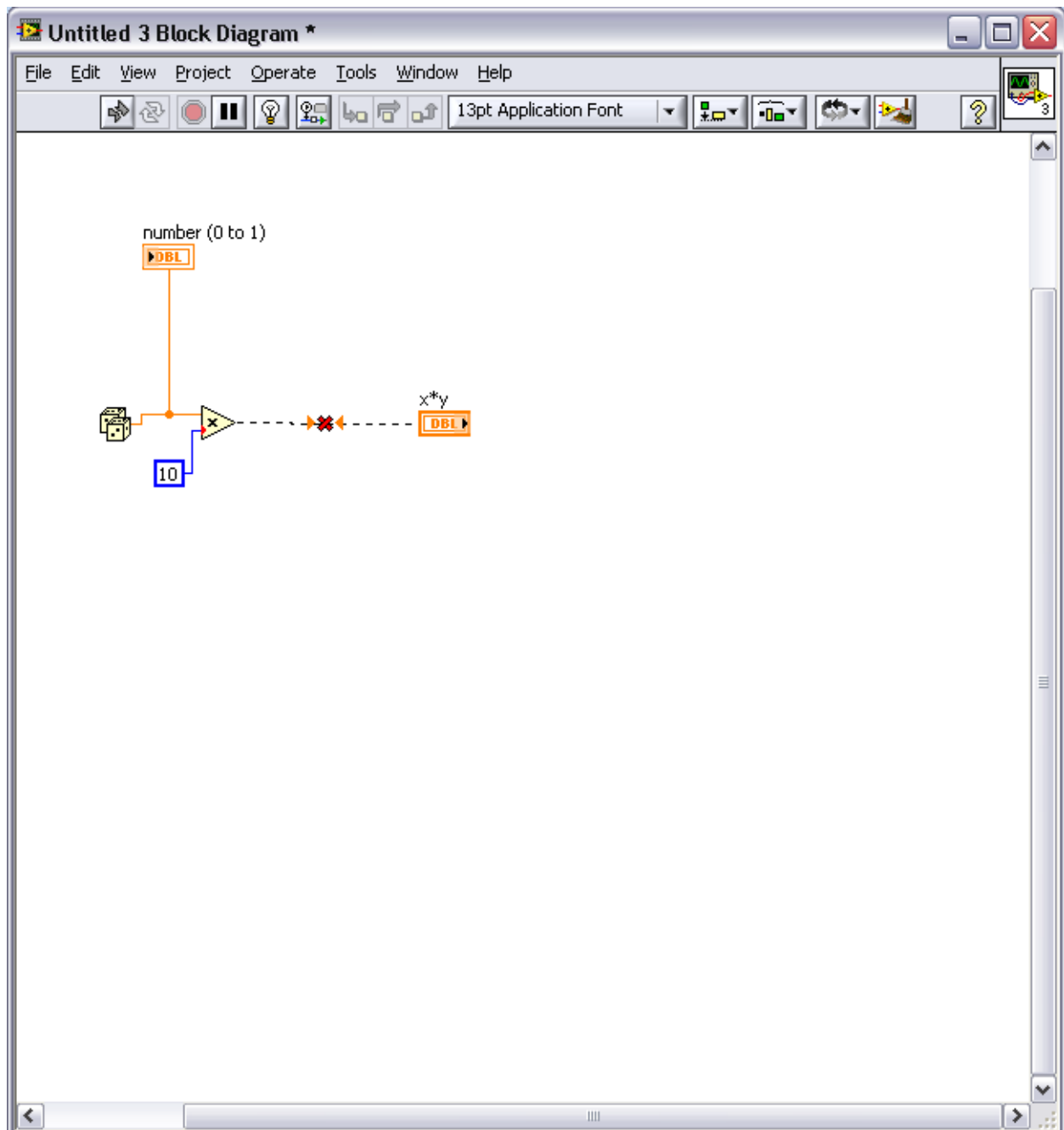


The above VI no longer works, and has a broken run arrow. It worked at one time, but would take close to 100% of the computer's resources when running. A general approach to debugging and fixing this VI would be to first **clean up the block diagram**, followed by clicking the **broken run arrow**. With that information it should be possible to make the needed changes and run the VI. Finally, in order to reduce the use of computer resources it would be beneficial to add a **wait** function inside the while loop. It would also be good programming practice to replace the Boolean constant with a **control**.

*A good general approach to debugging a messy VI that gives explicit errors when run is to first clean up the block diagram, then approach the errors. As an additional note, For loops and While loops will take all available computing power to complete their tasks unless they are told to wait between iterations. Waiting between iterations gives the computer enough time to complete other tasks while continuing to run your VI.*

*A successfully debugged VI:*





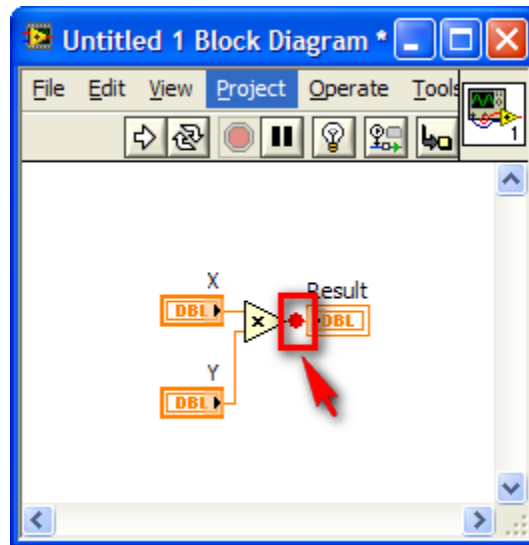
If you created this block diagram, what would be the two indications that there is a problem?

*A dashed wire with an X and a broken run arrow. These are the two visual ways that LabVIEW shows the presence of an error.*

*Based on this block diagram, identify the problem.*

*The output of the multiplier is connected to a control. Change the control to an indicator to fix the problem. Outputs must be displayed using indicators.*

The following code multiplies two numbers together and displays the answer in the result indicator:



When this code is run, the VI seems to pause. How can this be fixed?

*The VI pauses because a breakpoint was placed in the code. In order to remove the breakpoint, the user can right click the breakpoint and select **Breakpoint » Clear Breakpoint***

*Breakpoints are usually used to debug code to see if certain parts of code will run when executed.*

Briefly describe the functionality of a shift register.

*Shift Registers are used when you want to pass values from previous iterations through the loop to the next iteration.*



True or False: During the first iteration of a while loop, the iteration terminal will return 0.

**True:** *The iteration count always begins at 0, not 1.*

True or False: LabVIEW has built-in functions that can auto-align, auto-distribute, and auto-resize front panel and block diagram objects.

**True:** *LabVIEW has built-in functions that can auto-align, auto-distribute, and auto-resize front panel and block diagram objects.*

What is the difference between the I16 and U16 data types? Additionally, what is the range of data for each of them?

*An I16 data type can hold both positive and negative valued data between  $-(2^{16}/2)$  and  $(2^{16}/2)-1$ .*

*(i.e. -32768 through 32767)*

*Zero is part of the positive range so it is important to subtract the one from the positive side of the data range.*

*An U16 data type can hold only positive data between 0 and  $2^{16}-1$ .*



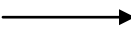
*(i.e. 0..65535)*

Tunnels on structures serve what purpose(s)?

- a. Feed data into the structure.**
- b. Feed data out of the structure.**
- c. Control loop execution.**
- d. None of the Above.

*Tunnels feed data into and out of structures. When a tunnel passes data into a loop, the loop executes only after data arrives at the tunnel.*

Please match the complex number representation to its respective name.

- |                           |   |   |
|---------------------------|---|---|
| a. Complex Double (CDB)   |  | 1. Real & imaginary values in 32-bit IEEE format  |
| b. Complex Single (CSG)   |  | 2. Real & imaginary values in 64-bit IEEE format  |
| c. Complex Extended (CXT) |  | 3. Real & imaginary values vary based on platform |

*In Windows, Complex Extended numbers are in the 80-bit IEEE extended-precision format.*

Which of the following are valid Loop Structures within the LabVIEW environment?

- a. Conditional – *There is no such thing as a Conditional loop.*
- b. For**
- c. Timed**
- d. Case – *A Case Structure is not a method of looping code.*
- e. Sequence – *A Sequence Structure only forces the order of operation; there is no loop functionality.*
- f. While**

True or False: The Wait Until Next ms Multiple function waits until the millisecond counter counts to an amount equal to the input you specify.

*The Wait Until Next ms Multiple function waits until the millisecond counter counts to an amount equal to the input you specify.*

**False:** *The function described is the Wait (ms) function, not the Wait Until Next ms Multiple. The Wait Until Next ms Multiple function monitors a millisecond counter and waits until the millisecond counter reaches a multiple of the amount you specify.*

From the following, select some good design techniques for LabVIEW programming.

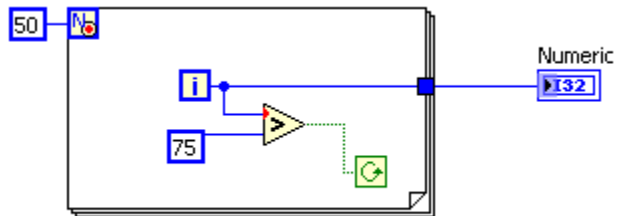
- a. Labels and Captions**
- b. Use flamboyant colors**
- c. Spacing and alignment**
- d. Text and fonts**
- e. System controls**
- f. Nested loops and structures
- g. Menus**
- h. Decorations**
- i. Automatic resizing of objects**

*Using Color is suggested but only whenever it is used to help make front panel items more visible or appear more important.*

*The use of nested loops and structures can cause troubleshooting problems. They also can easily cause delays and data flow problems in your code.*



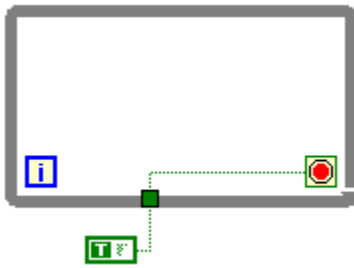
What will the indicator display after this program has run?



**The indicator will display zero.**

*The conditional terminal is set to continue if true. The first value will be false; therefore, the loop will only execute once. If the conditional terminal was set to stop if true, the indicator would display 49. The loop would stop after it executed 50 times.*

How many times will this loop execute?



- a. 1
- b. 0
- c. Infinite

*While loops always execute one time. If the Boolean was false, the loop would become an infinite loop.*

True or False: Captions appear on the block diagram.

***False:*** Labels appear on the block diagram; however, captions only appear on the front panel.

When LabVIEW coerces data, LabVIEW places what kind of indication to shown that conversion has taken place?

- a. A red wire going into the terminal where conversion is occurring.
- b. A red dot on the terminal where conversion is occurring.**
- c. LabVIEW produces an error saying that it cannot coerce data.
- d. LabVIEW does not indicate coercion but simply performs it.

Default values for controls can be changed to different values by what methods?

- a. Save the VI once the value has been changed.
- b. Right-clicking on the control in the block diagram and then selecting Data Operations>>Make Current Value Default.**
- c. Right-clicking on the control and changing it to a constant
- d. None of the above.

*This feature can be done on either the front panel or the block diagram on any type of control.*

Which of the following are possible front panel design options?

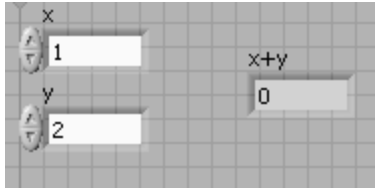
- a. Force the VI to behave like a dialog box
  - b. Choose to hide all labels
  - c. Force all front panel objects to rescale when you resize the window
  - d. **All of the above**
- 
- a. *Can be achieved by selecting File >> VI Properties then choose Window Appearance from the drop down menu and select the Dialog option*
  - b. *This can be done by right clicking the front panel object and deselect Visible Items >> Label*
  - c. *Navigate to File>> VI Properties and select Window Size in the drop down menu. Then check the box for "Scale all objects on front panel as the window resizes."*

Choose all of the following characteristics that describe a While Loop.

- a. Acts as in “if statement”
- b. Similar to a Do Loop or Repeat-Until Loop in text based**
- c. Iterates a set amount of times
- d. Executes a sub-diagram until a condition occurs**
- e. Always executes at least once**

*a. Applies to Case Structure*

*c. Applies to For Loop*



X is an integer with representation I32.

Y is an integer with representation I64.

What will be the representation of x+y?

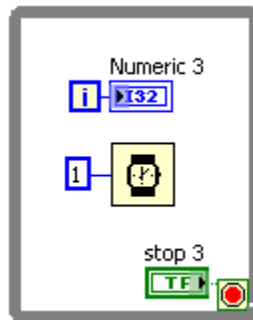
*X+y will have I64 representation. When LabVIEW coerces data, it always coerces it to the larger data representation.*



Match the Following Numeric Data Types to their corresponding storage size.

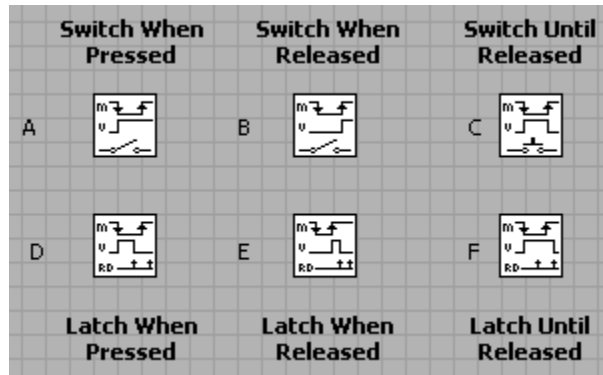
- |       |   |                                    |
|-------|---|------------------------------------|
| a. 16 | → | 1) Single Precision Floating Point |
| b. 32 | → | 2) Double Precision Floating Point |
| c. 32 | → | 3) Byte Integer                    |
| d. 8  | → | 4) Word Integer                    |
| e. 64 | → | 5) Long Integer                    |

What is the purpose of the wait function in the VI below? Hint: Think in terms of performance.



*Placing a 1ms delay will limit your loop to at most 1000 iterations per second. Without a delay, the loop rate can be in the millions of iterations per second range depending on what is inside it. So that means that your CPU has about 1/1000th as much work to do and can go off and tend to other tasks.*

Please match each Boolean mechanical action to the way it operates.



**A** The state is changed when the button is pressed. It remains pressed until button is pressed again.

**E** The state is changed when the button is released. Change back when value is read.

**D** The state is changed when the button is pressed. Always reads a value.

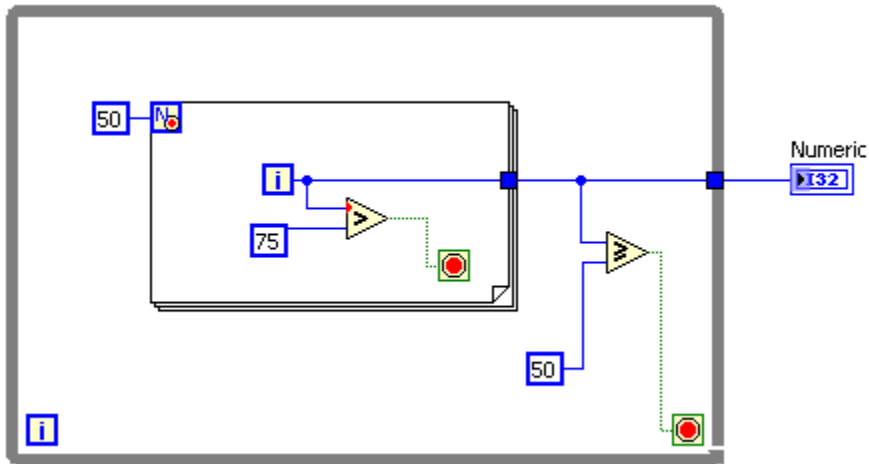
**B** The state is changed when the button is released. Do not read until mouse releases. Continue to read until button is pressed again.

**C** The state is changed when the button is released. Continue to read until button is released.

**F** The state is changed when the button is released. Continue to read until button is released. Always reads a value.

*See the Mechanical Action of Booleans example in the LabVIEW example finder for an interactive display of each type of mechanical action.*

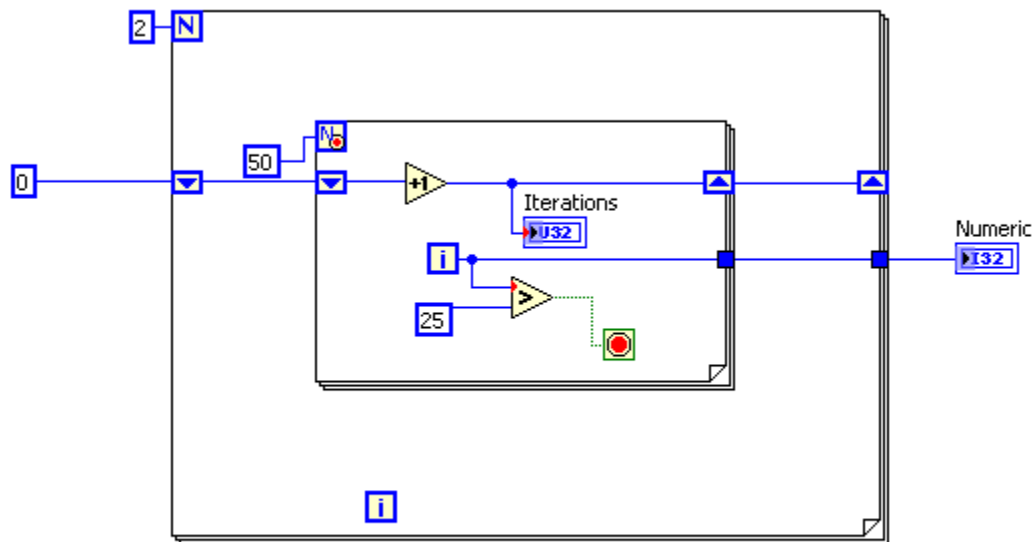
What number will be displayed in Numeric after the program completes



- a. 1
- b. 2
- c. 49
- d. 50
- e. **The loop is infinite, and the program will not complete**

*When the For Loop executes its last execution, the iteration terminal is at 49. When the While Loop iterates, the iteration terminal inside the For Loop will restart at zero. Therefore, the iteration terminal will never be greater than 49.*

After the program completes, what will be displayed in Iterations and Numeric Indicators?

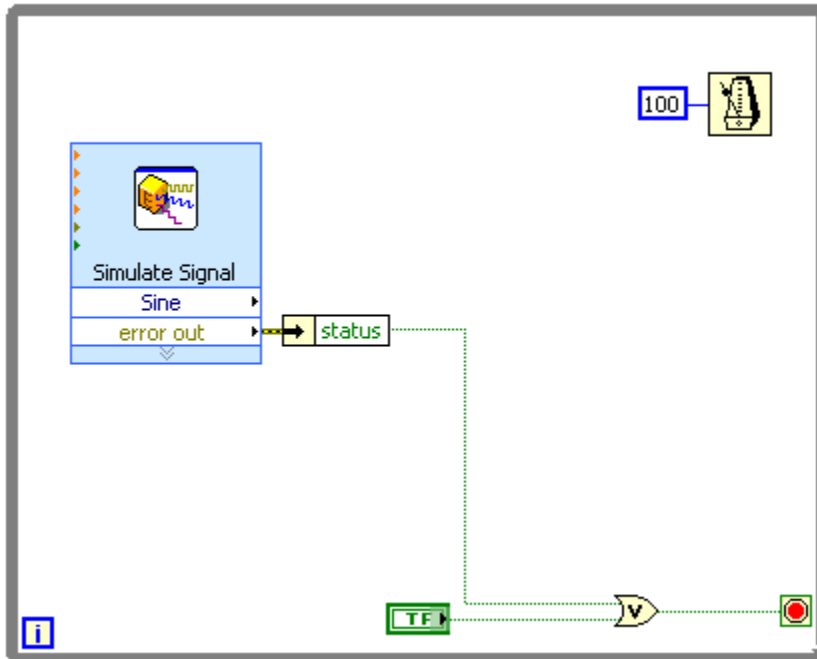


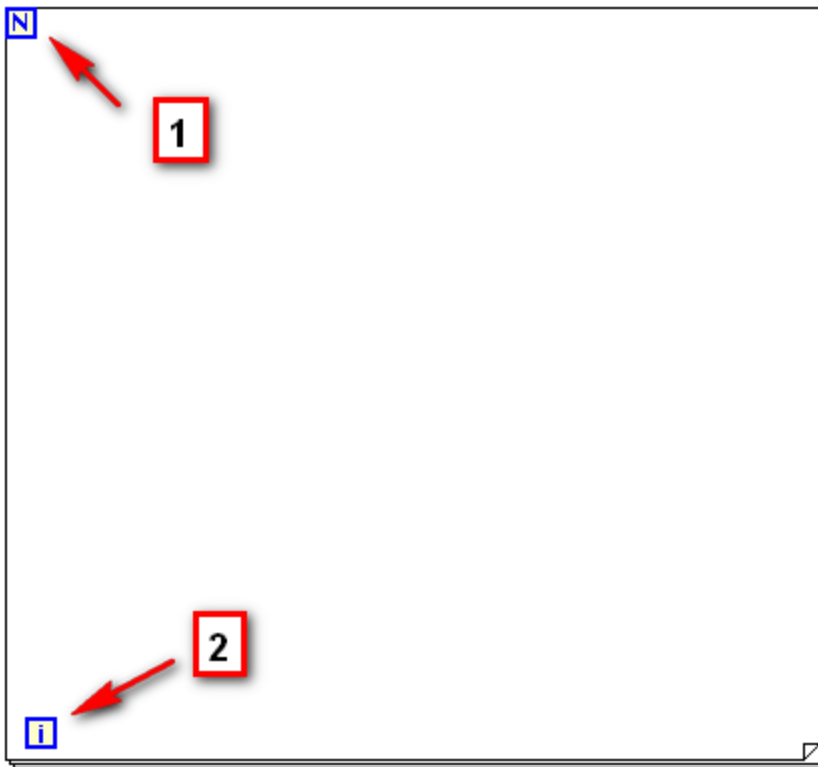
- a. 52, 25
- b. 54, 26**
- c. 50, 25
- d. 100, 49

*The inner for loop will stop when the iteration terminal gets to 26. This means that Iterations will be at 27 when the inner for loop is done executing. Since the outer for loop executes twice, Iterations will display 54 when the program is done*

Describe below how a While Loop can be used for handling errors by indicating where the error cluster should be wired as well as the effect of utilizing a While Loop in this way.

*The Status of an Error Cluster can be wired to the stop terminal of a While Loop using the Unbundle By Name function and the Or function. The Unbundle by Name function is used to obtain the error status, which is either true or false. The Or function allows both the occurrence of an error as well as a pressed stop button to stop the while loop. If an error occurs, the loop stops.*



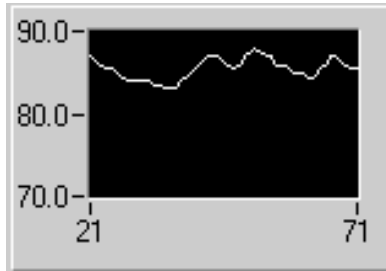


From the above figure, identify the type of loop as well as label the terminals indicated by numbers 1 and 2. Describe the function of both 1 and 2 and indicate at what value each terminal begins counting.

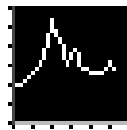
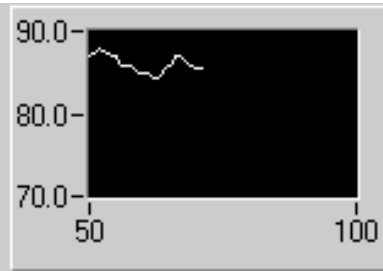
*The loop is a For Loop. Number 1 indicates the count terminal whose value indicates how many times to repeat the sub-diagram. It begins counting at 1. Number 2 indicates the iteration terminal. It is an output terminal that contains the number of completed iterations. The iteration count begins at 0.*

Match the name of the update mode to each of the following screenshots. The thumbnail image below each is the LabVIEW right-click menu representation of each. (Scope, Sweep, Strip)

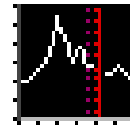
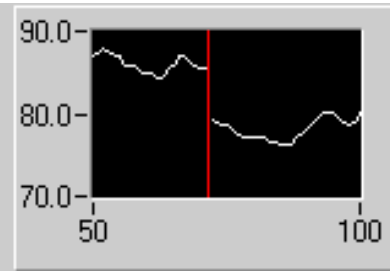
a. Strip



b. Scope



c. Sweep





Describe the difference between an initialized array and an uninitialized array.

*When an array is initialized, you defined the number of elements in each dimension and the components of each element. An uninitialized array contains a fixed number of dimensions but no elements.*

A cluster is similar to a **record** or **struct** in text based programming languages.

*Clusters group data elements of various types (e.g Boolean, string, double).*

True or False: You can mix different data types into an array?

**FALSE:** An array can only be of one data type.

True or False: You can index through cluster elements using a for loop?

**False:** *You can index through elements in an array, but not a cluster because there is no indexing available for clusters.*

A LabVIEW Error Cluster contains which of the following elements? Explain the function of each included element.

- a. Timestamp
- b. **Status - Boolean value that reports True if an error occurs.**
- c. **Code - 32-bit signed integer that identifies the error.**
- d. Index
- e. Description
- f. **Source - String that identifies where the error occurred.**

Which of the following cannot be an element of an array?

- a. File refnums
- b. Queues
- c. Strings
- d. **.Net Container**

*You cannot add .Net container because it is not a control, but a container of different data types.*

A cluster already contains a string control. Which of the following choices cannot be added to this existing cluster? (Choose as many as are appropriate.)

- a. An additional string control
- b. A string indicator**
- c. A numeric control
- d. A time stamp control
- e. A time stamp indicator**

*A cluster can contain only indicators or only controls at a given time. The cluster components will all switch to the type of the most recently added object.*

What is/are the benefit(s) of using clusters? (choose all that apply)

- a. Enables faster data processing
  - b. Eliminates wire clutter on the block diagram**
  - c. Reduces the number of connector pane terminals**
  - d. Can be used in place of an array
- 
- a. Clusters do not affect processing time*
  - d. Can be used with arrays, but not in place of arrays*



Select the ways to make a custom control? Choose all that apply.

- a. **Right-click on the front panel control or indicator and select **Advanced>>Customize**.**
- b. Right-click on a block diagram control or indicator and select **Advanced>>Customize**.
- c. **Select a front panel control of indicator and select **Edit>>Customize Control** from the menu bar.**
- d. **Use the **New>>Control** from the project explorer.**
- e. Right-click anywhere on the front panel and select **Advanced>>Customize**.
- f. Double-click on a control or indicator to open its properties box.

If you enable auto-indexing for more than one tunnel or if you wire the count terminal, the actual number of iterations becomes the \_\_\_\_\_ of the choices.

- a. Greater
- b. Smaller**
- c. You cannot do this in LabVIEW.
- d. LabVIEW will not display any error, but this function has an indeterminate output.

*LabVIEW wouldn't know what data to use to fill the end of the shortest array. It ignores the data in the indices of any array that is greater than the length of the shortest array.*

Which of the following is true about type defined custom controls?

- a. If you change an instance of a type defined custom control in one VI, you will not need to change all others to match it because the change happens automatically.
- b. If you change the data type of the .ctl file of the control/indicator then all instances will be changed that are still linked to that .ctl control/indicator.**
- c. If you change the default value of a type defined custom control, it will change the default value for all instances of that control.
- d. If you change the data range of a type defined custom control, it will change the data range for all instances of that control.

*Not A because the change only happens automatically if you use choice B's methods.*

*Not C and D because they are properties of strict type defined custom controls.*

Select appropriate answer(s): You cannot create an array of arrays. However, to accomplish a similar end result, you can...

- a. **Use a multidimensional array.**
- b. Use a “Flat Sequence Structure” with a 1-D array in each of the sequence panels.  
*This will only create x number of unconnected 1-D arrays, one after another, where x is the number of panels in the sequence.*
- c. Create a nested, auto-indexing “For Loop” with a “String to Byte Array” VI within it.  
*“String to Byte Array” is simply a conversion VI. It will not combine data or create multidimensional arrays.*
- d. **Create an array of clusters where each cluster contains one or more arrays.**





A **two-dimensional** array contains both row and columns?

*A one dimensional array has either rows or columns, not both. Arrays with more than two dimensions also contain row and column, but in other dimensions.*

You are planning on creating a new VI with a team. You are in charge of the initial planning, such as defining the inputs and outputs. Your VI will use many similar inputs but may undergo many revisions from your teammates. These revisions would most likely include a change of data representation for different inputs. What control type should be used for these inputs?

- a. Control *If any revisions are made, this type requires changes to each input*
- b. Type Definition** *Correct*
- c. Strict Type Definition *A strict type definition does not allow for data representation changes*
- d. All are suitable *Only a type definition is suitable for all requirements of these inputs*

You have an array that contains values and you need to find the max and min and their associated indexes, which of the following is the best block to use?

- a. 
- b. 
- c. 
- d. 

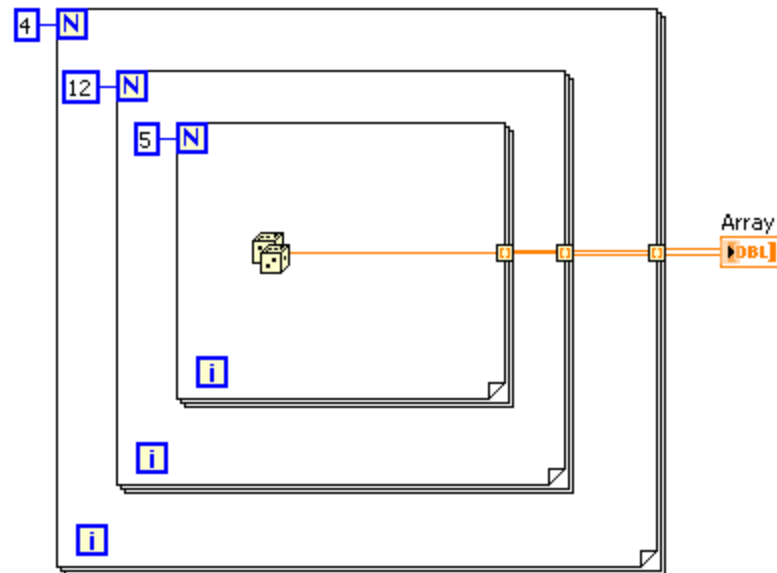
Match the buttons with their names



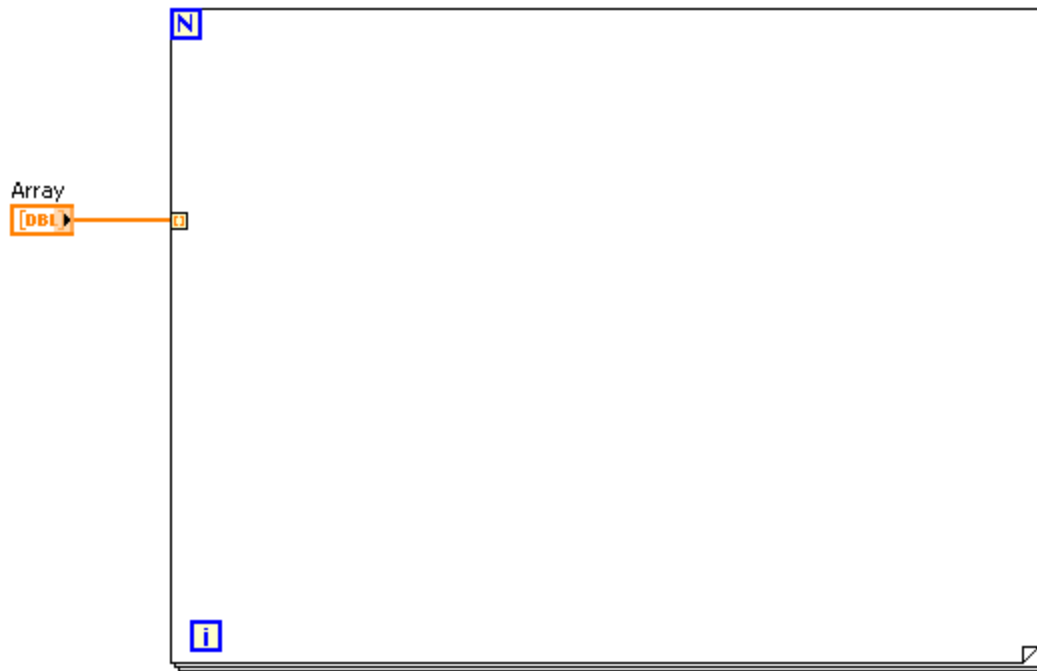
a.      b.                      c.                                      d.                                      e.                      f.                      g.                      h.

1.   c   Type Definition Status
2.   b   Edit Mode
3.   h   Reorder Objects
4.   e   Align Objects
5.   a   Customize Mode
6.   f   Distribute Objects
7.   d   Text
8.   g   Resize Objects



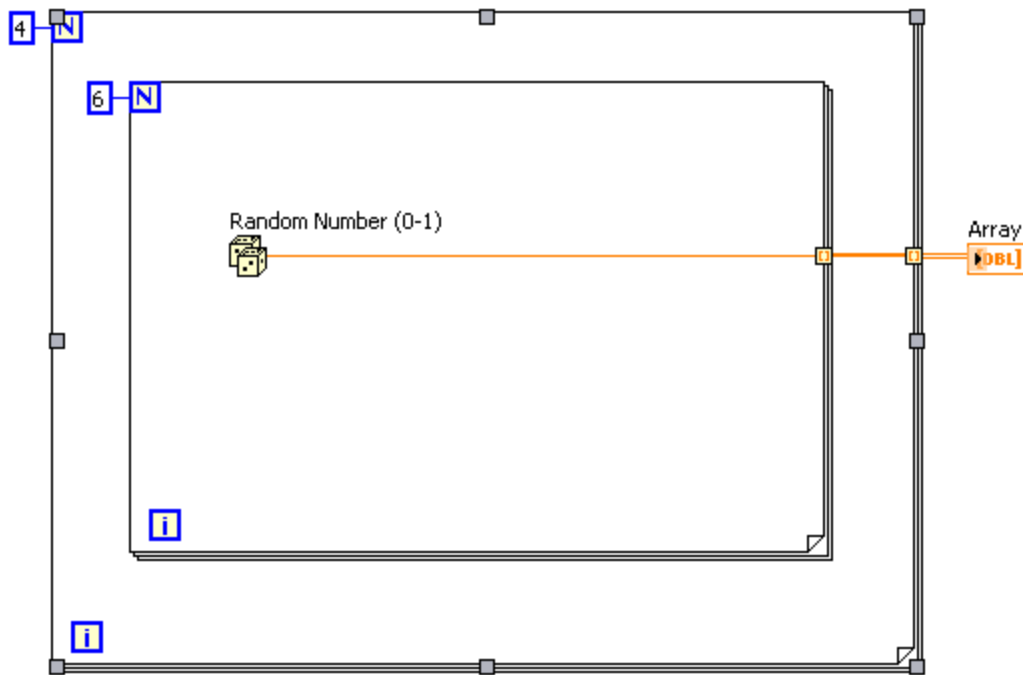


The above block diagram produces a 3 dimensional array with the dimensions 5 x 12 x 4. There are 5 columns in this array.



What is significant about the tunnel seen in the figure above. What does this tunnel state mean in terms of how the loop iterates in regard to the array?

*The tunnel is set for auto-indexing. This means that each iteration of the loop is linked to an element in the array.*



What is the dimension of the outputted array? How many rows and how many columns will appear in the array?

*The resulting array is two- dimensional. It has 4 rows and 6 columns.*

Why do high-level file I/O VIs cause a LabVIEW program to run slower if placed in a loop?

*These VIs perform open and close operations each time they run. Use lower level file I/O VIs if you plan on putting them inside a loop.*

True or False: A high-level VI performs open and close operations best when placed in a loop.

**False:** *A high-level VI opens and closes the VI, which means if it was in a loop, the operating system would be required to interact with the VI each time the file is opened or closed ultimately using more system resources and time.*

When using a Read from Spreadsheet File VI to read an ASCII file with columns of data, it must be **tab** delimited in order to be read properly.

*Without tab delimiters, only the first column of data will be read.*

In what situations should you use low-level VI's and high-level VI's when writing to a file?

*You should use low-level VI's when writing to a file repeatedly, such as in a loop. Use high-level VI's when writing to a file in a single operation.*

*Low-level VI's are more efficient and are designed specifically for a single operation. High-level VI's perform open, read/write, and close operations, which can slow down the program if used incorrectly.*

What is the correct order of operations for a typical file I/O operation?

- a. **Open File, Read/Write File, Close File, Check for Errors**
- b. Open File, Read File, Close File, Check for Errors
- c. Open File, Read/Write File, Check for Errors
- d. Open File, Read File, Close File, Open File, Write File, Close File, Check for Errors

*b. Does not include write*

*c. Does not close the file*

*d. Inefficient use of .vi's*



You are developing an application that logs temperature data from 3 different sensors. Each sensor acquires 100 samples one after the other and records the sample to a file immediately after acquiring it. What level of File I/O VIs will be the most efficient?

**a. Low Level File I/O**

*Since you are writing after sample, using low level file I/O VIs and disk streaming will be the most efficient*

**b. Medium Level File I/O**

*LabVIEW only considers high and low level types of file I/O*

**c. High Level File I/O**

*Since high level file I/O VIs open and close a file every time it is used, it would be an inefficient use of system resources.*

Refnums are required for operations involving which of the following? Choose all that apply.

- a. Outputting data to a waveform graph *Does not require opening any kind of reference*
- b. Opening a file**
- c. Starting a network connection**
- d. Accessing a device**

List the steps of the file I/O process in the order they occur.

*Open, read/write, close*

True or False: Disk streaming uses less system resources when performing File I/O functions due to fewer interactions with the operating system.

**True**

List the three main considerations when deciding to access text files from another application.

- a.* If you need to perform random access read or writes
- b.* If disk space and file I/O speed are not crucial**
- c.* If you do not need to perform random access read or writes**
- d.* If numeric precision is not important**
- e.* If numeric precision is important

Low-level file I/O Vis and functions each perform how many pieces of the file I/O process?

- a. 2-open and close
- b. 2-read and write
- c. **1-each perform only one piece**
- d. They perform all of the pieces

*Low-level file I/O Vis and functions each perform only one piece of the file I/O process. For example, there is one function to open an ASCII file, one function to read an ASCII file, and one function to close an ASCII file. All other answer choices may be tempting but are made-up.*

Match the following to VIs to their description.

1. Write to Spreadsheet File

*c. Converts a 2D or 1D array of single-precision numbers to a text string and writes the string to a new ASCII file or appends the string to an existing file.*

2. Read From Spreadsheet File

*b. An Express VI that reads data from a text-based measurement file (.lvm) or a binary measurement file (.tdms)*

3. Write to Measurement File

*d. An Express VI that writes data to a text-based measurement file (.lvm) or a binary measurement file (.tdms) format.*

4. Read from Measurement File

*a. Reads a specified number of lines or rows from a numeric text file beginning at a specified character offset and converts the data to a 2D single-precision array of numbers.*

Please match each file format to its respective description

- |           |   |  |
|-----------|---|--|
| a. Binary | → | 1. Called a text file and is the standard for most programs      |
| a. TDMS   | → | 2. Underlying format of all other file formats                   |
| b. ASCII  | → | 3. Includes additional information for data such as a time stamp |
| c. LVM    | → | 4. Consists of a properties file and an index file               |



Describe what a refnum is and how it is used within LabVIEW.

*A reference number, or refnum, is a unique identifier for an object, such as a file, device, or network connection. When you open a file, device, or network connection, LabVIEW creates a refnum associated with that file, device, or network connection. All operations you perform on open files, devices, or network connections use the refnums to identify each object. Use a refnum control to pass a refnum into or out of a VI. For example, use a refnum control to modify the contents of the file that a refnum is referencing without closing and reopening the file.*

*Because a refnum is a temporary pointer to an open object, it is valid only for the period during which the object is open. If you close the object, LabVIEW disassociates the refnum with the object, and the refnum becomes obsolete. If you open the object again, LabVIEW creates a new refnum that is different from the first refnum. LabVIEW allocates memory for an object that is associated with a refnum. Close the refnum to release the object from memory.*

*LabVIEW remembers information associated with each refnum, such as the current location for reading from or writing to the object and the degree of user access, so you can perform concurrent but independent operations on a single object. If a VI opens an object multiple times, each open operation returns a different refnum. LabVIEW automatically closes refnums for you when a VI finishes running, but it is a good programming practice to close refnums as soon as you are finished with them to most efficiently use memory and other resources. Close refnums in the opposite order that you opened them. For example, if you obtain a refnum to object A and invoke a method on object A to obtain a refnum to object B, close the refnum to object B first and then close the refnum to object A.*

*If you open a refnum inside a For Loop or While Loop, close that refnum for each iteration of the loop because LabVIEW repeatedly allocates memory for the refnum and does not free the memory until the VI finishes running.*

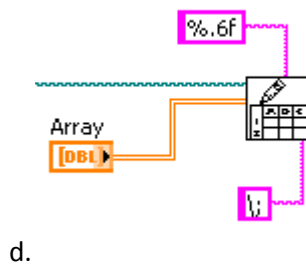
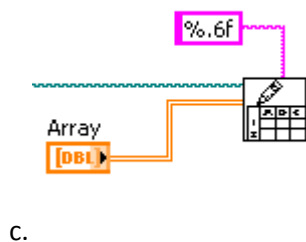
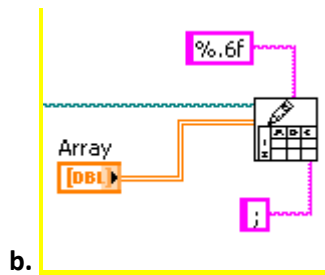
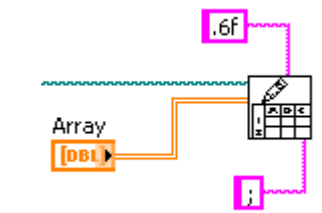
Briefly describe each file type.

- a. Binary-underlying file format of all other file formats*
- b. ASCII-Specific type of binary file also called a text file*
- c. LVM-LabVIEW measurement data file that can be opened with a spreadsheet or text-editing application*
- d. TDMS-Specific type of binary file created for National Instruments products*

Which of the following describes a refnum? (Select all correct answers)

- a. **A unique identifier for an object (file, device, network connection)**
- b. **A temporary pointer to an open object**
- c. A number that identifies the location of an element within an array
  - *This is the definition of an index*
- d. The number that is unique to your LabVIEW license
  - *This answer describes your LabVIEW serial number or activation code.*

You need to write a 2D array to a semicolon delimited spreadsheet file with 6 digits of precision after the decimal point. Which of the following configuration would be the best to use?

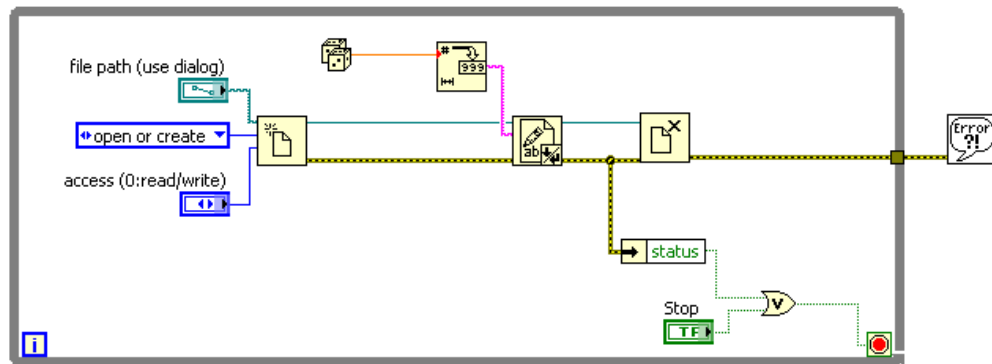
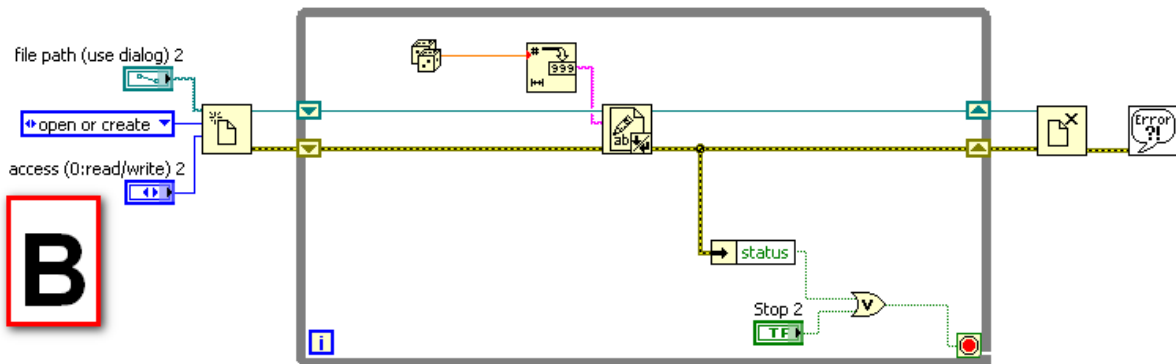


Not a because missing % on the format sting

Not c because leaving delimiter unwired defaults the output to tab delimited

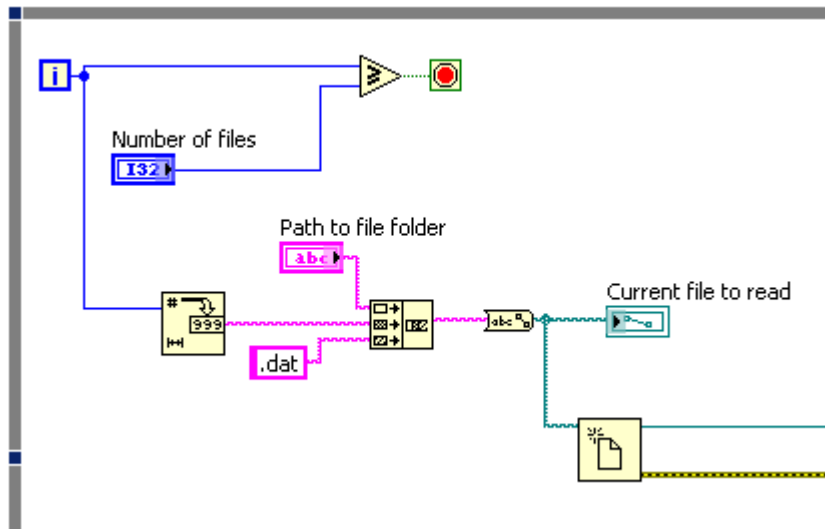
Not d because delimiter includes will be "\;"

Which picture below (A or B) implements disk streaming? Briefly, what is the benefit of disk streaming?

**A****B**

Choice B implements disk streaming. Disk streaming is a technique for keeping files open while you perform multiple write operations.

You have multiple ASCII, tab delimited files that you need to read data from for analysis. Luckily, the files are in the same directory and are sequentially labeled (1.dat, 2.dat, 3.dat, etc...). Which of the following choices is a method to open each file sequentially? *Note: Each choice is using a while loop.*

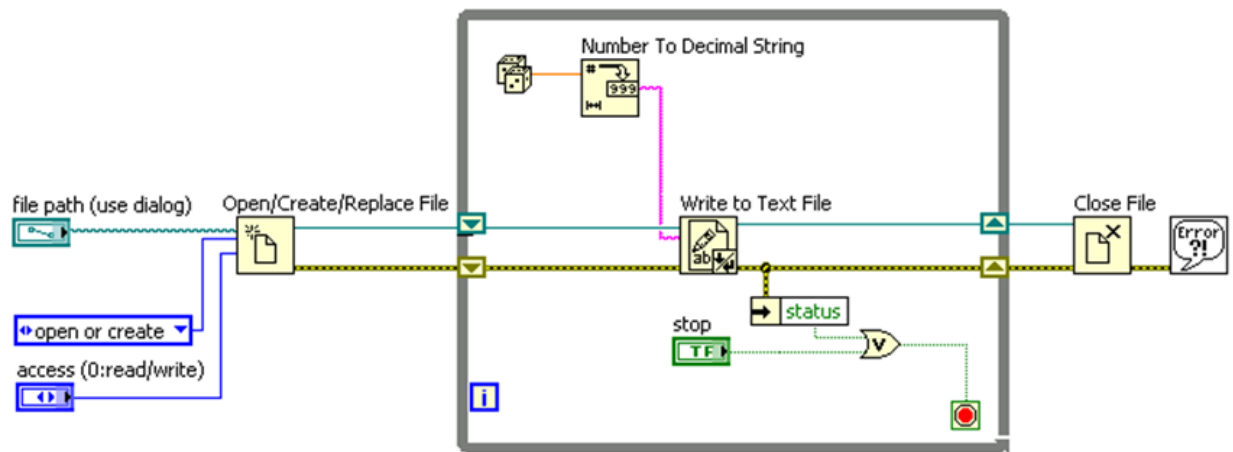


Choice a uses the < node as a means to test if you are done indexing files; using this method file mean that the loop will only execute once since number of files will be greater than two.

Choice b has ".txt" in the concatenate string node; you want to index ".dat" files.

Choice d is missing the path to the folder containing the files and the VI would not run if a terminal is unwired to a concatenate string node.

Why is it advantageous to use low-level VI's when writing to a file in a loop?



*Each time a function opens or closes a file, it needs to interact with the operating system and uses additional system resources. Using low-level VI's allows the program to open the file once, write everything within the loop, and close the file at the end. This minimizes interaction with the operating system and is the most efficient coding method for basic file I/O.*

A VI within another VI is called a **subVI**. It corresponds to a **subroutine** in text-based programming languages.



True or False: You must customize the VI icon in order to maintain functionality of that VI when it is used as a subVI.

**False:** *Customizing the icon is recommended, but optional. Using the default LabVIEW icon does not affect functionality.*

Why is it good programming practice to create a custom icon for a subVI?

*It is good programming practice to create a custom icon for a subVI because it helps identify the operations the subVI is used for. A picture that describes the use of the subVI allows someone looking at the code to quickly understand its purpose without needing to open it and analyze what operations are being performed.*

True or False: Creating a subVI can be done by using the Positioning tool and selecting a section of code within your VI and going to **Tools>>Create SubVI**.

**False:** *The correct menu to go to for this feature is **Edit>>Create SubVI**. LabVIEW will automatically setup the selected portion of code into a SubVI by creating controls and indicators, configuring the connector pane, and wiring the new subVI to the appropriate items on the block diagram.*

If your front panel contains more than 28 controls and indicators that you want to use programmatically, what should you do?

*Group some of them into a cluster and assign the cluster to a terminal on the connector pane.*

True or False: The connector pane can be accessed by right-clicking the VI icon in the top right-hand corner of the block diagram window.

***False:*** The connector pane is accessed by right-clicking the VI icon in the top right-hand corner of the Front Panel.

What is the maximum number of connections on a connector pane?

- a. Infinite-the connector pane automatically adjusts to the number of controls/indicators on the front panel
- b. 36
- c. Twice the number of controls-for every input, there is an output
- d. **28**

Which choice below is an efficient option when performing a task frequently?

- a. Run the VI every time that an operation needs to be performed
- b. Copy and paste the operation to every location it needs to be used
- c. It is bad programming practice to have an operation occur more than once
- d. **Use subVIs or loops to perform an operation repetitively**

True or False: It is best to avoid using error clusters in your subVIs.

**False:** *It is in the developer's best interest to include error clusters in their subVI's and to wire them to terminals in the connector pane. Error clusters are used to not only pass along critical error information, but are also very helpful in defining the sequence of execution within a VI.*



Please select all of the suggested practices when using the connector pane.

- a. Use as many terminals as possible
- b. Connect controls (inputs) on the left terminals and indicators (outputs) on the right terminals**
- c. Label controls and indicators according to their purposes**
- d. Delete all unused terminals

- a. *The fewer the terminals, the easier it is to use the subVI*
- b. *This is a good practice*
- c. *This is a good practice*

*Deleting unused terminals is not necessary. It is also helpful to have some terminals available in case changes are made to the subVI in the future.*

Modularity provides the following advantages.

- a. **Breaks up code into manageable pieces.**
- b. **Make code easier to understand.**
- c. Prevents viewing of lowest level functions
- d. Makes it hard for other coders to modify your code.

*Not c because one can still view the lowest level functions unless they are locked.*

*Not d because coders can use modularity to their advantage to work on small pieces of code.*

How is the front panel of a subVI accessed?

- a. Go to View > Select subVI > See Front Panel
  - b. Double-click the subVI**
  - c. Select the subVI and then go to Window > Show Front Panel
  - d. Right-click the subVI and select Open Front Panel**
- 
- a. *This option does not exist*
  - b. *This is a correct answer*
  - c. *This option will open the current VI's front panel*
  - d. *This is a correct answer*

A subVI is typically used when:

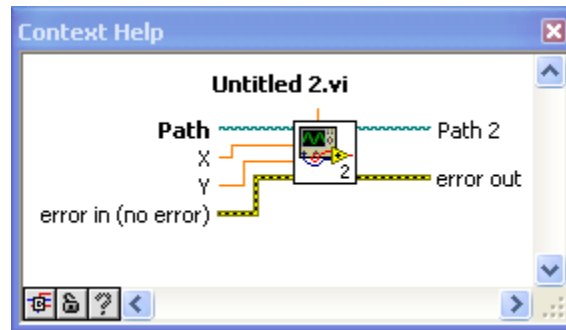
- a. An operation is performed only once
- b. An operation is performed repetitively**
- c. An operation is performed within a loop
- d. An operation is modified within a VI

- a. *Not necessary to create a subVI*
- b. *Correct answer*
- c. *If the operation is performed once per iteration, then it is most likely not necessary to put it in a subVI*
- d. *If an operation is changed programmatically it is not a repetitive action and would be difficult to implement as a subVI*

True or False: The icon in the upper right hand corner of a VI can only be changed to a pre-existing photo on the machine that the VI was originally developed.

***False:*** The icon can be customized by the user to be whatever they want. It can also be edited by any machine that the VI can be opened on.

Please match the input terminal types to their characteristics.



- |                |  |  |
|----------------|--|--|
| a. Required    |  | 1. No errors are thrown but may not execute correctly when not wired |
| b. Recommended |  | 2. The VI can execute properly when not wired                        |
| c. Optional    |  | 3. When not wired, the run arrow is broken.                          |

The following Calling Program Code and Function Code are shown below:

Function Code	Calling Program Code
<pre>function triangle_area(in1, in2, out) {   out = 0.5 * in1 * in2; }</pre>	<pre>main {   triangle_area(base, height, area) }</pre>

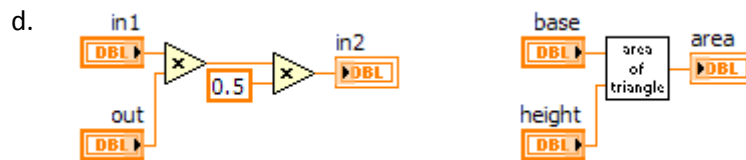
What is the correct implementation of the SubVI?

*Answer a is incorrect because the SubVI code is wrong. The function code should multiply 0.5, in1, and in2; however, the code multiplies in1 and in2, but adds 0.5.*

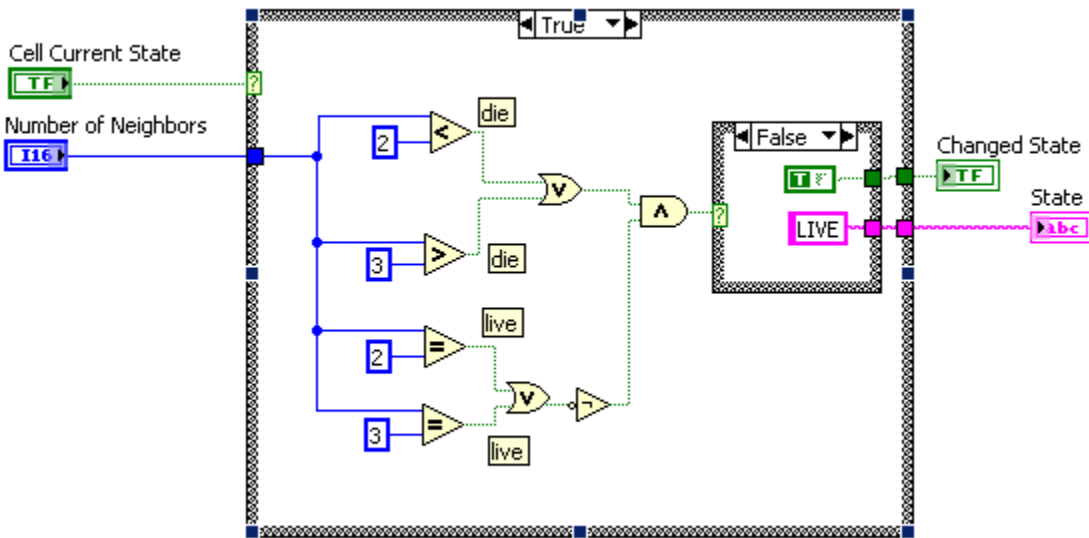
*Answer b is incorrect because the SubVI code should multiply in1, in2, and 0.5 together instead of adding them. Additionally, the calling VI code is incorrect because area should be an output and base should be an input.*

*Answer c implements the function code and the calling program code correctly. The SubVI should multiply in1, in2, and 0.5 and send it to out and the calling VI should give the SubVI the height and base and receive the area.*

*Answer d is incorrect because out should be an output and in2 should be an input.*



Which of the following is the best connector pane layout for this subVI?



c.

*Not a because the State string is not added to the connector pane.*

*Not b because Number of Neighbors integer wire could be added to the left side of the connector pane and again the State string is not added.*

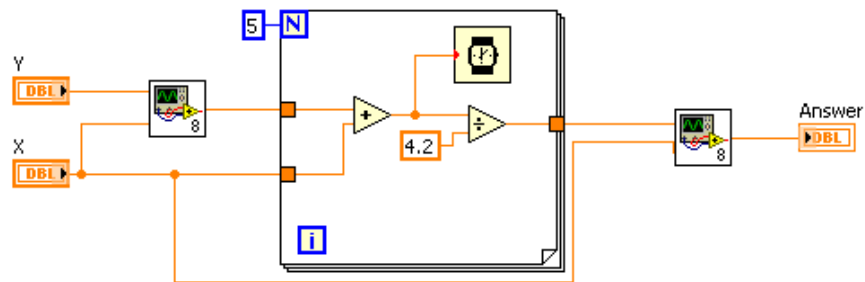
*The answer is c because the inputs are on the left side of the connector pane and the output are on the right side of the connector pane.*

*Not d because the output of the VI should be on the right side of the connector pane since there is room.*



Please select the best use of a subVI for the following code.

The correct answer is c.



- Turning the entire VI into a subVI is inefficient and does not add modularity*
- If a subVI is created from a section of code that is only used once, it does not add modularity, although it does make it easier to look at code*
- This is the correct answer. It adds modularity and makes it easier to look at the code.*
- This is an incorrect use of subVIs as it changes the execution of the code*

Please name at least four bus structures that different DAQ devices support.

*The six different bus structures that DAQ devices support are: PCI, PCI Express, PCMCIA, USB, PXI, and Compact PCI. The USB, PCI, and PCI Express cards are used in a typical desktop PC. The PCMCIA devices are used in laptops. PXI and CompactPCI cards are used in portable systems.*

You want to have a warning light on when a particular sequence happens in your system. You should use Generate \_\_\_\_\_ task timing.

- a.     **1 Sample**
- b.      $n$  Samples
- c.     Continuously

*When you want to generate a signal that doesn't change, you only have to generate it once. This is why you should use Generate 1 Sample and not Generate Continuously.*

You are developing a VI that tests an oscilloscope. You need to send a 1 kHz non-finite AC signal when the user presses an external button.

Should you use analog input measurement or **analog output generation**?

*Analog Output since the VI needs to send a signal to an external device, analog output generation is required.*

Should you acquire/generate 1 sample,  $n$  samples, or acquire/**generate continuously**?

*Since the AC signal is non-finite, the VI needs to continuously generate samples. Generating  $n$  samples should be used for generating a finite AC signal. Generating 1 sample is best for constant signals, where level is more important than the rate of the signal.*

If you are acquiring/generating  $n$  or more samples, what other parameters do you need to specify?

*You should include a timing VI that specifies the sampling rate at 1 kHz*

*When generating more than 1 sample, timing parameters need to be specified, such as the rate of generation.*

Is a task trigger required?

*Since a user controls when the AC signal is sent, a start trigger needs to be specified.*

What is the communicating medium between the device and the application software?

- a. The cable
- b. The VI created in software
- c. **The driver engine**
- d. The device configures itself to communicate

True or False: A 4-bit counter can count up to 16.

**False:** *The formula for counters is:*

$$2^{(\text{Counter Resolution})} - 1 = 2^4 - 1 = 15$$

*Zero is included in the numbers that a counter uses.*

Which of the following terms apply to the state of a digital signal? Choose all that apply.

- a. Red and green
- b. On and off**
- c. Go and stop
- d. High and low**
- e. + and –
- f. 1 and 0**

*2, 4, and 6 are the only correct choices. The other choices are all designed to be distracters.*

What is the highest value a 24-bit counter can count to? (Show the formula you use clearly)

- a. **16777215**
- b. 16777216
- c. 8388608
- d. 24

*The correct answer is a. This is calculated from the formula  $2^{(\text{counter resolution})} - 1 = 2^{24} - 1$ . The other choices are manipulations of this formula that could be common mistakes.*



Please select all functions provided by the DAQ Signal Accessory from the list below.

- a. **Analog Input**
- b. **Function Generator**
- c. Packaged DAQ Card
- d. **Digital Trigger**

*The DAQ Signal Accessory requires a DAQ card to operate*

NI-DAQmx is supported on what operating systems?

- a. Windows
- b. Linux
- c. Mac OS
- d. Both A and B**
- e. Both B and C
- f. None of the above.

*NI-DAQmx is only supported in Windows and Linux. NI-DAQmx Base is required in Mac OS, but also is supported in the other two operating systems.*

Please select all of the advantages of using the DAQmx drivers over Traditional NI-DAQ.

- a. Traditional NI-DAQ is packaged with it
- b. Increased performance**
- c. DAQ Assistant is included for configuring tasks and channels**
- d. Simpler API**

You have a 5V peak-to-peak square wave you are trying to view on a test panel in MAX but all that is appearing is a flat line at 0V. Which of the following could be possible problems?

- a. LabVIEW is currently open, but not running a VI.
- b. You are sampling the square wave too quickly.
- c. The square wave source is not connected to the data acquisition device.**
- d. Wire is connected to different channel than the one you are viewing.**
- e. You have an incompatible version of DAQmx installed.

*Not A because LabVIEW is not running a VI that could be using a channel on your DAQ device.*

*Not B because you should still see a signal, though it might not look like a square wave.*

*C is a possible reason and is D*

*Not E because if a version of DAQmx is incompatible, then you will not be able to view the devices test panel. However, this is not to discourage using the latest version of DAQmx to fix bugs.*

List the three parts to every DAQ system. Also explain how data flows throughout the system.

- a.     **Terminal Block** *Provides a place to connect the signals.*
- b.     **Cable**   *Connects the terminal block to the DAQ device.*
- c.     **DAQ Device**   *Converts between analog and digital signals*

*Note: There are actually two more hardware components that the book does not list. This would be the transducer and computer.*

*Transducer:     Converts the measured signal into a signal that the DAQ device can measure*

*Computer:       Interprets and displays digital data to the user.*

The use of Traditional NI-DAQ is required when

- a. Porting an old application to NI-DAQmx. (*You use Traditional NI-DAQ whenever you're upgrading from NI-DAQ 6.9.x and have existing applications that you don't want to port over to NI-DAQmx.*)
- b. You're using Windows Vista. (*Windows Vista does not support Traditional NI-DAQ (Legacy).*)
- c. Doing data acquisition with any DAQ device (new or old). (*Only use Traditional NI-DAQ whenever your device is not supported by NI-DAQmx.*)
- d. **Using a version of LabVIEW earlier than version 7.0.**

Why should you use Measurement & Automation Explorer (MAX) when installing a DAQ device?

- a. It cleans up the DAQ device installation files (*MAX does not do this*)
- b. It rewrites the registry files to be more efficient (*MAX reads records from the Windows Registry, but does not rewrite them*)
- c. It allows 3<sup>rd</sup> party software to interface with other NI programs (*MAX does not interact with 3<sup>rd</sup> party software*)
- d. **It completes the device configuration and assigns a device number to the device.**

A data acquisition system uses a DAQ device to pass a **conditioned** electrical signal to a computer for **software** analysis and data **logging**.

*A DAQ system is the avenue for a conditioned signal to be transferred to the computer, where the signal can be analyzed and recorded. A DAQ system usually includes three types of hardware- a terminal block, a cable, and a DAQ device (which goes into your computer or is part of a modular system, such as a PXI [PCI eXtensions for Instrumentation])*



Match the counter parts to their description.

- a. Count Register
- b. Source
- c. Gate
- d. Output

- 1. c An input signal that determines if an active edge on the source changes the count
- 2. b An input signal that can change the current count
- 3. a Stores the current count of the counter
- 4. d An output signal that produces pulses

Which of the following timing methods can run the fastest and is the most accurate?

- a. **Sample clock**    *The sample clock uses HW timing which can run the fastest and most accurate*
- b. Timed Loop        *Not the fastest, software timed*
- c. OS Timer          *Also software timed*

Controlling the speed of a DC motor	<b>b. Analog Output</b> <i>Controlling a DC motor requires an analog voltage to be sent to it</i>	<ul style="list-style-type: none"><li>a. Analog Input</li><li>b. Analog Output</li><li>c. Digital Input</li><li>d. Digital Output</li><li>e. Counter Input</li><li>f. Counter Output</li></ul>
Sending three pulses after every button press	<b>f. Counter Output</b> <i>Counters can generate a pulse train</i>	
Monitoring the states of 8 switches	<b>c. Digital Input</b> <i>Digital Input fits here because you can monitor each switch on a separate line.</i>	
Taking temperature readings	<b>a. Analog Input</b> <i>Reading analog voltages from a temperature sensor means analog input fits here</i>	
Signaling an LED when a certain limit is reached	<b>d. Digital Output</b> <i>A LED has two states, On or Off. Digital Output</i>	
Determining the number of times a quadrature encoder is turned	<b>e. Counter Input</b> <i>Counters can measure the number of events that occur; in this case, the event is every turn of the encoder</i>	

In Measurement and Automation Explorer you can simulate which of the following DAQ device(s)?

- a. **SCXI devices**
  - b. GPIB Devices
  - c. **USB devices**
  - d. **PXI Devices**
- 
- a. MAX can simulate SCXI chassis, modules, and connector blocks.
  - b. This is not an option in the simulated devices list in MAX.
  - c. MAX can simulate USB DAQ devices.
  - d. MAX can simulate PXI DAQ devices.

Which of the following statements correctly describes the functionality of an ADC?

- a. The ADC takes in a series of ones and zeroes and converts them into a voltage signal.

*The logic in this statement is reversed; the ADC takes in a analog signal and converts it to a series of zeroes and ones, a digital signal.*

- b. A sample clock controls the rate at which the ADC takes snapshots of the input signal.**

**This statement is correct, the ADC samples the analog signal on each rising or falling edge of the sample clock.**

- c. A benefit of the ADC is that it can convert the input signal without loss of precision.

*This is false, the incoming signal has infinite precision but the ADC must approximate this signal with fixed precision so a loss occurs.*

- d. The ADC samples the digital signal on each rising or falling edge and converts each sample to a series of ones and zeroes.

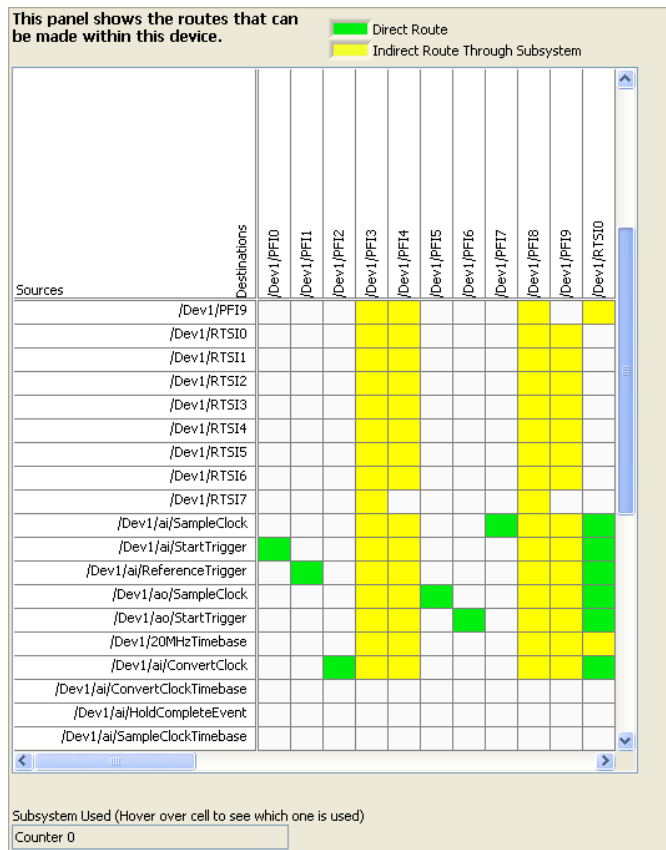
*This statement just doesn't make sense, it is not a digital signal that gets sampled, but an analog one and the timing is based on the sample clock.*

- e. A digital trigger controls the rate at which the ADC takes snapshots of the analog signal.

*A sample clock controls this rate, not a digital trigger.*

You are acquiring some type of data. However, it seems like you are having some incorrect values from the thermocouples you are reading. You are using an unshielded terminal block with 50 terminals. The terminal block is connected to the DAQ card through an unshielded cable. Explain how you would try to get a better, more clear signal.

- **Use shielded cables-** *shielding reduces noise introduced from the environment, so using shielded cables would eliminate a possible source of noise. Noise is caused by stray electromagnetic waves in the environment and prevents clarity in the signal.*
- **Use shielded terminal blocks** -*shielding reduces noise introduced from the environment, so using shielded cables would eliminate a possible source of noise. Noise is caused by stray electromagnetic waves in the environment and prevents clarity in the signal.*
- **Use a terminal block with more terminals-** *A terminal block with 68 terminals offers more ground terminals to connect a signal to than a terminal block with 50 terminals. Having more ground terminals prevents that need to overlap wires to reach a ground terminal, which can cause interference between the signals.*
- **Use a shorter cable-** *the longer a cable is the more opportunity there is for stray electromagnetic waves to interfere with signal transmission even if the cable is shielded because shielding is not 100% effective.*



Can Dev1/PFI9 be internally routed to Dev1/PFI6?

**No.** When the block is white, the devices cannot be internally routed.

Is Dev1/ai/StartTrigger directly or indirectly routed to Dev1/PFI0?

**Directly routed.** When the block is colored green, the devices have a direct route to each other

If the mouse was hovering over the top right yellow block, which system is used to internally route Dev1/PFI9 to Dev1/RTSI0?

**Counter 0.** When you hover the mouse over the yellow blocks, the system that is used to internally route the signal is displayed in the text field below the table.

Can this system be used in other applications when it is used to internally route lines?

**No.** When a system is being used to route two signals together internally, it cannot be used simultaneously for another function.

What is the data transfer rate of GPIB communication?

*1 Mbyte/s and higher. This rate will be diminished if a cable longer than 4m. The maximum cable length is 20m.*



What is the character transmission rate of a serial port if the baud rate is set at 9600 and the character length is 12?

*9600/12 = 800 characters per second. The formula for characters per second is baud rate divided by character length. This is the maximum character transmission rate. This could be limited by the hardware on either end of the serial link.*

In order to achieve the high data transfer rate that the GPIB was designed for, you should limit the number of **devices** on the bus, as well as the **physical distance** between devices. Finally, faster data rates can be achieved with the use of **HS488** devices and controllers.

*Having a limited number of devices on the bus reduces the amount of data transfer, talk time, and distance between the devices. A lower physical distance between each device results in less time to complete a data transfer. HS488 is an extension to GPIB that most NI controllers support that provides increased data rates.*

You have a device that needs to communicate with your computer. The device and the computer are 40 ft. apart. The device has both GPIB and RS 232 Serial ports. The transfer rate of the device is relatively low at about 100 Kb/s. Which communication protocol should you use and why?

*You should use RS 232 serial communication. Serial communication is better over long distances if transfer rates are low.*

*GPIB devices should be a maximum of 4 m apart to stay within protocol. Eight meter cables are available but the transfer rates will be severely diminished. RS 232 cables can be a maximum of 50 ft. apart. If greater distance is required, RS 422 and RS 485 can be up to 4000 ft. apart.*

Different types of serial communication protocol can have a different number of devices send commands (driver) and receive commands (receiver). What are the maximum number of devices that can be a driver and receiver when using different types of serial communication protocol? Find values for RS232, RS422 and RS485. You may need to use other resources than your book.

What are the maximum number of devices that can be a driver and receiver when using serial communication? Find values for RS232, RS422 and RS485. You may need to use other resources than your book.

*RS232 1 driver and 1 receiver*

*RS422 1 driver and 10 receivers*

*RS485 32 drivers and 32 receivers*

*The text book mentions that RS485 can communicate with 32 different devices and RS232 can only do point to point. The student will need to find other resources to finish the problem.*

True or False: There are buses made to communicate with instruments through Ethernet, USB, or IEEE 1394 (FireWire) ports using Serial or GPIB commands. When using these buses, you need to remember to program them differently than when using a standard serial or GPIB bus.

*There are busses made to communicate with instruments through Ethernet, USB, or IEEE 1394 (FireWire) ports using Serial or GPIB commands. When using these busses, you need to remember to program them differently than when using a standard serial or GPIB bus.*

**False:** *When using these ports, program them just as you would if they were using the standard serial or GPIB bus. High level drivers such as VISA abstract these low level bus drivers from the user so the programming remains consistent across all busses.*

True or False: The instrument I/O Assistant is a good option when an instrument driver is not available.

**True:** *The Instrument I/O Assistant is a LabVIEW Express VI which you can use to communicate with message-based instruments and convert the response from raw data to an ASCII representation. You can communicate with an instrument that uses serial, Ethernet, or GPIB interface.*

RS-232 uses two voltage states MARK and SPACE.

Specify which voltage is positive and which is negative.

SPACE is positive.

MARK is negative.

Complete the following truth table for RS-232:

Signal > +3 Volts = 0

Signal < -3 Volts = 1

VISA is a \_\_\_\_ level API that calls \_\_\_\_ level drivers.

- a. **High/Low**
- b. Low/High
- c. Low/Low
- d. High/High

*In LabVIEW, VISA is a library of functions that calls the low level drivers of the specific devices.*



True or False: Data Transfer Termination, or just Termination, is necessary to inform all listeners on the GPIB bus that all data has been transferred.

**True:** *Without termination, devices on the GPIB do not know when one data transfer is finished and another can begin.*

Which of the following drivers does VISA not communicate with?

- a. GPIB
- b. Serial
- c. **DAQ** *VISA communicates with bus drivers for instrument control. DAQ is not a bus driver.*
- d. Ethernet

VISA also communicates with Firewire (IEEE 1394) bus drivers.

How many instruments can GPIB support?

- |  |  |
|--|--|
| a. 1 controller and 5 additional instruments         | <i>GPIB can hold up to 14 instruments</i>            |
| b. 1 controller, 4 talkers, and 4 listeners          | <i>Number of talkers &amp; listeners not defined</i> |
| <b>c. 1 controller and 14 additional instruments</b> | <i>Correct</i>                                       |
| d. 1 controller, 5 talkers, and 5 listeners          | <i>Number of talkers &amp; listeners not defined</i> |

Instrument driver VIs can be broken down into six categories. Match these categories with their description.

- a. Initialize
- b. Configure
- c. Action/Status
- d. Data
- e. Utility
- f. Close

  **f**   terminates the software connection to the instrument

  **d**   transfers data to or from the instrument

  **c**   commands the instrument to perform an action or update its condition

  **a**   establishes communication with the instrument

  **b**   software routines that configure the instrument to perform specific operations

  **e**   performs a variety of auxiliary operations

Under which folder should one copy an instrument driver library folder for them to appear in the function palette for LabVIEW on a Windows based system?

- a. *C:\Program Files\National Instruments\LabVIEW 8.6\*
- b. *C:\Program Files\National Instruments\LabVIEW 8.6\instr.lib***
- c. *C:\Program Files\National Instruments\LabVIEW 8.6\instrumentation*
- d. *C:\Program Files\National Instruments\LabVIEW 8.6\vi.lib*

*You can download instrument drivers (VI and functions for instruments) from [www.ni.com/idnet](http://www.ni.com/idnet). Be sure to check out the readme files for the last information about the drivers.*

Some devices bypass the serial port or the computer's GPIB device but still communicate with instruments through serial and GPIB commands. What other communication busses can be used?

- a. **Ethernet**
- b. **USB**
- c. DVI                      *DVI port is only used for video output*
- d. **IEEE 1394**

*Some devices are designed to communicate with serial or GPIB instruments through Ethernet, USB, or IEEE 1394. However, they typically still use the same serial or GPIB commands.*

Identify the communication system that each of the following are associated with; choose either GPIB or Serial or both.

- a. `ibwrt`      GPIB, this is a specific GPIB function to write data to the GPIB
- b. `\r`      Both, this is a terminal formatting command that can be sent on either system.  
In GPIB it needs to be sent via the `ibwrt` command
- c. `\n`      Both, this is a terminal formatting command that can be sent on either system. .  
In GPIB it needs to be sent via the `ibwrt` command
- d. `ibread`      GPIB, this is a specific GPIB function to read data from the GPIB
- e. `*IDN?`      Both, this is an ASCII string that can be sent in either system. In GPIB it needs to be sent via the `ibwrt` command

You are communicating with a microcontroller via RS-232 and you are trying to receive the string “Hello World” on the serial port on your computer using LabVIEW. However, the string you are receiving is something similar to “¶¶”. Which of the following could be possible problems with your system?

- a. **Bad connection between microcontroller and computer.**
- b. The communication port is not open.
- c. **Data communication rate is set incorrectly.**
- d. The data rate is set correctly, but the read serial buffer loop is not correctly timed to pull data from the received serial buffer quickly enough.

A possible reason could be A because a faulty connection can lead to bad data and should be checked. The answer is not likely B because you are receiving data from the device. However, C is very likely the problem because you are improperly reading data from the serial line and could lead to misinterpreting the signal. Furthermore, the answer is not D because if the data rate is set correctly, then you would be reading "Hello World" but out of order with the letters being switched or missing some letters but you would still get some recognizable characters.



Please match the each GPIB device category to its definition.

- |               |   |                                |
|---------------|---|--------------------------------|
| a. Controller | → | 1. Defines communication links |
| b. Talker     | → | 2. Reads data from the GPIB    |
| c. Listener   | → | 3. Writes data to the GPIB     |

Please define the four parameters that must be specified for serial communication.

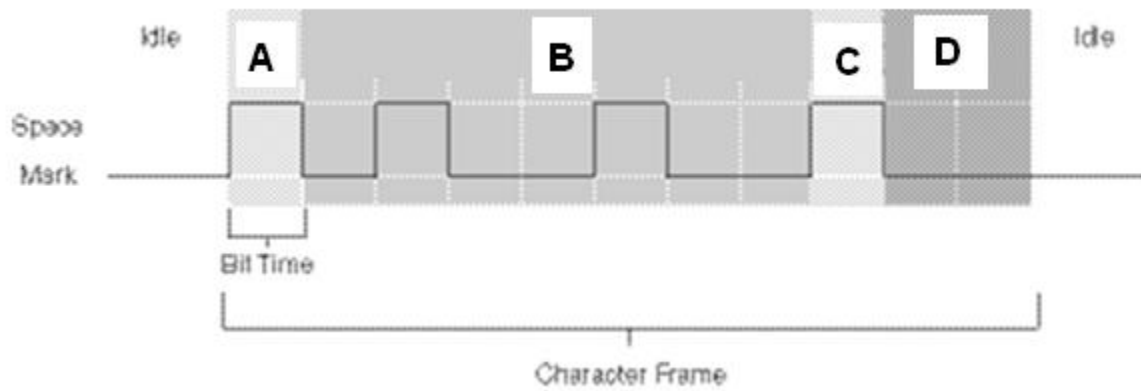
- Baud Rate:** The baud rate defines how quickly data moves between instruments that are using the serial communication.
- Data Bits:** It is necessary to define the number of data bits that encode a character so the data can be ready properly by the receiving device.
- Parity Bit:** Used for error checking, the parity bit specifies if the number 1s among the data bits and the parity bit should add up to be even or odd. If the received data does not follow what is defined by the parity bit, there is an error in the data.
- Stop Bits:** The stop bits, represented by a negative voltage, signify the end of a character frame and stat whether more characters will be transmitted. There are usually 1, 1.5, or 2 stop bits.

*These are explained on page 9-4 in the LabVIEW Basics I manual in Chapter 9, Section C.*

Match the characteristics to the type of instrument control bus.

1. GPIB
  2. Serial
- 
- a.     GPIB     8-bit parallel communication interface
  - b.     GPIB     Data transfer rate of 1Mbyte/s and higher
  - c.     Serial     Uses a transmitter to send data one bit at a time
  - d.     Serial     Data sent over a single communication line
  - e.     Serial     Use this method when data transfer rates are low
  - f.     GPIB     Categorizes devices as controllers, talkers, or listeners
  - g.     Serial     Each device has a unique primary address between 0 and 30
  - h.     Serial     Must specify baud rate, number of bits, parity bit, and number of stop bits

Match the bits to their respective position on the character frame.



  B   Data bit(s)

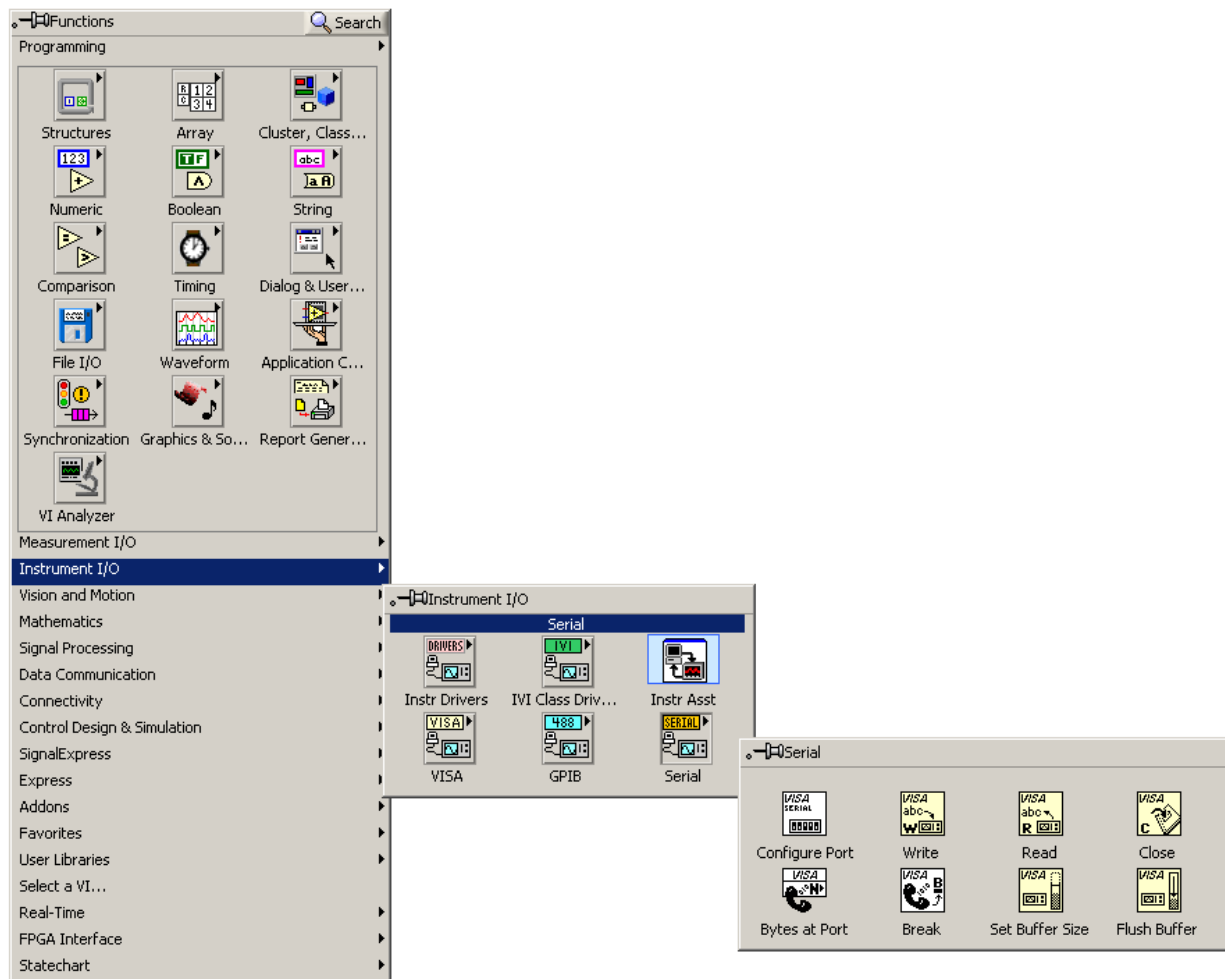
  D   Stop bit(s)

  A   Start bit(s)

  C   Parity bit(s)

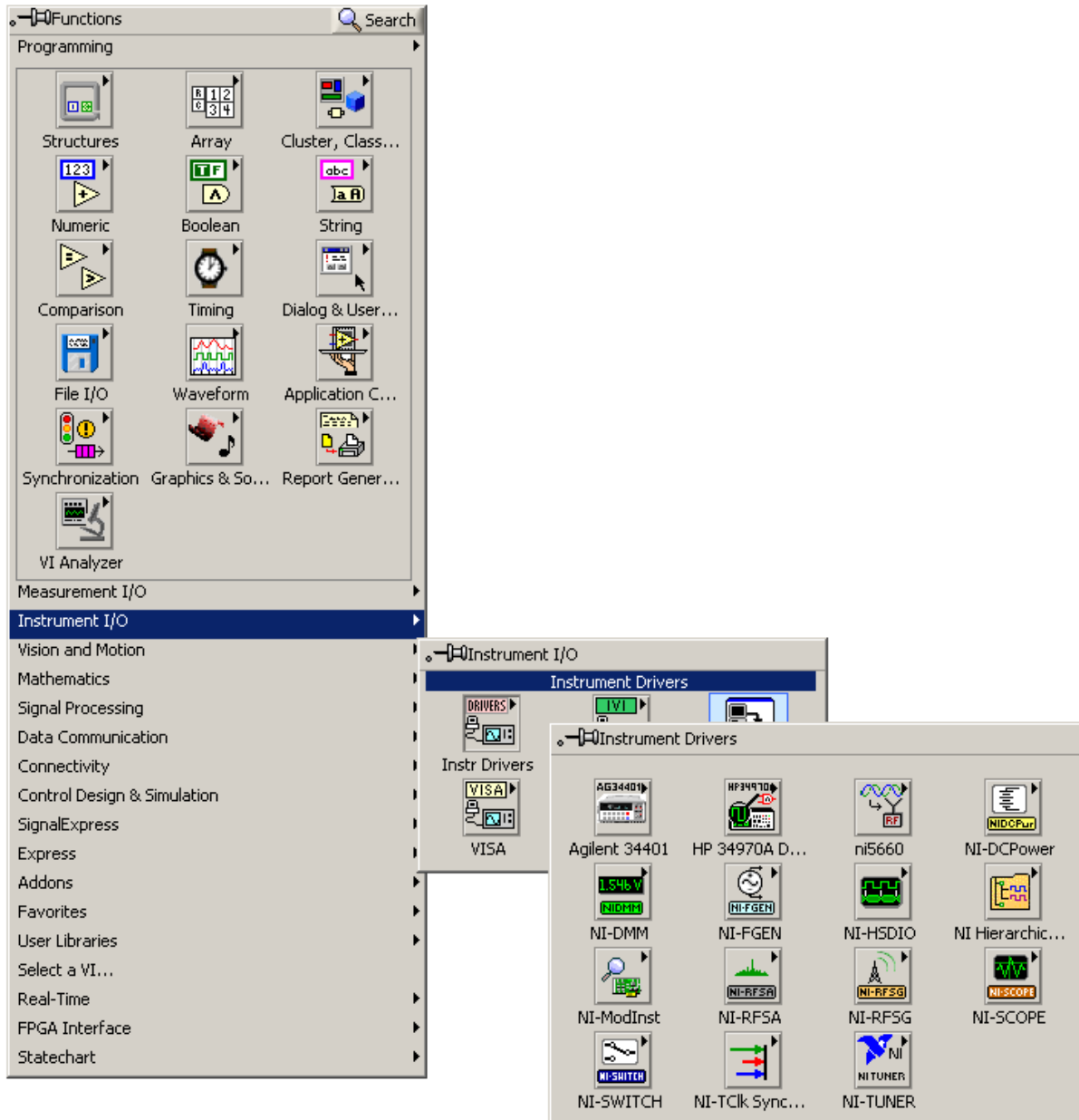
In the **Functions>>Instrument I/O>>Serial** palette one can find VIs for serial communication.

*Must be done in the block diagram.*



In the **Functions>>Instrument I/O>>Instrument Drivers** palette one can find VIs, and sometimes examples, for instrument drivers.

*This has to be done in the block diagram*



True or False: A state machine performs an action for each state in a transition diagram?

**True:** *Actions may be data processing and/or determining the next transition.*

Describe the difference between a Moore machine and a Mealy machine. Which type does LabVIEW most commonly implement?

*In a Moore machine, the actions are performed for each state in the state transition diagram. In a Mealy machine, the actions are performed during each transition of the state transition diagram. The State Machine design pattern implements the Moore machine.*



Is it a good idea to utilize a State Machine structure in a VI instead of a simpler sequence structure even if that is all that is required for the task? If so, why?

*Yes. Expandability is a key to good programming. Often times, a program is written with certain intent and then the scope of the project changes or an additional task is required of it. A little extra time up front can save the programmer from having to completely re-write a program.*

You have a program that utilizes state machine architecture. You made an enumerated type control with each of the states and copied it throughout your program. You later decided to add a case to your program. When you update the control all of the wires that are connected to the copies of the control break. Why does this happen and what would be a solution?

*When you updated the states in the control, it didn't update the states of the copies of the control. The solution would be to save the original control as a type defined enumerated type control. This would ensure that the values of the enumerated type control would update on all of the copies*

*This is a very common occurrence when implementing state machines in a program. Type defined controls are very useful when the constants used throughout the program are going to be changed frequently.*

True or False: An advantage of LabVIEW's dataflow programming is that the programmer does not have to be concerned with the proper order of execution of commands; it is handled automatically by the compiler.

**False:** *Dataflow programming allows the programmer to focus on the data and the corresponding operations primarily. However, the compiler has no way to know in what order the programmer wants sections of the block diagram without data dependency to run; it is important to program the order of execution for these independent operations.*

The best method for controlling the initialization and transition of state machines is the enumerated type control . These are widely used as case selectors in state machines.

*Type defined enumerated type controls are best because they automatically update if you add or remove a case. Standard enumerated type controls can be used but will have broken wires if a state is added or deleted.*

What are the effects (both positive and negative) that result from using sequence structures? Choose all that apply.

- a. Take advantage of the inherent parallelism in LabVIEW
- b. Guarantee the order of execution**
- c. Can stop execution part way through the sequence
- d. Prohibit parallel operations**
- e. Cannot stop the execution part way through the sequence**

What are the reasons to avoid overusing sequence structures? Choose all that apply.

- a. **Sequence structures prohibit parallel operations**
- b. Sequence structures force the operation of block diagram objects

*This is the function of the sequence structure*

- c. **You cannot stop a sequence structure partway through**
- d. **Sequence structures hide code**

Pick the methods used to transition among states. Choose all that apply.

- a. Sequence structure
- b. Creating a control
- c. **Case structure**
- d. **Transition array**
- e. Transition cluster
- f. **State Diagram Toolkit**

Choose the answer choice which describes an attribute of State Programming.

- a. Nothing in the code can be changed
- b. Programming will not rely on conditions
- c. The program will run completely and cannot be stopped before the end of the sequence
- d. Some items in the sequence can be set to execute only when certain conditions are met**

*Only d relates to state programming. Choices b and c relate to sequence structures and a is completely made up.*



What is the best method for controlling the initialization and transition of a state machine?

- a. Ring  
*A ring could be used, but it is not the common method.*
- b. Enumerated Type Control**
- c. Bracelet  
*This is not a LabVIEW function, only jewelry.*
- d. AI Control  
*This is a Traditional DAQ VI that controls the analog input tasks and specifies the amount of data to acquire.*
- e. CTR Control  
*This is a Traditional DAQ VI that controls and reads groups of counter. Control operations include starting, stopping, and setting the output state.*

True or False: Parallel tasks can run simultaneously even if they have data dependencies.

**False:** *In LabVIEW, tasks can run in parallel if they do not have a data dependency between them, and if they do not use the same shared resource*

Choose four components which are required for a state machine:

a. Stop Button

*A stop button can be used, but it is not required. A state machine with a comparison can stop a state machine.*

**b. Case Structure**

c. Dialog Box

*A dialog box can be used to transition states, but it is not required.*

**d. Shift Register**

**e. State Functionality Code**

f. Comparison Statement

*A comparison statement can be used to transition states, but it is not required.*

**g. While Loop**

There are many methods to perform sequential tasks in LabVIEW. Which of the following are applicable methods?

- a. Placing each task in separate while loops from left to right in your main VI and wiring them in the order you want them executed.
  - b. Use of a sequence structure will force the order of operations.**
  - c. Placing your tasks in order from top to bottom in your VI.
  - d. The use of Error Clusters to control data flow.**
- 
- a. While loops do not control sequence, they only perform repetitive tasks. Data will only leave the loop whenever the loop completes its final iteration.*
  - b. Correct answer.*
  - c. Even though many cultures read from top to bottom, that doesn't mean LabVIEW is designed to do the same. LabVIEW will perform whatever task you program it to execute first whether it's on the left, right, top, or bottom of your VI.*
  - d. Correct answer.*

Which of the following is generally considered to be a more appropriate method to incorporate correct execution order of functions or sub-VIs that come without an error cluster on the connector pane?

**a. An Error Cluster into a Case Structure**

*If the function or sub-VI has no error cluster on the connector pane, it is necessary to enclose it in some sort of structure that will allow an error cluster to pass through it, guaranteeing the proper execution order. When available for editing, a more appropriate method of fixing a sub-VI is to edit the sub-VI to include error cluster inputs and outputs.*

**b. A Sequence Structure**

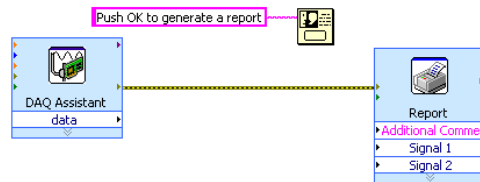
*A negative to using a sequence structure is that it prohibits parallel operations throughout the VI. In addition to other lags within the program, this may impede the VI from closing when expected.*

LabVIEW's state machine design pattern template implements an algorithm described by what type of state machine?

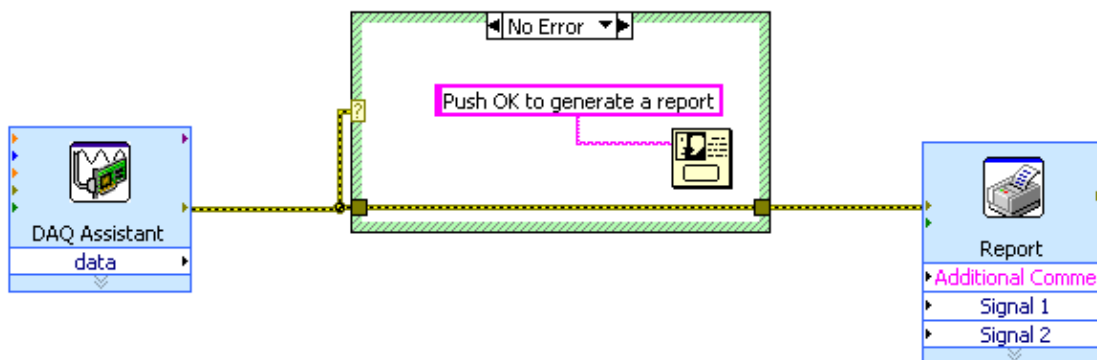
- a. Mealy machine
- b. Moore machine**
- c. Both A and B
- d. None of the above

- a. *A Mealy machine performs an action for each transition.*
- b. *The correct answer, a Moore machine performs a specific action for each state.*
- c. *Incorrect answer*
- d. *Incorrect answer*

Force the One Button Dialog VI to execute between the express VIs. Do this without the use of a sequence structure. Notice that the One Button Dialog VI doesn't use error clusters.

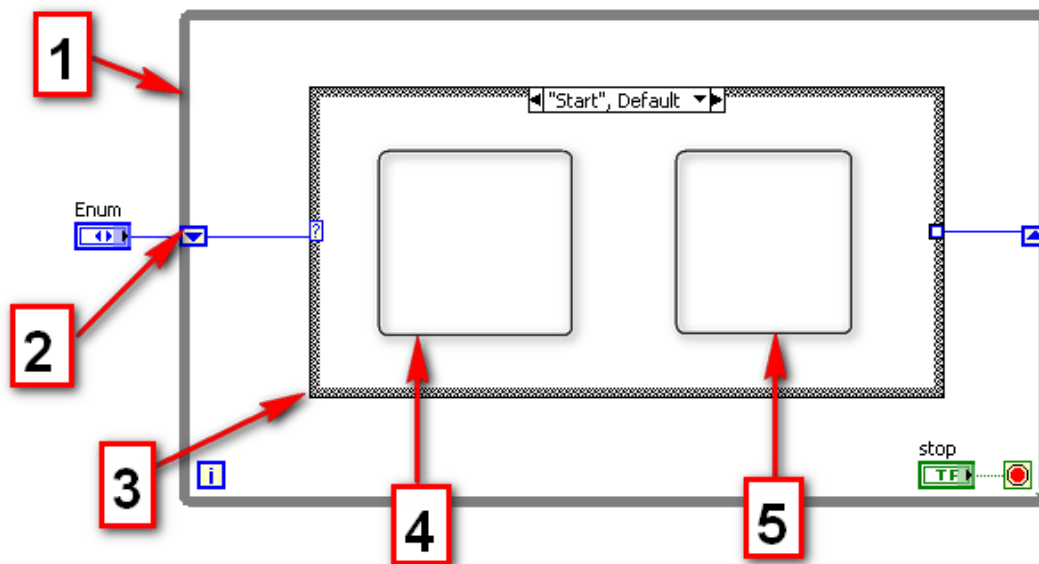


Use a Case Structure as shown below



*Sequence structures prohibit parallel operations. It is better practice to use sequence structures sparingly. In this problem, a case structure can accomplish the desired functionality.*

*Note: There are other ways to force execution order.*



A State Machine Infrastructure has five components which are each identified in the figure above. Label each component of the LabVIEW block diagram. The two empty boxes symbolize where the two types of state machine code should be placed. Give the names for both types of code.

1. While Loop
2. Shift Register
3. Case Structure
4. State Functionality Code
5. Transition Code



A Timing / Wait mechanism is usually required within a loop to keep the code from using all of the processor time.

**False:** *A Wait (ms) VI wired with a constant of zero has the same effect as no wait VI at all.*

*A constant value of zero will yield control of the processor to other tasks if they have requested CPU time. This is not an ideal setting or use of the wait (ms) VI, but it will at least allow for other programs to get infrequent attention as opposed to none.*

Determine whether a state machine or a simple VI architecture is appropriate in each case:

For single components within larger applications: **Simple VI**

*This case describes an application where the user has broken large portions of the functionality into separate VIs, and needs to run each portion in sequence.*

For dividing a VI into a sequence of smaller tasks: **State Machine**

*The state machine fits here because it allows for the programmer to produce more readable code and to provide order to the tasks the VI accomplishes.*

For VIs that act as a user interface: **State Machine**

*The state machine fits here because it allows the programmer to develop states based on the different actions of the user interface. For example, if the user has the option to acquire or record data, the programmer can simply write states that handle both cases and have a transition state that selects the appropriate case (acquire or record).*

For a single task: **Simple VI**

*This is the general definition of the simple VI architecture.*

You have a system that measures a voltage signal at 1000 KHz. For each point that is measured, you want to perform some complex software manipulation. The problem is that the software manipulation takes longer than the time between each sample. What type of loop architecture would best suit this problem?

- a. Master/Slave
- b. Producer/Consumer**

*When the Master/Slave architecture is used, the master loop MUST take longer to execute than the slave loops. In this case, the software manipulation (slave loop) was taking longer than the data acquisition (master loop). In this case a Producer/Consumer loop would be best because of the use of queues.*

Why should you insert a timing function in a continually executing VI?

*If you don't regulate the timing on a continually executing VI, LabVIEW will use all of the processor time and other background processes may not run.*

Each state of the state machine is:

- a. A separate VI
- b. A separate array
- c. A separate while loop
- d. A separate case in a Case Structure**

*A and b are completely made up and are meant to make the student ponder if a state has a to be stored in some separate way. Choice c is incorrect because a state machine only contains one main execution loop. D is the correct answer as cases are used to separate states.*

True or False: In a Master/Slave Loop Design, the slave loop must run at the same rate as the master.

**False:** *In a Master/Slave Loop Design, the slave loop must run at the same rate as the master.*

*The slave loop can operate at a different rate as long as data loss doesn't occur.*

**Queues** are used for communication between producer and consumer loops

*Queues buffer the data created by the producer loop so the consumer can process the data at a separate rate.*

A buffer is a memory device that stores temporary data among two devices.

Queue functions allow you to store a set of data that can be passed among multiple loops running simultaneously or among VIs.

*The definition of a buffer is a memory device that stores temporary data among two devices. Queue functions are used to store data for the purpose of passing that data between loops or VIs that are running simultaneously.*



Use the **master/slave** design pattern when you need a VI to respond to user interface controls while simultaneously collecting data.

Use the **producer/consumer** design pattern when you must acquire multiple sets of data that must be processed in order.

*The definition of master/slave is that it is a design pattern used when you need a VI to respond to user interface controls while simultaneously collecting data.*

*The definition of producer/consumer is that it is a design pattern used when you must acquire multiple sets of data that must be processed in order.*

You have a program that you would like to have run at one hour intervals throughout the day. The program takes five minutes to complete. Your friend tells you that you can use the Wait (ms) function and use 3300000 (55 minutes) as the constant. What is a better way to establish a wait time?

*Use the Elapsed Time express VI. Have the program only execute when the “time has elapsed” Boolean output is true.*



*The Elapsed Time express VI allows the program to do other execution while waiting. This VI uses time stamps. This can be very useful in other timing applications as well.*

Parallelism refers to:

- a. Two things happening side by side
- b. Two things happening at the same time on different computers
- c. **Executing multiple tasks at the same time**
- d. The front panel and block diagram running at the same time

*In LabVIEW, loops can be used to accomplish two tasks running simultaneously. This is referred to as parallelism.*

Choose the benefits of implementing a state machine design pattern from the list below.

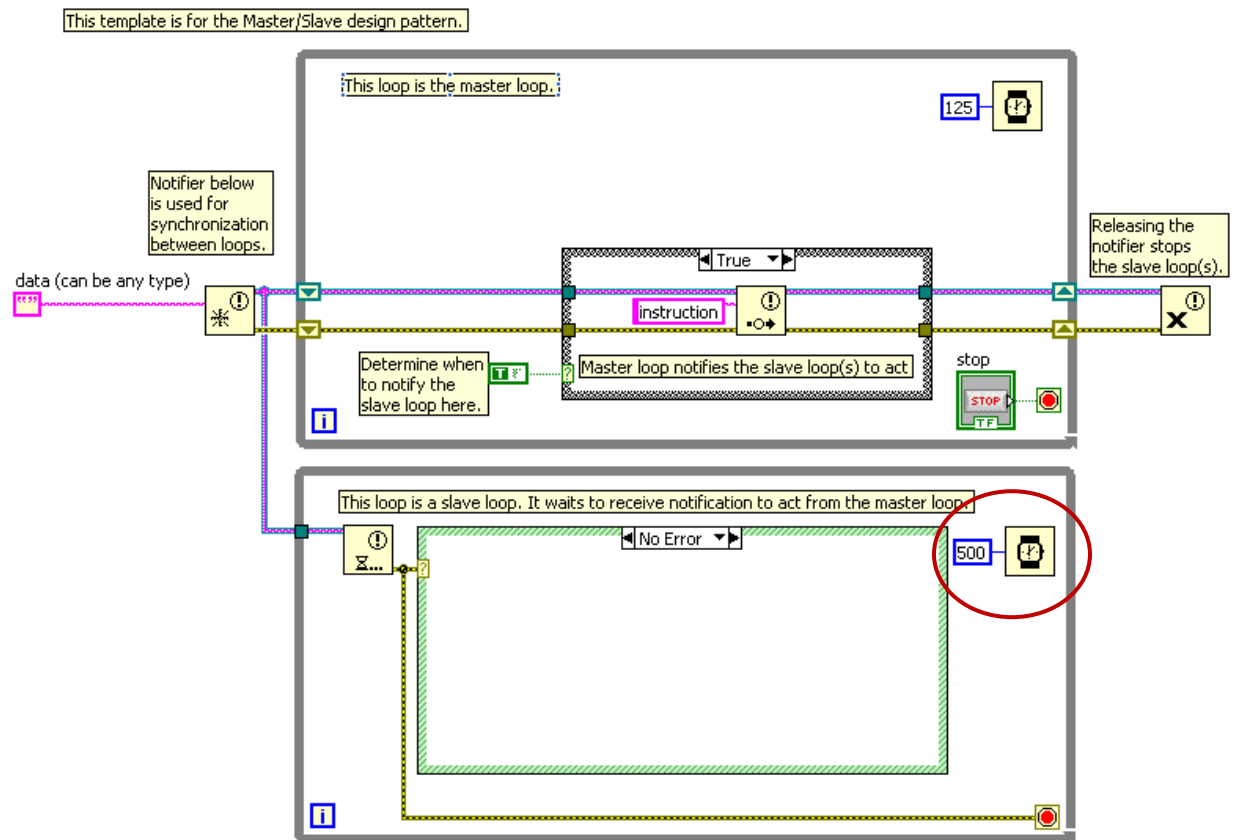
- a. Make the block diagram larger and easier to see
- b. Make the block diagram smaller**
- c. Each case determines the next state**
- d. Executes every frame in sequence

*Making the block diagram smaller means that you programmed more efficiently, which is always good programming practice. A state machine allows each case to determine the next state, unlike a Sequence Structure which must execute every frame in the sequence.*

Match the words with their descriptions.

- 
1. Buffers
2. Queues
- a. are functions that allow you to store a set of data that can be passed among multiple loops running simultaneously or among Vis.
- b. are memory devices that store temporary data among two devices.

Below is a program that uses master/slave architecture. The run arrow is not broken; however, this program will not function correctly. What is wrong with it?



The slave loop executes slower than the master loop. When this program executes, the master loop will tell the slave loop to execute 3 times before the slave loop finishes its first iteration.

This program was modified from the Master/Slave Design Pattern template. Slave loops do not need to have timing in them. They will always wait for the notifier to tell them to iterate

True or False: It is necessary to create a Project File in order to create a Shared Variable.

**True:** *Shared Variables are only created in the Project Explorer within your project file.*

A structure that uses an uninitialized shift register with a **for loop/while loop** is called a functional **global** variable.

*A functional global variable uses the shift register to store data as long as the VI is in memory.*



What is another method of accomplishing the same task that a shift register performs in LabVIEW?

*Use the “Feedback Node” VI. This VI can be found in the Programming → Structures palette on the Block Diagram.*

Determine if the each of the following require a local or global variable:

Sending a stop Boolean to two while loops running in parallel	<b><u>Local</u></b>
Sending a stop Boolean to two VIs running in parallel	<b><u>Global</u></b>
Storing data for access among several VIs	<b><u>Global</u></b>
Storing data in a front panel control or indicator	<b><u>Local</u></b>
Updating a numeric indicator if the value is out of range	<b><u>Local</u></b>
Accessing some acquired data with another simultaneous VI	<b><u>Global</u></b>

*A local variable is stored in a front panel control or indicator and is best used within the same VI.*

*A global variable are stored as a VI and are best used with multiple VIs running simultaneously.*

A Local Variable is accessible from multiple VI's. True or False?

- a. True
- b. False**

*Local Variables store data in front panel controls and indicators within a single VI. Global variables are accessible from multiple VI's.*

Please select all of the reasons why it is advantageous to use descriptive labels for controls and indicators.

- a. The code is more readable**
- b. Simple identification mistakes can be avoided**
- c. Creating a variable is more intuitive**
- d. It serves as additional documentation**

*All answers are correct.*

What are some good techniques to avoid race conditions?

- a. **Specifying execution order**
- b. **Controlling and limiting shared resources**
- c. **Protecting critical sections within your code**
- d. Using local variables instead of global variables

*Local variables and global variables can both have race conditions. If you have a case where both variables would work; there isn't an advantage of using local instead of global in terms of race conditions*

(1)**Local** variables store data in front panels controls and indicators.

(2)**Global** variables and (3)**single process shared** variables store data in special repositories that you can access from multiple VIs.

(4)**Functional global** variables store data in While Loop shift registers

- a) Global - **2**
- b) Local - **1**
- c) Functional global - **4**
- d) Single processed shared- **3**

To mark a VI as non-reentrant means that...

- a. After it is executed once in the code, that sub-VI cannot be called again until LabVIEW is restarted.

*There is no native function in LabVIEW that behaves as described.*

- b. It can only be accessed by one process at a time.**

- c. Calls to multiple instances of it can execute in parallel with distinct and separate data storage.

*This is the behavior of a reentrant VI.*

In LabVIEW, what defines the execution order of block diagram elements?

- |  |  |
|--|--|
| a. Sequential order of commands          | <i>Only in text-based languages</i>                            |
| b. Command list                          | <i>There is no command list in LabVIEW</i>                     |
| <b>c. Flow of data</b>                   | <i>Correct-Command is executed when it receives all inputs</i> |
| d. VI last selected in the block diagram | <i>This has no effect on execution order</i>                   |



With specific respect to Producer/Consumer Architecture, Queues can be used to pass data from one loop to another one running in parallel. What happens when the buffer is written to too quickly and overflows? How about when the buffer isn't written to quickly enough and it is empty?

- a. The program will crash if either event occurs.  
*Not true.*
- b. **If consumption is slower than production, the queue will become full and the producer code will be forced to wait until the consumer has de-queued an element before a new element can be queued. If consumption is faster than production, the queue will become empty and the consumer code will be forced to wait until the producer has queued an element before the consumer code can execute.**
- c. If consumption is slower than production, a buffer overflow error will occur. If consumption is faster than production, the code will execute the consumer code on the last received data.  
*Not true.*

*Queues store their data in FIFO buffers. The size of the buffer can be set.*

Fill in the blank in this sentence.

A \_\_\_\_\_ provides the elegant implementation of the master/slave design pattern for data transfer because it sends a signal when the data is ready and removes any issues with race conditions.

*Answer: Notifier*

It is important to use variables correctly in your VI. To initialize a local or global variable properly, verify that it contains the expected data value before the VI runs. If you do not initialize the variable before the VI reads it for the first time, the variable will contain *the default value of the associated front panel object*.

Finish the sentence.

Explain the differences between Master/Slave and Producer/Consumer loop architecture.

*Points that should be included in the response*

- *P/C uses queues while M/S uses notifiers*
- *P/C buffers the data so it all gets processed*
- *P/C is based off M/S architecture*
- *In M/S the master loop must execute slower than the slave.*
- *In M/S the slave loop will execute only when the master tells it to. In P/C the consumer loop will execute as long as there is data in the queue.*
- *P/C is best suited for data acquisition if the data is created faster than it is processed*
- *M/S is best suited when you have certain loops that you will tell to iterate at different times*

Name one way to protect the critical section of your code and give a brief description of this method.

*Two Possibilities:*

*Functional Global Variables: Place the critical section of the code in a non-reentrant subVI and use the functional global variable architecture (using shift registers). This way you may only call it from one location at a time and will keep the code from being interrupted by another process calling the subVI.*

*Semaphores: allows one task to acquire it at a time so you can prevent critical sections of code from interrupting each other by enclosing each between an Acquire Semaphore and Release Semaphore.*

Which of the following statements about the use of variables in LabVIEW (and other dataflow languages) are true?

- a. Block diagrams can become more difficult to read.**

*Trying to follow where variables are written across multiple VIs and sub-VIs can be complicated with increasingly large programs.*

- b. Variables make it easy to share data between multiple VIs running on the same computer.**

*Rather than using wires to route the data from one place to another within the program, sometimes many sub-VIs deep (true dataflow programming), a variable can be used. Also, when dealing with parallel loops, there is no other way of successfully getting certain information from one loop to the other(s).*

- c. Variables make it easy to share data between multiple VIs running on different computers across a network.**

*Network variables allow any computer running a LabVIEW VI on the same network to access the data they contain. LabVIEW does have security settings to prevent unauthorized access to your data.*

- d. Variables can cause unexpected VI behavior.**

*If multiple processes are writing to the same variable at the same time, a race condition has been created. This will prevent the user from knowing the state and value of a certain variable's value.*

- e. Variables can slow performance.**

*Dataflow programming is designed to not use variables. Consequently, the use of them, though sometimes necessary, can impede performance.*

- f. There are some cases when a variable is the only way to accomplish a task.**

*Passing data between loops that run in parallel.*

What is this a picture of?



- a. **Local Variable not associated with a control or indicator**
- b. **Local Variable that has had its control or indicator deleted**
- c. Local Variable that was given a wrong data type
- d. Global Variable that lost its connection between VIs

*If a local variable is given the wrong data type, the wire will become broken*

What is the difference between these local variables?



- a. The top variable is set to read and the bottom is set to write
- b. The top variable is set to write and the bottom variable is set to read**
- c. The bottom variable is a cluster of Boolean values
- d. The bottom variable corresponds to an indicator and the top corresponds to a control

*Variables can correspond to indicators or controls with no appearance change*



True or False: A Local Variable can be written to or read from regardless of whether it is a control or indicator.

**True:** *With a local variable you can access a front panel object as both an input and an output.*

Name one good programming technique that will help you avoid race conditions.

*A correct solution will mention one or more of the following topics:*

- *Control, Limit shared resources for example when two tasks have read and write access to a resource like a shared variable*
- *Identify and protect critical sections of your code*
- *Specify execution order*

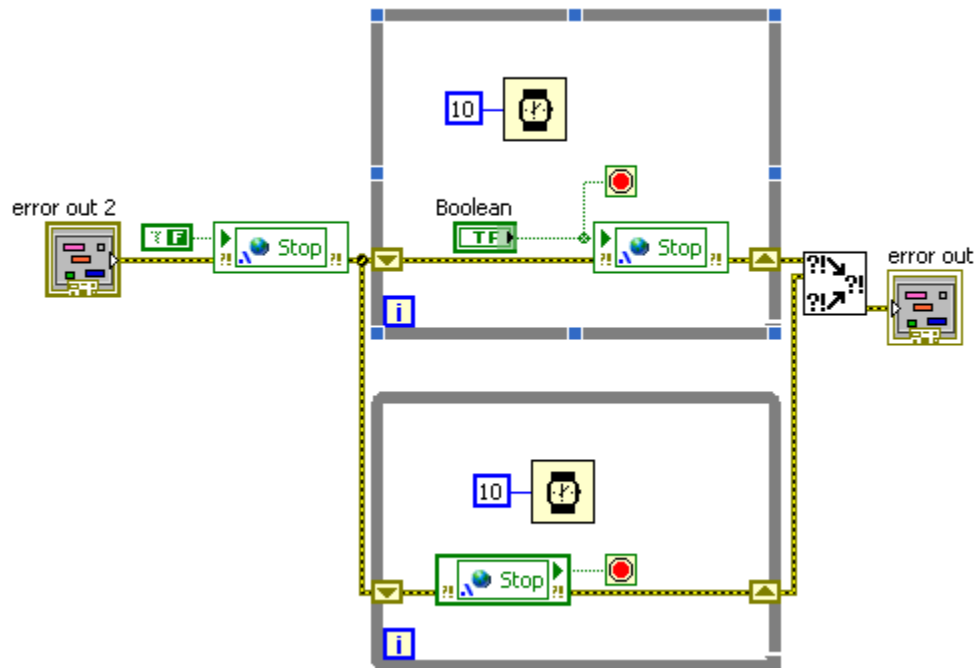
When the user creates a global variable, LabVIEW automatically creates a special global VI which only has a \_\_\_\_\_.

- a. Block Diagram *There is no block diagram for the global VI created*
- b. Front Panel**
- c. File Path *The user must later save the global VI after adding controls or indicators*
- d. Data Type *The controls or indicators added to the front panel will define the data types of the global variables it contains.*

Match the following types of Labview Variable to their usage:

- 1) C *Local Variables share data within a single VI*
- 2) B *Global Variables allowing sharing of data among many VI's of one computer*
- 3) A *Shared Variables allow sharing data across VI's on different computers on the same networks*

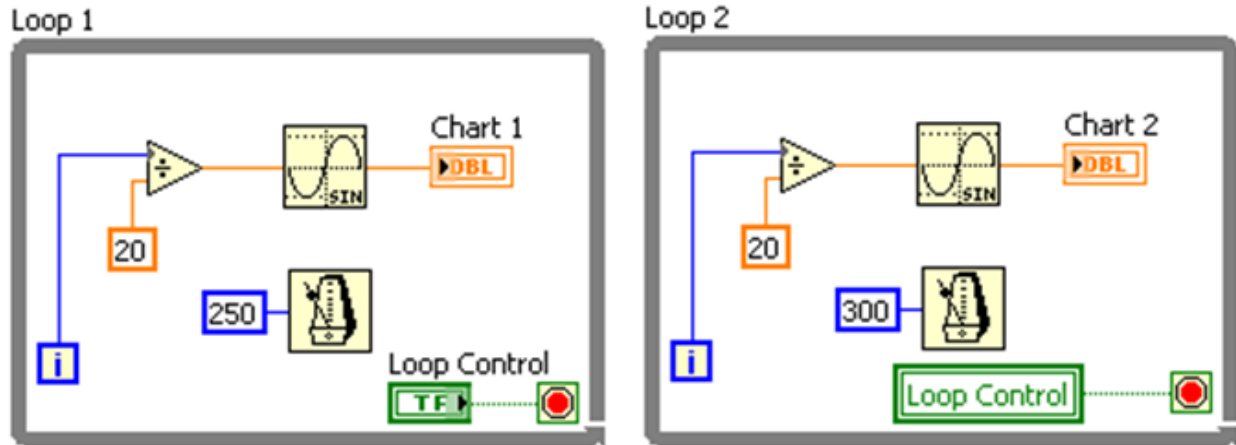
The block diagram below displays a common mistake when using variables. The shared variable Stop synchronizes the stop conditions for two loops. Determine the behavior of this code and diagnose the problem and solution.



*This example operates fine the first time it runs because the default value of a Boolean is false. However, each time this VI runs, the Stop control writes a True value to the variable. Therefore, the second and subsequent times that this VI runs, the lower loop stops after only a single iteration unless the first loop updates the variable quickly enough.*

*The solution above has code added to initialize the shared variable. By doing so, we insure that the second loop does not immediately stop.*

Which loop will be the first to execute its first iteration?



- a. Loop 1
- b. Loop 2
- c. They will execute in parallel
- d. No way to tell

*Since there is no data input to either of the loops, and there is no data dependency between them, either loop could be the first execute its first iteration. LabVIEW tends to execute from left to right when there is no data dependency, but this is not guaranteed.*

An event is an **asynchronous** notification that something has occurred.

In an **event-driven** program, events that occur during execution directly influence the execution flow. In a **procedural** program, all execution is predetermined and follows a sequential order.

*Event-driven programs hold code for specific events, and change which code is executed based on the events that take place. LabVIEW Full and Professional Development Systems provide event structures that allow the user to create these types of programs.*



Should the event structure be located in the producer loop or the consumer loop, if you are using an event structure and the producer / consumer architecture to continuously and asynchronously monitor and update the front panel with user inputs?

*Producer loop. By keeping the user interface event structure in the producer loop, the user will be able to continuously use the interface. If commands are given too quickly, they will be stored in a queue until the consumer loop can get to them.*

Complete the following sentence.

There are two types of user interface events-   **notify**   and   **filter**  .

Match your answers from the above sentence to the definitions below.

  **Notify**   events are an indication that a user action has already occurred, such as when the user has changed the value of a control.

  **Filter**   events inform you that the user has performed an action before LabVIEW processes it, which allows you to customize how the program responds to interactions with the user interface.

*These are the definitions of Notify events and Filter events.*

A developer is designing a program that uses event structures. The front panel of the program has ten buttons on it, and each button corresponds to an action that the program must execute. The event structure has an event set up to run for each of the buttons. Each of these events takes about one minute to complete. The user wants to be able to tell the program to complete three of these tasks in sequence; however, he or she must wait for each action to complete before he or she can select the next task from the front panel. What should the developer of the program do to fix this issue?

*The developer should implement a Producer/Consumer (Events) Design Pattern. He or she should put an Enqueue Element VI in the case structure for each case. Then the consumer loop should be built with the Dequeue Element VI. All of the programming should be moved to the consumer loop.*

*In this architecture when the user specifies an event, the program doesn't execute the event. It sends the event to a queue. The main programming takes place in the consumer loop and will run in the background while the user interface is always responsive. The user interface doesn't lock up because it takes very little time to send data to a queue and complete the event.*

True or False: You can turn off front panel locking for both notify and filter events. This dialog box is found in the edit events dialog box

**False:** *You CANNOT turn off front panel locking for filter events. This dialog box is located in the edit events dialog box.*

True or False: You want to make the close button on the front panel of a VI not work and display a message when clicked. You would want to use a filter event.

**True:** *Filter events let you change the default behavior of an action. In this case, the close button will not close the window; rather, it will display a message. Notify events will only respond to a user input. The LabVIEW default action will still take place.*

Dynamic event registration avoids the limitations of static registration by integrating event registration with \_\_\_\_\_, which allows you to use Application, VI, and control references to specify at run time the objects for which you want to generate events.

- a. **The VI Server**
- b. MAX
- c. A cluster of global variables

True or False: If you wire a value to the timeout terminal in an event structure, you **MUST** have a timeout event.

**True:** *When you wire a number to the timeout terminal, LabVIEW will automatically create a timeout case if one was not already made.*

Which of the following events are supported by LabVIEW?

- a. A hardware timing signal that determines when data acquisition is complete  
*LabVIEW does not support external I/O events*
- b. **A mouse click that starts data acquisition**  
*This is an example of a user interface event, which LabVIEW supports*
- c. **A property node that signals a change in a variable's value**  
*LabVIEW supports programmatic events, in this case the property node triggers a value change event*
- d. An external trigger that signals when an error occurs  
*Again, this is an example of an external I/O event that is not supported by LabVIEW*
- e. **A key press that exits the VI**  
*This is another example of a user interface event*



LabVIEW does not support external I/O events. True or False?

**False:** *LabVIEW supports user interface and programmatically generated events, but not external I/O events.*

Please select all advantages of using user interface events.

- a. Events allow synchronization with user actions on the front panel**
- b. Events are automatically aware of the actions of other programs
- c. Events remove the need to poll the state of front panel objects**
- d. Events allow specific event-handling based on user actions**

*User interface events serve as interrupts that allow a program to take specific actions based on the user interface without the need to poll every element in the user interface. This increases modularity within the program while greatly reducing the computing power necessary to run the program.*

True or false: You must wire event structure output tunnels in every case.

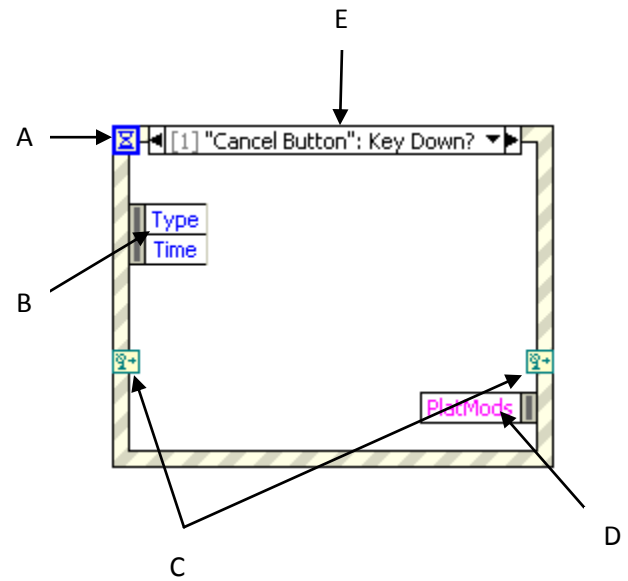
**False.** *Although you can force all of the output tunnels to be wired by right-clicking a tunnel and deselecting Use Default if Unwired, by default the output tunnels do not have to be wired for every case.*

The Event Structure works like a Case Structure with a built-in Wait on Notification function.

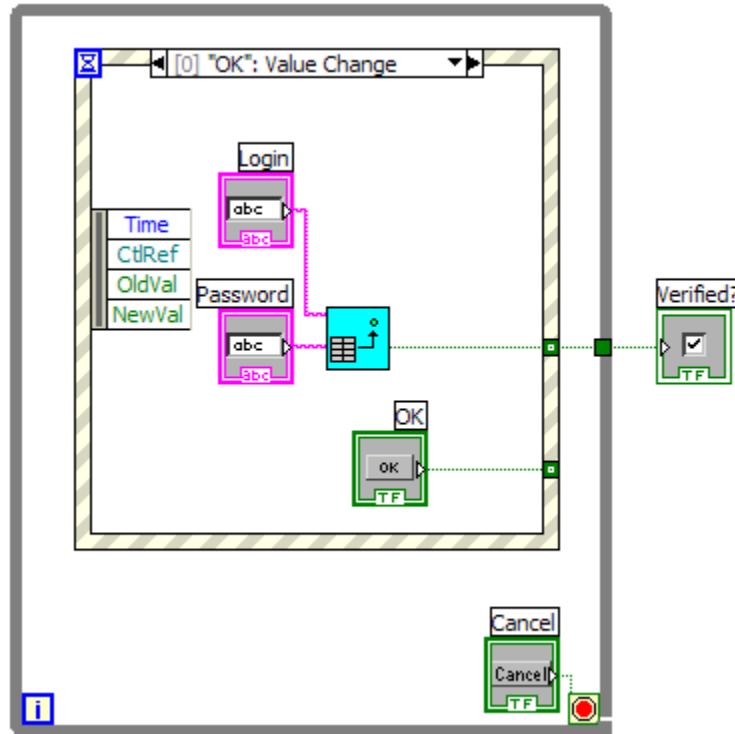
*Event Structures are very similar to Case Structures except that they have a built in Wait on Notification case so that they are ready to execute when an event occurs.*

Match the event structure components to the picture.

1.   C   Dynamic Data Terminals
2.   A   Timeout Terminal
3.   E   Event Selector Label
4.   B   Event Data Node
5.   D   Event Filter Node



The following code appears to be unresponsive when a user clicks the cancel button:



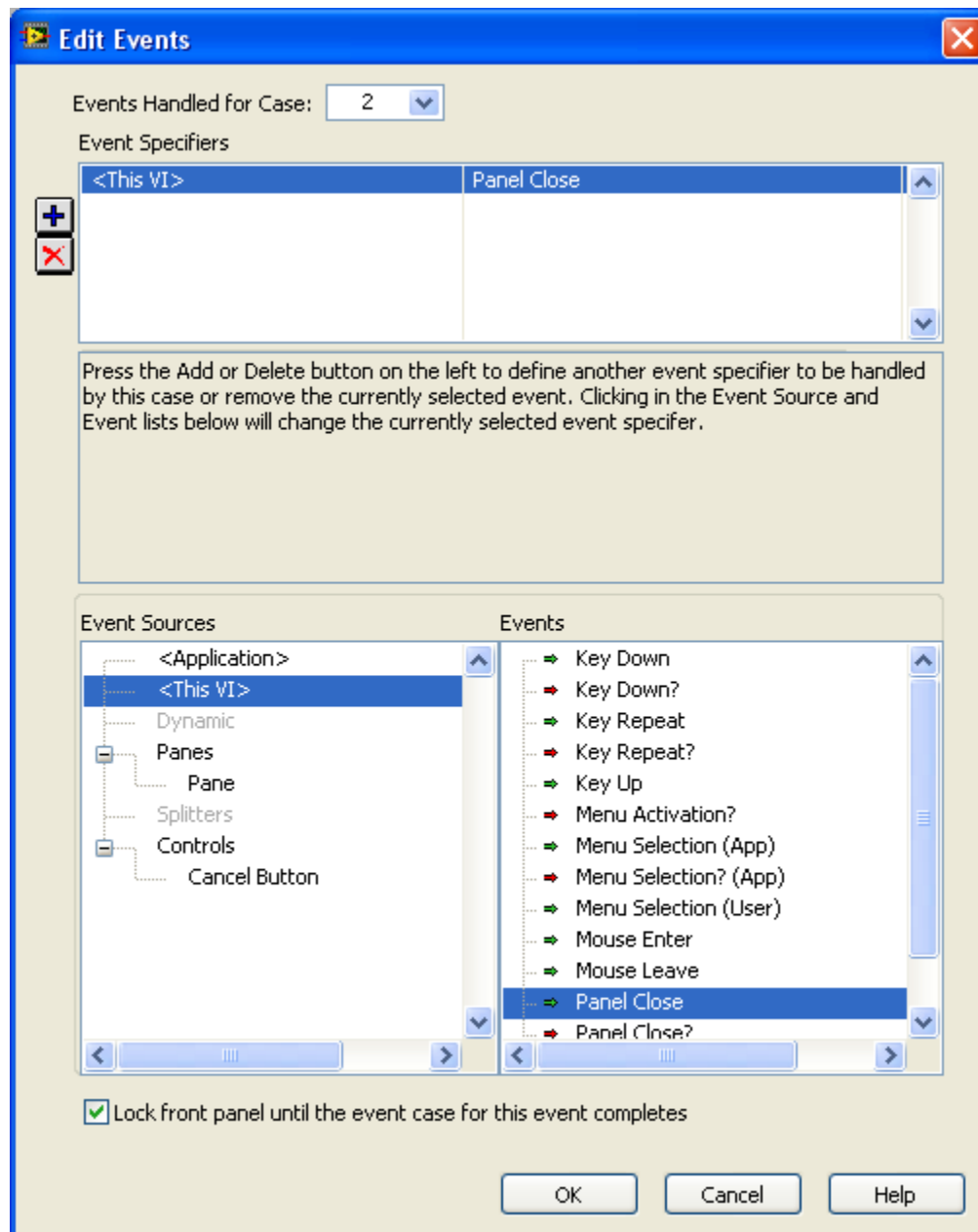
Explain why this error is occurring and how it can be fixed.

*Event structures lock up the front panel until the event structure is executed by default. There are two solutions, one is to disable the feature that locks the front panel. A more elegant solution is to develop a case that handles exiting the loop.*

*You can change the default behavior of the event structure to prevent it from locking the front panel by right-clicking the event structure and selecting “Edit Events Handled by This Case...” and removing the checkmark from the Lock front panel until the event case for this event completes checkbox. The second and preferred method for fixing this issue is to generate an event case that handles when the cancel button is clicked to end the loop.*

True or False: In the edit events dialog box, the red arrow represent filter events and the green arrow represent notify events.

**True:** *Notify events are an indication that a user action has already occurred. Filter events inform you that a user action has occurred before LabVIEW process it. Filter events can be used to change the default response to an action.*



True or False: Refactored code performs the same function as the inherited code.

**True:** *Refactoring does not change the observable behavior of the VI. Refactoring makes it easier to understand and maintain, which makes the code more valuable because it is easier to add features or debug. However, adding features and debugging are not part of refactoring.*



Creating well-designed software facilitates rapid **development** and decreases possible **decay**.

*Rapid development is side effect of good software design because less time is spent trying to determine what the code is doing and how it flows. Also, if decay happens, finding the source of the failure takes significantly longer.*

*The chance of decay is lessened because trouble spots are more easily identified.*

If the block diagram is not readable, but the VI works, should the VI be rewritten? Why or why not?

*Sometimes. There is value in a working VI, so it should not be rewritten unless it is simple and you understand the VI well. Another situation in which VIs should be rewritten is that the VI satisfies only a small portion of your needs.*

*Rewriting can also be done if the VI does not function.*

The key combination **CTRL+U** can be used to help align and reorder nodes on a block diagram.

*The block diagram cleanup tool is a new feature found in LabVIEW 8.6.*

True or False: When refactoring VIs, it is better to make small, incremental changes and test the VI after each change.

**True:** *Making small, incremental changes helps with managing the risk of introducing bugs. By testing after each change, bugs are detected sooner and are easier to find.*

Briefly describe the tradeoff between Refactoring a piece of code as opposed to Optimizing its Performance.

*These two mentalities are not the same. By Refactoring, the goal is to make the program easier to read, understand and maintain but does not change its execution speed.*

*When optimizing performance of code to make it run faster or use less memory, the code often can become more difficult to read and understand. That is the tradeoff.*

Match the following terms to the most appropriate option: Refactoring, Performance Optimization

**Performance Optimization** - makes the VI easier for a computer to process.

**Refactoring** - makes the VI easier for a human to understand.

True or False: The block diagram cleanup tool can assist in refactoring a VI?

**True:** *Remember that the block diagram clean up tool is a new feature found in LabVIEW 8.6. However, is it not the end-all, cure-all for organizing code.*

What problems should be solved by refactoring code?

- a. **Overly large block diagram**
- b. **Poorly named objects**
- c. Broken run arrow
- d. **Duplicated logic**
- e. **Unnecessary logic**

*A broken run arrow is not a problem that refactoring will solve. Refactoring doesn't change code; it just changes the appearance of the code.*



If possible, the size of a block diagram should be no larger than **one screen**.

*When block diagrams are larger than one screen, it is more difficult to read the code. Scrolling complicates understanding the flow of the VI. A block diagram's size can be reduced by using subVIs. If the block diagram is still larger than one screen, try to reduce scrolling to one direction only.*

Which of the following are attributes of good programming practice using LabVIEW?

- a. **Using descriptive labels for controls and indicators**
- b. **Creating custom icons for subVIs**
- c. **Saving subVIs with descriptive names**
- d. Keeping the block diagram to scrolling only one direction – *It is better to fit the block diagram onto one screen.*
- e. **Using subVIs**
- f. Placing all controls together and all indicators together on the block diagram- *Placement of controls and indicators varies from VI to VI depending on what the code is doing and when it utilizes its controls and indicators. However, controls should be more to the left and indicators should be more to the right, generally speaking.*

Choose the conditions that might warrant that the VI might need to be refactored.

- a. Dataflow moving from left to right.
- b. Dataflow moving from right to left.**
- c. Multiple subVIs.
- d. Lack of modularity.**
- e. Large white space in block diagram in False case of case structure.
- f. Lack of comments.
- g. Use of Express VIs.
- h. Unnecessary logic (e.g. not using auto indexing on for loops).**

*Remember that multiple issues can warrant refactoring code. Dataflow moving from left to right is good practice, whereas dataflow from right to left should be avoided. Using multiple subVIs is good practice and enhances modularity. A lack of modularity could mean that code should be refactored. Large white space, lack of comments, and using express VIs alone do not warrant refactoring. Unnecessary logic should be removed, so long as the resultant code is still easy to understand.*

Determine a meaningful name for each of the following controls or VIs. Why would creating meaningful names be important to a VI?

- a. A Boolean control that signals when the application has finished acquiring temperature values for 5 minutes  
**Some potential meaningful names include “Acquisition Done?”, “Acquired Temperature?”, or “Finished Acquiring Temperatures?”**
- b. A SubVI that takes a username and password and outputs a true if the password is correct for the username  
**Some potential meaningful names include “Login”, “Login SubVI”, “Authenticate User”, or “Check Username/Password”**
- c. A VI that takes an array of numeric values and determines the mean, median, mode and standard deviation of the values  
**Some potential meaningful names include “Get Statistics”, “Mean, Median, Mode, Std. Deviation”, or “Find Stats”**
- d. A graph indicator that displays voltages acquired over 200 samples from a thermocouple  
**Some potential meaningful names include “Thermocouple Voltage”, “Thermocouple Samples”, or “Thermocouple Readings”**

*Creating meaningful names is important in VI design because it helps to make the VI more readable and understandable in the event that multiple developers work on the same code.*

You have inherited a functional VI from a co-worker. You are required to add functionality to this VI, but it is very difficult to understand the code. Please order the following actions in the sequence you would use them to implement your additional functionality and for use in the future. All actions may not be required.

- a. Analyze the code to understand its purpose
- b. Remove unnecessary logic
- c. Program the additional functionality
- d. Replace formulas with basic math functions
- e. Relabel controls and indicators with meaningful names
- f. Replace repetitive code with subVIs
- g. Simplify algorithms
- h. Replace sequence structures and local variables with data flow wires

1.     **a**     *It is important to understand the code before changing it*
2.     **e**     *This simplifies changes, programming, and increases readability*
3.     **b**     *This simplifies the code and increases readability*
4.     **h**     *This increases efficiency and takes advantage of data flow*
5.     **f**     *This cleans the code further and simplifies further programming*
6.     **g**     *This increases readability and execution speed*
7.     **c**     *It is important to do steps 1-6 before adding to a poorly written VI*
8.

*Replacing formulas with basic math functions will only clutter algorithms further and does not considerably increase efficiency.*

Briefly describe one situation where creating a subVI would improve code design.

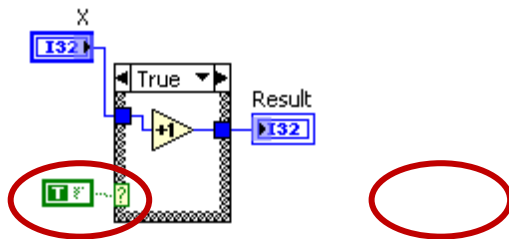
A correct answer will mention either of the following:

- A repetitive piece of code should be placed in a subVI (duplicate logic)
- A disorganized or complicated section may be placed in a subVI to improve readability

The main VI has become very large and difficult to navigate (larger than screen size, lots of scrolling)

You want to increment X when True, do nothing when False, and show the result in the Result indicator.

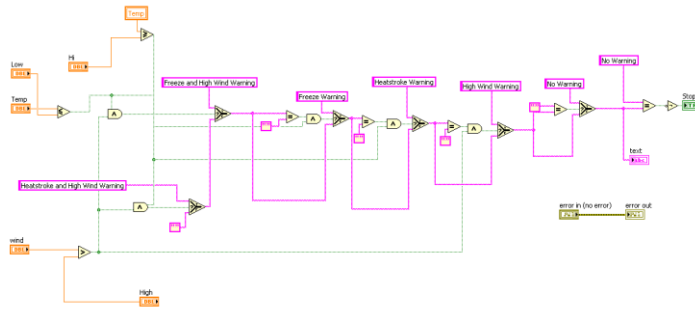
What is wrong with the following VI (True and False cases shown)? How should it be fixed?



*A true constant is wired into the case selector of the case structure. Therefore, the false case will never run. Since the false case does nothing, the case structure is unnecessary.*

*The VI can be fixed by removing the case structure.*

What should be refactored in this block diagram?



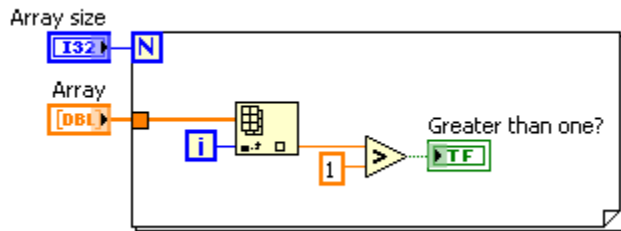
*Points that the student should make*

1. *Move controls to left*
2. *Move indicators to the right*
3. *Give objects better names*
4. *Make data flow from left to right*
5. *Reduce the number of bends in wires*
6. *Do not allow wires to run under objects*
7. *Remove unnecessary logic (not after the equals should be changed to not equal function)*
8. *Replace duplicated section of code with sub VI*
9. *Delete variable and use wires*

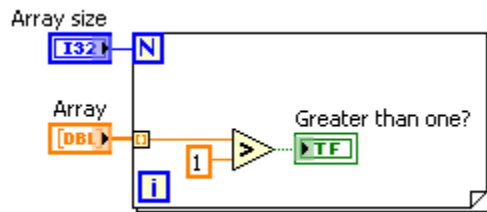


Given the following code, which of the following would be a good way to remove the unnecessary indexing logic yet still accomplish the task of indicating that the current indexed element is greater than one?

Original code.



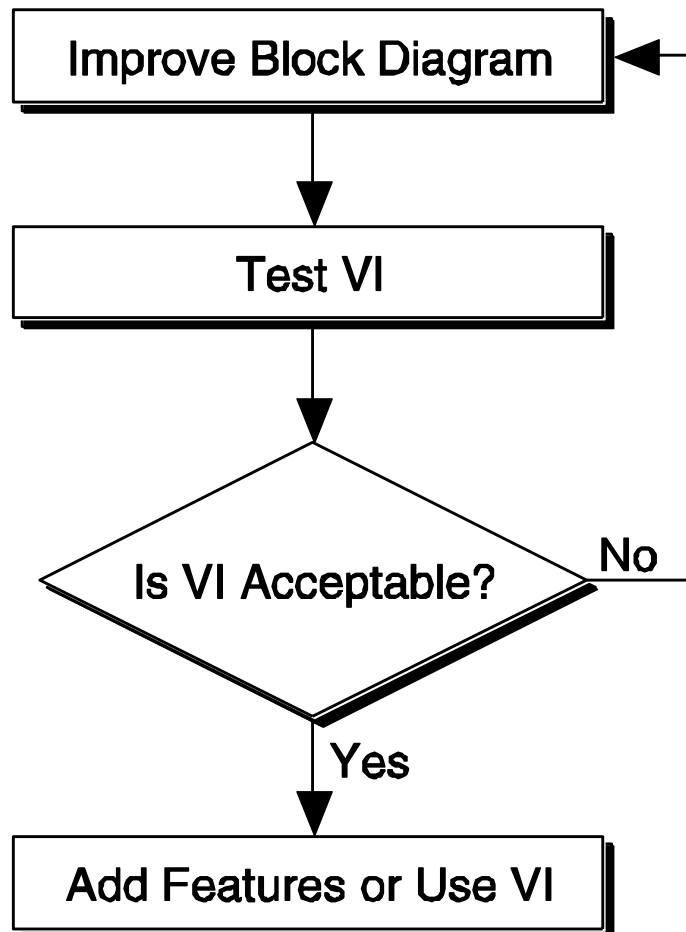
Best answer



e.

*Remember that LabVIEW for loops can auto index arrays. All the other answer still have unnecessary logic by trying to manually index arrays (some of them incorrectly).*

Please fill in the flow chart below with the proper steps for refactoring a VI.



*When refactoring a working VI, always test each small change before proceeding to further changes in order to confirm that your most recent changes were successful.*

True or False: Each front panel object is allowed to have one property node associated to it.

**False:** *You can create multiple property nodes for the same front panel object.*

The **Quit LabVIEW** function stops all executing VIs and ends the current instance of LabVIEW.

Illustrate how LabVIEW employs object-oriented programming by giving an example object from a LabVIEW class that you know of.

*The student should give an example of a Control or a VI and indicate that this object belongs to either the Control or VI class.*

*For example:*

*A Stop button is an example of an object of the Boolean Class. And the Boolean class is sub-class of Control. So the stop button is also a member of the Control Class.*

Determine whether a property node or invoke node would be appropriate in each case:

- |                        |  |
|------------------------|--|
| <b><u>Property</u></b> | Resize a front panel control                                 |
| <b><u>Invoke</u></b>   | Center the front panel when the VI runs                      |
| <b><u>Property</u></b> | Hide a graph before data acquisition                         |
| <b><u>Property</u></b> | Make an LED begin blinking when an invalid string is entered |
| <b><u>Invoke</u></b>   | Export an image of a graph indicator                         |
| <b><u>Invoke</u></b>   | Reinitialize all front panel controls to their default value |

*Property nodes access the attributes of objects for modification*

*Invoke nodes perform the methods of objects on an application or VI.*

True or False: You can change the color of decorations programmatically using properties nodes.

**False:** *You can change the color of decorations using the change color tool while editing the front panel or block diagram, but you can't change them with property nodes.*

Choose all that apply to LabVIEW VI Server Architecture.

- a. **It is an object-oriented feature of LabVIEW**
- b. It is a JAVA feature of LabVIEW
- c. **Invoke and properties nodes use the VI Server**
- d. **Provides programmatic access to LabVIEW**
- e. Provide dataflow programming for LabVIEW
- f. Only works on Window versions of LabVIEW

*Choice b is incorrect because JAVA is an object-oriented language, but it is not what VI Server is. Choice e is incorrect because VI Server relies on the dataflow feature of LabVIEW. Choice f is incorrect because VI Server is platform-independent and works on Mac OS and Linux as well.*



When a property node is placed in a subVI, which of the following is true? Choose all that apply.

- a. The front panel object that the property node is linked to is no longer seen on the front panel  
*-true but e is correct answer*
- a) A copy of the property node is placed in the subVI  
*-the property node is transferred into the subVI, it is not just a copy.*
- b) A control reference is needed to explicitly link the property node  
*-true but e is correct answer*
- c) One of the above  
*-two are true*
- d) Two of the above**  
*- a and c are true*
- e) None of the above  
*- a and c are true*

True or False: Property nodes execute top to bottom.

**True:** Property nodes execute top to bottom.

Please select the correct statements regarding the characteristics of generic and specific refnum classes.

- a. A specific refnum class will only have properties specific to that class
- b. Both generic and specific refnum classes share some properties**
- c. Specific refnum classes have more properties available than generic classes**
- d. A generic refnum class is more restrictive than a specific refnum class

- a. *Specific refnum classes still share some basic properties with other classes*
- b. *Correct*
- c. *Correct*

*Specific refnum classes are actually more restrictive than generic refnum classes since they accept fewer types of objects*

Explain the purpose of property nodes.

Short answer: *Property nodes allow the programmer to modify the front panel programmatically.*

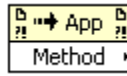
Long answer:

*Use the Property Node to get and set various properties on an application or VI. The following are examples of how Property Nodes can enhance ease of use in an application or VI:*

- *Set the text focus to the main, most commonly used control.*
- *Disable or hide controls that are not currently relevant or valid.*
- *Guide the user through steps by highlighting controls.*
- *Change colors to bring attention to error conditions.*

*You can read or write multiple properties using a single node. However, some properties are not readable and some are not writable. A small direction arrow to the right of the property indicates a property you read. A small direction arrow to the left of the property indicates a property you write. You can right-click the property and select Change to Read or Change to Write from the shortcut menu to change the operation of the property.*

What is the name of this block diagram object? Hint: It appears this way when first dropped into the block diagram, before it is wired to a control reference.

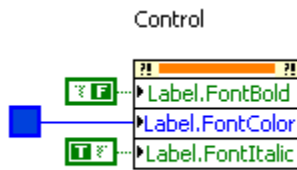


- a. Formula Node
- b. Script Node
- c. **Invoke Node**
- d. Property Node
- e. Constructor Node

Which of the following statements are true? Choose all that apply.

- 1) A class is part of an object- *the opposite is true: objects are members of a class*
- 2) **An object is a part of a class**
- 3) **Objects can have methods and properties**
- 4) **Properties are attributes of an object**
- 5) Methods are specific to objects – *methods can be overwritten to be specific to an object, but all objects within its class have the same methods.*
- 6) **Properties and methods are defined in a class**

This is a property node for a front panel control. What will this node do?



- a. Change the control label font to bold, italic and the color blue

*There is a false constant input to the bold terminal*

- b. Change the control label font to italic and the color blue**

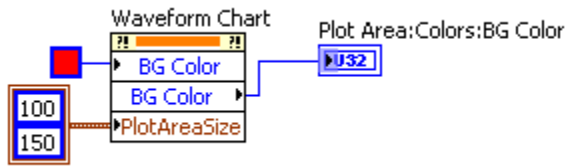
- c. Nothing, there is no wire at the top of the property node for the reference.

*There is no reference terminal on this node. The property node doesn't need a reference if it is an explicit property node created from the front panel object.*

- d. Change the text of the control to italic and the color blue

*The properties are for the label not the text.*

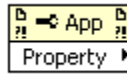
What does this property node do?



- a. **Changes the background to red, displays the background color as an integer, changes the width of the display to 100 and the height to 150.**
- b. Changes the background to red, displays the background color as an integer, changes the width of the display to 150 and the height to 100.
- c. Changes the background to red, displays the background color as a color box, changes the width of the display to 150 and the height to 100.
- d. Changes the background to red, displays the background color as a color box, changes the width of the display to 100 and the height to 150.

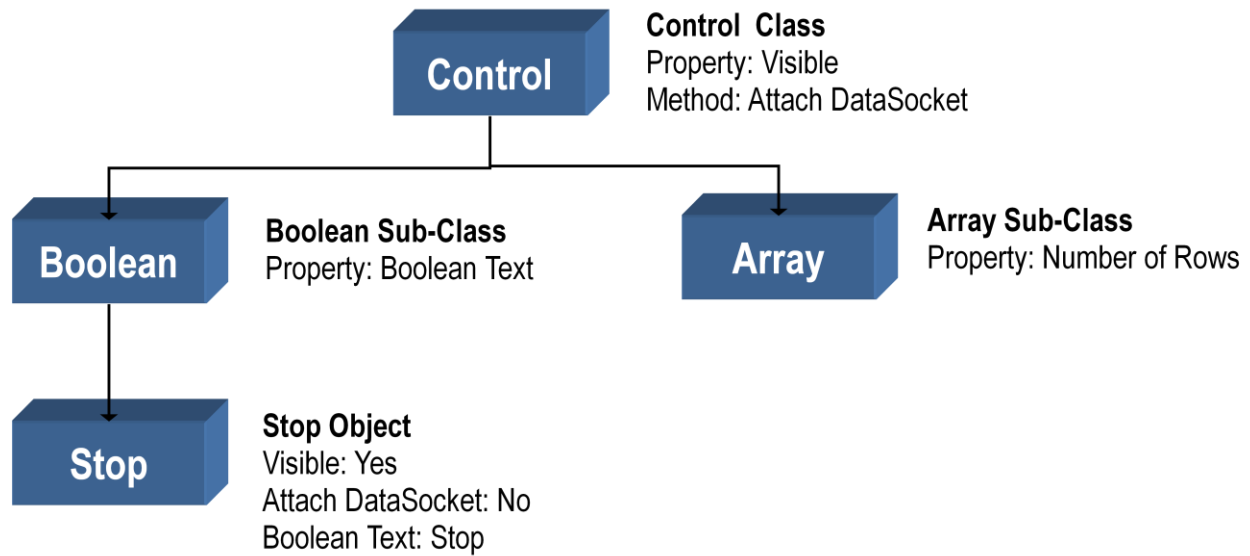


What is the name of this block diagram object? Hint: It appears this way when first dropped into the block diagram, before it is wired to a control reference.



*Property Node*

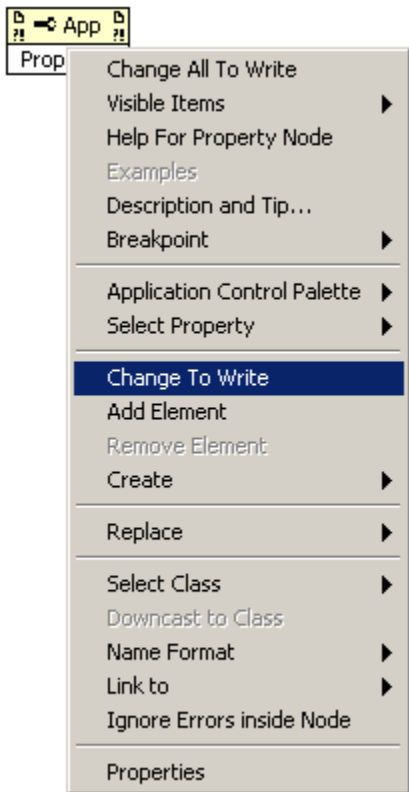
Please fill in whether a blue box contains a Class, Sub-Class, or Object



*Please reference page 5-2 of LabVIEW Basics II for a more detailed description of Classes, Sub-Classes, and Objects*

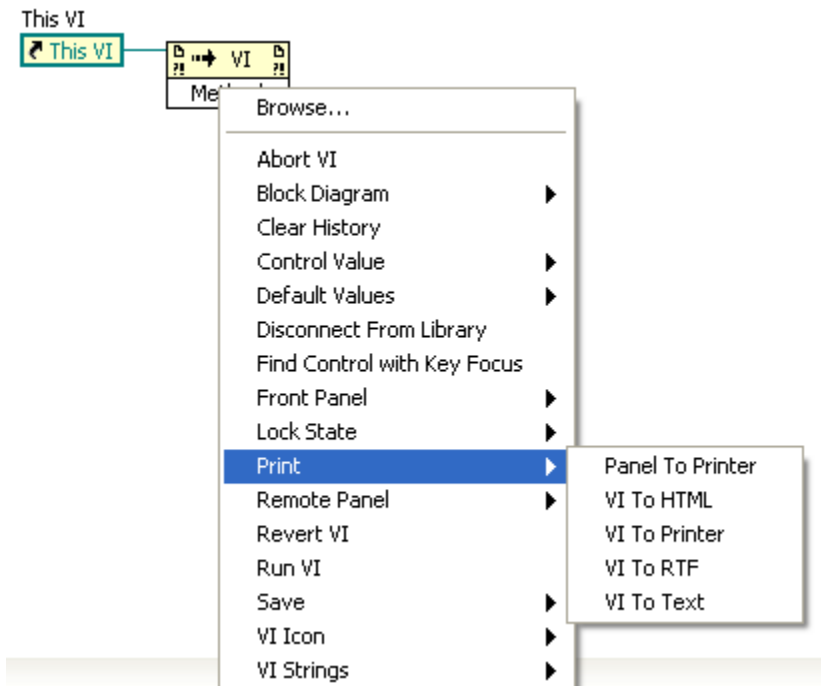
How can you change a property node from read to write?

*You can right-click on the element of a property node and select “Change to Write”.*



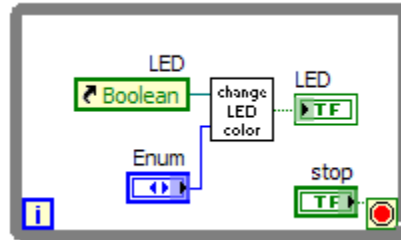
Which of these options are available when creating a method for a VI class?

- a. **VI to HTML**
- b. **VI to RTF**
- c. **VI to printer**
- d. **VI to text**
- e. Block diagram to printer
- f. **Panel to printer**

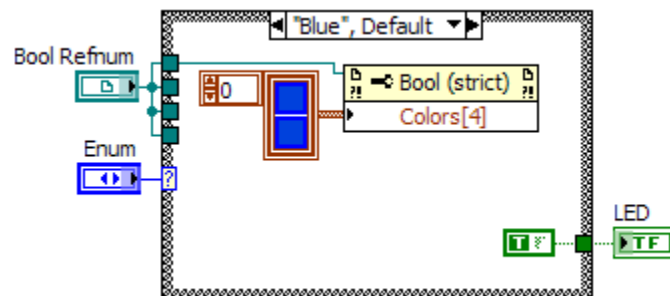


The following code changes the color of an LED based on an enumeration of four colors (Blue, Green, Yellow, Red):

*Main Code (Revised with solution)*



*Change LED Color SubVI Code (Revised with solution)*



When this code is run, the LED on the front panel fails to change color for every value of the enumeration. Explain why this error occurs and how to fix it.

*This error occurs because the Boolean control of the main code is not explicitly linked to the property node in the SubVI. In order to link the control to the property, a control reference for the LED must be created and wired to a generic property node within the SubVI (Control Palette » Programming » Application Control » Property Node).*

The TDMS Viewer VI is one way to view data within TDMS files.

TDMS stands for “Technical Data **Management** Streaming”

Why should one be aware of the endianness of a binary file?

*Student's answer should indicate that he/she understands that a value read assuming the incorrect endian format could lead to incorrect values being read.*



The **TDMS Read** VI reads TDMS files and outputs the group and channel names as well as the data from the TDMS file.

*The TDMS Viewer only displays the information on screen but is unable to pass data to other VIs.*

True or False: You can read and write from a TDMS file concurrently?

**True:** *Concurrent read/write access to TDMS files is a huge advantage over other file types.*

Which of the following are properties of datalog files? Choose all that apply.

- a. **Number of Records**
- b. Channel Grouping
- c. Set properties
- d. **The Record(s)**
- e. **Current read position**
- f. Data output

What are the accepted data types when writing to a binary file? Chose all that apply.

- a. **Double**
- b. **Integer**
- c. **Boolean**
- d. **Waveform**
- e. **Dynamic**

*All data types are accepted, but will be converted to their binary equivalent.*

True or False: TDMS is a data model created by National Instruments hence it is not possible to create third party programs to write and read TDMS files.

**False:** *The internal structure of the TDMS file format is publicly documented so it is possible to create third party programs to write and read TDMS files.*

True or False: The .tdms\_index can be regenerated from the .tdms file, but the converse cannot be done (can't recreate .tdms from .tdms\_index).

**True:** *The .tdms\_index file is created when using the .tdms file. The .tdms file is where the measurement data is stored. No measurement information is stored in the .tdms\_index file.*

Each character in an ASCII string takes up exactly \_\_\_\_\_ of memory. Choose amongst the following to fill in the blank.

- a. 4 bytes
- b. 1 bit
- c. 1 byte**
- d. 8 bytes
- e. 4 bits
- f. 1 nibble
- g. 1 megabyte

Which of the following file formats is the most compact and fastest for storing data?

- A) ASCII
- B) TMDS
- C) Binary**
- D) PDF

*Binary files are the most efficient because they use less disk space and because you do not have to convert data to and from a text representation when you retrieve or write data.*

*A binary file can represent 256 values in 1 byte of disk space.*



Compare and contrast when to use text and binary files.

*Students' answers should contain the following points*

1. *Use text files when:*
  - a. *Disk space and file I/O speed is not crucial*
  - b. *You do not need random access reads or writes*
  - c. *Numeric position is not important*
  - d. *Users need to be able to read the file*
  - e. *You want to access it from another application (e.g. Word, Excel)*
2. *Use binary files when*
  - a. *Save numeric data*
  - b. *You need to access specific numbers or randomly access numbers*
  - c. *Need the file to be small and fast*

What is best file type to use when collaborating with multiple people with different data analysis software (non-NI)?

- a. **Tab-delimited ASCII**
- b. Custom binary format
- c. TDMS
- d. Datalog

*Tab-delimited ASCII is the best format because of each of the other file types, more documentation is needed to allow others to correctly view the information stored in the file.*

True or False: All files written to your computer's hard drive are a series of binary bits.

**True:** *At their lowest level, all files written to your computer's hard drive are a series of binary bits. However, many formats for organizing and representing data are available.*

What does TDMS stand for?

Technical Data Management Streaming

Which of the following best describes the principles of datalog files?

- a. Datalog files are best used to store arrays of clusters because they can store data in either binary or ASCII format
  - b. Datalog files are best used to store arrays of image data because they provide efficient storage and random access
  - c. **Datalog files are best used to store arrays of clusters but if you lose the definition of the cluster, the files becomes difficult to decode**
  - d. Datalog files are best used to store arrays of clusters because they are easily accessible in every environment including LabVIEW
- 
- a. *Is not true because datalog files store binary format*
  - b. *Datalog files are the easiest method to log cluster data*
  - c. *This is true, the user must know the contents of the cluster type stored in the file*
  - d. *This is false, this storage format is complex and difficult to access in any environment other than LabVIEW*

*Fill in the blanks in the following paragraph:*

*When reading a binary file, there are two methods of accessing data; Sequential and **Random** access.*

*Using the **Sequential** access method, you read each item in order, starting at the beginning of a file.*

*Alternately, in the **Random** access method, you access data at an arbitrary point within the file.*

Match each data type to its binary representation in LabVIEW:

Boolean	<u>c</u>	a. 32-bit integer
String	<u>e</u>	b. Little Endian or Big Endian Bytes
Double Precision Numeric	<u>d</u>	c. 8-bit value
Array Header	<u>a</u>	d. 64-bit value
Multi-byte integer	<u>b</u>	e. A series of unsigned 8-bit integers

You are using LabVIEW to collect temperature data from multiple sensors. You are asked to run a 1 hour test everyday for one week. You decide you need to log the data in a file that can be later used for analysis. Which file format is best to use? Why? (Choose from text, binary, datalog, or TDMS)

*TDMS. TDMS is best when storing test or measurement data. Also, TDMS allows easy input of the type of test/measurements, which readings are associated with which sensor, and the time of the test.*

*Text file format is not the best choice because precision might be lost when storing numeric data and there is not a need to share the files, nor have another user read the files.*

*Binary files are precise, however they do not allow for easy documentation of time, sensor ID, or type of test/measurement performed.*

*Datalog files are a possibility since the files will be read by LabVIEW, however if the definition of cluster contents are every unknown it is very difficult to decode the cluster. In short, TDMS is the better choice.*



Compare and contrast text (ASCII) files and binary files.

<i>Text (ASCII)</i>	<i>Binary</i>
<ul style="list-style-type: none"><li>-Best to use when:<ul style="list-style-type: none"><li>- Need to make data available to other users if disk space and file I/O are not crucial</li><li>- Do not need to perform random access reads/writes</li><li>- If numeric precision is not important</li><li>- Want to access from another application(ie word processing or spreadsheet application)</li></ul></li><li>-Easiest to use and share</li><li>-Need to convert to string format</li><li>-Can contain different data types</li><li>-Take up more memory</li><li>-Difficult to randomly access numeric data</li></ul>	<ul style="list-style-type: none"><li>-Loss of precision not an issue</li><li>-Best to use when:<ul style="list-style-type: none"><li>- Save numeric data</li><li>- Need to access specific number from a file or randomly access number from a file</li></ul></li><li>-Machine readable only</li><li>-Most compact and fastest format for storing data</li><li>-Can contain multiple data types</li><li>-More efficient</li><li>-Can represent 256 values in 1 byte of disk space</li><li>-Often contain byte-for-byte image of the data as it was stored in data (except for cases like extended and complex numeric values), which makes reading from the file much faster because conversion is not necessary</li></ul>

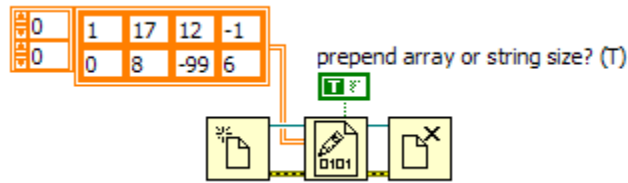
The TDMS file format contains two files, the .tdms\_index file and the .tdms file. Explain the purposes and properties of each file. Also include any information pertaining to the sharing and distribution of these files.

*The .tdms file is a binary file that contains the data and properties from the task/measurement. The .tdms\_index file is a binary index file that speeds up access while reading and provides consolidation information on all the attributes and pointers of the file. The .tdms\_file is created from information found in the .tdms file and as a result, can be re-generated if lost from the .tdms file. This is useful whenever moving, copying, or re-distributing the files to other locations. The TDMS file format is also publicly documented so the contents of the file can be easily accessed by other methods and programs, not just through LabVIEW.*

*The most important points that need to be made by the student are as follows:*

- *.tdms file is a binary file that contains the actual data from the experiment.*
- *.tdms\_index is made from the .tdms file and is only an indexing file that speeds up read access.*
- *.tdms\_index file is recreateable from the .tdms file.*
- *Openly/publicly documented so data access is easy.*

The following code writes a 2D array of double precision numbers to a binary file:



If the binary file is then read with a byte offset of 32, what is the first value that is read?

- 1
- 0
- 1
- 6
- 17
- 99
- 12
- 8
- None of the above, the byte offset is too large

*The byte offset is 32. Arrays are stored as a sequential list of their elements and since the array is full of double precision numbers, each value is stored as 8 bytes (64-bits). Additionally, each header is stored as 4 bytes (32-bits), so you must account for that data as well in the offset. After doing some arithmetic:*

Values	2	4	1	17	12	-1	0	8	-99	6
Bytes	4	4	8	8	8	8	8	8	8	8
Offset	0	4	8	16	24	32	40	48	56	64

*The first value that will be read after a byte offset of 32 will be -1.*

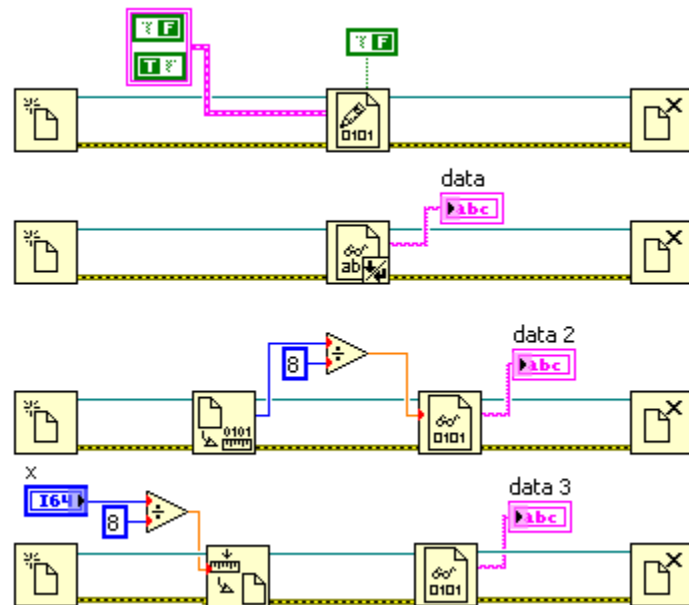
Please match the file types and uses to their respective code.

a. Read Binary File

b. Random Access

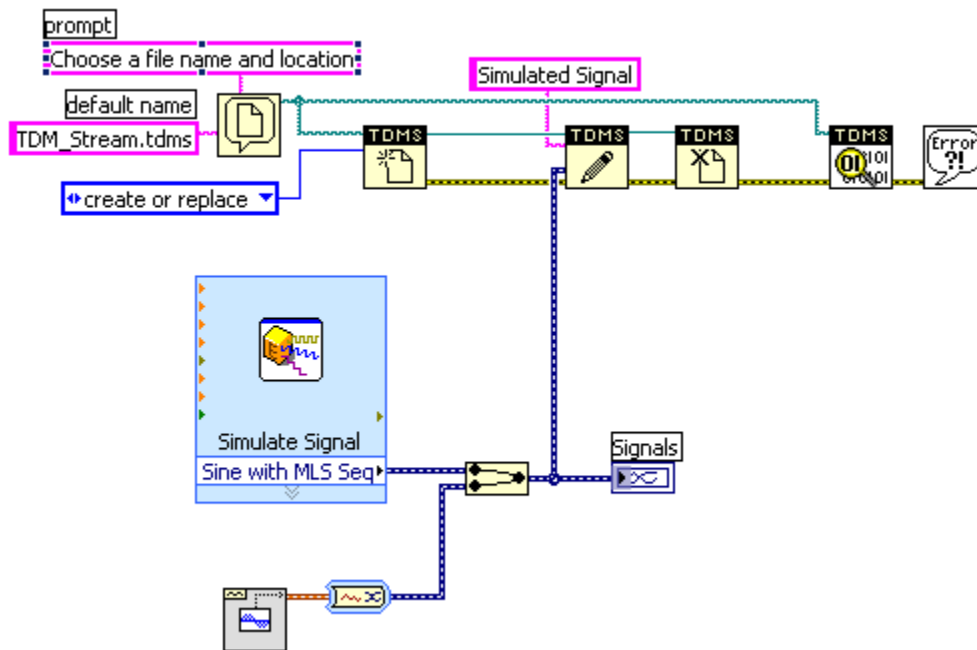
c. Read ASCII File

d. Sequential Read



Modify the block diagram below to add the output of the Square Waveform.vi block to the TDMS file as an additional channel.

*The data input terminal of the Write to TDMS block takes in dynamic data, so the simplest way would be to merge the two signals together. However, it is better to convert the waveform output of the Square Waveform.vi to dynamic data type before merging signals. Remember that TDMS can take in many data types (arrays, waveforms, timestamps, etc.)*



How do you force VI programmers to leave revision comments on a VI they are editing?

*Go to File->VI Properties and select Revision History from the drop down menu and make sure the "Prompt for a comment when the VI is closed" is checked.*

What are relative paths and why are they an important consideration when building applications for distribution?

*A relative path is one that does not start from the root directory and uses a base directory that is unlikely to change. Student's answer should indicate that relative paths are less likely to receive errors when addressing files that were made with their installer.*

Your application requires that you have a very simple standalone program to log data from a USB-6008 on a computer with no National Instruments products installed, what should be included in your installer?

- *Your built executable and its associated files generated during the build (lvanlys.dll, etc.)*

*You will need to add in the following additional installers*

- *The LabVIEW runtime engine for the version of LabVIEW you are building your application*
- *The DAQmx Driver*



What is the main step that needs to be taken before being able to building an executable?

*The user must create Build Specifications within the Project Explorer.*

Using constants for file names can be a bad idea when building executables. Why is this a bad idea?

*This file path will never be able to be changed in the executable. If the path changes or this application is moved to another computer, the file path could be non-existent.*

For large installers, it is possible to split an installer across multiple types of media such as CDs or DVDs. This option is located in the **Advanced** category of **My Installer Properties**, which is reached from the Project Explorer.

*Media spanning is a useful tool that allows the distribution of large applications. It is selected in the Advanced category of the My Installer Properties dialog window, which appears when constructing an installer from the Project Explorer.*

You can view the block diagram of an executable made from LabVIEW?

**False:** *You can only view the block diagram of a VI, not an executable.*

The VI Hierarchy is helpful for maintaining an organized view of your VI. Please select all attributes of the VI Hierarchy.

- a. **Accessed from View > VI Hierarchy**
- b. **Shows all connections between VIs and subVIs as well as the additional functions they use**
- c. Shows files that the VI has written to or read from      *VI Hierarchy does not show this*
- d. **Shows globals and type definitions**

What versions of LabVIEW can build executables?

- a. Base
- b. Full
- c. **Professional**
- d. **Developer Suite**

*The Base and Full version of LabVIEW do not have the application builder activated. You can purchase the application builder at <http://sine.ni.com/nips/cds/view/p/lang/en/nid/10731>.*

In a stand-alone application, the top-level VI is usually tasked with quitting LabVIEW when the program is finished running. What is the easiest method of doing this?

- a. Issuing a “Quit” command to the LabVIEW .dll from the top-level VI.  
*This does not exist.*
- b. Placing a “True” Boolean constant wired to a Close.LabVIEW property node on the top-level VI.  
*This does not exist.*
- c. **Call the “Quit LabVIEW” function on the block diagram of the top-level VI.**

You are planning on building an installer to use on multiple computers. These computers will not have LabVIEW installed on them. What must you include as an additional installer?

- a. The original VI
- b. Full version of LabVIEW
- c. LabVIEW Run-Time Engine**
- d. The built in sub VIs that come with LabVIEW



Which of the following choices describe the most compelling situation to create an Installer instead of simply compiling your VI to an executable file?

- a. **The need to distribute your application to a system that does not have LabVIEW installed.**
- b. The program contains many VIs.  
*A program with many VIs can be compiled into a single executable file.*
- c. The entire Block diagram of the main VI cannot be seen without scrolling its window.  
*This has no effect on the application builder aspect of LabVIEW.*
- d. The program uses one or more libraries.  
*Libraries can also be incorporated into a single executable.*

*Any computer that is expected to run an executable made in LabVIEW needs to have either a copy of LabVIEW or the LabVIEW Run-Time Engine installed. The latter can be downloaded and installed for free from the National Instruments website.*

*An installer is most commonly used to package the Run-Time Engine with the executable into a single source for all of the needed files to run your application.*

What is required for running a stand-alone LabVIEW application?

*The Run-Time Engine is required for running a stand-alone application written in LabVIEW. The Run-Time Engine is distributed by installers or downloaded from [ni.com](http://ni.com).*

When using a custom scale in your DAQ Assistant, does the Application Builder automatically include the scale used or do you have to manually add it to the builder?

*Whenever you add your source files, a .ini file will be automatically included into the Data subfolder. This file should share the same name as your custom scale and it will then be included in the application and installer.*

True or False: In order to run an application, LabVIEW must be installed.

**False:** *Only the LabVIEW Run Time Engine is required to run an application written in LabVIEW. The Run Time Engine should be included with the installer or downloaded from ni.com.*

You created a stand-alone LabVIEW application. The application runs without warnings and functions as it should. However, the top level front panel remains open even after the application has finished processing. How can you fix this situation?

*To completely quit and close the top-level VI, you must call the Quit LabVIEW function on the block diagram of the top level VI.*

Which of the following are true about the VI Revision History?

- a. You can view the development history of the VI by selecting Tools » VI Revision History

*The correct menu is Edit » VI Revision History*

- b. Revision numbers increment every time you save the VI**

- c. You can set Revision History options for specific VIs**

- d. Revision numbers for a VI always start at 1

*Revision numbers always start at 0*