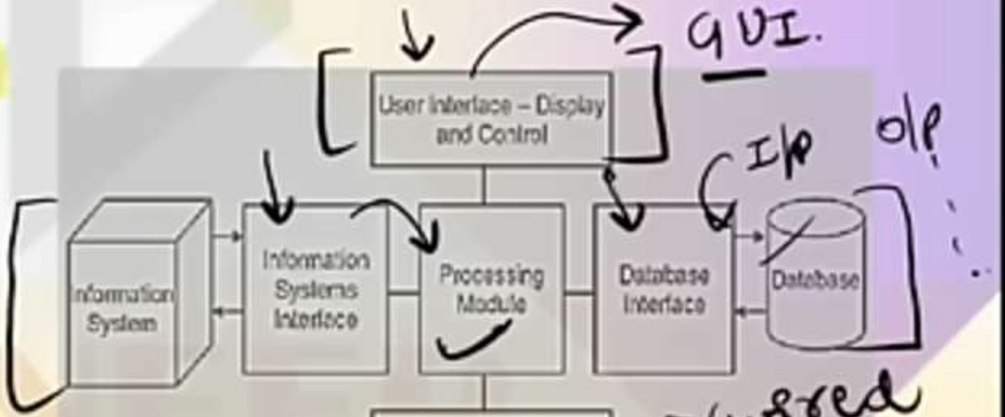
Virtual Instrumentation

A virtual instrumentation system is software that is used by the user to develop a computerized test and measurement system, for controlling an external measurement hardware device from a desktop computer, and for displaying test or measurement data on panels in the computer screen.

Architecture of Virtual Instrumentation

- Sensor module
- Sensor interface
- Processing module
- Database interface
- User interface



ENGINEER BEING

Sensor module

The sensor module detects physical signal and transforms it into electrical form, conditions the signal, and transforms it into a digital form

Sensor interface

Through a sensor interface, the sensor module communicates with a computer. According to the type of connection, sensor interfaces can be classified as wired and wireless.

Processing module

Once the data are in a digital form on a computer, they can be processed, mixed, compared, and otherwise manipulated, or stored in a database.

Management Systems (DBMSs)

They provide efficient management of data and standardized insertion, update, deletion, and selection.

Advantages of VI

- Flexibility
- Simplification
- Cost-effectiveness
- Scalability
- User-friendliness
- Portability
- Data analysis

ENGINEER BEING

Ameter

Comparison of Traditional instruments and virtual instruments

VI (Important)

Traditional instruments	Virtual instruments
Vendor Defined. Function specific, stand alone with limited connectivity.	User-defined. Application Oriented sysem with connectivity to networks, peripherals, & applications
Hardware is the key. Expensive, Closed, fixed functionality	Software is the key. Low cost, reusable. Open flexible funcitonality leveraging off familiar computer technology .
Slow turn on technology. Minimum economic scale .	fast turn on technology. Maximum economic scale.
High development and maintenance cost.	Software minimizes development & maintenance costts.

Graphical Programming

imp

VIs

LabVIEW

{Arrays vs clusters}

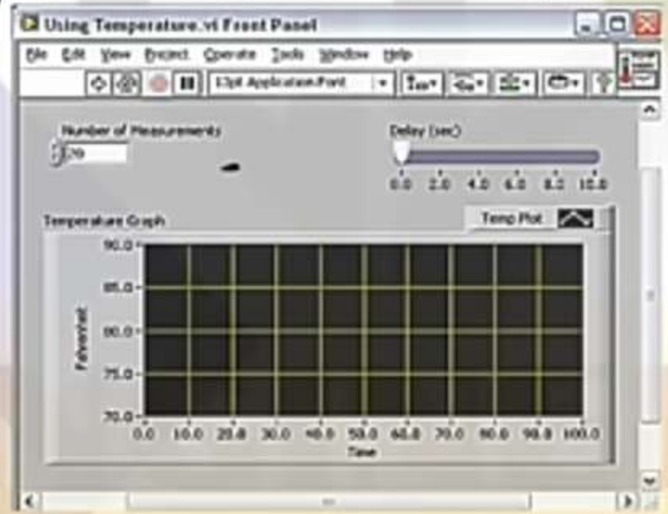
LabVIEW is a graphical programming language that uses icons instead of lines of text to create applications.

i/p ← control & indicator → o/p

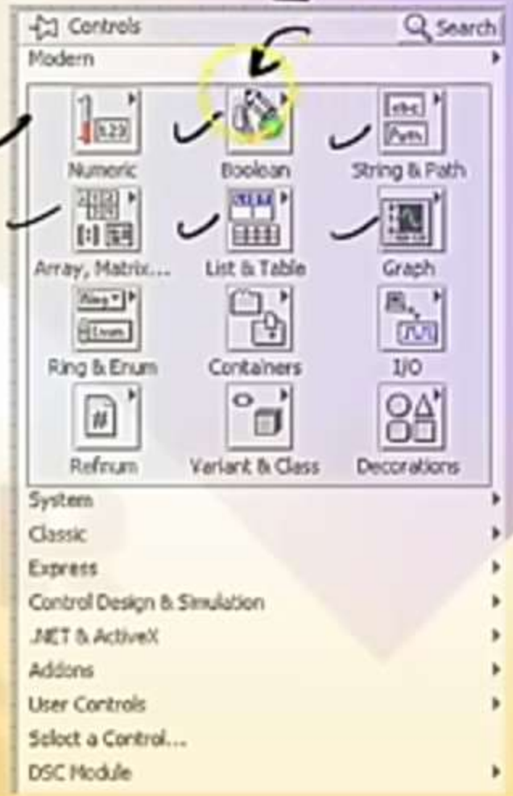
Each VI contains three main parts: {GUI}

- Front Panel
- Block Diagram
- Icon/Connector

(Laboratory Virtual Instrument Engineering Workbench)



{C, C++ } ≡ text



ENGINEER BEING

Boolean → T, Yes
F, NO

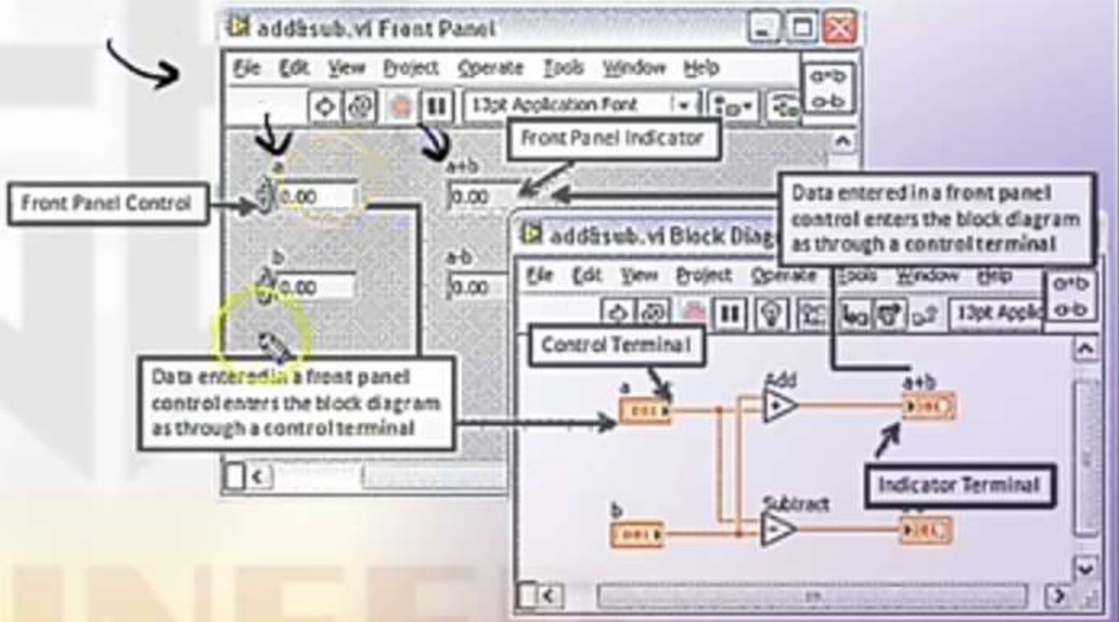
Vertical Toggle Switch

Round LED

I/p

O/p

Block Diagram.



ENGINEER BEING

Graphical Programming

LabVIEW is a graphical programming language that uses icons instead of lines of text to create applications.

LabVIEW programs are called virtual instruments (VIs).
Controls are inputs and indicators are outputs.
Each VI contains three main parts:

- **Front Panel** – The front panel is the window through which the user interacts with the program most powerful features that Lab VIEW offers engineers and scientists is its graphical programming environment to design custom virtual instruments by creating a graphical user interface on the computer screen.

- **Block Diagram** – The code that controls the program. The block diagram window holds the graphical source code of a Lab View's block diagram corresponds to the lines of text found in a more conventional language like C or BASIC – it is the actual executable code.

- **Icon/Connector** – An icon is a graphical representation of a VI. The icon can contain both text and images.

Advantages of Lab VIEW

- Graphical user interface:
- Drag-and-drop built-in functions
- Modular design and hierarchical design
- Multiple high level development tools
- Professional Development Tools

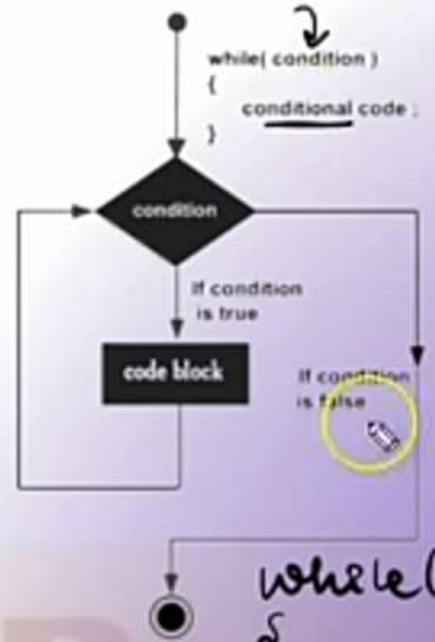
VI

While Loops in LabVIEW

A While Loop is a structure you use to execute a block of LabVIEW code repeatedly until a given condition is met.

When the VI runs, the code inside the While Loop executes, and then the terminal condition is evaluated.

The While Loop will be a familiar concept for experienced programmers as it operates similarly in other computer languages.



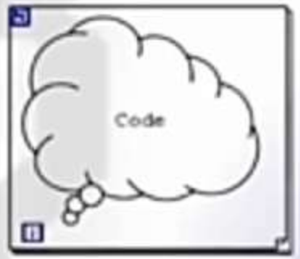
LabVIEW While Loop flowchart

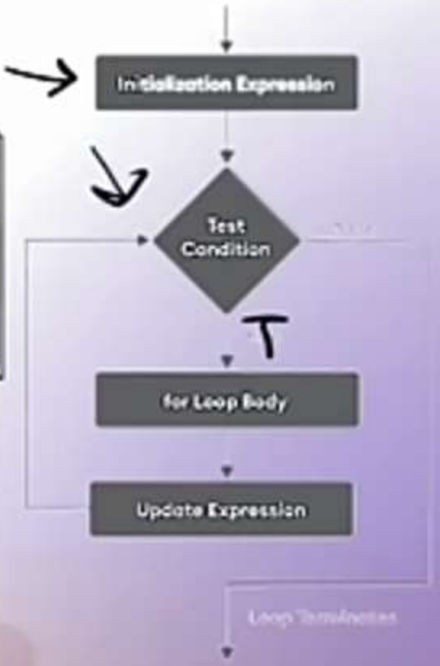
For Loops in LabVIEW

A For Loop is a structure you use to execute a block of code a set number of times. When the VI runs, the iteration count is evaluated, and then the code is executed.

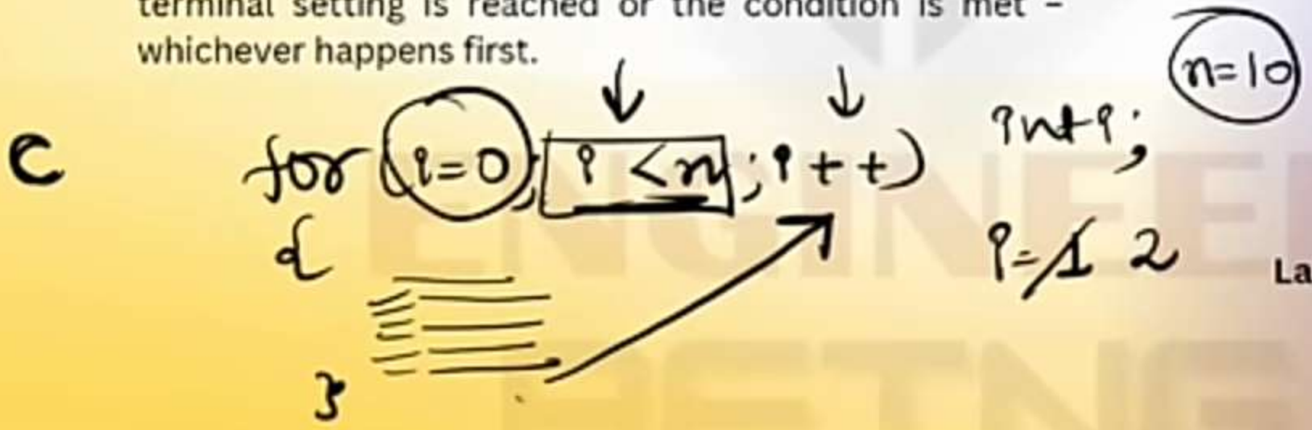
A For Loop can be configured to conditionally stop code execution in addition to its iteration-based exit.

In these cases, the code will execute until the count terminal setting is reached or the condition is met – whichever happens first.





LabVIEW for Loop flowchart



Arrays in Lab view

ENGINEER BEING

In LabVIEW, an array is a data structure that allows you to store and manipulate a collection of values of the same data type. Here are the basic steps to create and manipulate an array in LabVIEW:

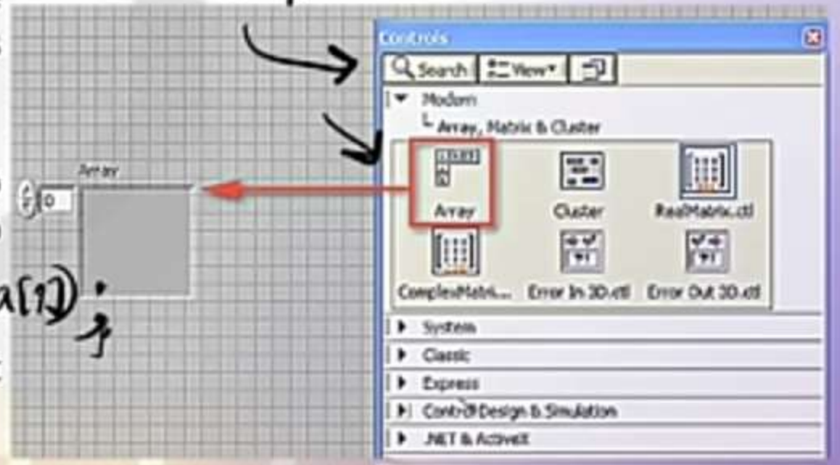
Create an Array: You can create an array in LabVIEW by right-clicking on the block diagram and selecting "Array" from the "Add" menu. Then, select the data type for the array elements and specify the array size

Populate the Array: You can use a "For Loop" structure to iterate through each element of the array and assign a value to it.

Access Array Elements: The index starts from 0 for the first element and increases by 1 for each subsequent element.

Manipulate the Array: You can perform various operations on an array in LabVIEW, such as sorting, filtering, and searching.

Display the Array: You can display the contents of an array in LabVIEW using various visualization tools, such as a waveform graph.



ENGINEER BEING

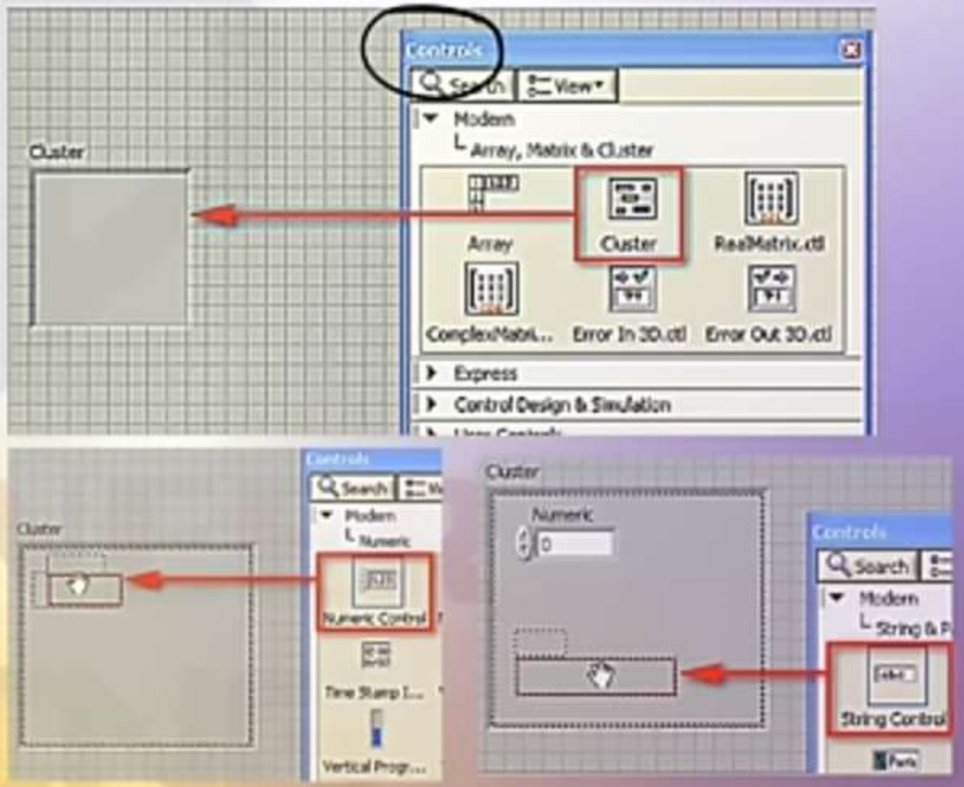
Clusters in labview

a[]={'a', '1', "Engbeing"}

In LabVIEW, a cluster is a data structure that groups together multiple data elements of different data types into a single unit. Clusters are useful for organizing and passing around related data, as well as for grouping together inputs and outputs of a VI. Here are the basic steps to create and manipulate a cluster in LabVIEW:

Create a Cluster: You can create a cluster in LabVIEW by right-clicking on the block diagram and selecting "Cluster" from the "Add" menu. Then, you can add data elements to the cluster by dragging and dropping controls or constants into the cluster.

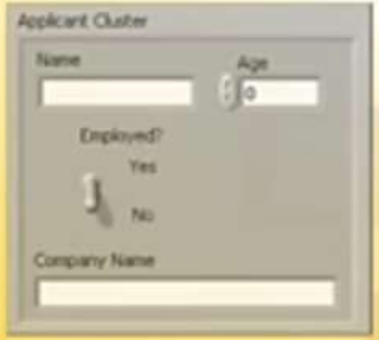
Assign Values to Cluster Elements: Once you have created the cluster, you can assign values to its elements. You can use the cluster's name followed by the element's name to access and assign a value to that element. For example, if the cluster is named "myCluster" and has an element named "myInteger", you can assign a value to it using the expression "myCluster.myInteger = 42".

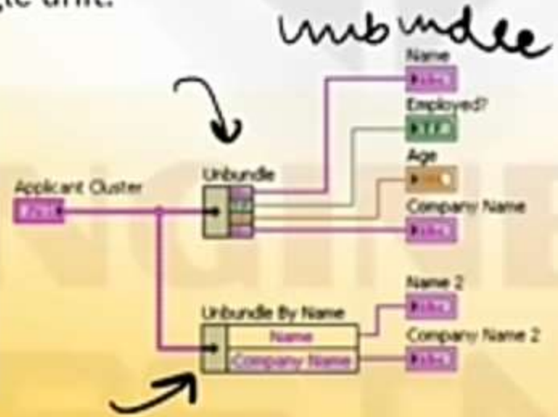


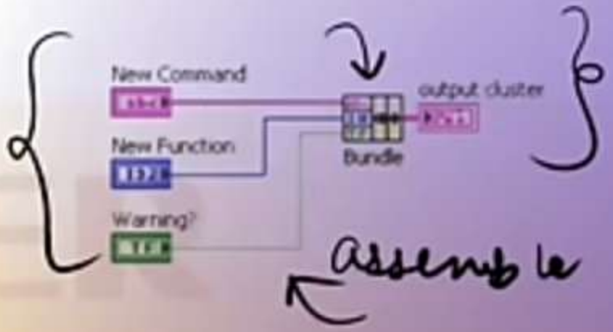
Access Cluster Elements: You can access the elements of a cluster in LabVIEW using the same syntax as when assigning values. Function in clusters are the Bundle, Unbundle, Bundle By Name, and Unbundle By Name functions.

Display the Cluster: You can display the contents of a cluster in LabVIEW using various visualization tools, such as a numeric indicator or a string indicator.

Overall, clusters are a powerful data structure in LabVIEW that allow you to group together multiple data elements of different data types into a single unit.







ENGINEER BEING

Graphs & Charts in Lab view

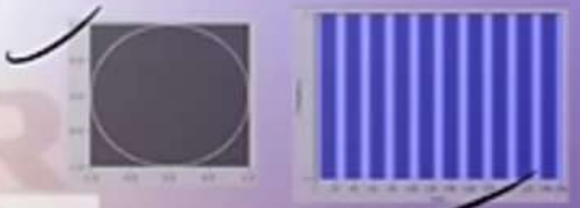
In LabVIEW, graphs are used to display data in a visual format that is easy to understand. There are several types of graphs available in LabVIEW, including waveform graphs, XY graphs, and charts.

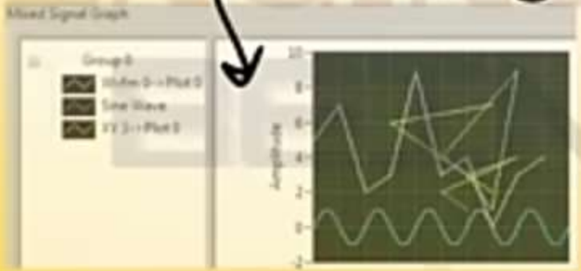
steps to create and manipulate a graph in LabVIEW:

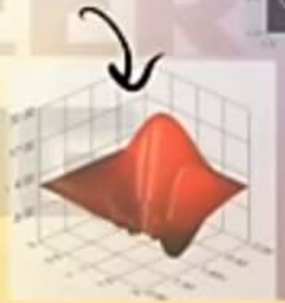
- Create a Graph
- Configure the Graph
- Update the Graph
- Manipulate the Graph
- Save and Export the Graph

Types of graphs and charts:

- Waveform Graphs and Charts
- XY Graphs
- Intensity Graphs and Charts
- Digital Waveform Graphs
- Mixed Signal Graphs
  - 2D Graphs
  - 3D Graphs





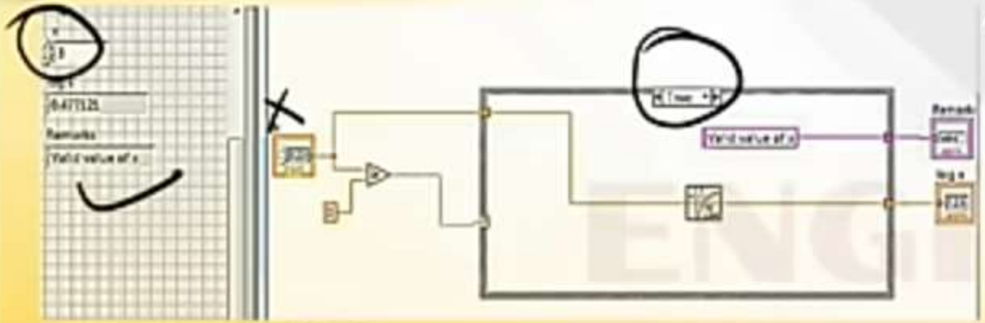


Structures: case, sequence, and formula nodes

In LabVIEW, structures are used to control the flow of data and program execution within a VI. Three common structures in LabVIEW are the Case structure, Sequence structure, and Formula node. Here's an overview of each structure:

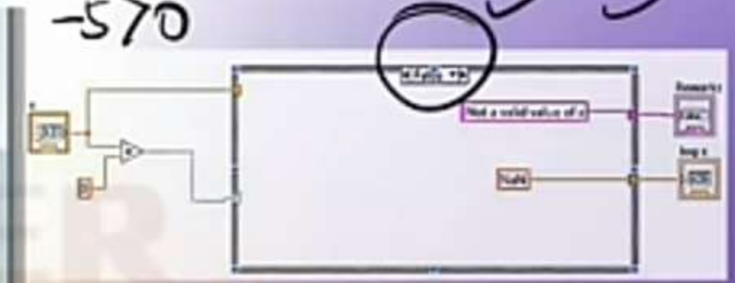
Case structure: Case structures are widely used in the scenarios where the program or the users have to take a decision. The decision is categorized into two options, i.e. True or False.
At any point of execution, only one condition (or case, i.e. True or False) will be executed.

{T|F}



270

-570



(if-else) age=

if (age > 18)
vote
else
not allow to vote

Sequence structure: A Sequence structure is used to execute a set of instructions in a specific order. The structure contains multiple frames, each representing a step in the sequence. The frames are executed in the order in which they appear in the structure.

Within Sequence structure, we have two different subtypes available, and they are:

- **Flat sequence structure**

In a flat sequence structure, all the frames are available in the block diagram. If there are a lot of frames, then the structure may take a considerable amount of space.

- **Stacked sequence structure**

In a stacked sequence structure, all the frames are in sequential order, but they are stacked on top of each other like a case structure.

Formula node: The Formula Node is a programming construct in LabVIEW that allows you to write mathematical expressions using a syntax similar to traditional programming languages. The Formula Node is a text-based programming construct that can be used to perform complex mathematical calculations, data analysis, or signal processing.

ENGINEER BEING

Need of software based instruments for Industrial Automation

Software-based instruments are becoming increasingly popular in industrial automation for several reasons:

1. **Flexibility:** Software-based instruments can be easily programmed and reprogrammed to perform different functions, making them highly flexible and adaptable to changing production needs. This allows companies to quickly modify their production lines and adapt to new market demands.
2. **Programmable Logic Controller (PLC):** A PLC is a ruggedized, digital industrial computer control system that is pre-programmed to carry out automatic operations in industrial processes. The PLC continuously monitors and receives information from input devices or sensors, processes the information, and triggers the connected output devices, to complete the task in the industrial process or machinery.
3. **Supervisory Control and Data Acquisition (SCADA):** SCADA systems control and monitor industrial processes. The system acquires and processes real-time data through direct interaction with devices, such as sensors and PLCs, and records events into a log file.
4. **Data analysis:** Software-based instruments can provide real-time data analysis and visualization, allowing companies to identify trends, monitor performance, and make informed decisions that can improve productivity and reduce costs.

5. Remote monitoring and control: Software-based instruments can be accessed and controlled remotely, allowing companies to monitor and control their production lines from anywhere in the world. This can be especially useful for companies with multiple production facilities or for companies that need to perform maintenance on their equipment from a remote location.

Overall, software-based instruments offer a range of benefits for industrial automation, including flexibility, cost-effectiveness, integration, data analysis, and remote monitoring and control. As a result, they are becoming an increasingly important component of modern industrial automation systems.