Table of Contents

Lab 04
Introduction to LabVIEW

4.1 Objective

To introduce LabVIEW- a system engineering software

4.2 Introduction to LabVIEW

LabVIEW is a system engineering software that allows users to quickly build dynamic systems, deploy their test on existing systems, measure the responses, and design control applications. The quick development time can be attributed to their extensive library of device interfaces (low-level code) to some extent. It can also be attributed to their graphical programming language (G), which will be the focus of this lab. LabVIEW has been chosen as the preferred environment for this course for two particular reasons:

1. It allows graphical programming which mirrors block diagrams and is utilized to design and analyze control systems.
2. It incorporates hardware-in-the-loop in a seamless manner (we hope to cover this part if time permits).

This handout is not intended to be an exhaustive tutorial on LabVIEW. It simply aims to introduce LabVIEW in such a way that the students can later utilize it for control systems lab and at the same time encourage more investigation on the student's part into LabVIEW and controls. You can learn more by watching the video: *https://youtu.be/yUHvD_ajpn0*

4.3 G programming language

G has a flow-chart like data flow model. This is similar to the block diagrams used in analyzing control systems, where a system is represented by a series of subcomponents connected together by different types of interconnections. The G programming language allows us to do the same. Lines indicate data-type. In principle, G is capable of the same things as any other programming language such as C or C++. Consider a simple programming problem, where given the base and height of a triangle, we wish to compute its area. In C++, the following program could implement this.

```
float    base ;
float    height ;
float   area ;
cin >> base;
cin>>height;
area = 0.5* base*height ;
cout<<area ;
```

A possible implementation of the same in LabVIEW can look like Figure 1.

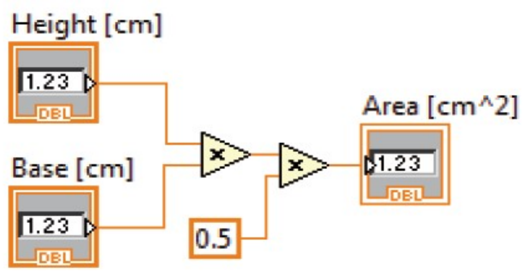


Figure 1: LabVIEW implementation for computing area of triangle

4.4 LabVIEW Environment

A LabVIEW program is called a *virtual instrument* (*VI*). Each VI, when opened up, will have two windows - a UI window or the **front-end**, and a **block diagram** window where the graphical code goes. Right clicking anywhere on the UI window or the block diagram will display a **Controls palette** or **Functions Palette** respectively. You create a front panel for your VI by using **Controls** and **Indicators** from the Controls palette. Controls are used to get input, while indicators provide output to the user. Any control or indicator you place on front panel will generate a corresponding **terminal** in the block-diagram window. In Figure 1, base and height are controls and area is an indicator.

Table 1: LabVIEW Blocks

LABVIEW Blocks	**Where to find?**	**Purpose**
Random Number	Functions >> Mathematics >> Numeric >> Random number	Generates random number ranging from 0 to 1
Convolution	Functions>> Signal Processing >> Signal Operation >> Convolution	This block convolves two polynomials.
Polynomial Roots	Functions>>Mathematics>> Polynomial>> Polynomial roots	This block calculates roots of the polynomial
Polynomial Evaluation	Functions >> Mathematics >> Polynomial >> Polynomial Evaluation	This block evaluate polynomial at specified value
Polynomial with Matrix	Functions>>Mathematics >> Polynomial >>Evaluate Polynomial with Matrix	This block evaluate polynomial at values specified in matrix
MATLAB script MATLAB Script	Functions >> Mathematics>> Script and formulas >> Script nodes>> MATLAB script	MATLAB codes can be run on LabVIEW using MATLAB Script

Task 1: To calculate the roots of a polynomial

Table 1 lists few blocks in LabVIEW related to mathematical functions. Follow the given paths and find these blocks in LabVIEW.

Let F(x) be a polynomial defined by the given equation: $F(x) = 6x^2 - 5x + 1$. Using calculator, find the roots of this polynomial.

x1	0.5
x2	0.33

A VI can be developed in LabVIEW to calculate roots of the polynomial.

1. Open LabVIEW. Open a Blank VI.
2. Right-click anywhere on the block panel.
3. Go to *Functions >> Mathematics >> Polynomial >> Polynomial roots.* Add this block to your VI. When you hover over the input or output ports, it displays the name of port.
4. Right-click on the input lead of root block, select *create control*.
5. Similarly, add indicator by right clicking on output lead.
6. Connect input of *Roots* block to *Numeric Control* and output to *Numeric Indicator*.
7. Press CTRL+E to switch between panels. In Front panel stretch control and indicator, to create array as shown in Figure 2.
8. Enter values to control. Here, the order of coefficients is in ascending order of the variable's power.
9. Click the white arrow at the top to run the VI.

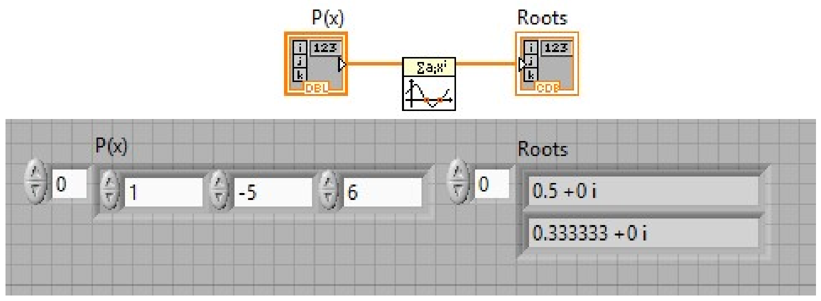


Figure 2: Block panel and front panel view for Task 1

Note: In LabVIEW, every object and wire have a data type associated with it. If you right-click on the control, you can set its data type from the representation menu. In the block diagram window, you will notice that each data type is represented by a different color. This is helpful as you will not be able to connect different colored blocks to each other in general, without a conversion block in between.

Now modify the VI to complete the following.

a. Calculate the roots of $F(x) = x^4 - 11x^3 + 38x^2 - 52x + 24$

(1) 6 (2) 1 (3) 2+2.8463i (4) 2-2.8463i

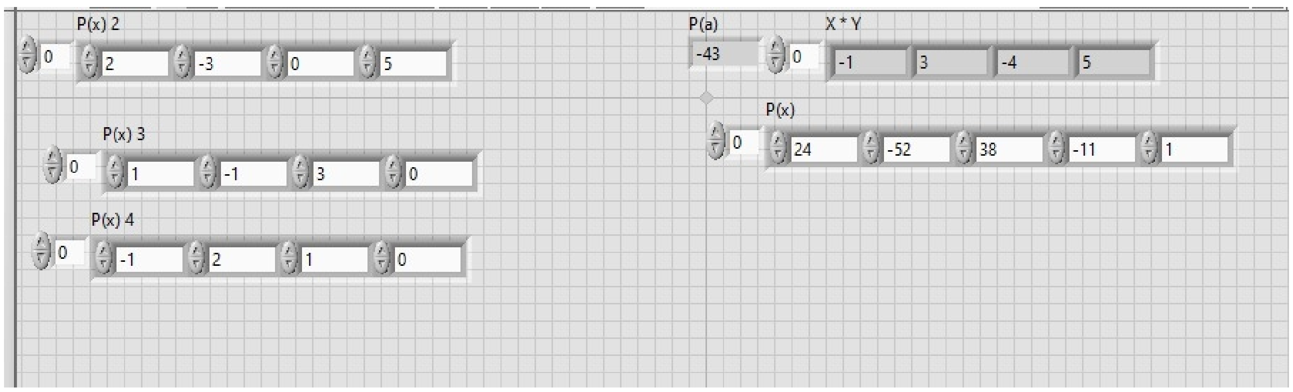
b. Calculate $P(-2)$ and $P(-3)$, for polynomial $P(x) = 5x^3 - 3x + 2$. For this, add Polynomial Evaluation block:

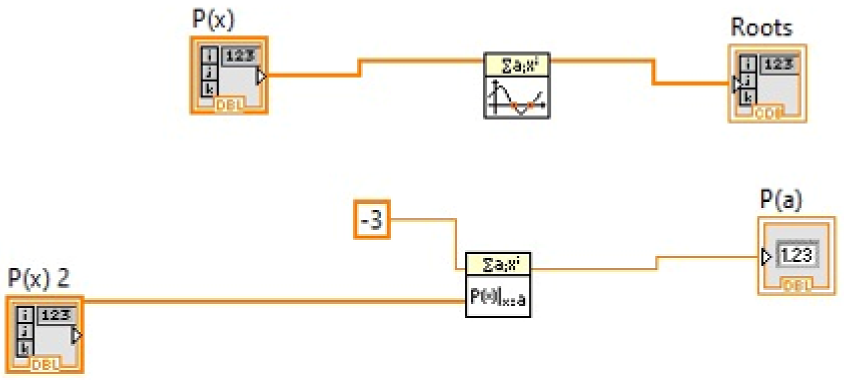
$P(-2)$	-32

$P(-3)$	-124

c. Multiply the given polynomials $P(x) = 3x^2 - x + 1$, $Q(x) = x^2 + 2x - 1$. Use Convolution block.

Product of P(x) and Q(x)	$3x^4+5x^3-4x^2+3x-1$





Attach capture of all the in-lab tasks

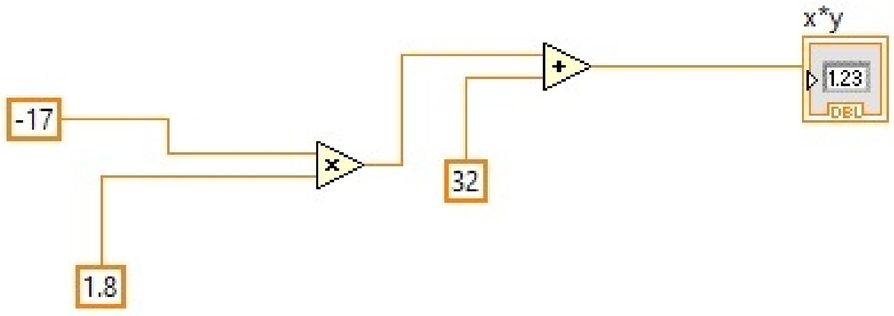
Task 2: To convert temperature from Celsius to Fahrenheit

Write a LabVIEW program, which accepts temperature in Celsius from the user and outputs the equivalent temperature in Fahrenheit. For more information, visit: *https://youtu.be/PqxStfwjQoQ*

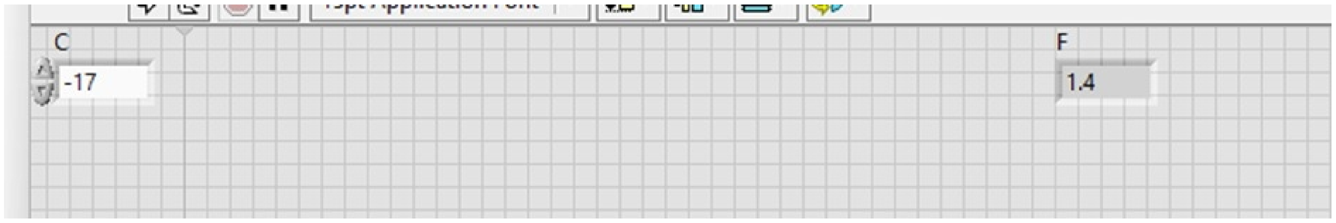
$$F = \frac{212-32}{100}C + 32$$

Run the VI and find equivalent temperature in Fahrenheit for the temperature given in Celsius.

Temperature in Celsius	Temperature in Fahrenheit
0 ℃	32℉
100 ℃	212℉
-17 ℃	1.4℉



Above block diagram is implemented which take temperature in Celsius and give output in Fahrenheit as can be seen in the pic below



As a result of this exercise, you should be able to answer the following questions:

a) What are a few examples of controls and indicators available in LabVIEW?

Some examples of controls are as follow: Numeric control, Boolean control, String control, Knob control etc.

Some examples of Indicators are as follow: Graph indicator, Chart indicator, String indicator, Tank indicator etc.

b) How do you create a constant in LabVIEW?

Constants are used to specify fixed value which do not change during execution. The steps for creating a constant are as follow: Open or Create VI > Select the block diagram > Choose a Data type > Place the constant on the Block Diagram > set the value > wire the constant with whatever the with appropriate component as per the requirement.

c) What are terminals in LabVIEW? What do tiny arrows on controls and indicator terminals represent?

Terminals are connection points or pins associated with the controls, indicator and various functions or objects on the block diagram. Terminals serve as the points where data flows into and out of these elements, allowing you to connect them together to create a functional data flow within your VI.

Whereas tiny arrows represent the direction of data flow for these elements. These arrows indicate whether terminal is an input or output.

d) What is a node in LabVIEW?

Node is typically a specific function or operation that you place on block diagram. It is any programming element in G on block diagram with the exception of control terminal, wires, free labels (bookmarks and other decorations).

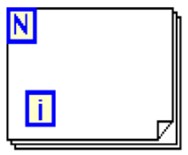
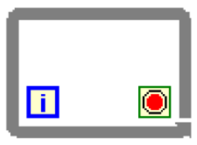
e) How do you search for a control, VI, or function?

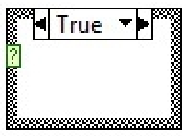
By following the steps given below you can find a control, virtual instruments or function:

Click the search box at the top left corner > enter what you want to find > select the item

4.5 Loop Structures

Structures are graphical representations of the loops. Loops use structures on the block diagram to repeat code and to execute code conditionally or in a specific order. Like other nodes, structures have terminals that connect them to other block diagram nodes, execute automatically when input data are available, and supply data to output wires when execution completes. You will find loops in ***Functions>>Programming>>Structures***. For Loop, While Loop and Case Structures are available in LabVIEW as listed below.

For Loop executes a sub-diagram a set number of times specified by a constant or control at its count terminal **(N).**	
The **WHILE LOOP** executes a graphical code while a condition is true or until a condition is met. A LabVIEW while loop always executes at least once. *Stop If True*: When a conditional terminal is in the Stop If True state, the While Loop executes its Block diagram until the conditional terminal receives a TRUE value from the front panel. The VI checks the conditional terminal at the end of each iteration. *Continue If True*: When a conditional terminal is in the Continue If True state, the While Loop executes its sub-diagram until the Conditional Terminal receives false value	

A **CASE STRUCTURE** has two or more sub-diagrams, or cases. Only one sub-diagram is visible at a time, and the structure executes only one case at a time. An input value determines which sub-diagram executes. You must wire an integer, Boolean value, string, or enumerated type value to the selector terminal. You can specify a default case for the Case structure to handle out-of-range values or explicitly list every possible input value.	

For more information, visit: *https://youtu.be/hnx9WI2D9zU*

Task 3: To implement While loop

Follow the given procedure and modify Task 2 to calculate temperature in Fahrenheit scale, for temperature in Celsius scale generated by ***random number function***. It lasts when user wants to abort.

1. Place a while loop from the function palette on the block diagram.
2. Right click on the conditional terminal and Select Create>>Control. This provides a True or False Control for the conditional terminal.
3. Place the blocks as shown in the Figure 3. To control the rate at which loop executes, ***Wait (ms)*** or the ***Wait Until Next ms Multiple*** functions can be used.
4. Connect them with wire and run the VI.

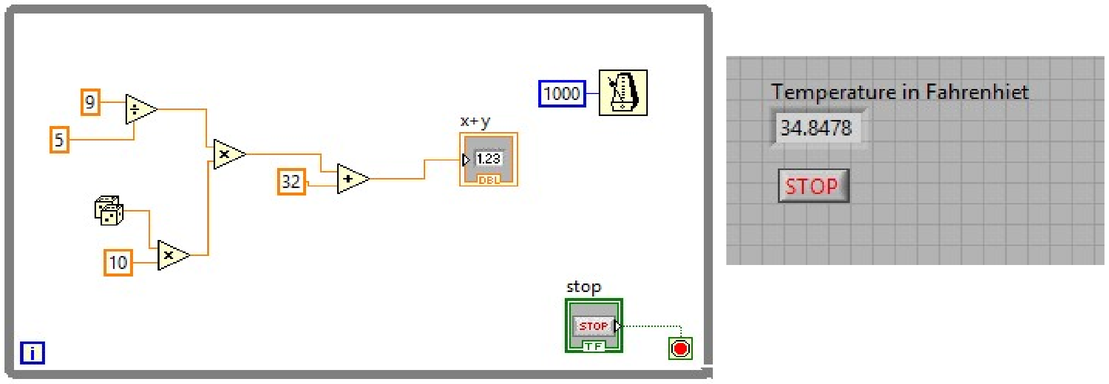
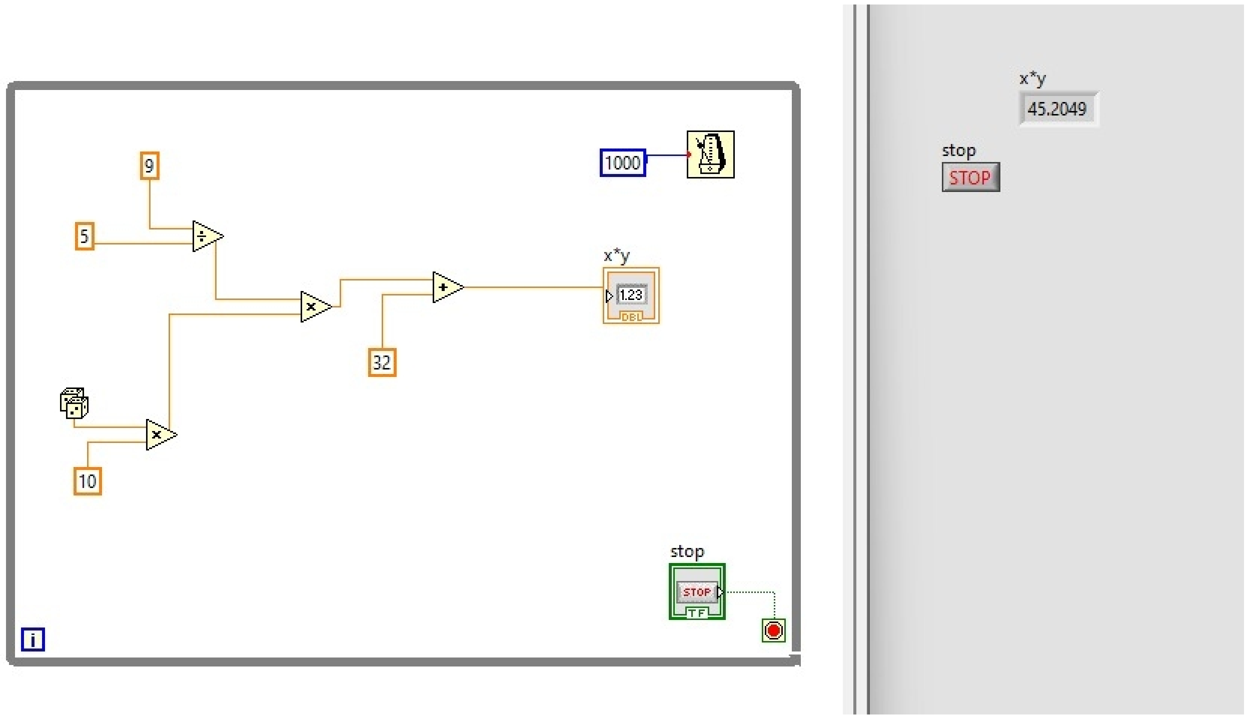


Figure 3: Sample VI for Task 3

Show implemented VI and its output:
(Following pic shows the implementation of this task)



X*Y is representing the temperature in Fahrenheit.

Answer the given questions:

a) How do you know the number of iterations of a loop?

> The symbol ‘i’ in the bottom left corner is the iteration number, To know iterations of the loop, just right click on the ‘i’ box and select create >> indicator that’s how you can see number of iterations.

b) How do you specify a stopping condition?

> For ‘while’ loop You can specify a stopping condition by creating a Boolean expression or condition that, when become true the loop stops
>
> For ‘for’ loop, the setting ‘n’ will be stopping condition when n iterations will be done, loop will stop as n is predefined

c) How can we add delay to a while loop so that the loop is evaluated once every 20ms, and save processor resources?

> You can do this by using the “wait (ms)” function. This function will introduce a delay in ms between loop iterations. You can find this function by right-clicking on block diagram selecting “timing”> “wait(ms)”. Double click on the wait (ms) function and write 20 ms in the given space

Task 4: To implement case structure

In this task, you will build a VI to perform addition, subtraction, multiplication and division on two numbers taken as input from the users. Only one of the functions can be performed at a time according to user's wish.

1. Place a case structure from the function palette on the block diagram.
2. Right-click on the arrow of case structure, in the menu select, "Add case after" and type the following in each case: " + ", "-" , " * " , "/ "
3. Go to front panel, place two numeric control, numeric indicator and a string control
4. (Control>>Modern>>String and path>>String control)
5. In the "+ "case, place "add" block. Connect both numeric controls to its input, and its output to numeric indicator, as shown in figure 4 below.

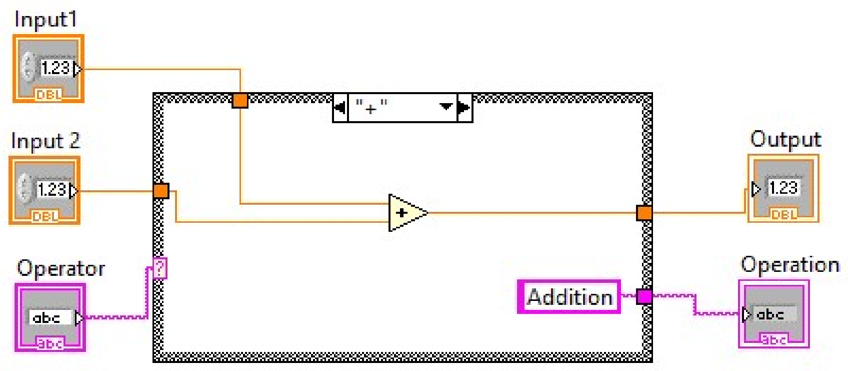


Figure 4: Sample case-structure

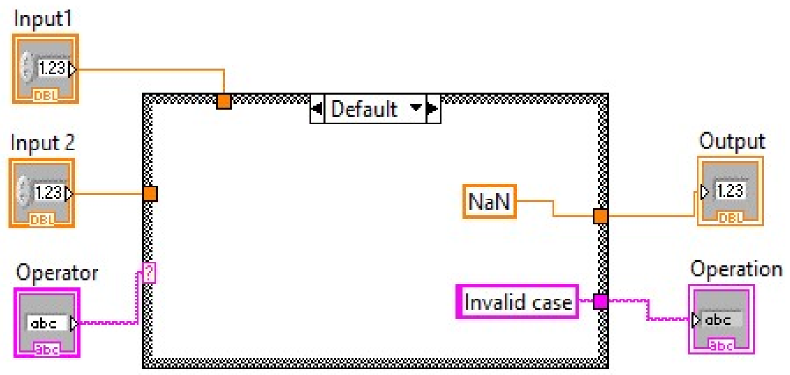
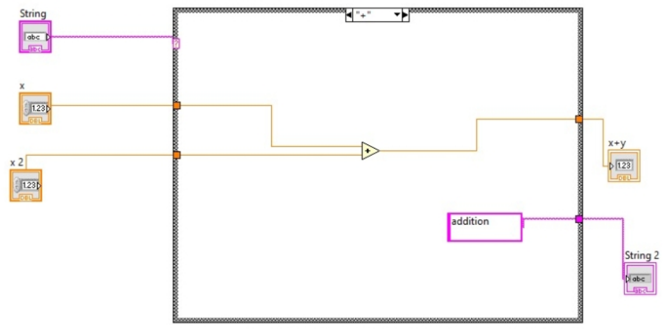
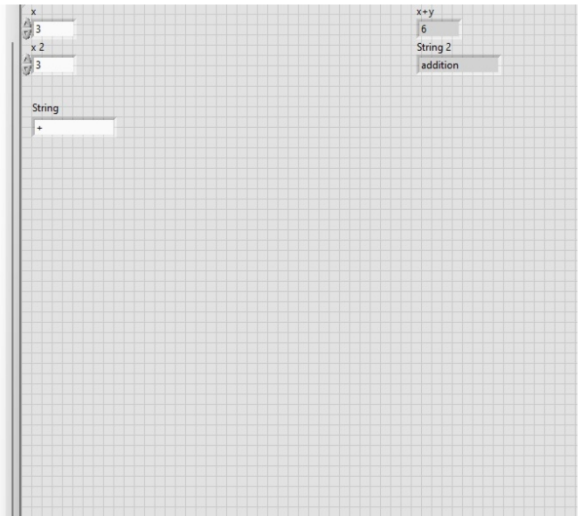


Figure 5: Default case

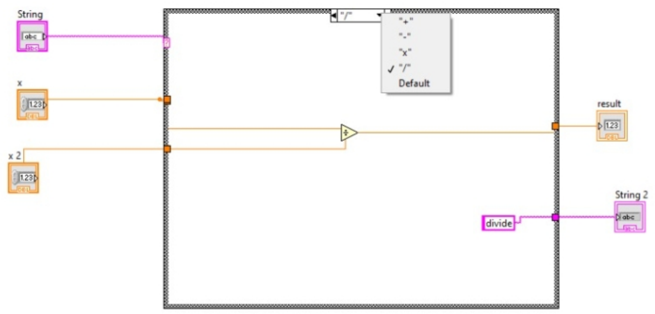
6. Do the same for the other three cases. Add another case default. For default case, connect as illustrated in figure 5. On block panel, add appropriate inputs and run VI.

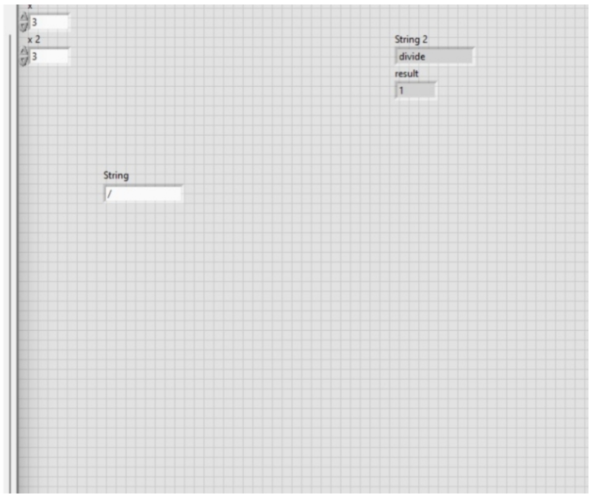
Answer the following:





Above shows the implementation for addition case

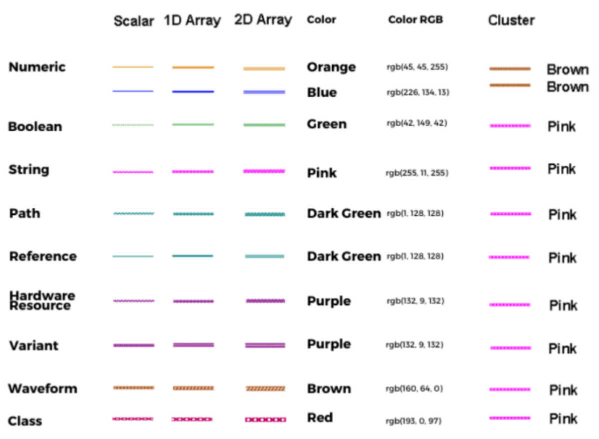




Above shows the implementation for division case, also as per drop down box we have created other cases as well including the default case.

a) What does the color of connecting lines in block diagram window represent?

The specific color represents different data types or signal types being transmitted between nodes. These colour help you identify the data types. Like as per our case pink represents string and orange represents numeric. Consider the following chart:



	Scalar	1D Array	2D Array	Color	Color RGB	Cluster
Numeric	—	—	—	Orange	rgb(45, 45, 255)	— Brown
	—	—	—	Blue	rgb(226, 134, 13)	— Brown
Boolean	—	—	—	Green	rgb(42, 149, 42)	— Pink
String	—	—	—	Pink	rgb(255, 11, 255)	— Pink
Path	—	—	—	Dark Green	rgb(1, 128, 128)	— Pink
Reference	—	—	—	Dark Green	rgb(1, 128, 128)	— Pink
Hardware Resource	—	—	—	Purple	rgb(132, 9, 132)	— Pink
Variant	—	—	—	Purple	rgb(132, 9, 132)	— Pink
Waveform	—	—	—	Brown	rgb(160, 64, 0)	— Pink
Class	—	—	—	Red	rgb(193, 0, 97)	— Pink

(Source: https://knowledge.ni.com/KnowledgeArticleDetails?id=kA00Z0000019LsVSAU&l=en-PK)

4.6 Script Nodes and Graphing

LabVIEW has a Math-script node, which can be used to include MATLAB code or other text based programming languages' code in a VI. Using the Math-script node one can add MATLAB script to VIs as well as create and run .m files from within LabVIEW.

LabVIEW also has multiple indicators for graphing, all found under "Graph" tab in the Controls Palette. Two popular instances are a "Waveform Graph" and a "Waveform Chart", the former allowing point by point plotting and latter for plotting an entire array of data.

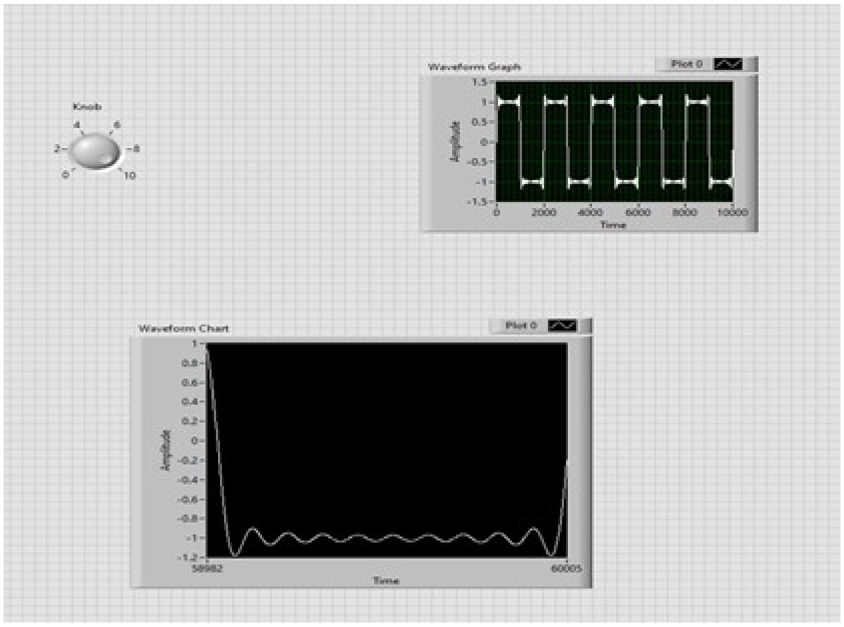
To demonstrate both of these we are going to create a VI, which the Fourier series of a square wave.

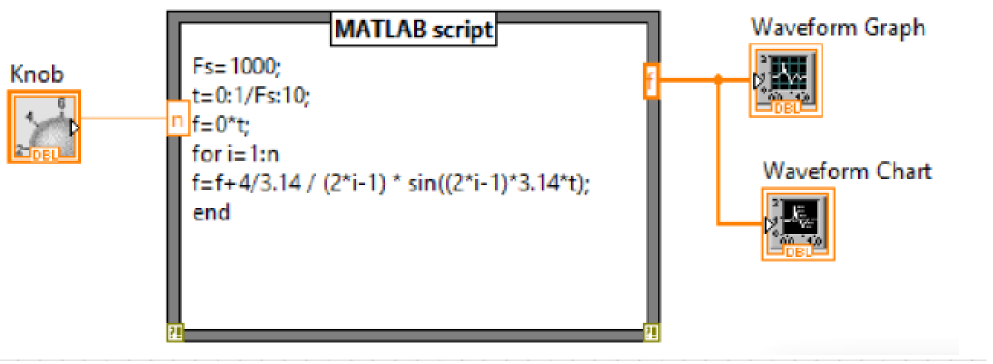
Task 5: To practice MATLAB-Script and Graphing in LabVIEW

1. Right-click on the Block panel, select Mathematics >> Script and formulas>>Script nodes>>MATLAB script
2. Select MATLAB Script node and create a node on front panel.
3. Type the following code in the node.

```
 Fs=1000;

t = 0:1/Fs:10; f = 0*t;

for i = 1:n

f = f + 4/3.14 /(2*i-1)*sin((2*i-1)*3.14*t);

end
```

4. Right-click on the MATLAB Script node and add an input with the name **n**, and an output **f** (right click on the output **f** >> Choose Data type>> 1-D array of real).
5. In front panel, add graph from Control palette>>Express>>Graph Indicators>>Waveform Graph
6. Similarly add control; In Control palette>>Express>>Numeric Control>>Knob. It will be used to define the number of terms in the Fourier sum.
7. In the block panel, connect input to numeric control and output to waveform graph. You can set the scaling of the graph by accessing the properties by right clicking on the waveform graph.
8. Run Your VI.
9. Now replace "Waveform Graph" by "Waveform Chart" and observe the change in output. Watch https://youtu.be/HtjgeoJc4zA to learn more.





```
Fs= 1000;
t=0:1/Fs:10;
f=0*t;
for i=1:n
f=f+4/3.14 / (2*i-1) * sin((2*i-1)*3.14*t);
end
```

Answer the following:

1. What is the difference in the output of "Waveform Graph" and "Waveform Chart"? What is an XY graph?

   Waveform chart is used to plot and remember data points which are added usually one at a time. The Waveform Graph is used to plot an entire array of data all at once. In waveform graphs, when the number of data points displayed exceeds the specified length, it automatically overwrites the oldest points with the new ones, creating a scrolling effect. In waveform charts, older data points are not removed or overwritten but it has feature that you can navigate through older data and focus on specific time intervals.

   XY graph is a graphical user interface element used for plotting and visualizing data points in 2D Cartesian coordinate system. It can be used for plotting experimental data with independent variables, displaying sensor measurements, and visualizing relationship between two variables.

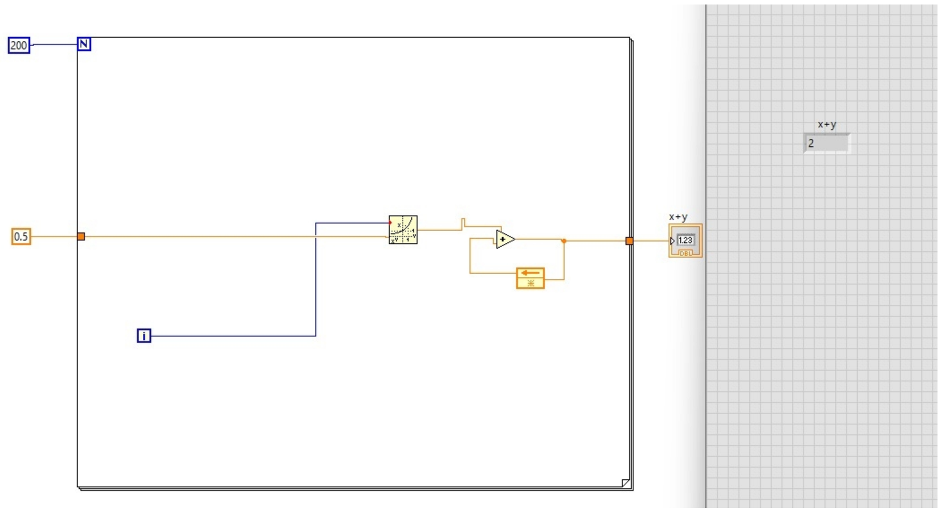
2. What are the different data types available in LabVIEW?

   There are numerous data types available in labview like Numeric, Boolean, string, array, cluster, enumerated, timestamp datatype etc.

4.7 Post Lab Tasks

Task 6: To calculate sum of geometric series

It is well known that $1 + x + x^2 + \ldots = \frac{1}{1-x}$ when $|x| < 1$. Write a LabVIEW program that takes a value for x as an input, and a number of iterations. The program will use a loop to calculate the sum of the geometric series for the specified number of operations. Demonstrate your program with $x = 0.5$ and 200 iterations. *You will have to either use for or while loop. Implement a simple loop first to see how loops work in LabVIEW and then figure out how to pass data between different iterations.*

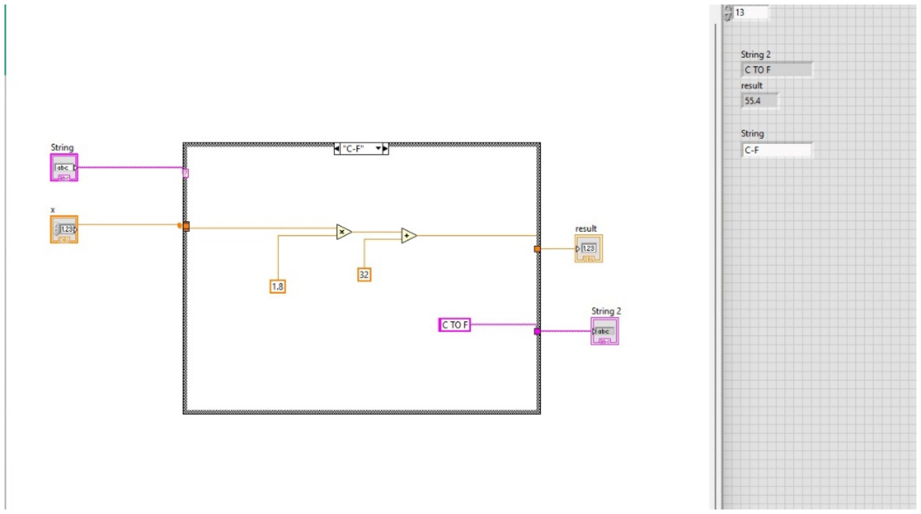


Task 7: To make a unit converter

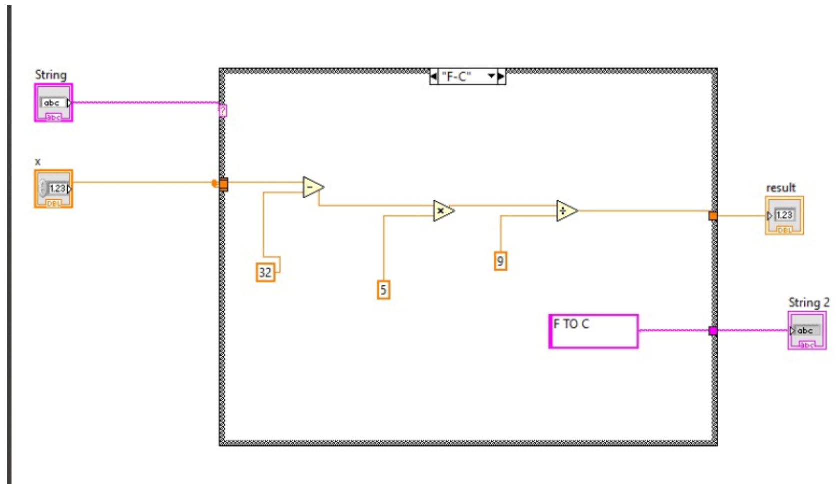
Using case structure, build a VI to convert temperature from °C to °F and °F to °C depending upon the user's choice and display the result using a single indicator for both conversions. Develop a mechanism to distinguish between the results by their respective units. The formula for conversion is given below.

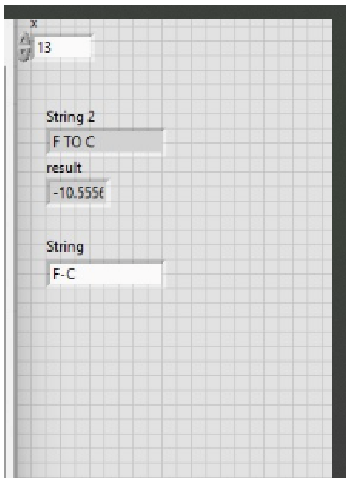
$$°F = \frac{9}{5}{}^{0}C + 32$$

CENTIGRADE TO FARHENHEIT



FARHENHEIT TO CENTIGRADE





Task 8: Answer the following:

1. How do you create an array? What is a cluster in LabVIEW?

ANS: There are two ways to create an array in LabVIEW:

1. Place an array shell on the front panel and then place an element, such as a numeric, Boolean, or waveform control or indicator, inside the array shell.
2. Use the Array Constant function on the block diagram.

To create a cluster in LabVIEW, you can use the Cluster function on the block diagram. A cluster is a collection of data elements that can be of different data types. The elements of a cluster can be accessed by name or by index.

2. How does data pass into and out of an execution structure, and between different iterations of a loop?

ANS: Input terminals: These terminals are used to pass data into the structure.

Output terminals: These terminals are used to pass data out of the structure.

Between different iterations: The data that is passed between different iterations of a loop is typically passed through a shift register. A shift register is a special type of tunnel that stores data from one iteration of a loop to the next iteration.

3. How do you create a local varaibel?

ANS: There are two ways to create a local variable in LabVIEW: 1). Right-click an existing front panel object or block diagram terminal and select Create >> Local Variable from the shortcut menu. 2). Select a local variable from the Functions palette and place it on the block diagram.

Assessment Rubric

Lab 04
Simulating Dynamic Models using Simulink

Name: Afsah Hyder	**Student ID: 07065**

Points Distribution

Task No.	**LR2** **Simulation/Model/ Code**	**LR5** **Results/Plots**	**LR 10** **Analysis**	**AR 6** **Class Participation**
Task 1	4	4	-	-
Task 2	8	4	6	-
Task 3	8	4	6	-
Task 4	8	4	2	-
Task 5	4	4	2	-
Task 6	8	4	-	-
Task 7	8	4	-	
Task 8	-	-	8	
SEL	-	-	-	/20
Course Learning Outcomes	-			CLO 4
Total Points	/100		/20	
	/120			

For details on rubrics, please refer to *Lab Evaluation Assessment Rubrics*.