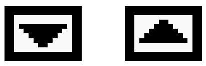
Week 2

Shift Registers:

Shift registers are similar to static variables in text-based programming languages. Use shift register when you want to pass value from previous iterations through the loop to the next iteration. For example, at every iteration of a loop you are acquiring one piece of data and need to calculate the average of the values of the data at the end of every five iteration. Note that the calculation will be correct only if the values of the data when the loop runs. Shift register transfer values from one loop iteration to the next. It appears as a pair of terminals directly opposite of each other on the vertical sides of loop border.



Note: Feedback Nodes can be considered as the alternative method used by LabVIEW for retaining information from a previous iteration. Refer to the *Feedback Node* topic on the *LabVIEW Help* for more information about feedback nodes.

Create a shift register by right-clicking the left or right border of a loop and select **Add Shift Register** from the shortcut menu.
The stacked shift registers let you access data from previous loop iterations. Moreover, stacked shift registers remember values from previous multiple iterations and carry the values from previous iterations to the next iterations. To create a stacked shift register, right-click the left terminal and select **Add Element** from the shortcut menu.

Exercise 1:

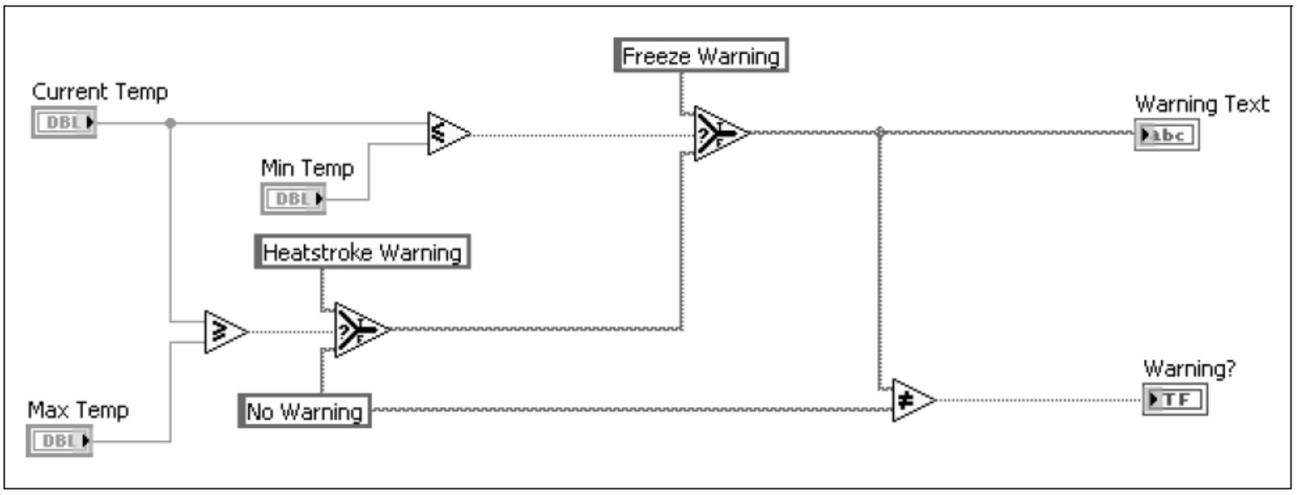
Use shift registers to upgrade the temperature history monitor from Exercise 4 of week 1 to monitor the average of the last two observations.

Exercise 2:

Using case structure for error handling

A VI has been created where a user can input a temperature, a maximum temperature, and a minimum temperature. A warning string is generated based on the relationship of the given inputs. However, a situation could occur that causes the VI to work incorrectly. The user could enter a maximum temperature that is less than the minimum temperature. Modify the VI so that a different string is generated to alert the user about the temperature error: "Upper Limit < Lower Limit." Set the Warning?

Hints: use case structure



Create a SubVI:

A VI within another VI and is called a subVI. A subVI corresponds to a subroutine in the text-based programming languages. The upper right corner of the front panel and block diagram displays the icon for the VI. This icon is the same as the icon that appears when you place the VI on the block diagram.

Exercise 3:

Create a subVI to convert degrees Celsius to degrees Fahrenheit.

Create this icon $\boxed{C \to F}$ as its visual representation.

Exercise 4:

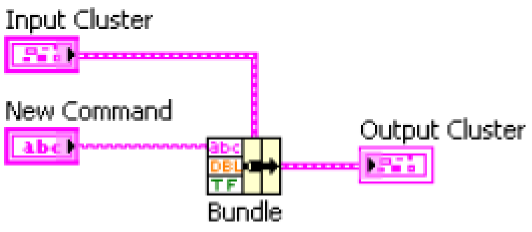
Create a VI that prompts the user to enter a room temperature in Fahrenheit, displayed from the output. Use your converter from the previous exercise to present the data in Centigrade.

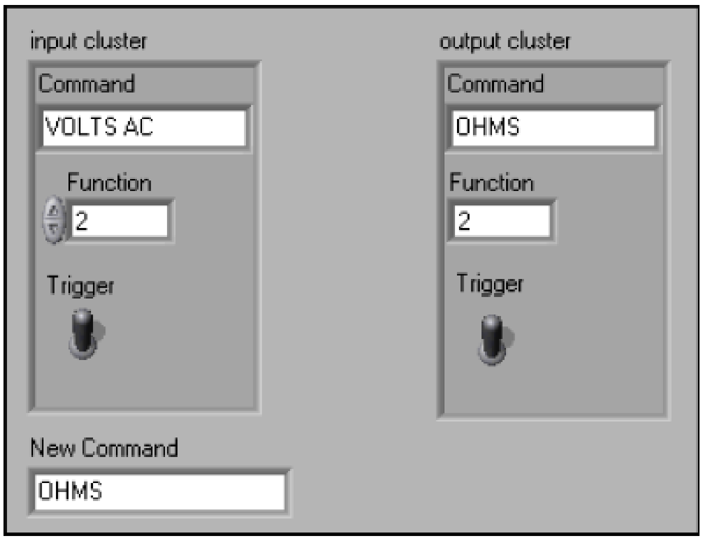
Clusters

The purpose of clusters in Labview is to group data elements of mixed types into a bundle (e.g. stack of wires in a telephone cable). Each wire in the cable consisting of different wires represents a different element within the cluster. You can also think of a cluster being to a *record or a construct* in text-based programming languages. Bundling several data elements into clusters eliminates wire clutter on the block diagram and reduces the number of connector pane terminals that a subVI may need.

Exercise 5:

Use a bundle function to assemble a cluster to change the value of individual elements in a cluster without having to specify new value for all elements.





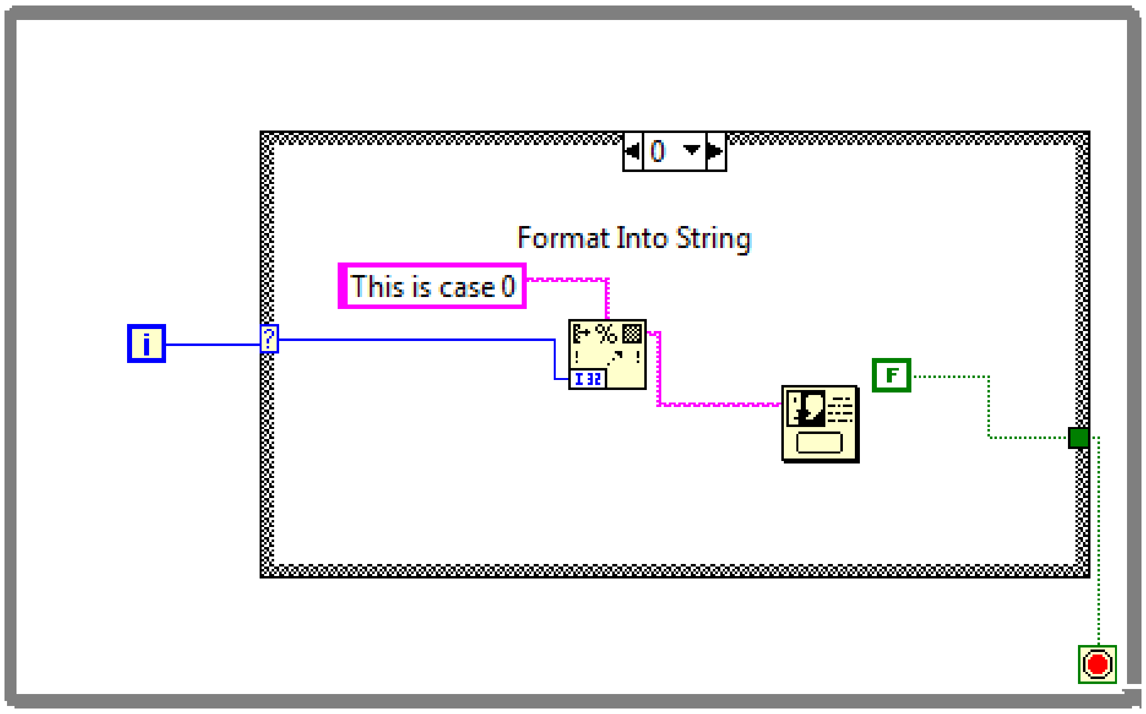
State machines

The state machine is one of the fundamental frameworks that NI LabVIEW developers frequently use to build applications quickly. Developers use state machines in applications where distinguishable states exist. Each state can lead to one or multiple states and may function as a sign for the end of a process flow. A state machine relies on user input or in-state calculation to determine which state to go to next. You can perform many different operations and implement actions. These actions depend on the previous and current inputs as well as the states of the system.

In LabVIEW software, you can create a basic state machine with a while loop, a shift register, a case statement, and some form of case selector.

Exercise 6:

Use while loop and case structure to create a simple machine state which pops up 3 different dialog box messages in every loop circle i.e. “This is case 0” , “this is case 1” and so on **.**



Hints: use ‘format into string ‘and ‘one button dialog’.