Exercise 0C:

Welcome to the Code School course on Intro to Counters. The logic below is a simple counter circuit. There is a lot of new functionality here, so we'll break it down piece by piece over the next few lessons.

First, notice the new tag type in the tag list. Rather than having a value next to it, the 'Counter' tag has an arrow. Click the arrow and you'll see that the counter tag is made up of several other tags, called children. Child tags are referenced using dot notation. For example, the 'EN' value within the 'Counter' tag would be referenced as 'Counter.EN'.

The 'Counter' tag is of the type Counter. When creating a new tag, it is possible to choose a type for that tag by using the drop down below the tag name input box. We will go over this in more detail in a future lesson.

For now, toggle the 'Input' tag on and off five times. Once you see the 'Counter Done' tag turn ON, press Check to go to the next lesson!

Exercise 1C:

Click the down arrow next to the 'Counter' tag to display its children tags. Two of the children tags, 'EN' and 'Q' are Boolean tags. Boolean tags can have two states, ON or OFF. In the previous courses all of the tags we worked with were Boolean tags.

The 'EN' tag is turned ON by the counter whenever there is power flowing into the timer. Notice how the 'EN' tag turns ON when the 'Input' tag is ON.

The 'Counter' also has two children tags which have the tag type 'Number'. Rather than simply being ON/OFF like Boolean tags, tags with the type 'Number' can store any numeric value. The 'ACC' tag is short for accumulated, and stores the total number of times the input to the counter has turned ON. The 'PRE' tag is short for preset.

The 'Q' tag turns on when the accumulated counter value has reached the preset value (in other words, when the 'ACC' tag is greater than or equal to the 'PRE' tag). In this example, we use 'Counter.Q' on the second rung to turn ON the 'Counter Done' tag after the input has been pressed several times.

Currently, the 'Counter Done' tag turns ON after the 'Input' tag has been toggled 5 times. Modify the value of 'Counter.PRE' so that the 'Input' tag has to be toggled 10 times before the 'Counter Done' tag turns ON, then press Check to go to the next lesson!
Exercise 2C:

The ladder logic in this lesson is similar to the previous two lessons. The 'Counter.Q' tag is used to turn the 'Output' tag ON after the 'Input' is toggled 5 times. There is one problem though, once the 'Counter.ACC' tag is greater than or equal to 5 and the 'Output' tag turns ON, there isn't a good way to reset the counter and turn the 'Output' OFF again.

We are going to modify this logic to allow for resetting the timer by using the 'Reset Counter' tag. First, create a new rung (on the 'Other' tab, press the 'Add Rung' button). Now we need to drag a normally open contact (the left-most element on the 'Contacts' tab) and a reset element (the right-most element on the 'Time/Count' tab) from the toolbox to our new rung. Make sure the reset element is on the right side of normally open contact after you have placed them on the new rung. Click the drop-down box above the normally open contact and select the 'Reset Counter' variable. Do the same thing for the reset element and select the 'Counter' variable.

The reset element causes the accumulated ('Counter.ACC') value of the counter to be reset to zero. Try it out by toggling the value of the 'Input' tag 5 times until the 'Output' tag turns ON, then press the 'Counter Reset' button to see the 'Output' tag turn OFF. Press Check to go to the next lesson!

Exercise 3C:

In this lesson you are going to implement a counter element on your own by modifying the existing logic, which currently just turns the 'Output' tag ON anytime the 'Input' tag is ON. We want to modify this logic so that the
'Input' tag needs to be toggled three times before the 'Output' tag turns ON.

Drag a 'Count Up' element from the Time/Count tab of the toolbox onto the 'Output' coil on the first rung of the ladder logic. Now move the coil onto the second rung of ladder logic (this will temporarily cause the 'Output' tag to turn ON). Drag a normally open contact from the toolbox onto the 'Output' coil on the second rung. The normally open contact should be on the left of the 'Output' coil.

Now we are going to create a new tag. Using the drop down menu in the tag list, select 'Counter' as the tag type. Now type in the name 'Counter' and press Add to create the new tag. In general, you can use any name you'd like here, but for this lesson it is important that the tag is named 'Counter'. In the tag list, expand the 'Counter' tag by clicking the downward facing triangle next to it, and then set the value of the 'PRE' child tag to three.

Use the drop down box within the counter element you created on rung 1 to select the 'Counter' tag you just created. Similarly, select the 'Counter.Q' tag using the drop down box above the normally open contact you created on rung 2.

Test your logic by toggling the value of the 'Input' tag. You should see the 'Output' tag turn ON after the 'Input' tag has been toggled three times. Press Check to go to the next lesson!

Exercise 4C:

In this lesson we will combine the output from two counters. The goal is to turn on the 'Output' tag after the 'Input 1' tag has been toggled 3 times, AND the 'Input 2' tag has been toggled 5 times.

Click the arrow next to each counter in the tag list to see their child variables. Notice that 'Counter 1.PRE' is set to 3 and 'Counter 2.PRE' is set to 5.

The third rung already has the elements needed to complete this logic, you just need to assign the variables. Assign the 'Counter 1.Q' and 'Counter 2.Q' variables to the normally open contacts on the third rung, and assign the 'Output' tag to the coil.

Now test your logic by toggling the value of the inputs until the 'Output' tag turns ON. When you are finished, press Check to go to the next lesson!
Exercise 5C:

In this lesson we will combine the 'Counter 1.Q' bit with 'Input 2' using OR logic, so that the 'Output' tag turns ON when either 'Input 1' has been toggled 3 times, OR 'Input 2' is ON.

The necessary elements have already been placed on the second rung. You'll need to assign 'Counter 1.Q' to one of the normally open contacts, and assign 'Input 2' to the other. Lastly, assign the 'Output' tag to the coil.

You should see the 'Output' tag turn ON when either 'Input 1' has been toggled 3 times (causing 'Counter 1.Q' to turn ON), or 'Input 2' is ON. Press the Check button when you are ready to continue with the next lesson!

Exercise 6C:

In this lesson we will use one counter to reset another counter. This allows for building logic which causes the 'Output' to turn ON when 'Input 1' is toggled 3 times and turn OFF when 'Input 2' is toggled 3 times. We will achieve this by resetting both counters when the 'Counter 2.Q' bit is ON.
The fourth rung has all of the elements needed for this logic. You'll need to assign 'Counter 2.Q' to the normally open contact, and assign one counter to each reset element.

You should see the 'Output' variable turn ON after toggling 'Input 1' 3 times. Toggling 'Input 2' 3 times should reset both counters and turn the 'Output' variable OFF. Press Check to continue with the next lesson!

Exercise 7C:

In this lesson, you'll write your own counter logic. The logic you write should turn the 'Output' ON when 'Input 1' has been toggled 3 times, AND 'Input 2' is OFF. In this case, turning 'Input 2' ON should stop the 'Output' from coming on (you'll need to use a normally closed contact for this).

The 'Counter 1.PRE' tag has already been set to 3 for you, but you'll need to create all of the logic. Feel free to review previous lessons for ideas, but note that when you load a previous lesson your current work will be lost unless you save it. You can save using the button on the 'Other' tab of the toolbox. Clicking the save button generates a unique URL which will always link to the ladder logic which you saved.

Also, while testing you may want to reset the accumulated value in 'Counter 1'. The simplest way to do this is to click the triangle next to 'Counter 1' in the tag list to show the children tags. You can directly change the 'Counter 1.ACC' value back to 0 to reset the counter.

After you have tested your logic, make sure 'Input 1' is OFF, then press the Check button to continue to the next course!
Exercise 0T:

Welcome to the Code School course on Intro to Timers. We'll start with a simple application of the on delay timer.

On delay timers work similarly to counters. Click the triangle next to the 'Timer' tag in the tag list to see it's child tags. Like the counter, the on delay timer has an 'EN' tag (enable), a 'Q' tag (done), an 'ACC' tag (accumulated time), and a 'PRE' tag (preset time). The timer does have one new tag, the 'TT' tag, which is ON when the timer is timing. The on delay timer is timing anytime the input to the timer is ON ('EN' is ON) and the 'ACC' tag is less than the 'PRE' tag.

Turn ON the 'Input' tag and watch the the 'ACC' tag increase as the timer keeps track of the elapsed time. Notice that once the 'ACC' tag value is greater than the 'PRE' tag value the timer stops counting, and the 'Q' tag turns ON to indicate the timer is done.

Now turn the 'Input' tag OFF and notice how the 'ACC' tag value returns to zero, and the 'Q' tag turns OFF. One difference between the on delay timer and the counter is that the on delay timer automatically resets when the input is OFF (when the 'EN' tag is OFF).

To complete this lesson, turn the 'Input' tag ON again and wait three seconds until the 'Output' tag turns ON. Now press Check to continue with the next lesson!

Exercise 1T:

The logic in this lesson is the same as the previous lesson, but the on delay timer has been replaced with a retentive timer. Retentive timers retain their accumulated value when the input is OFF, much like a counter.

Expand the 'Timer' tag in the tag list to see the children variables. Toggle the 'Input' tag ON and OFF while watching the 'Timer.ACC' tag. Notice how, unlike the on delay timer, with the retentive timer the accumulated value does not reset to zero when you turn the 'Input' tag OFF.

Since the retentive timer doesn't reset automatically, we will need to use a reset element to clear the accumulated value. First, create a boolean tag called 'Reset'. Then, add a new rung and drag a normally open contact and the reset element (right-most element on the Time/Count tab of the toolbox) onto the new rung. Make sure the normally open contact is on the left side of the reset element. Now assign the 'Reset' tag to the normally open contact, and assign the 'Timer' tag to the reset element.

Test your logic by turning on the 'Input' tag until the 'Output' tag turns ON. The 'Output' tag should stay ON after you turn the 'Input' tag OFF, but turning the 'Reset' tag ON should reset the timer and cause the 'Output' to turn OFF. Press Check to continue with the next lesson!
Exercise 2T:

Sometimes it is desirable to provide a warning before turning on an output. An example of this might be turning on a siren for a few seconds before turning on the output for a conveyor belt. That is what we will do in this example.

The logic to turn on the 'Output' tag after the 'Input' tag has been on for 3 seconds has been created for you. There is also a 'Warning' tag, which represents the siren we want to turn on during those 3 seconds before the 'Output' tag is turned ON. Drag a normally open contact and a coil onto the third rung, making sure the coil is on the right of the contact. Now assign the 'Warning' tag to the coil, and 'Timer.TT' to the contact.

Test your logic to make sure that when you turn the 'Input' tag ON, the 'Warning' tag turns ON for 3 seconds before the 'Output' tag turns ON. Press Check to continue with the next lesson!

Exercise 3T:

There are many ways to use timers to create a pulsed output. This is useful for a variety of reasons. One example would be if you want to create a flashing light to indicate a machine fault. The logic below causes the 'Fault Light' tag to pulse when the 'Fault' tag is ON.

Expand the timer tags in the tag list, and set 'On Timer.PRE' and 'Off Timer.PRE' to 5 seconds. This will make it easier to see what is happening as the logic is executing. Turn the 'Fault' tag ON and try to figure out how this
This logic contains a couple new concepts, so we'll go through it rung by rung. The first rung starts the 'On Timer' when the 'Fault' tag is ON. You'll notice there is also a normally closed contact on the first rung, but we'll come back to that. The second rung starts the 'Off Timer' when the 'On Timer' is done. The 'Off Timer.Q' tag is used on the first rung to reset the 'On Timer' when the 'Off Timer' is done. This is what creates the flashing behavior. The third rung turns the 'Fault Light' ON whenever the 'On Timer' is timing.

Make sure you have changed the preset values for both timers to 5 seconds, and spent some time reviewing the functionality of this logic, before pressing Check to continue with the next lesson!

Exercise 4T:

We have already covered the functionality of the on delay timer and the retentive timer. The final timer type is the off delay timer. This type of timer is useful in cases where you want the output to stay on for longer than the input is on.

Test this logic by toggling the value of the 'Input' tag. You'll see that the 'Output' tag stays on for 5 seconds after you turn the 'Input' tag OFF.

Modify this logic so that the 'Output' tag stays on for only 3 seconds after you turn the 'Input' tag OFF (change the 'Timer.PRE' tag from 5 to 3 in the tag list), and then press Check to go to the next lesson!
**Exercise 5T:**

It is possible to combine timers with counters to create more complex logic. In this lesson we will use a counter to count the number of times the 'Input' tag has been on for at least 3 seconds. Once the counter counts to 3, we will turn the 'Output' tag ON.

Drag a normally open contact and a count up element onto the second rung, making sure the counter is to the right of the contact. Assign 'Timer.Q' to the contact and 'Counter' to the counter. On the third rung, drag a normally open contact and a coil. Assign 'Counter.Q' to the contact and 'Output' to the coil.

Expand the 'Timer' and 'Counter' tags in the tag list so you can see the accumulators as you test this logic. Test the logic by toggling the 'Input' tag. If the 'Input' tag is on for at least 3 seconds you should see the 'Counter.ACC' tag increase by 1. Once the accumulated counter value reaches the preset value the 'Output' tag should turn ON.

After testing your logic, press Check to continue with the next lesson!

**Exercise 6T:**

In this lesson, we are going to combine two timers in order to turn the 'Output' tag ON when both input tags have been on for at least 3 seconds.

The current logic turns the 'Output' tag ON when 'Input 1' is ON for 3 seconds. You'll modify the logic to also require that 'Input 2' is on for 3 seconds before turning the 'Output' ON.

Drag a normally open contact and an on delay timer onto the empty second rung. Assign 'Input 2' to the contact and 'Timer 2' to the timer. Now add a normally open contact to rung 3, and assign the 'Timer 2.Q' tag. The contact you add to rung 3 should be right next to the existing contact for the 'Timer 1.Q' tag, like a boolean AND.

Test your logic by ensuring that the 'Output' tag turns ON only after both 'Input 1' and 'Input 2' have been on for 3 seconds. The 'Output' should immediately turn OFF if either input turns OFF. This is the last lesson in the Intro to Timers course, so once you are finished press Check to continue with the next course!