Title: **Input Table and Order of Operation** Job: 5

Course: Introduction to Automation Unit: Introduction to PLCs CLO: 1, 4

Name \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Station \_\_\_\_\_\_\_ Date \_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Objectives**

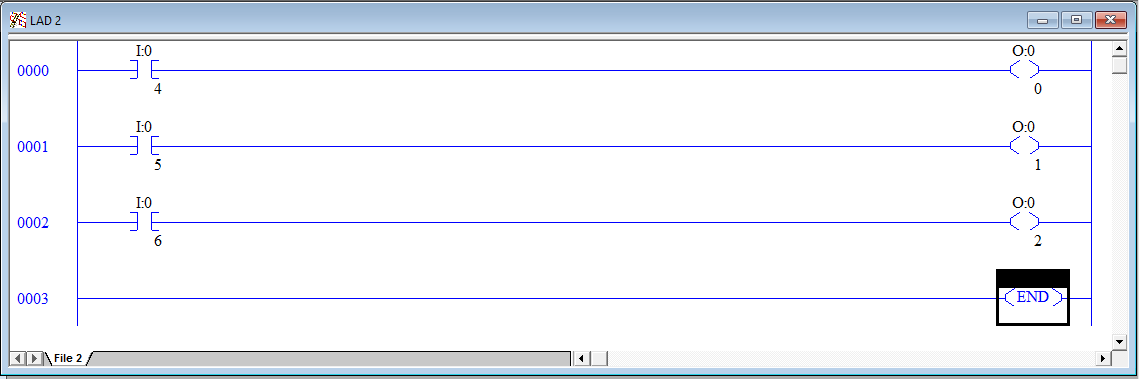
1. Student shall identify the various features within RSLogix 500 Pro software.
2. Student shall locate and select the appropriate *System Options* for composing ladder logic within RSLogix 500 Pro.
3. Student shall compose their first ladder logic program within RSLogix 500 Pro.

**Assessment**

Students shall demonstrate a comprehension of the objectives listed above by scoring a minimum of 75% on this shop job. Grading shall be based on the Introduction to PLC rubric.

**Instructions**

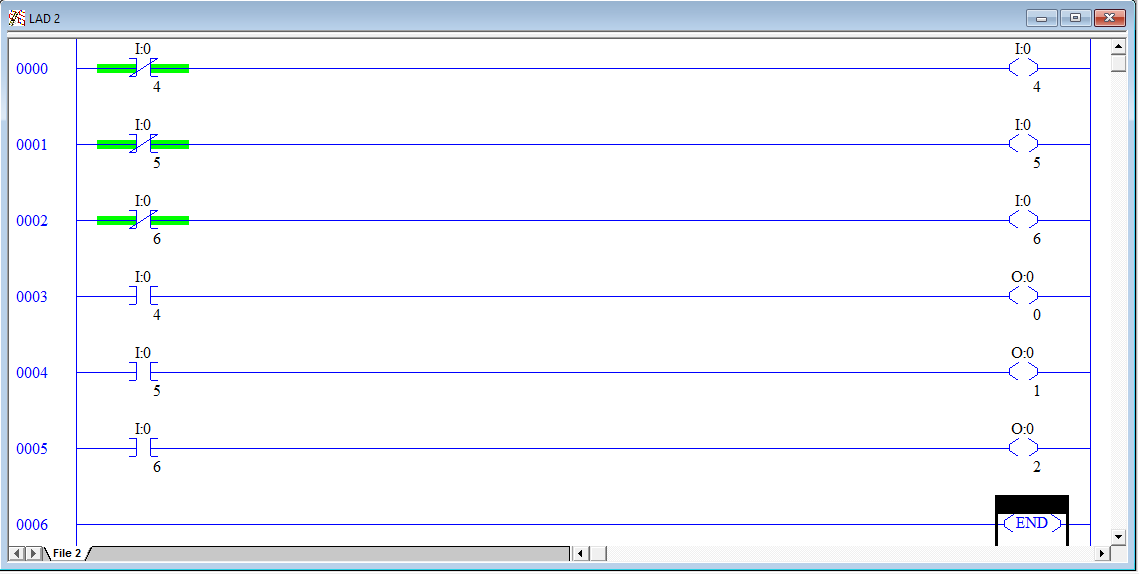
Create a new ladder programmed named I2P Job 5 - Input Table and Order of Operation. Construct the following ladder diagram. After composing the ladder diagram below, press the  *Verify File* button and ensure that the program has no issues. If the program is warning and error free, go to *Comms > Download > Download Program*.



Press the normally closed pushbutton and observe that the green light goes out. Press the normally open pushbuttons and observe that the yellow and red lights come on. This behavior is expected since the normally closed pushbutton will pass 24VDC to the light until it is pressed where the normally open pushbuttons need to be pressed to pass 24VDC to their pilot lights. This is behavior that was observed in the Manual Motor Controls units. This lesson is to highlight that once an input is read from the field, it can easily be manipulated. Proceed to the next step to see an illustration of this functionality.

Insert the additional rungs above the existing program as illustrated in the program below. Notice that this program’s first three rungs read from an input and write directly back to the same input. In this case, examine if open (XIO) is used. Previously we’ve used this “normally closed” symbol to represent a logical NOT. This is the case here as well. Rung 0000 assigns the NOT of the forth input back to itself. Below is a truth table that shows the result of Rung 0000.

|  |  |
| --- | --- |
| I/4  Before Rung 0000 | I/4  After Rung 0000 |
| 0 | 1 |
| 1 | 0 |



Observe the states of the lights. How do they differ from the previous program?

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Press the normally closed pushbutton and both the normally open pushbuttons and observe each light.

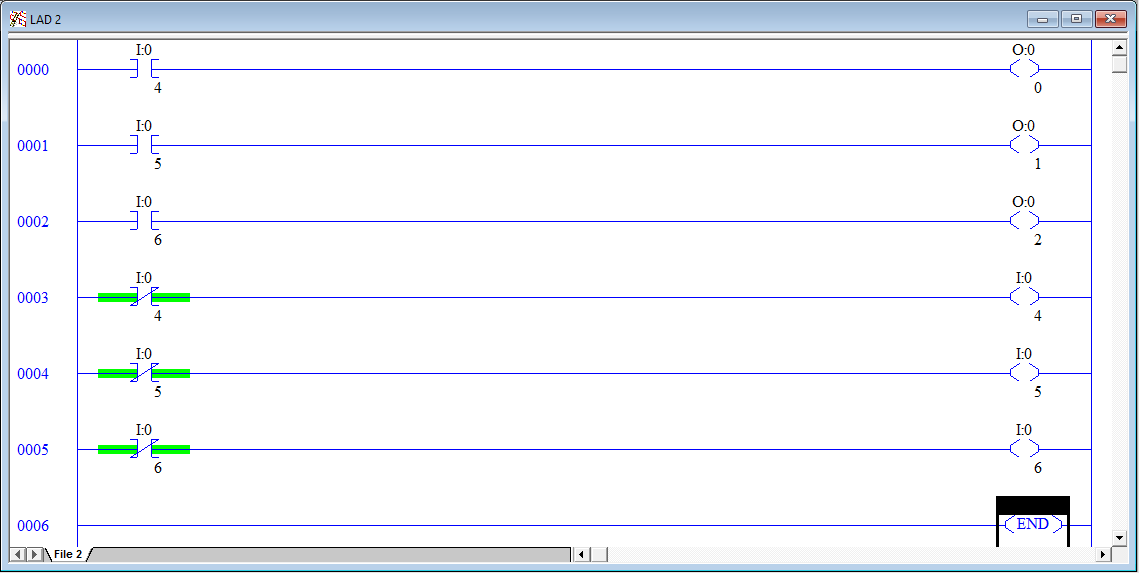
Why did the lights change in behavior?

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Now we shall experiment with the order of operation and see how that can affect the behavior of a programs outputs. Modify the program to change the rung order as seen in the illustration below.



Verify and download the newly modified program. Observe the states of the lights. How do they differ from the previous program? How do they compare to the first program?

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Press the normally closed pushbutton and both the normally open pushbuttons and observe each light.

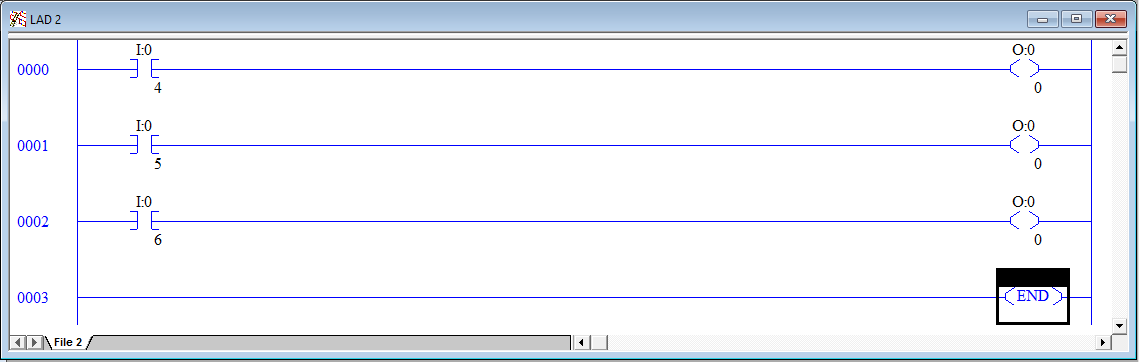
All the code is the same as before, just in a different order. Why did this cause a different result?

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Lastly, we shall look at how this has an effect on outputs. Create the following changes to the program so it matches the illustration below. (**Note:** All the outputs are the same register)



Verify and download the newly modified program. Observe the states of the lights. How do they differ from the previous three programs?

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Press the normally closed pushbutton and observe the lights. Note the behavior.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Press the upper normally open pushbutton and observe the lights. Note the behavior.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Press the lower normally open pushbutton and observe the lights. Note the behavior.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Explain why there is a change in the behavior?

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Discuss your findings with your instructor Instructor Initials \_\_\_\_\_\_\_\_\_\_\_