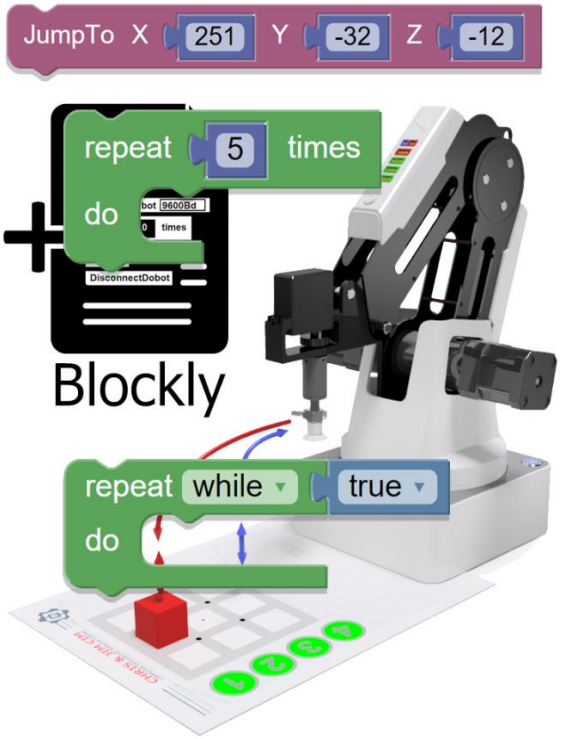
2 Blockly - PnP with Jumps and Loops

NAME: _____ *Date:* _____ *Section:* _____

INTRODUCTION

When programming a robotic arm, it often becomes necessary to repeat operations a set number of times or indefinitely. This can be accomplished by adding different styles of loops to our program. It is also a good programming habit to optimize or reduce your lines of code when appropriate.



In this activity you will learn one way to simplify your code as well as add different styles of loops to a basic Pick and Place operation in blockly.

The two main types of loops are:

- Forever loops
- While loops
- Repeat

Caution: NEVER wire anything to the Dobot Magician while it has power on. ALWAYS shutdown the Dobot before making connections or damage to the robot could occur.

KEY VOCABULARY

- Forever Loop
- While Loop
- Repeat
- True
- Jump
- Placeholder
- Condition

All blockly commands have been put into a separate document called *Blockly Vocabulary*, and can be referred to at any time throughout all of these activities.

EQUIPMENT & SUPPLIES

- Robot Magician
- Dobot Field Diagram
- 1” cubes or cylinders.
- DobotStudio software
- Suction Cup Gripper

ESSENTIAL QUESTIONS

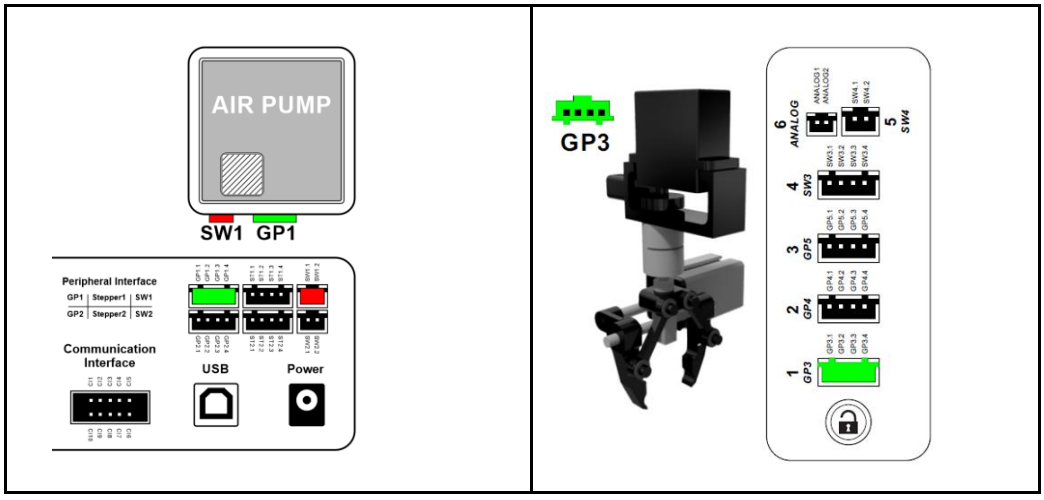
Essential questions answered in this activity include:

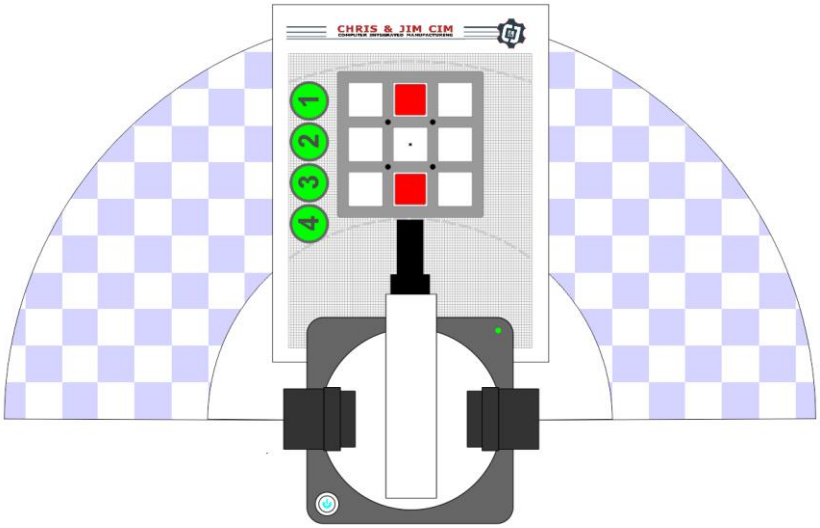
- What’s a *Jump*, and why is it important?
- How to make the robot repeat a motion or a task?
- When would I use a *While* statement?
- How do I use an *Until* statement?
- How do I set jump height?

PROCEDURE

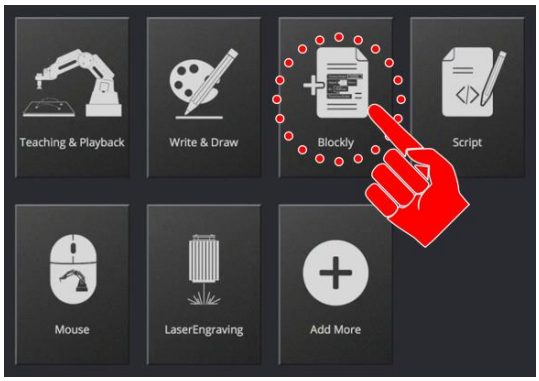
Caution: NEVER wire anything to the Dobot Magician while it has power on. ALWAYS turn it off before making connections or damage to the robot could occur. Be sure to ask your instructor if you have any questions.

1. Print the Blockly - Pick and Place Field Diagram
2. Set up the robot with a suction cup and place a cube in one of the red squares on the field diagram provided





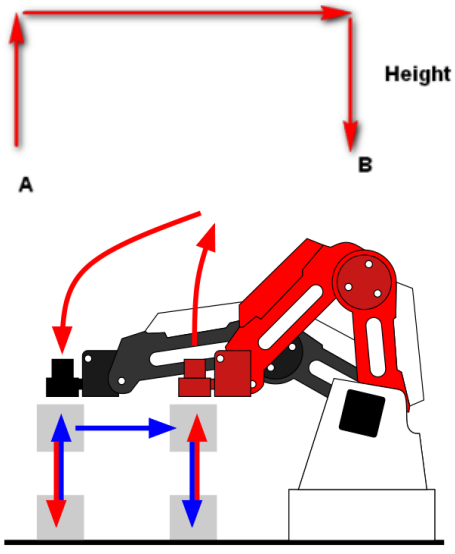
3. Open up Blockly in the software



When DobotStudio is closed with a file open, it will reopen with the last file used. Insure the file open is the one you want to edit, if it does not, you may end up overwriting another program

Open your Pick and Place Blockly Activity.

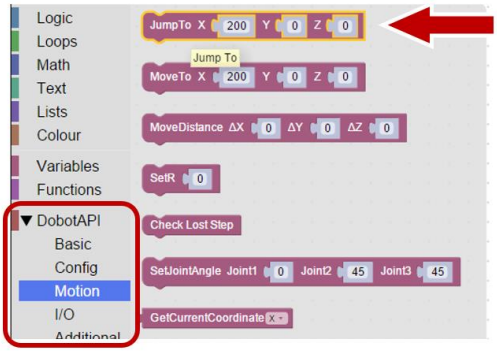
In this activity, we will add a *JUMP* movement as well as a *LOOP*. The JUMP command in blockly works the same as it does in Teach and Playback. This will allow us to remove a few lines of code create the same operation is less steps.



JUMP Movement - A *JUMP* movement combines three steps into one. It combines the raise up, over, and back down to a new point. This type of movement simplifies repetitive movements such as a dipping operation or soldering operation. The Z Height is defined in the JUMP parameters in the settings menu.

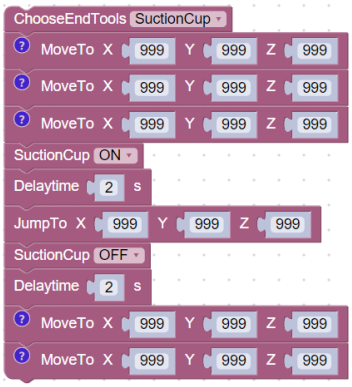
A *JUMP* movement does not replace the initial ABPick to ATPick as well as the finial ABPlace

Remove the lines of code from your previous activity and replace them with a ***JumpTo*** command. The JumpTo command is found in the same DobotAPI/Motion Tool box



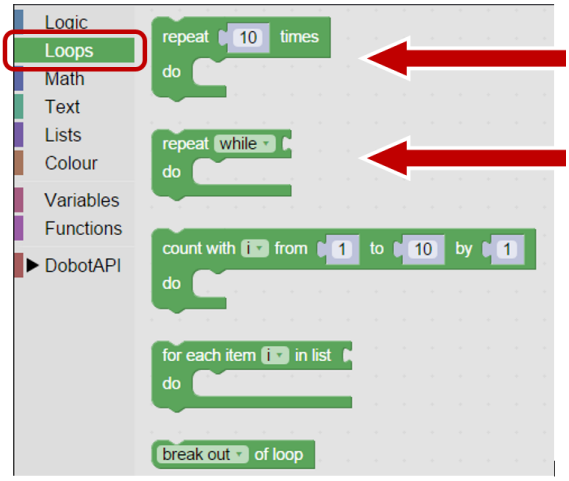
Adjust the X, Y, and Z movements as needed

Run your code to ensure proper operation.

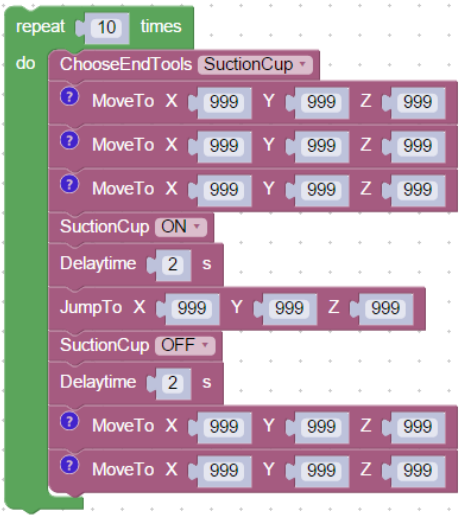


If your set up did not work correctly the first time, what did you have to do to make it work?

Next we are now going to make the program loop, or restart, for a certain number of times. To do this we will be using the ***repeat*** block and the *while* loop, both can be found in the *Loops* section on the left.



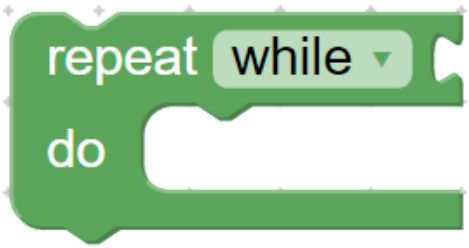
The ***repeat*** block is very easy to use. Drag the ***repeat*** block into a program environment. It can be dragged into the empty space or dragged directly onto the top connection for the ***ChooseEndTools***. If the Repeat loop is dragged onto the empty space, grab the top line of the existing program and pull it into the repeat loop. The ***repeat*** loop will automatically expand to encompass the entire code. The program will now run for the set number of times specified in the repeat.



Run the program again. To ensure the *Repeat Loop* works as designed. You will need to manually replace the cube or cylinder at the start of each of the loops.

If your set up did not work correctly the first time, what did you have to do to make it work?

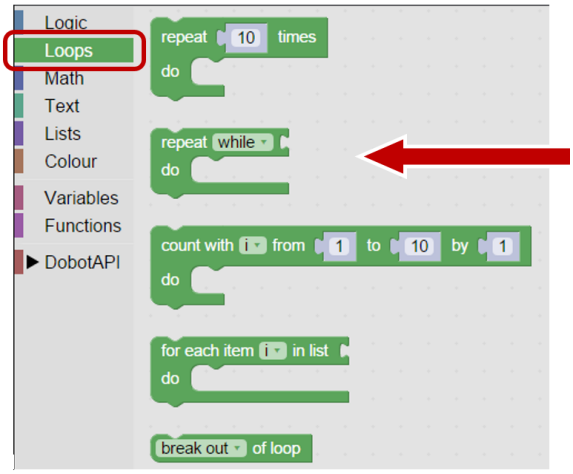
Many times in programming, we need to add a *CONDITION* to our loops. A condition is the scenario that must be met to start a loop. Many times we need something to keep repeating until we stop the program, this type of loop is considered a forever loop and can be made using *while loops.* To create a forever loop for this program, we will add a loop that needs a condition instead of just having it repeat a set number of times.



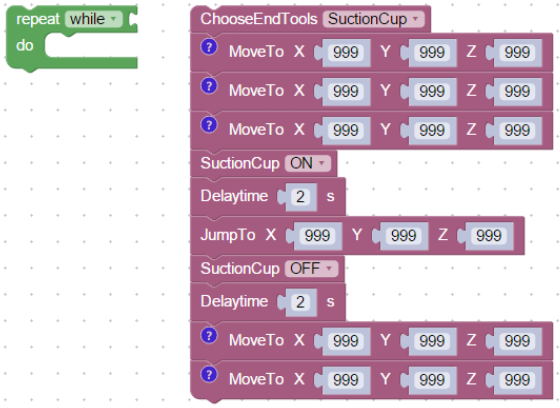
In order to do this, first drag the code out of the ***repeat*** block and then delete it.

*If you select the **Repeat** Loop to be deleted before the program section is pulled out of the loop, it will delete the loop and everything it contains.*

Go to the *Loops Section* and drag the ***Repeat While*** loop over and drag the rest of the program into it.

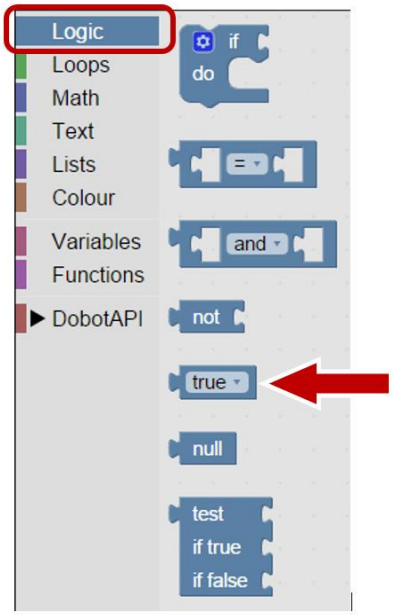


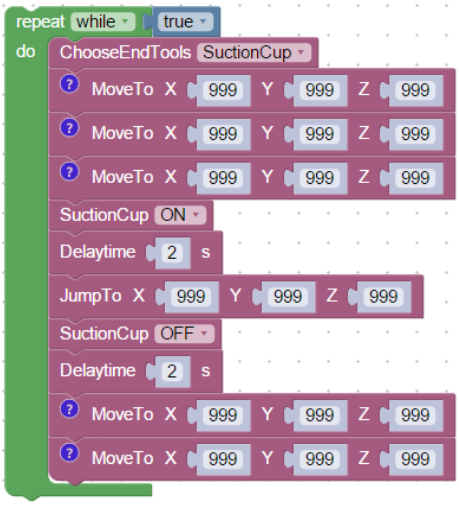
Repeat While statements are different than ***Repeat*** a number of times as a Repeat while requires a condition to be true or false to determine if they run or not, and the Repeat Times only requires a number of times to repeat/loop; no conditions must be met.



A *while loop* will run until it's condition is false so in order to run something forever, the statement must always be true. For example, 1 = 1, 4<5, 100>3, 1 is not 2. These statements will always be true no matter what happens and we could use these for our while loop but there is an easier way. We can simply have the statement be *True*, as True can never be False.

In the *Logic Section*, you will see a ***True*** block. Drag this over to the puzzle link next to the top of the while loop and it should snap into place. This will cause the loop to run forever as statement can never be false





Once again, run the program and you should see that once the program drops off the block and returns to it's safe position, it goes back to the start of the program to try to pick up another block and that this will continue repeating indefinitely until you hit the stop button or the robot is turned off.

If your set up did not work correctly the first time, what did you have to do to make it work?

CONCLUSION

1. *What is the difference between a forever loop and a repeat loop?*

2. *Can a loop be placed inside another loop? Give an example of how you might use this when programming in Blockly.*

3. *Describe in your own words how a JumpTo command works. Why is it good programming practice to use JumpTo's?*

GOING BEYOND

Finished early? Try some of the actions below. When finished, show your instructor and have them initial on the line.

_____ 1. Nesting Loops

Produce 2 different pick and place routines

Routine 1 will repeat three times

Then routine 2 will repeat twice

This process will loop routine 1 and then 2 forever

_____ 2. Produce a repeat loop for a dipping operation. Use the JumpTo when appropriate