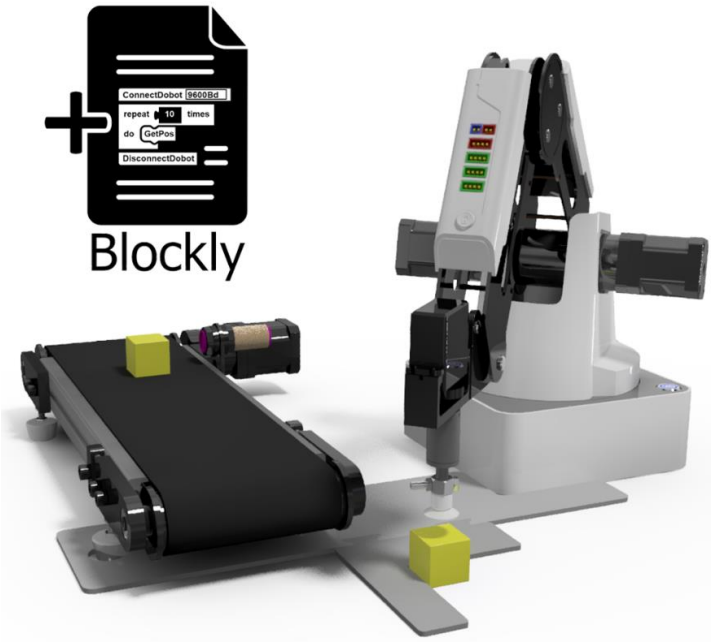
6 Blockly - Start & Stop Conveyor

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

INTRODUCTION

Robotic arms need to communicate with each other as well as other peripherals such as conveyor belts or linear rails in order to move materials or products through stages of a work cell.

In this activity you will learn how to program a robot to control a conveyor belt. We will use an Infrared Sensor to create a closed loop system.



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

- Stepper Motor
- Conveyor Belt
- Output
- IR - Infrared Sensor

EQUIPMENT & SUPPLIES

- Robot Magician
- Dobot Field Diagram
- 1” cylinders or cubes
- Dobot Conveyor
- DobotStudio software
- Suction Cup Gripper
- *Dobot Input/Output Guide*
- IR Sensor

ESSENTIAL QUESTIONS

Essential questions answered in this activity include:

- Is the conveyor an input or an output to the robot?
- Is the Infrared sensor an input or an output?
- How do I wire the components together?
- What can I do with the infrared sensor?
- How do I code the conveyor in blockly?
- How do I code the infrared sensor in blockly?

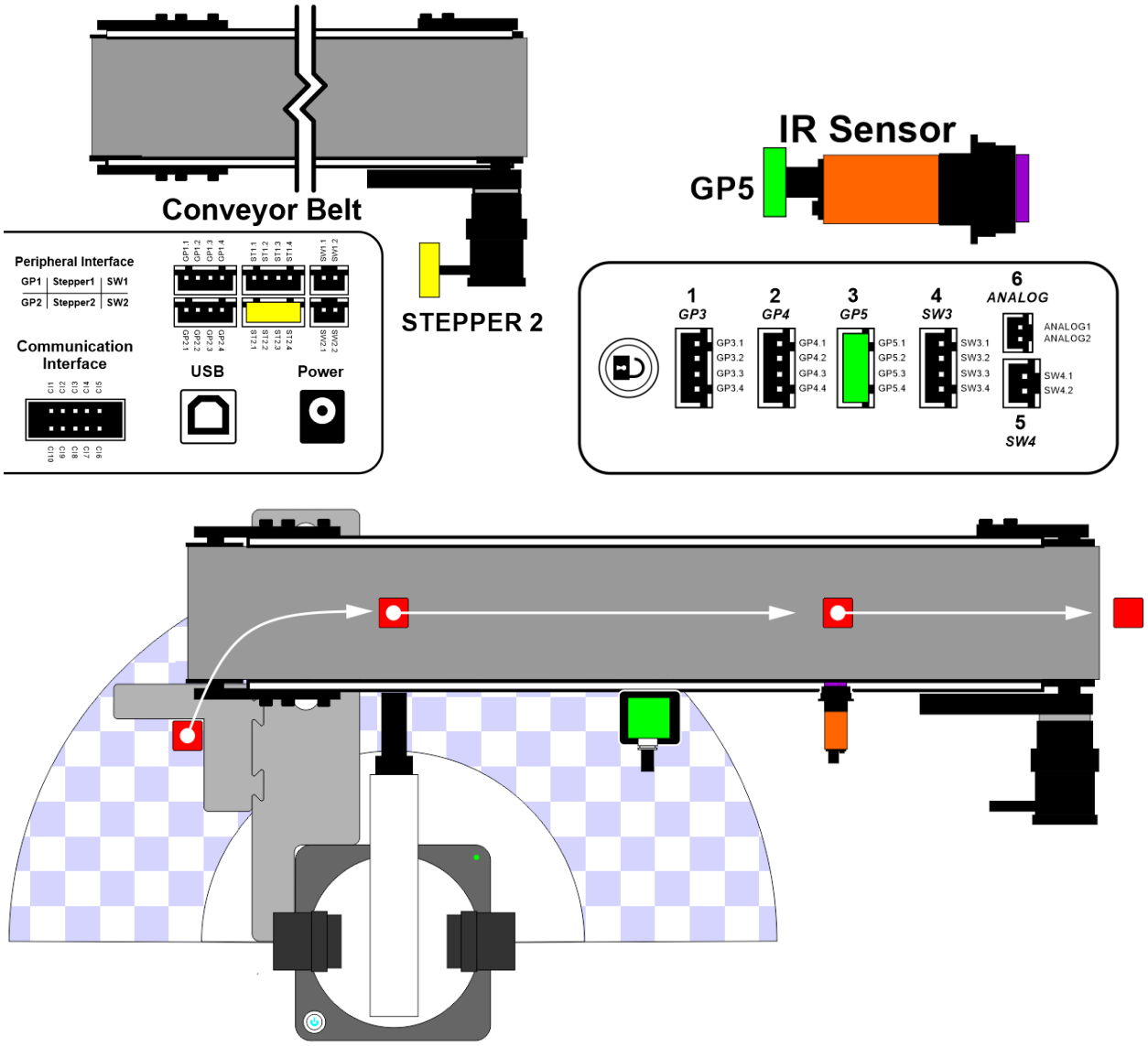
PROCEDURE

Order of operations

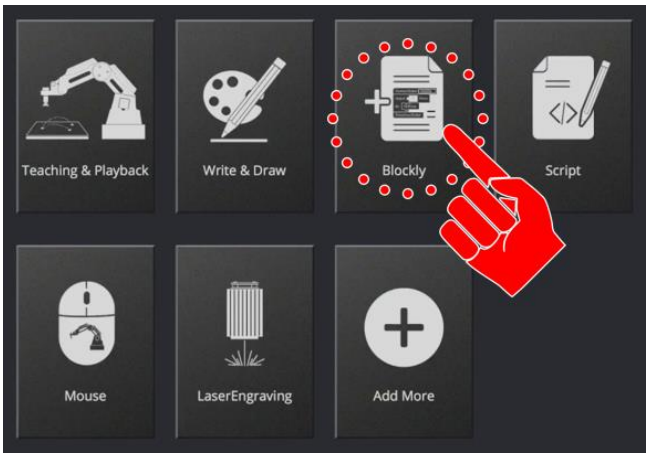
- The Robot will pick up a cube from a known location and place it on the conveyor belt.
- The Robot will return home and then start the conveyor belt.
- The conveyor belt will run until the block arrives at the IR Sensor for inspection.
- The block will be manually removed from the belt, inspected and then returned to the belt.
- Once the block is returned to the belt, the belt will run again until the parts runs off the belt and into storage.
- The process will loop forever.

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. Set up the robot with the suction cup gripper.
2. Plug the Conveyor Belt into **STEPPER2** of Robot and plug the IR sensor into **GP5** of the Robot


Conveyor Belt
STEPPER 2
IR Sensor
GP5
Peripheral Interface
GP1 | Stepper1 | SW1
GP2 | Stepper2 | SW2
Communication Interface
USB
Power
1 GP3
2 GP4
3 GP5
4 SW3
6 ANALOG
ANALOG1
ANALOG2
5 SW4

Open up Blockly in the software

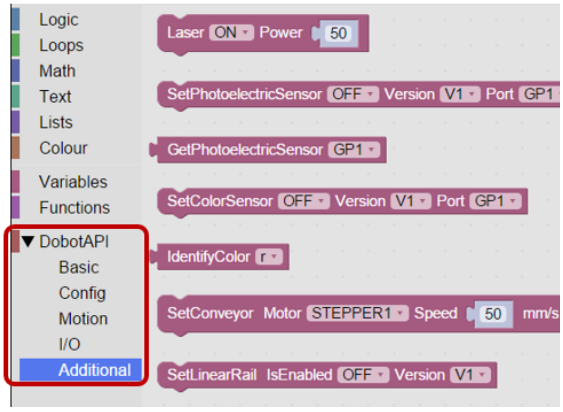


When you re-open this program check that the name of the file on top matches the code in the file, if it does not, you may end up overwriting another program

The first step will be to setup all of our INPUT and OUTPUT ports.

Drag over the ***SetPhotoelectricSensor*** block from the DobotAPI/Additional tool box

Set the sensor to ON / V2 / GP5

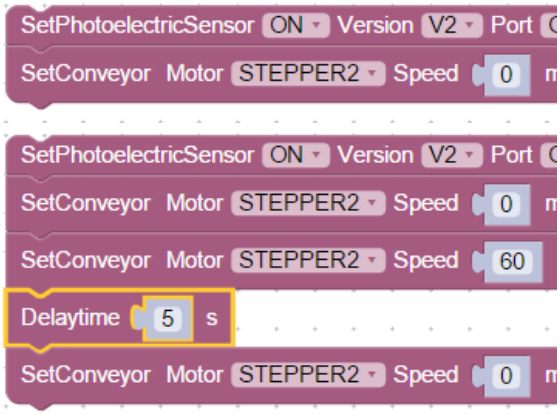


Drag over the ***SetConveyor*** from the DobotAPI/Additional tool box

Set the conveyor to Stepper 1 / 60 mm/s

Once the header code in established, we are going to write a quick code that will turn on the conveyor for 5 seconds

In this example, I have used 60 mm/s for the speed of the conveyor. Play with this value until you have a controllable speed for this activity.

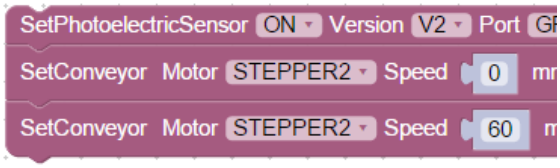


Once the program is completed, run it and see if it works correctly. If it does not work, troubleshoot it until it does.

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

We will now edit our current code to get the conveyor belt to stop when an object is detected by the IR Sensor.

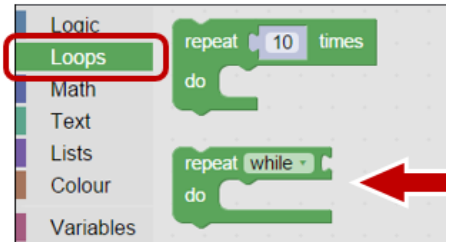
Remove the ***DelayTime*** from the program.



Create a loop that will keep checking the IR Sensor value until an object is detected

Bring over a ***RepeatWhile*** loop.

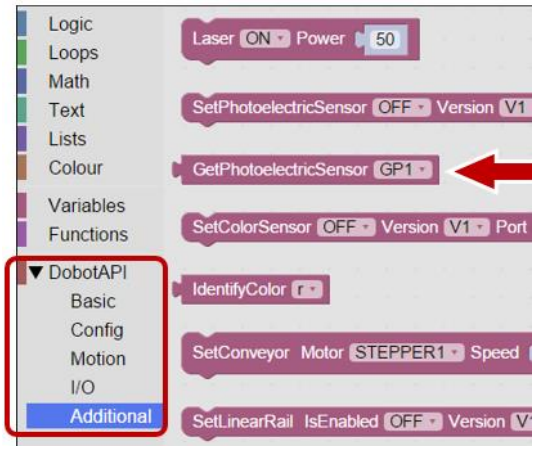
Change the loop into a ***RepeatUntil*** Loop



Create a *condition* that evaluates the IR sensors value. We are looking for the sensors value to change from 0 to 1.

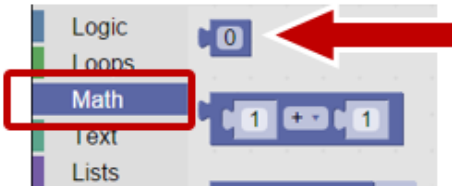
The IR Sensor reads true/high when an object is present. The small LED on the back of the IR Sensor will also light up.

Drag over the ***ReturnTrue*** condition from the Logic Toolbox.



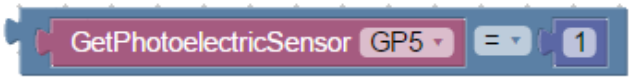
Drag over the ***GetPhotoelectricSensor*** from the DobotAPI/Additional Toolbox.

Drag over the ***Number Block*** from the Math Toolbox.



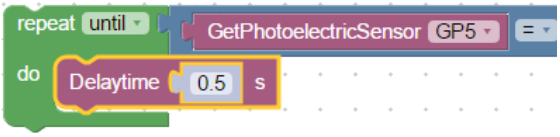
Add all three blocks together to complete the statement.

Make sure to change the ***GetPhotoelectricSensor*** value from GP1 to GP4 and the number value to 1



Add the *condition* statement to the ***RepeatUntil*** loop

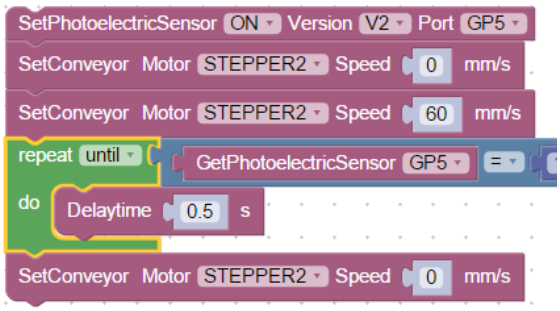
Add a small ***DelayTime*** into the ***RepeatUntil*** loop in order to give the operation something to process.



Put together the code that we have so far.

Run the program. Place a cube in front of the sensor. The conveyor should stop.

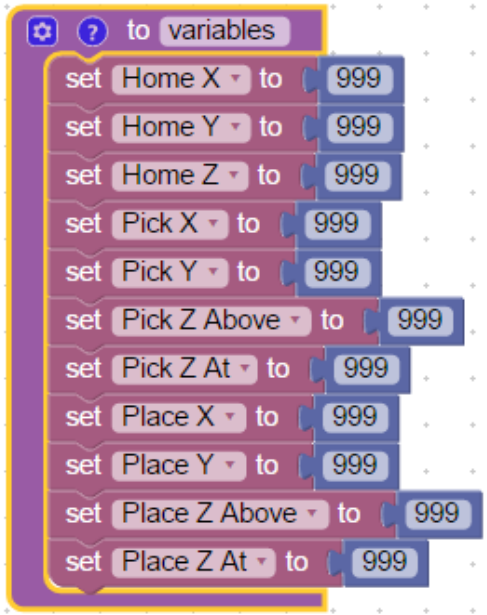
If it does not work, troubleshoot it until it does.



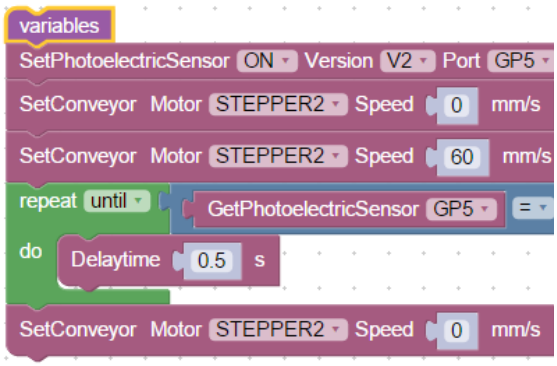
If your set up did not work correctly the first time, what did you have to do to make it work?

Now that we can start and stop the conveyor using the IR Sensor we can find our positions and set our variables for this activity.

Place the *variables* and positions in a function as has been done in previous activities.

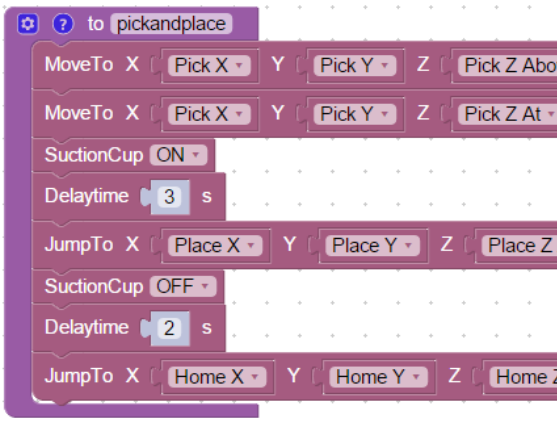


Add the *variables* ***Function*** to the beginning of your program

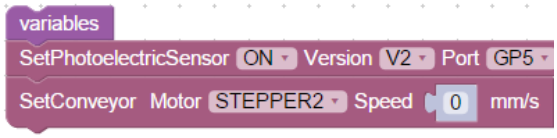


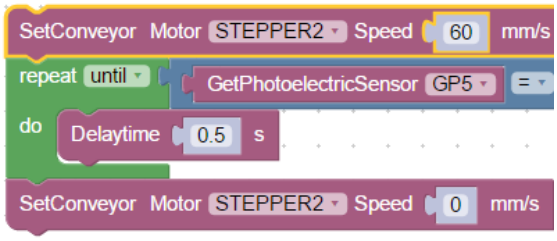
Create a ***Function*** for the Pick and Place operation.

- Pick up the cube
- Place it on the belt
- Return home



Pull the main program apart between the first two ***SetConveyor*** blocks.



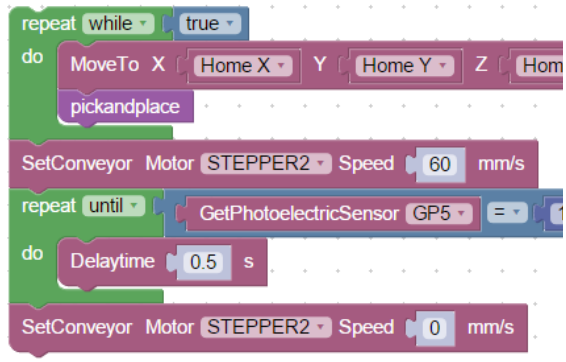


Drag over and create a ***Forever*** Loop.

Start developing the loop

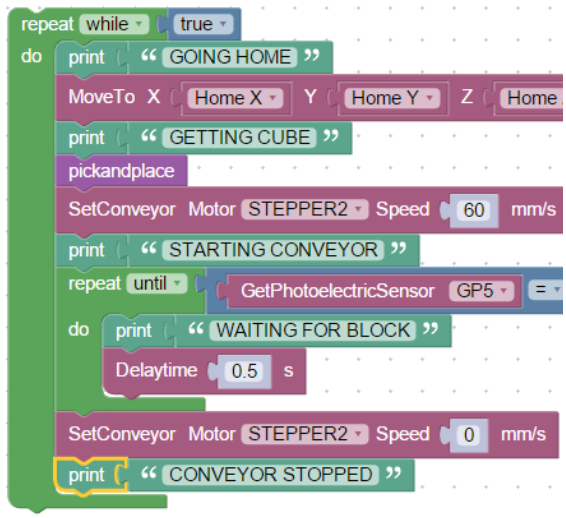
Consider the order of operations

- Start with the **robot at a Home** position
- **Pick and Place** - Get the block and put it in the belt and return to home.
- **Start the belt** and run it **until the cube reaches the sensor**
- **Stop** the belt

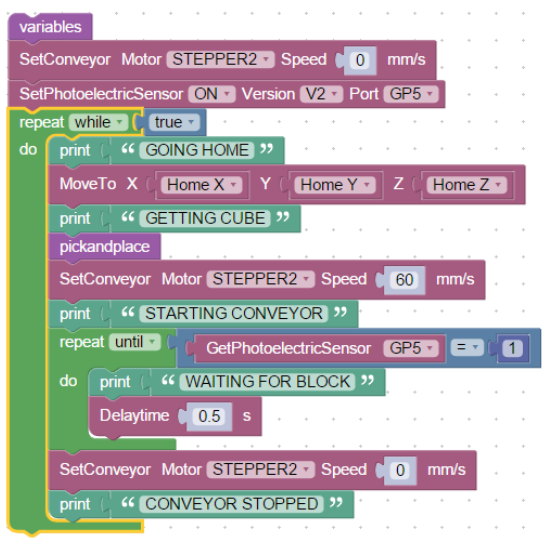


As the program starts getting more complex, it may help to start adding print commands to the code to be able to troubleshoot the code

Add them anywhere in the program where an operation may be changing or a value may be read.



Assemble the code that we have so far.



Once the program is completed, run it and see if it works correctly. If it does not work, troubleshoot it until it does.

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

Reminder:

This activity requires the following operations:

- The Robot will pick up a cube from a known location and place it on the conveyor belt.
- The Robot will return home and then start the conveyor belt.
- The conveyor belt will run until the block arrives at the IR Sensor for inspection.
- The block will be manually removed from the belt, inspected and then returned to the belt.
- Once the block is returned to the belt, the belt will run again until the parts runs off the belt and into storage.
- The process will loop forever.

The next step is to create a loop that will allow the block to be removed and then wait for it to be returned. The issue is that we have no idea when the block has been removed, how long it will take for the inspection, or when it will be returned.

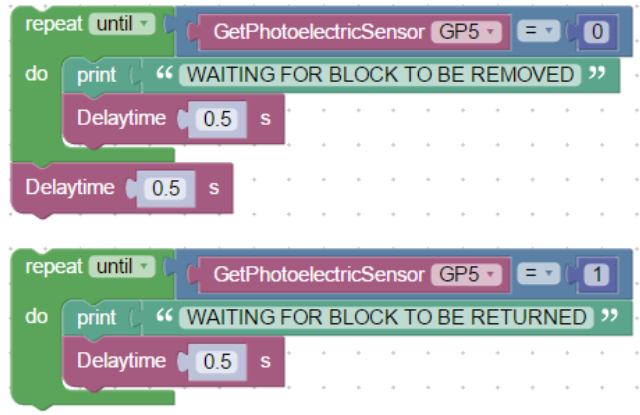
We need to create a closed loop system that will look for the following conditions without respect to time.

The Block has been REMOVED
The Block has been RETURNED

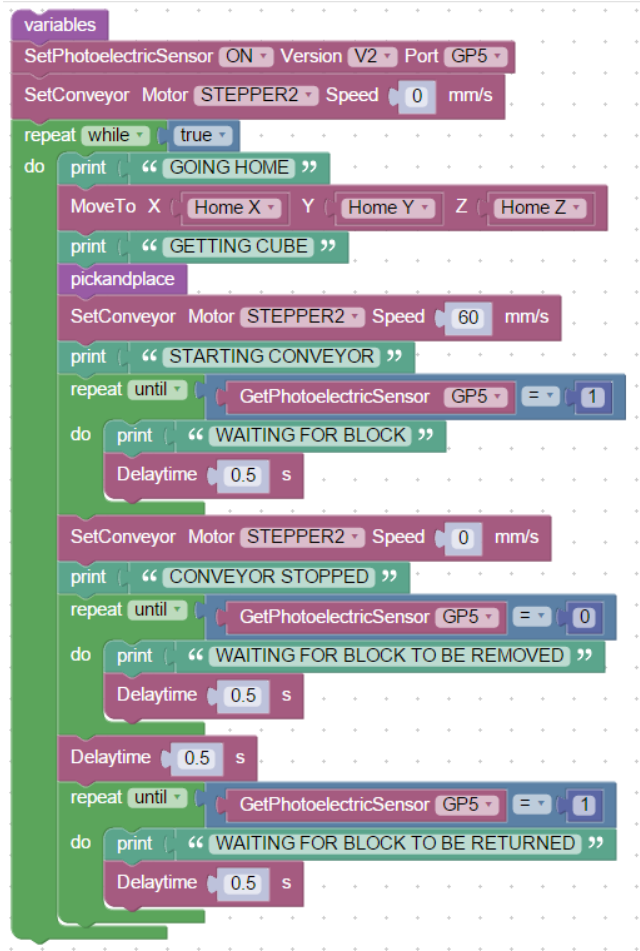
These two conditions can be developed using two separate ***RepeatUntil*** Loops just as we did for stopping the conveyor.

One that will wait for a '0' and one that will wait for a '1'

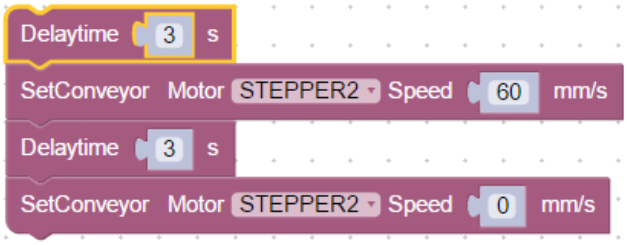
Place a small **DelayTime** in between the two loops to keep them separate and then add them into our main loop



Some of this program can be condensed into separate functions in order to simplify the main program.



Our final task is to start the conveyor again and run the part off the end and into a container. Use the same code you started with to accomplish this task. The time needed to run the part off the conveyor will be different for each project, depending on where the sensor is mounted.



Place a small ***DelayTime*** in front of the code to allow time for the user to get their hand away from the belt before it starts running.

Add this last group inside the end of your Forever Loop.

```
to variables
  set Home X to 196
  set Home Y to 0
  set Home Z to 93
  set Pick X to 168
  set Pick Y to 168
  set Pick Z Above to 40
  set Pick Z At to -44
  set Place X to 264
  set Place Y to 34
  set Place Z Above to 44
  set Place Z At to 17
```

```
to pickandplace
  MoveTo X Pick X Y Pick Y Z Pick Z Above
  MoveTo X Pick X Y Pick Y Z Pick Z At
  SuctionCup ON
  Delaytime 3 s
  JumpTo X Place X Y Place Y Z Place Z At
  SuctionCup OFF
  Delaytime 2 s
  JumpTo X Home X Y Home Y Z Home Z
```

```
variables
SetPhotoelectricSensor ON Version V2 Port GP5
SetConveyor Motor STEPPER2 Speed 0 mm/s
repeat while true
do print "GOING HOME"
   MoveTo X Home X Y Home Y Z Home Z
   print "GETTING CUBE"
   pickandplace
   SetConveyor Motor STEPPER2 Speed 60 mm/s
   print "STARTING CONVEYOR"
   repeat until GetPhotoelectricSensor GP5 = 1
   do print "WAITING FOR BLOCK"
      Delaytime 0.5 s
   SetConveyor Motor STEPPER2 Speed 0 mm/s
   print "CONVEYOR STOPPED"
   repeat until GetPhotoelectricSensor GP5 = 0
   do print "WAITING FOR BLOCK TO BE REMOVED"
      Delaytime 0.5 s
   Delaytime 0.5 s
   repeat until GetPhotoelectricSensor GP5 = 1
   do print "WAITING FOR BLOCK TO BE RETURNED"
      Delaytime 0.5 s
   print "PREPARING TO MOVE - STAND CLEAR"
   Delaytime 3 s
   SetConveyor Motor STEPPER2 Speed 60 mm/s
   print "SENDING PART TO STORAGE"
   Delaytime 3 s
   SetConveyor Motor STEPPER2 Speed 0 mm/s
```

Once the program is written, try it and make sure that it works as expected. If it does not work correctly, troubleshoot until it does.

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

CONCLUSION

1. *Why is it better to have the infrared sensor stopping the conveyor belt rather than just running it for time?*

2. *How would the program be different if the conveyor belt could not be run as a linear rail?*

3. *What's one way to determine where the robot is at any given time in the program?*

GOING BEYOND

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

_____ 1. Use functions to make your program as short as possible.

_____ 2. Use the color sensor and make the robot report what color block is being sent down the conveyor.