Calibrate Distance and Square (Straight on 'til Morning)

Second star to the right and **straight on 'til morning**

Time required: 60 minutes

NOTE: The mBot is not an accurate robot. As the batteries discharge and the conditions change, it will behave differently. We can change power and time. Just try to get close. We will use this program to calibrate our mBot for future programs.

This program will allow us to calibrate our mBot for the following:

1. Adjust the **COMP** compensation factor percentage for the robot to go straight.
2. Distance by adjusting the **time** constant to drive 48”.
3. Adjust turns by adjusting **turnTime360** to have the mBot start and start in approximately the same orientation while making a square.

Debouncing

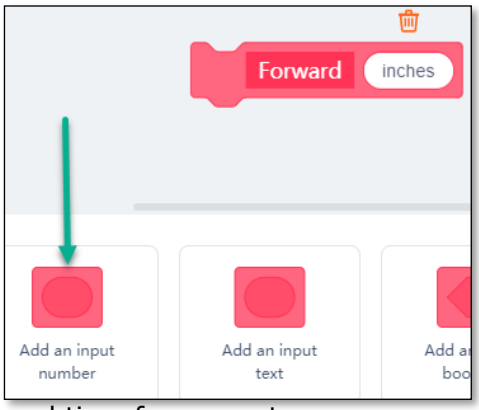
Putting in a short wait allows for more accuracy scanning for the remote signal and movement. A small wait while changing directions allows the robot to temporarily stop, then turn more.

When you press a button on the remote, it is impossible to press it once. It makes contact several times, bouncing off the contact. It may make contact on one side, then the other, then it settles down. The mBot scans for each ir code so fast that it can mistake one code for another. Debouncing is putting a tiny delay in between scanning for the ir codes to ensure smooth operation.

Code Blocks with Arguments/Parameters

Code blocks with arguments allow us to create reusable code that will behave differently depending on what we feed them. Arguments/parameters allow us to change the function of the block based on the information we pass to the block.

1. To make a code block, go to **My Blocks**, Click **Make a Block**.
2. Name the Block **Forward**.
3. Click Add an input number. Type **inches**. Click **OK**.
4. To use the block, go to **My Blocks**, drag the block to where you want to run it.



We will determine how much time it takes your mBot to move a certain distance at a certain power. We can input that distance and time for accurate movement.

avgSpeed (inches per second) = (Distance(inches) / Time)

Example: (48/7.4) = 6.5 inches per second

Tutorial Assignment

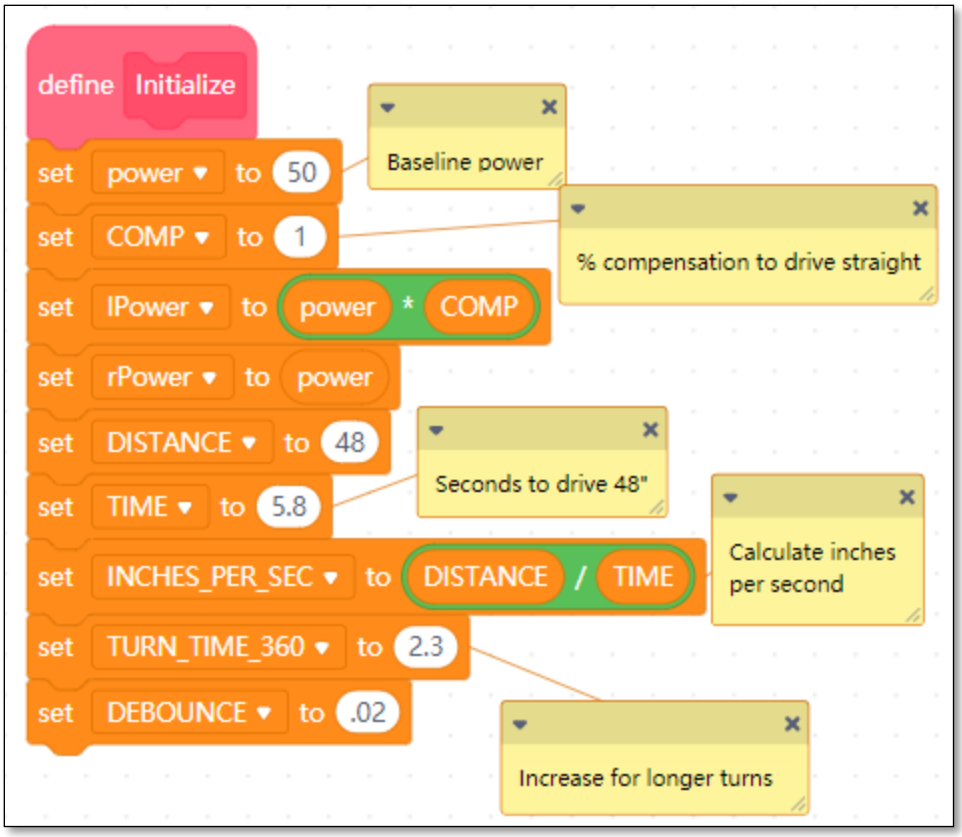
1. Start mBlock. Save the program as **Calibrate Distance and Square**.
2. Complete and test the sample program and add the requirements listed.

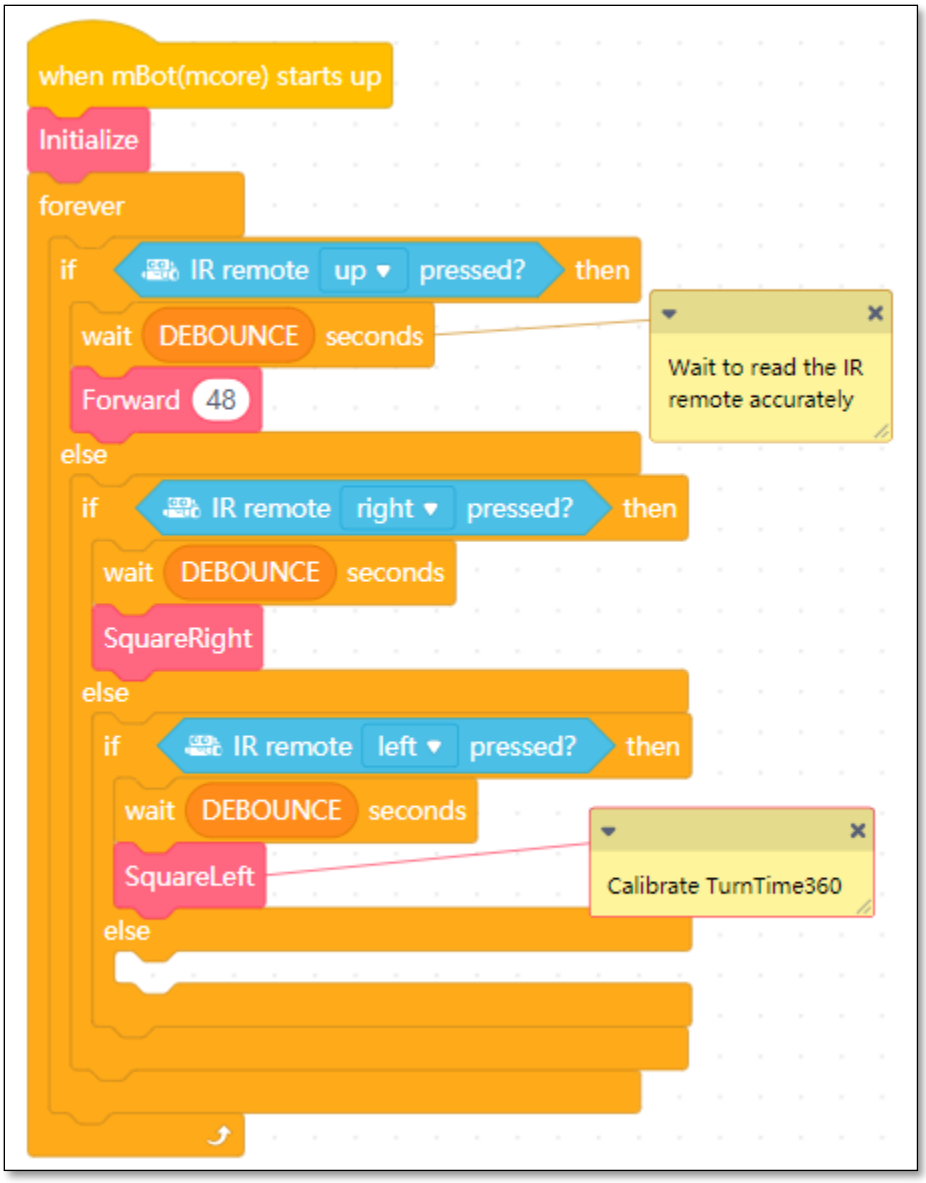
Requirements

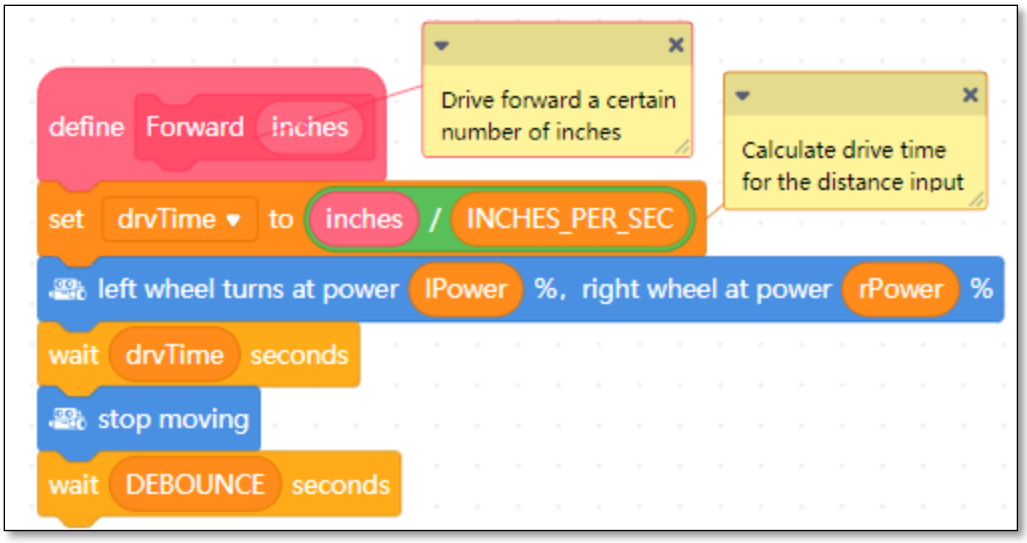
NOTE: The mBot will never be able to do these requirements perfectly. Just get close.

Create a program with the following blocks each triggered by a different remote button.

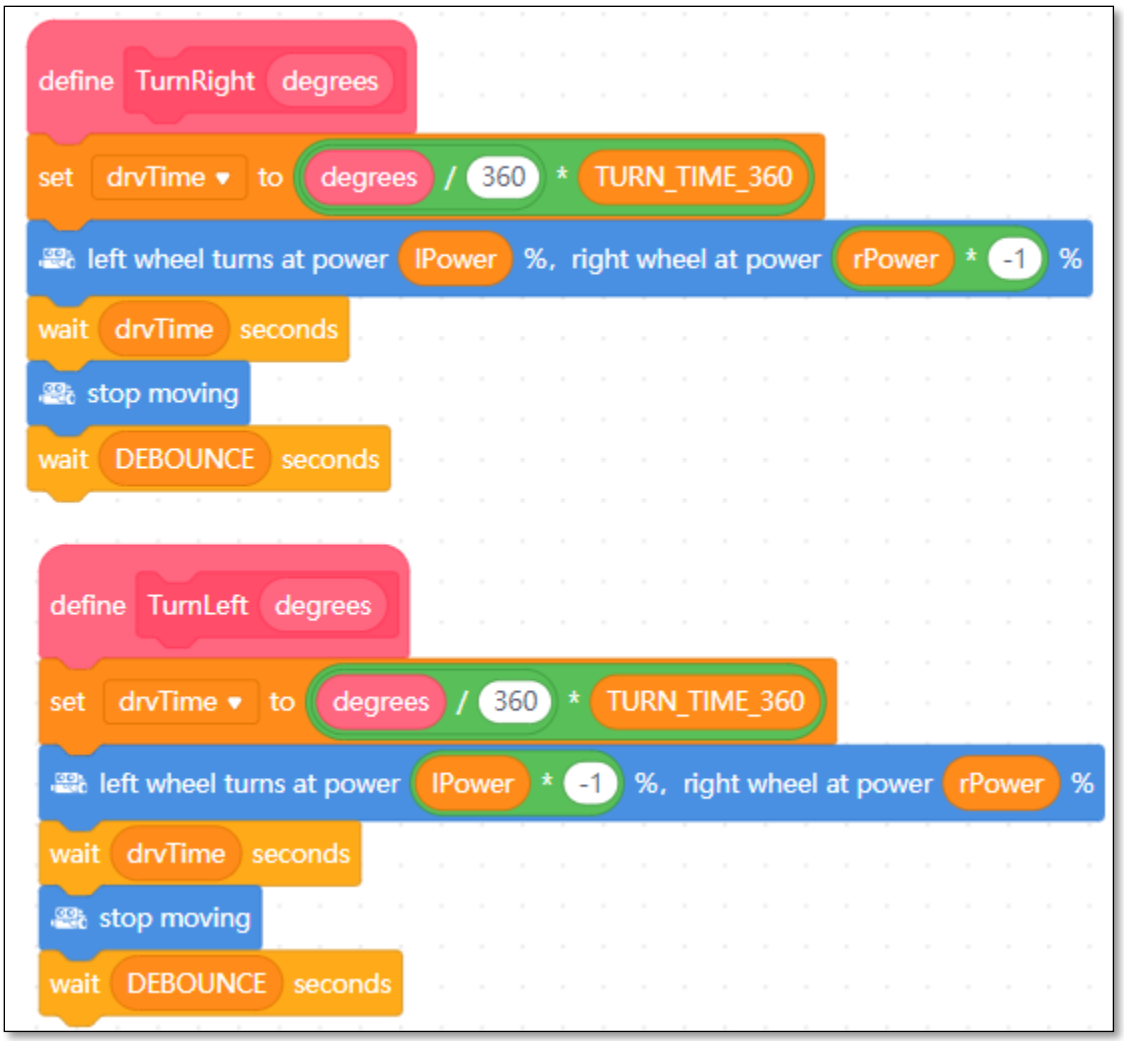
- A block that moves forward 48”.
    a. If your mBot turns to the left, adjust **COMP** to 1.05 to start. Keep adjusting the COMP until your mBot runs fairly straight.
    b. Adjust **time** until the robot moves forward 48”.
- A block that does a square turning to the right.
- A block that does a square turning to the left.
    a. Adjust the **TurnTime360** until the right square and left square are more or less the same.

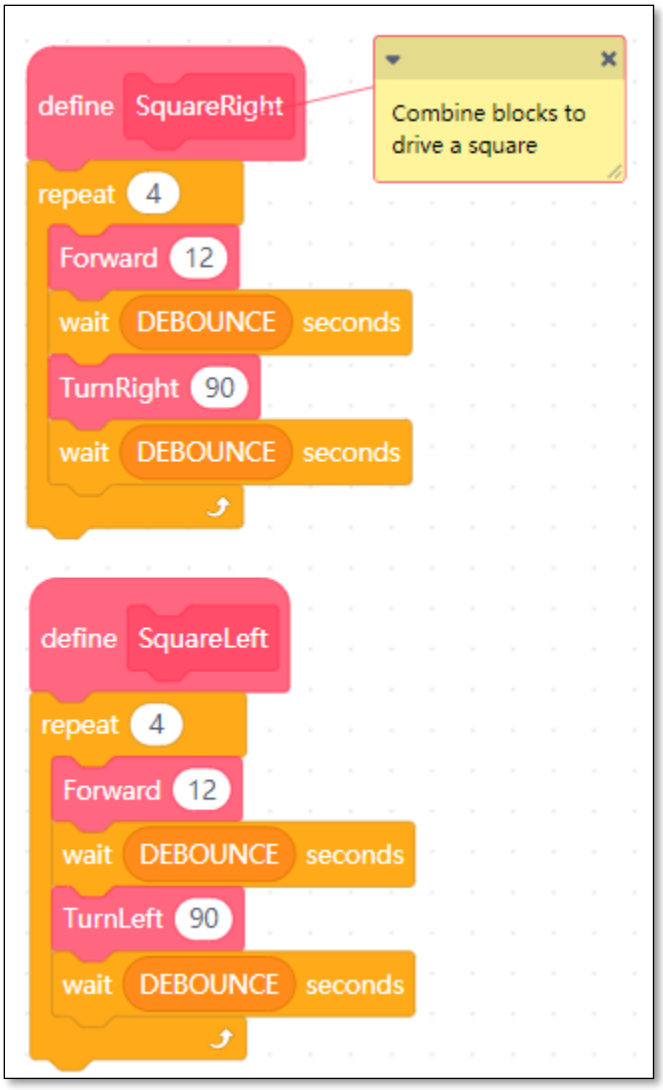

define Initialize
set power to 50
Baseline power
set COMP to 1
% compensation to drive straight
set IPower to power * COMP
set rPower to power
set DISTANCE to 48
set TIME to 5.8
Seconds to drive 48"
set INCHES_PER_SEC to DISTANCE / TIME
Calculate inches per second
set TURN_TIME_360 to 2.3
set DEBOUNCE to .02
Increase for longer turns


when mBot(mcore) starts up
Initialize
forever
if
IR remote
up
pressed?
then
wait
DEBOUNCE
seconds
Forward
48
Wait to read the IR remote accurately
else
if
IR remote
right
pressed?
then
wait
DEBOUNCE
seconds
SquareRight
else
if
IR remote
left
pressed?
then
wait
DEBOUNCE
seconds
SquareLeft
Calibrate TurnTime360
else



NOTE: Notice that in the set drvTime to: **degrees / 360** is grouped together.





Assignment Submission

- **All students** → Attach finished programs to the assignment in Blackboard.
- **In class assignment submission** → Demonstrate in person.
- **Online submission** → A link to a YouTube video recording showing the assignment placed in the submission area in BlackBoard.