# LOST IN SPACE

## MISSIONS

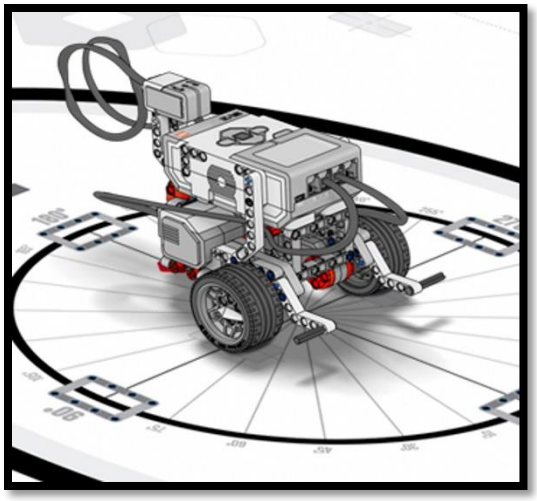
Your robotic spaceship has been hit by an asteroid. You must carry out a series of repairs and upgrades in order to survive and return home.

## CHALLENGE A:
### SPIRALING OUT OF CONTROL

**Your ship was struck by an asteroid and is spinning out of control. Use your gyro sensor to detect your movement and point your ship back to its original heading.**

*Specifics: You will place your robot on the turntable and start your program. When the table is spun in either direction, your robot should react by turning back to the direction it was before it began turning (0 degrees).*

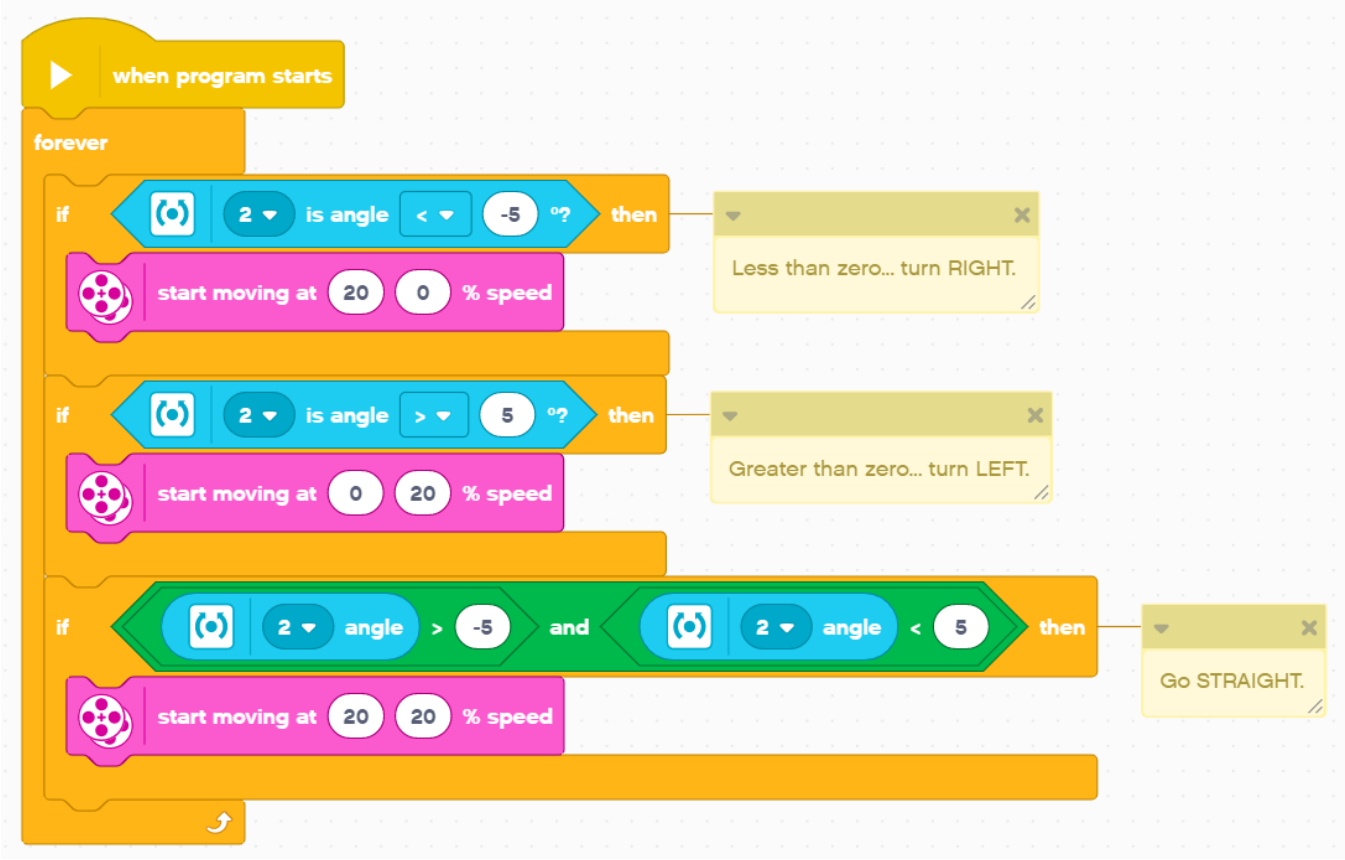
Planning:



- How will you compare your robot's current angle to its starting angle? (greater than / less than comparison blocks)
- What two possibilities might happen in this mission? (your angle is less than zero and you should turn left -OR- your angle is greater than zero and you should turn right)
- What type of turn would be best if you want to stay in your current position? (spin turn)
- You might want to allow for a small range of numbers where you will stop turning. (-5 to +5)

Practice:

Try making a code to drive in a straight line using the gyro sensor. Because you want to move forward the entire time, you will want to use a pivot turn instead of a spin turn for this example. When driving, move the back of the robot side to side to see it auto-correct its direction.




## CHALLENGE B:
## FINDING THE CLOSEST PLANET

**Your ship's navigation system was knocked out by the asteroid. You must quickly find the closest planet and land there to do repairs. Spin one time around while using your ultrasonic sensor as a radar system to find the closest planet. Once you find which planet is closest, you will fly to that planet and stop.**

*Specifics: The planets (blocks) will be placed randomly around your ship each time you test. You will use a gyro to control your spin, the ultrasonic sensor to check for distance, and at least two variables to keep track of the numbers you see (distance and direction). Create the two variable (distance and direction).*

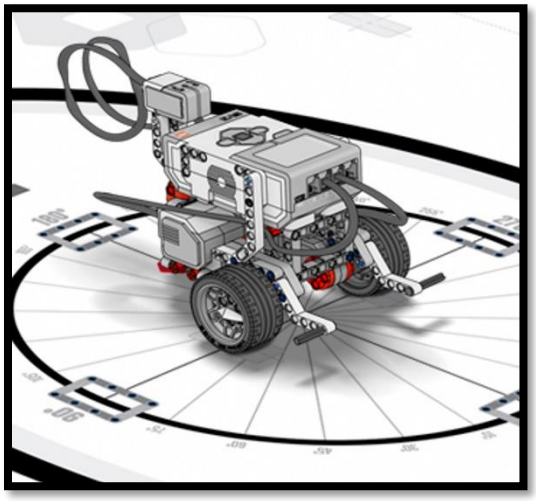
*Start by setting the distance variable to 100 and the direction variable to 0. This is because your ship is starting at 0 degrees direction and the farthest the ultrasonic sensor can see is 100 inches.*

*Turn in small, controlled amounts. Each time you stop, check the ultrasonic sensor's distance value. Compare it to the value you have stored in the distance variable. If it is smaller than what is currently in the distance variable, you should write this new ultrasonic distance into the distance variable. You should also write the ship's current direction (the gyro's degrees) into the direction variable so you can remember where you found this planet. If your ultrasonic value is not smaller than what is written in your distance variable, you do not need to do anything with your variables.*

*Keep turning (and updating your variables) until you have turned in a complete circle (360 degrees).*

*Your variables should now be holding the distance to the smallest planet and the direction it is in. Turn your robot until you are facing that direction. Use the distance variable (how far away the planet is in inches) to drive forward the correct number of rotations, then stop. Print, "You have arrived at your destination." on the screen.*

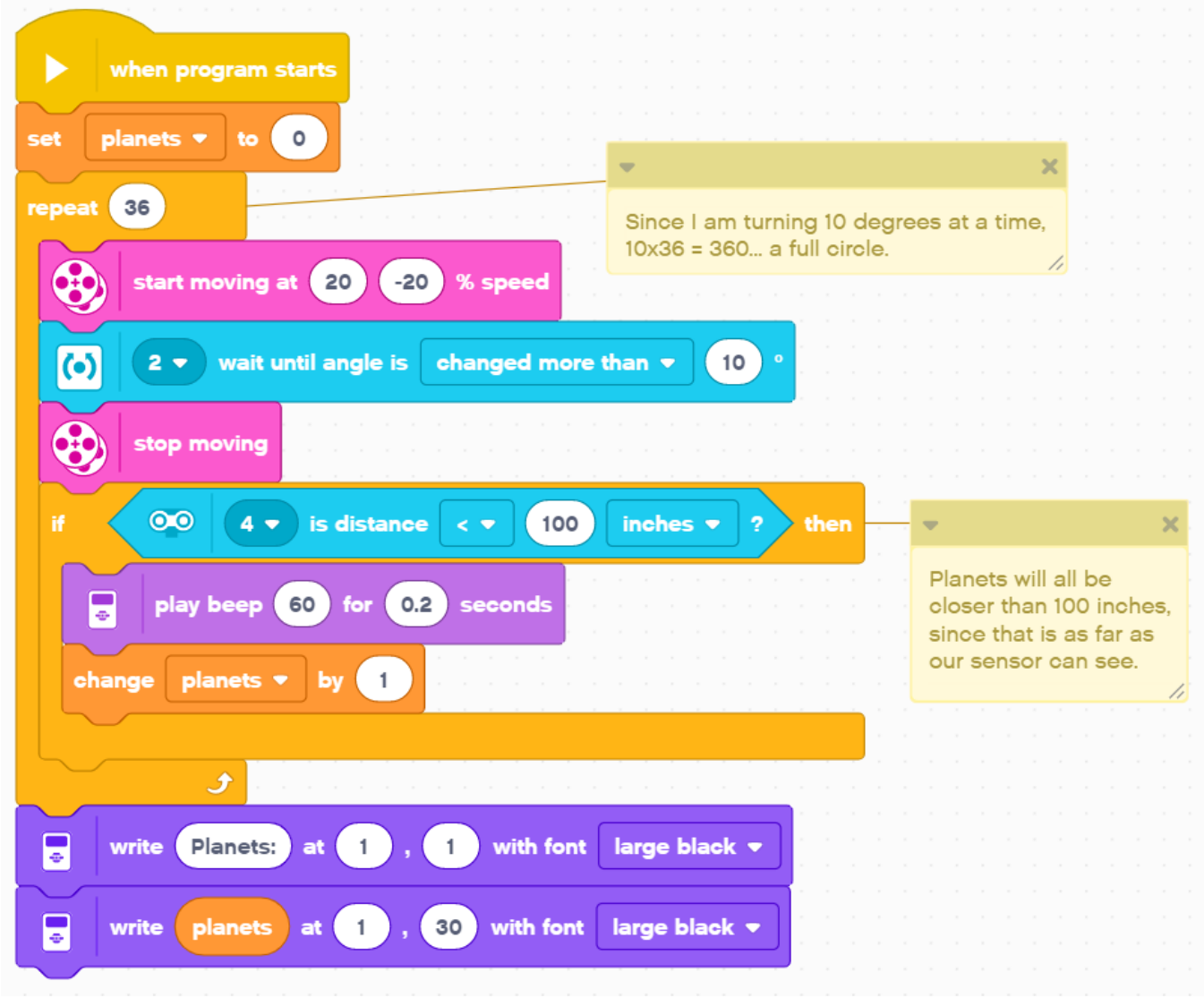
Planning:



- A variable is just a "storage container" that holds a number. You can name them anything you want.
    - You can SET the variable to any value with the SET TO block.
    - You can READ the value stored in a variable with the READ VARIABLE (oval) block.
    - You can use the CHANGE BY block to increase or decrease the number stored in the variable by a specific amount.
- Why do you need to use variables for this mission? (because the numbers your sensors will see are going to keep changing and you need to remember what you saw before)
- Why do you need to turn in a complete circle before you head to the nearest planet? (because until you have turned all the way around, your ship won't have seen where all of the planets are located or how close they are)
- How much should you turn before checking the ultrasonic sensor's distance? (you might try 10 degrees at a time, or if that is not accurate enough change to 5 degrees at a time)
- Which direction should you turn? (turning right from 0 to 360 would probably be easiest, then you can turn back to the left to line up with the closest planet)
- How far should you drive to get to the closest planet? (the wheels are 7 inches around, so divide your distance variable amount by 7 to know how many rotations to move forward)

Practice:

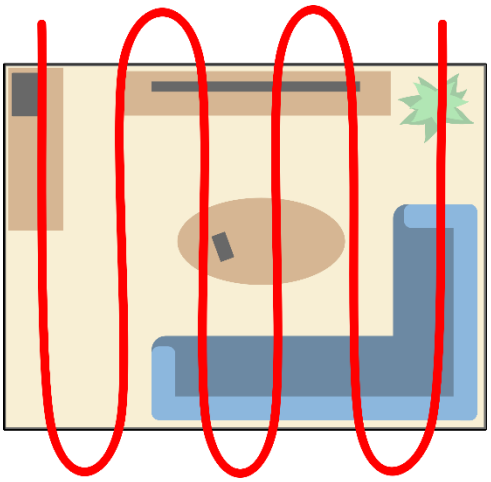
Try making a code to count the number of planets around your ship. You will need to create a variable called "planets" to keep count of how many blocks your ship sees. Set it to zero at the beginning, then start turning in 10 degree steps. (You will have to turn 36 times to make a complete circle.) Each time your ultrasonic sensor sees a value less than 100 inches, change your variable by +1. When you have finished turning in a circle, print the number of planets you counted on the screen.




## CHALLENGE C:
## SEARCH FOR LANDING SITE

**You fly across the strange planet looking for a safe place to land so that you can complete repairs. Complete a linear search pattern until your color sensor detects a flat landing site. Then land completely in that site.**

*Specifics: Your ship has narrowed down the search area to a spot outlined in blue. You should not search outside of this area. The flat landing site will be detected as red.*

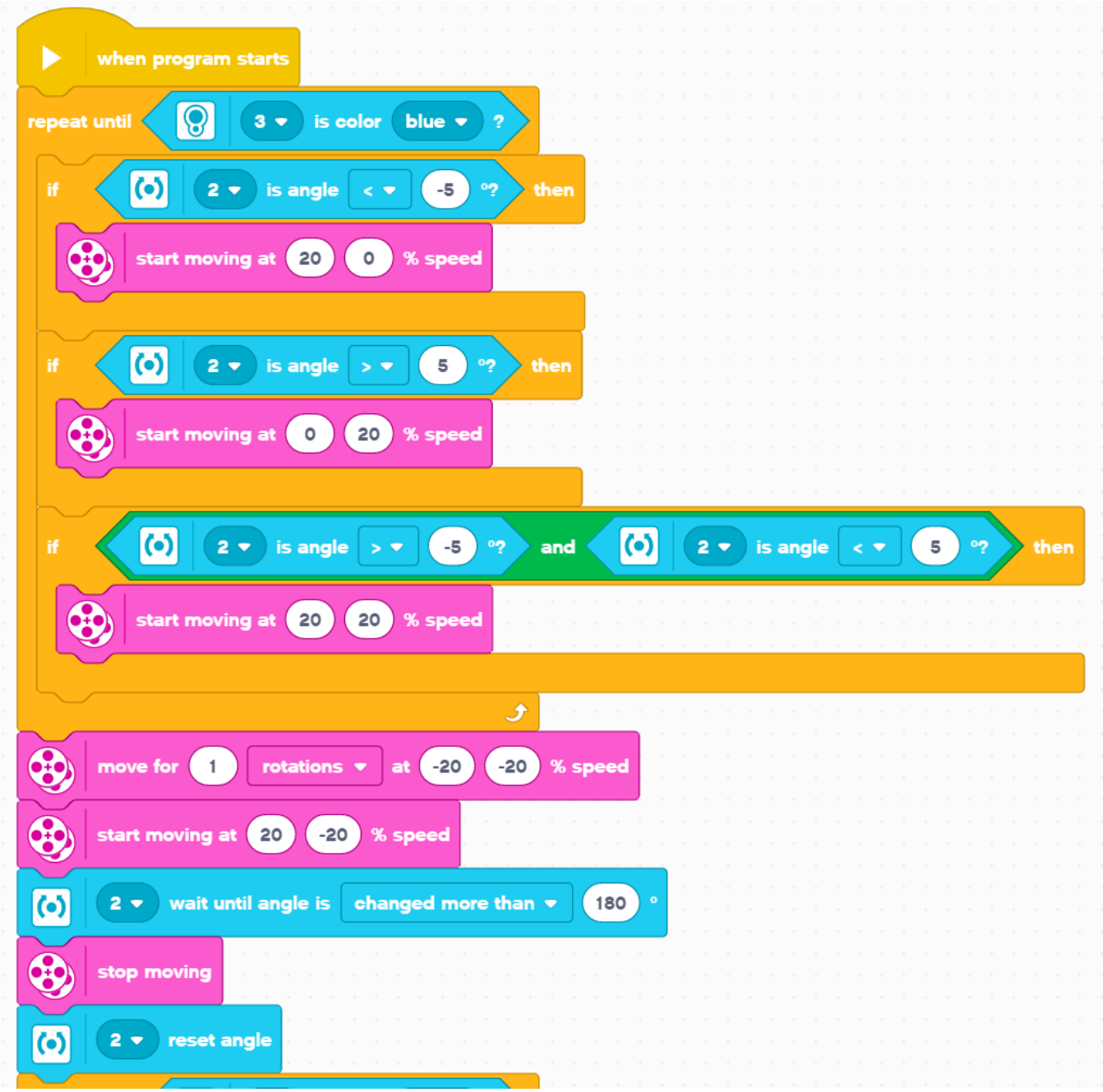


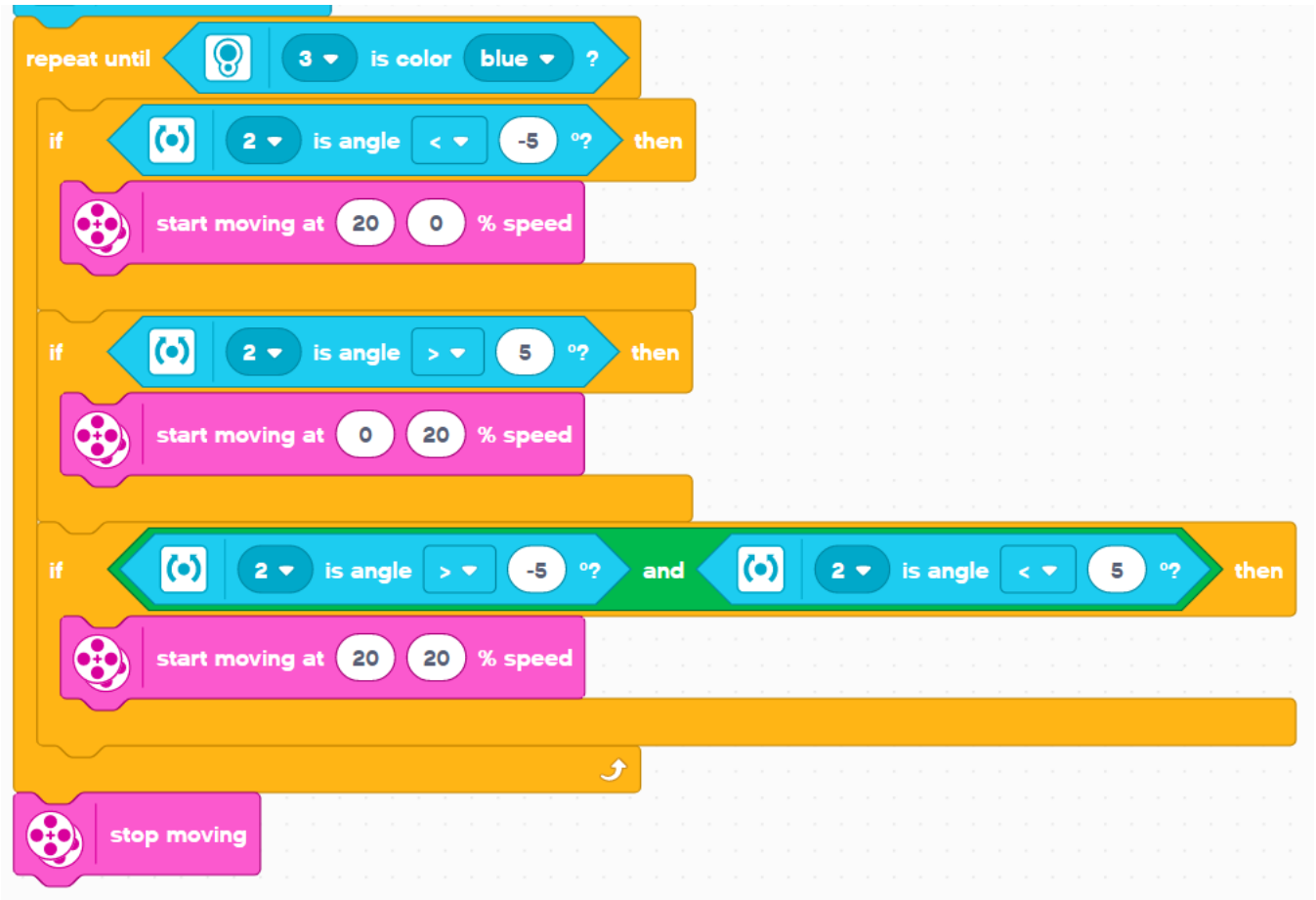
Planning:

- Should you randomly search the area for the landing site? (no, you might run out of fuel before finding it… use a back and forth pattern to cover the area)
- Before you begin searching, you should program your ship to recognize the search area (tan), boundaries (blue), and landing site (red).
- Should you use color values or reflected light values? (you can try color values, but if they are not accurate then switch to reflected light as it is more precise)
- Calibrating your sensor to the brightest and darkest reflected light percent might help your search.

Practice:

Try making a code to fly from one end of the search area and back. Before finishing your code, you must find out if color values will work or if you need to use reflected light values.






## CHALLENGE D:
## FIX THE RADIO TRANSMITTER

**You discover that your ship's communication equipment was damaged by the asteroid collision. Create a system for sending MORSE CODE (a system of short and long beeps called dots and dashes) so that you can transmit a message back to Earth.**

*Specifics: Use a touch sensor to create the dots and dashes. Each dash should be held 3 times as long as a dot. Pause between each letter in a word for the same time as you would hold a dash. Pause between different words for the same time as you would make two dashes. You will need to "write out" your message and practice delivering it before making your official transmission.*

Planning:

- Whenever you press the touch sensor, a tone should start.
- Whenever you release the touch sensor, the tone should stop.
- You should write each letter's dot and dash pattern on separate lines with WAIT written between letters.

MORSE CODE CHART:

| | | |
|---|---|---|
| A ● − | J ● − − − | S ● ● ● |
| B − ● ● ● | K − ● − | T − |
| C − ● − ● | L ● − ● ● | U ● ● − |
| D − ● ● | M − − | V ● ● ● − |
| E ● | N − ● | W ● − − |
| F ● ● − ● | O − − − | X − ● ● − |
| G − − ● | P ● − − ● | Y − ● − − |
| H ● ● ● ● | Q − − ● − | Z − − ● ● |
| I ● ● | R ● − ● | |

Practice:

When sending the message “GOT IT”, you should write it out like this:

− − ● WAIT (G)

− − − WAIT (O)

− WAIT (T)

WAIT

● ● WAIT (I)

− WAIT (T)

Try sending this message to another student and see if they can tell the difference between your dots and dashes, between the wait time between letters and the wait time between words.

# CHALLENGE E:
## DESTROY THE ALIEN COMMUNICATIONS TOWER

**Your transmissions have attracted the attention of some local aliens. They are preparing to call for a horde of reinforcements to protect their outpost on this planet unless you can destroy their communications tower in time. The tower is located behind a mountain range. A river runs around the base of the mountain, which would be easy to follow. When you reach the lake, look for the tower, then knock it down.**

*Specifics: Send your ship's rover around the mountain by following the river. Use your color sensor and a line-follower program which should repeat until you reach the lake.*

*Once at the lake, begin scanning for the tower using an ultrasonic sensor. Once located, make sure it is knocked down before returning to your ship.*

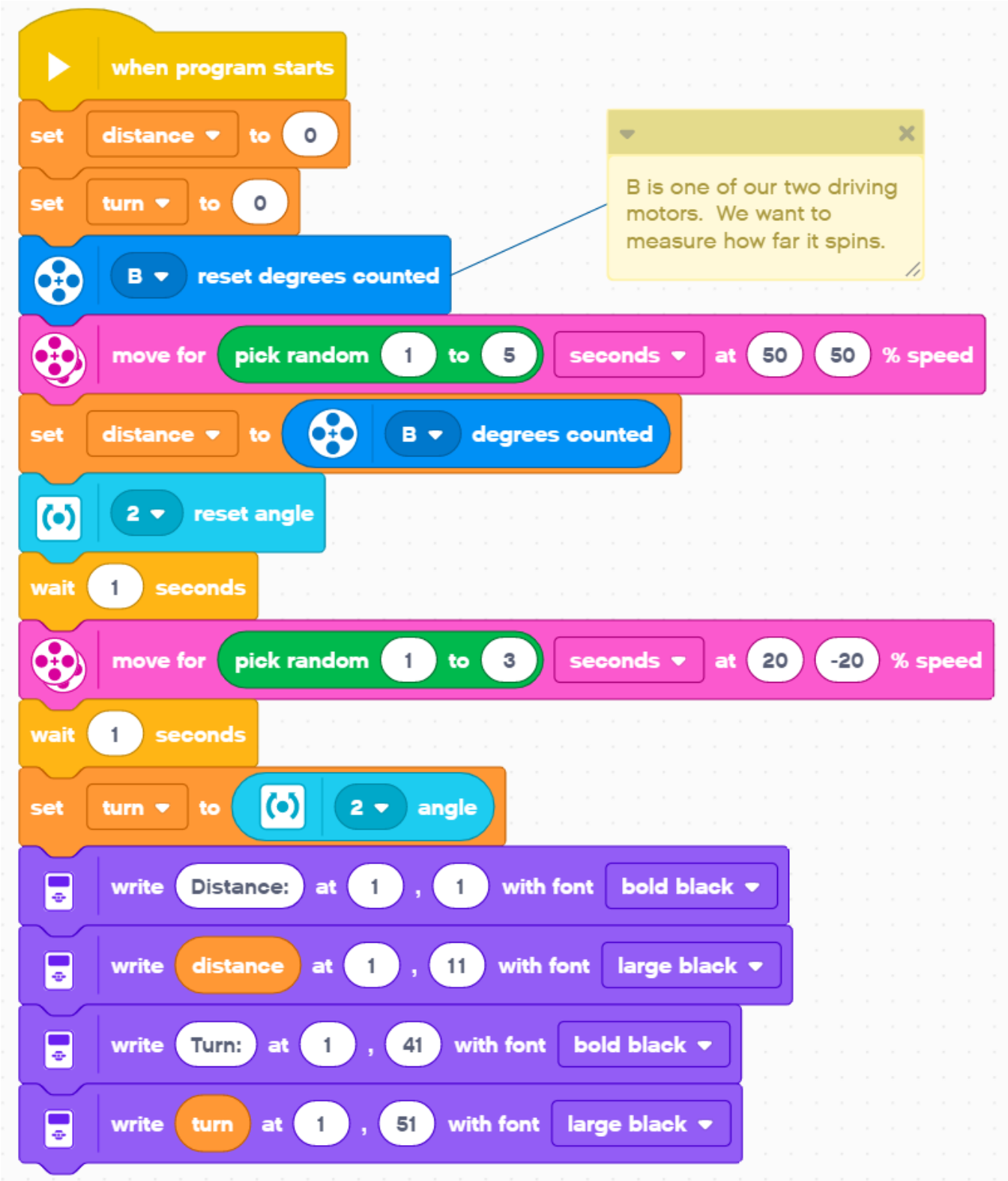
*(Using variables to record the amount you turn and the distance you drive will make your return trip easier.)*

Planning:

- When following a line, remember that you are actually turning onto and off of the edge of the line. Which type of turns should you use? (pivot turns allow you to go forward instead of spinning in place)
- What will you use to know when your rover reaches the lake? (a second color sensor might be helpful here)
- You have already used the ultrasonic sensor to detect planets. Now use it to turn until you find the tower.
- How can you drive in a straight line from the lake to the tower? (use the gyro program to drive in a straight line)
- What two variables do you need to help you find your way back? (store how far you turned in a variable called turn, and store how far you drove in a variable called distance)

Practice:

Try making a code to drive a random distance, then make a random turn. Store these values into a variable to print on your screen.


when program starts
set distance to 0
set turn to 0
B reset degrees counted
B is one of our two driving motors. We want to measure how far it spins.
move for pick random 1 to 5 seconds at 50 50 % speed
set distance to B degrees counted
2 reset angle
wait 1 seconds
move for pick random 1 to 3 seconds at 20 -20 % speed
wait 1 seconds
set turn to 2 angle
write Distance: at 1 , 1 with font bold black
write distance at 1 , 11 with font large black
write Turn: at 1 , 41 with font bold black
write turn at 1 , 51 with font large black

## CHALLENGE F:
## LOAD THE CARGO BAY

**In order to have enough fuel to make the return flight home, you need to load some of the trilithium ore you found on the planet into your cargo bay. Create a robotic arm or a device to move the ore into your ship.**

*Specifics: Build a robotic arm or device to pick up the red trilithium sphere and place it into your ship. Do not drop the trilithium as it is highly explosive.*

Planning:

- There are several "robotic arm" designs on the smithlearning.com/robotics site. However, you probably don't need something that complicated.
- You will need to design a place for the spheres to be stored on your ship.

## CHALLENGE G:
## NAVIGATE THE ASTEROID FIELD

**You have left the strange planet and are headed home. (Mission Control received your transmission for help and sent the necessary coordinates for your return trip.) The last obstacle is to get through the asteroid field safely.**

*Specifics: Use any sensors you like, but do not hit any of the asteroids or you will be right back where you started.*