Name: _____ Date: _____

VEXIQ Distance Sensor: VEXcode IQ Python

<u>In this Lesson:</u> You learn about how to use the Distance Sensor in a project in VEXcode IQ Python, to make your robot accomplish a task.

Definition(s):

● **Distance Sensor:** a sensor that uses a pulse of laser light to determine the distance from the front of the sensor to an object in front of it (as shown in this image from the video).

Notes:

● You can use the ***distance_object_detected*** command to make your robot drive until the Distance Sensor detects an object, then perform a behavior, like turn right, as shown in this project from the video.

```
3  # Begin project code
4
5  calibrate_drivetrain()
6  drivetrain.drive(FORWARD)
7  while not distance_7.is_object_detected():
8      wait(20,MSEC)
9  drivetrain.turn_for(RIGHT, 90, DEGREES)
```

● To make your robot get closer to the object it detects, you can change how far from the object the Distance Sensor is when it moves to the next behavior using the ***distance_object_distance*** command, as shown in this project from the video.

```
4
5  # Begin project code
6  calibrate_drivetrain()
7  drivetrain.drive(FORWARD)
8  while not distance_7.object_distance(MM) < 50:
9      wait(20, MSEC)
10 drivetrain.turn_for(RIGHT, 90, DEGREES)
11
```

Here, the robot will drive until the Distance Sensor is <u>less than 50mm </u>from the object before turning.

● An advantage of using the Distance Sensor this way is that you can place your robot anywhere on the Field, and it will carry out the same behaviors.

● Once configured, Distance Sensing commands will appear in the Toolbox, as shown in this image from the video.
    ○ To learn more about how to use each of these blocks in a project, select the Help icon beside each command to use the built-in Help in VEXcode IQ.

```
<> Code
Looks       brain.battery.capacity()                                    (?)
            inertial.calibrate()                                        (?)
Sound       inertial.set_heading(0, DEGREES)                            (?)
Events      inertial.set_rotation(0, DEGREES)                           (?)
            inertial.heading(DEGREES)                                   (?)
Control     inertial.rotation(DEGREES)                                  (?)
Sensing     inertial.acceleration(AxisType.XAXIS)                       (?)
            inertial.gyro_rate(AxisType.XAXIS,                          (?)
Variables   VelocityUnits.DPS)
            inertial.orientation(OrientationType.ROLL,                  (?)
Functions   DEGREES)
            distance_2nd.object_distance(MM)                            (?)
            distance_2nd.object_velocity()                              (?)
            distance_2nd.object_size()                                  (?)
            distance_2nd.is_object_detected()                          (?)
```