MISSION 7: Hot Pursuit **Lesson 1 (Objectives 1-3)**	**Time Frame: 45-50 minutes**
Project Goal: Students use proximity sensors to detect an object. **Learning Targets** • I can use proximity sensors to detect objects. • I can indicate a detected object using the proximity sensor LEDs. • I can experiment with light and dark surfaces to find the best power and detection threshold settings for each surface. • I can use a pre-defined function to scan multiple settings to find the best detection threshold for a given power setting.	**Key Concepts** • CodeBot uses the Infrared Proximity Sensor system to detect objects in its path. • A detection threshold of 0-100% controls how much light is needed for a True detection. If you decrease the thresh value, the 'bot works well, even on a white surface. • An emitter power level setting from 1 to 8 (high power) controls the brightness of CodeBot's IR "flashlight." • The prox.detect(power, thresh) function lets you adapt to different environments. • The prox.range(num_scans, power) function scans multiple settings to find the best threshold for the given power.
Assessment Opportunities • Mission 7 Lesson 1 Log • Submit completed program ***HotPursuit*** • Mission 7 Obj. 1-3 Review Kahoot!	**Success Criteria** ☐ Use the basic function prox.detect() to detect objects in front of the sensors. ☐ Use the reading from prox.detect() to turn on/off the proximity sensor LEDs. ☐ Measure the detection distance for a white surface using different power and thresh values. ☐ Measure the detection distance for a black surface using different power and thresh values. ☐ Use the function prox.range() to find the best threshold for detecting IR light on a white surface. ☐ Use the function prox.range() to find the best threshold for detecting IR light on a black surface.
Teacher Materials in Learning Portal • Mission 7 Lesson 1 Slides • Mission 7 Lesson 1 Mission Log • Mission 7 Lesson 1 Mission Log Answer Key	**Additional Resources** • Mission 7 Obj. 1-3 Review Kahoot! • HotPursuit_obj3 sample code (learning portal) • HotPursuit_ext1 sample code (learning portal)

Vocabulary

- **Proximity sensors:** Infrared (IR) sensors that can detect nearby objects based on the reflected IR light.
- **Detection threshold:** How much light is needed for a True detection. For proximity sensors, the range is 0%--100%.
- **Emitter power level:** The brightness of CodeBot's IR flashlight, with settings from 1 (low) to 8 (high power).

New Python Code

`p = prox.detect()`	Calling this function pulses the emitter and detects reflected IR light. It returns a tuple of two bools: (left, right).

`p = prox.detect(power, thresh)` `leds.prox(p)`	Turn on the LED below each sensor if an object is detected. Example: if p = (True, False), the left LED is turned on and the right LED stays off.
`p = prox.detect(power, thresh)` `leds.prox(p)`	Proximity detection with optional parameters (power, threshold). The power parameter is a range from 1 to 8 for the IR brightness. The threshold is the detection sensitivity, with a range from 0%-100%.
`sensed = prox.range(10, power)`	The function scans multiple sensitivity levels to find the lowest detection threshold (0%-100%). It has four optional parameters: num_samples, power, range_low, range_high. We use the first two. 10 is the highest num_sample possible.

Real World Applications

Detecting objects is used by many electronic objects you might use every day, without even thinking about it. Some examples are given below. Can you think of even more?

- touchless faucets
- soap dispensers and hand dryers
- automatic doors
- vehicle navigation and safety systems
- factory automation systems

Teacher Notes:

- The code for Objectives 1 and 2 is the same as CodeTrek. You can use the slide or CodeTrek.
- The code for Objective 3 is different from CodeTrek. You can type the code directly in the console panel without adding to the code. However, the last goal in the objective will not be met until you add a line of code. This is detailed in the slides.
- Students will need a white surface and a black or very dark surface. Regular printer paper and black construction paper work well.
- For Obj. 2, students need to measure distance. You can put a ruler next to the surfaces, or use the paper PDF in the teacher resources, which has a metric ruler. Just put the black construction paper over the white paper for the second table.
- This lesson can run long if students are doing a lot of testing. You can condense the lesson and have students do less testing, stretch the lesson a little bit into the next day, or use the extension for Objective 3 instead of typing all the code in the console panel.

Extensions / Cross-Curricular:

- Modify the while True loop to cycle through the power range and print the results to the console. This is a fast way to test all the powers for many different surfaces and can be an alternative to using the console panel exclusively in objective 3.
- **SCIENCE:** Learn more about IR light or proximity sensors. Research where proximity sensors are used.
- **MATH:** Create a chart from the data collected during objective 2 or objective 3. Use a different color for the different surfaces.
- Supports **language arts** through reading instructions, guided notes, and reflection writing.

Preparing for the lesson:

- Look through the slides. Decide what materials you want to use for presenting the lesson. The slides can be converted to Google Slides. They can be projected on a large screen.

- Be familiar with the mission log assignment and the questions they will answer. Prepare the assignment to give through your LMS.
- Have white and black surfaces available for each student or programming pair.
- Have a ruler available for each student or programming pair, or print the “White paper with ruler” PDF.
- If you have a word wall, or another form of vocabulary presentation, prepare the new terms.

Lesson Tips and Tricks:

💡 **Teaching tip:**

You can use a variety of discussion strategies to get the most engagement from your students. For example, you can have students write their answers before asking anyone for an answer. You can use one of many think-pair-share methods.

👫 **Pre-Mission Warm-up:** -- slide 2

Students can write in their log first and then share, or discuss first and then write in their log. The warm-up questions review line sensors and have students make a prediction for this mission. Students can share their answers, or compare with each other.

- Question: How does a line sensor work?
- Question: How do you think CodeBot can “see” objects?

💻 **Mission 7 Lesson 1 Activities:**

The Chrome browser works best, but other browsers also support CodeSpace. Each student will complete a Mission Log. Students could work in pairs through the lesson, or they can work individually. There is a lot of testing in this mission, so it is a good one for pair programming.

💡 **Teaching tip: Mission Introduction** -- slides 3-7

This mission is divided up into three lessons. The first lesson focuses on the first three goals. Students answer three questions in their mission log. If you think you will be short on time, they can skip the questions or just answer them as a class discussion.

💡 **Teaching tip: Objective #1** -- slides 8-10

These slides introduce the proximity sensors and how they work.

💡 **Teaching tip: Objective #1** -- slides 11-12

These slides show the function that the sensors use to detect an object. It returns a Boolean value for the left sensor and the right sensor as a tuple.

💡 **Teaching tip: Objective #1** Activity -- slides 13-16

Students start a new file and copy the code from the slide, or from CodeTrek (they are the same).

Students need to use a dark surface for CodeBot. Both LEDs should be turned off. Then students place a finger or small object in front of each proximity sensor to turn on the LED. A larger object can be used to turn on both LEDs at the same time. No measurement is taken for this objective.

The last part of the activity is to move CodeBot to a white surface, like printer paper. Both LEDs should turn on without having an object in front of them. Students record what happens in their mission log.

💡 **Teaching tip: Objective #2** -- slides 17-21

Parameters are added to the prox.detect() function so the programmer has more control over detection. The two parameters are explained.

💡 **Teaching tip: Objective #2 Activity** -- slides 22-26
Students modify their code by using the slide or CodeTrek. CodeBot is placed on a white surface. Then they run the code several times, changing the power and thresh each time. Students need a ruler to determine the farthest distance an object can be for detection. They fill out a chart for their results. If the LED is turned on immediately, they can record “True” in the table. Six different combinations are included in the table. They can do more, or record the most interesting six results.

💡 **Teaching tip: Objective #2 Activity** -- slides 27-29
The activity continues by placing CodeBot on a black surface. Another table is filled out for different power/threshold settings.

An optional extension of the activity is to point CodeBot in the air so it is not on any surface, and then run the code again for various power/threshold settings.

💡 **Teaching tip: Objective #3** -- slides 30-34
A new function is introduced that scans multiple settings to find the best detection threshold for a given power setting. The function has four optional parameters, but only two are used for the activity.

💡 **Teaching tip: Objective #3 Activity** -- slides 35-39
The instructions in CodeSpace have students modify the program and run it. The slides give an alternative by typing directly in the console panel. It can be easier to keep track of the results this way. Start with CodeBot on a white surface. Students fill out another table for their results. NOTE: results can vary widely, even with the same lighting and code run back-to-back. Students repeat the activity for a black surface.

The instructions tell students to use the up arrow to repeat an instruction in the console panel. They can use the back arrow to change the power setting before pressing <enter> to see the results.

If you are running short on time, you can do the extension for this objective. But make sure you use the console panel at least once first.

In order to meet the two goals for this objective, you have to use the console panel at least once and add one line of code to the program (slide 39).

💡 **Teaching tip: Extension** – slides 40-41
One extension is given for this lesson. You can use it as an alternative for objective 3, or for students who finish early. Code is given for this extension.

Optional: 🗝️ Mission 7 Obj 1-3 Kahoot! Review. A review Kahoot! Is available for these three objectives.

👫 **Post-Mission Reflection:**

The post-mission reflection asks students to review the objectives in this lesson.

You can use a cross-curricular activity for a post-mission activity.

End by collecting the Mission 7 Lesson 1 Log.

SUCCESS CRITERIA:

- ❏ Use the basic function prox.detect() to detect objects in front of the sensors.
- ❏ Use the reading from prox.detect() to turn on/off the proximity sensor LEDs.
- ❏ Measure the detection distance for a white surface using different power and thresh values.
- ❏ Measure the detection distance for a black surface using different power and thresh values.
- ❏ Use the function prox.range() to find the best threshold for detecting IR light on a white surface.
- ❏ Use the function prox.range() to find the best threshold for detecting IR light on a black surface.