



## MISSION 4: Animatronics Lesson 1 (Objectives 1-3)

Time Frame: 35-45 minutes

**Project Goal:** Students will use an infinite loop to constantly blink the user LEDs in a cool pattern.

### Learning Targets

- I can use a while True: infinite loop.
- I can increment a counter variable.
- I can use the debugger to track a counter variable.
- I can use an if statement to avoid an out of range error.
- I can reset a variable to its initial value.

### Key Concepts

- While loops are used to execute an algorithm constantly.
- Increments (and decrements) are used for updating variables like counters.
- CodeBot has 8 user LEDs, numbered 0 through 7. If you try to turn on a user LED with a different number, you get an out of range error.
- The debugger and console panel can be used to track the value of a variable.

### Assessment Opportunities

- Mission 4 Lesson 1 Log (digital)
- Submit completed program **SweepLEDs**
- Submit the program with extensions
- [Mission 4 Obj. 1-3 Review Kahoot!](#)

### Success Criteria

- ☐ Use an infinite loop
- ☐ Use a variable to light a user LED
- ☐ Increment the variable to light the LEDs in sequence
- ☐ Use an if statement to check if the variable is a specific number
- ☐ Reset the variable to its initial value

### Teacher Materials in Learning Portal

- Mission 4 Lesson 1 Slides
- Mission 4 Lesson 1 Log
- Mission 4 Lesson 1 Answer Key

### Additional Resources

- [Mission 4 Obj. 1-3 Review Kahoot!](#)
- **SweepLEDs** sample code (learning portal)
- **SweepLEDs\_extensions** sample code (learning portal)

### Vocabulary

- **Loop:** Repeating a block of code, as long as a condition is True.
- **While condition:** The Boolean value, variable or expression used in a while loop.
- **Infinite loop:** A loop that never ends because the condition is always true.
- **Updating a variable:** Assign a new value to a variable, based on the old value.
- **Increment:** Update a variable by adding one (or a specific number) to the old value.
- **Single equal (=):** Assignment symbol; used to assign a value to a variable.
- **Double equal (==):** Comparison operator used to determine if two variables or values are the same.

### New Python Code

<code>while True:</code>	Infinite loop; code block must be indented directly underneath the loop.
<code>leds.user_num(n_led, True)</code>	Use a variable to indicate the LED
<code>n_led = n_led + 1</code>	Update a variable; increment
<code>if n_led == 8:     n_led = 0</code>	Check if a variable is the same as a specific value. If so, reset the variable to its initial value.



## Real World Applications

Using loops and infinite loops are used in many real-world applications.

- Blinking traffic lights.
- Sensors that detect movement or objects need to constantly be checking.
- Robotic arms in factories repeat their actions continuously.

## Teacher Notes:

- Objective 3 uses the debugger to track the variable. This can be done together as a whole class. Use the debugger and a large display screen. Note that the loop doesn't stop at 7, there just aren't any more LEDs left.
- The debugger stops when an error occurs, and the value in the debug window will no longer show. Make note of the value before it disappears.
- Review the difference between = and ==.
- Students should refer to their data from Mission 3 Lesson 3 to help with moving forward and turning.

## Extensions / Cross-Curricular:

- Use two variables for the delay, one for turning on the LEDs and one for turning off the LEDs.
- Use one delay variable but different amounts of delay for turning on and off the LEDs.
- Increment the delay variable when resetting the n\_leds variable.
- Use two variables for the LEDs and create a sweeping pattern with two LEDs.
- **MATH:** Use math to calculate how many times an LED will blink, or how long it takes to do 20 sweeps, or what the delay should be to do a specific number of sweeps per minute.
- **LANGUAGE ARTS:** Write about a real-world application that uses an infinite loop.
- Supports **language arts** through reading instructions, guided notes, and reflection writing.

## Preparing for the lesson:

- Look through the slides and workbook. 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. The workbook (if used) can be printed or remain digital through your LMS and given to students.
- Be familiar with the mission log assignment and the questions they will answer. Prepare the assignment to give through your LMS.
- You might want to have more practice problems for Objective 2. The mission log assignment gives three problems. You can prepare more code segments for students to practice with.
- 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. You can have students write their answer and share with someone, and then have other students share answers they heard from their peers. You can randomly select students to answer.

### 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. These warm-up questions review code from Mission 3. Students can share their answers, or compare with each other. This code is used during the lesson.

- Question: What code turns on a user LED?
- Question: What code defines a variable for delay?



### **Mission 4 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.

#### **Teaching tip: Mission Introduction** -- slides 3-6

This mission is divided up into four lessons. The first lesson focuses only on the first goal. Students answer two questions in their mission log.

#### **Teaching tip: Objective #1** -- slides 7-10

Students start a new file and learn about infinite loops. They will blink a single LED in the loop. For added learning, students should change the value of `n_led` and see what happens. The slides give instructions for this. It will help them understand what the variable is used for. It isn't required to meet the goal.

#### **Teaching tip: Objective #2** -- slides 11-15

This objective introduces updating a variable by incrementing. The assignment statement may throw off students a little because it doesn't look correct in math. Some sample problems are given in the mission log to practice updating a variable. You might want to give them more problems to practice. Or have students come up with their own code segments and share with each other.

After practicing updating variables, students will follow CodeTrek to add to their code. They will get an error message when the program runs. They will write the error message in their mission log.

#### **Teaching tip: Objective #3 Debugger** -- slides 16-18

These slides and the first part of the objective give a visual of the variable by using the debugger. Its purpose is to show the value of `n_led` throughout the code until it throws an error. Caution: once the program throws an error, the program stops and the debugger window goes blank. You will need to notice the value of `n_led` before the final click when the error occurs.

This activity is not checked by the validator, and students don't have to do it in order to continue. As an alternative, you can do it together as a whole class, using a smart board or large screen. You can also help students use paper and pencil to keep track of the value of the variable and see when/why it throws an error.

#### **Teaching tip: Objective #3** -- slides 19-21


Students add to their code to fix the error. Students need to be careful with their indenting!

This objective uses the `==` for comparison. Students should recognize the difference between `=` and `==`. Something you can do is encourage students to use the wrong symbol in their code and see what the error message is. Making mistakes on purpose is a great learning tool.

At the end of this objective, students should have a nicely working program that sweeps the LEDs from right to left continuously.

#### **Teaching tip: Extensions** -- slide 22

If you have time, students can do an extension. If they do an extension, they might want to do a "File-Save As" so they have their original code, which they will use later. A code solution for extensions 2 and 3 is available in the teacher resources.

**Optional:**  Mission 4 Obj 1-3 Kahoot! Review.

A review Kahoot! is available for these three objectives. You can do the Kahoot together as a class, or assign it independently.

### **Post-Mission Reflection:**

The post-mission reflection asks students to review the difference between `==` and `=`.

You can use an extension or cross-curricular activity as post-mission activity.



End by collecting the Mission 4 Lesson 1 Log.

### **SUCCESS CRITERIA:**

- ☐ Use an infinite loop
- ☐ Use a variable to light a user LED
- ☐ Increment the variable to light the LEDs in sequence
- ☐ Use an if statement to check if the variable is a specific number
- ☐ Reset the variable to its initial value