Programmed Control with Arduino

Activity 6: Sequential Light (Iterative Structures)

In previous activities, we learned how to sequence a traffic light by writing instructions line by line. However, in professional engineering and programming, repeating identical blocks of code is inefficient. In this session, we will connect 5 LEDs and learn how to use the `for` iterative loop to control them in an agile way, drastically reducing the size of our program.

Learning Objectives

By the end of this session, you will be able to:

- Wire a linear bus of multiple digital outputs (5 LEDs in parallel) on the breadboard.
- Understand the logic and internal structure of the `for` iterative statement.
- Use local control variables (such as the index variable) to address physical pins.
- Optimize code by reducing the redundancy of repetitive instructions.

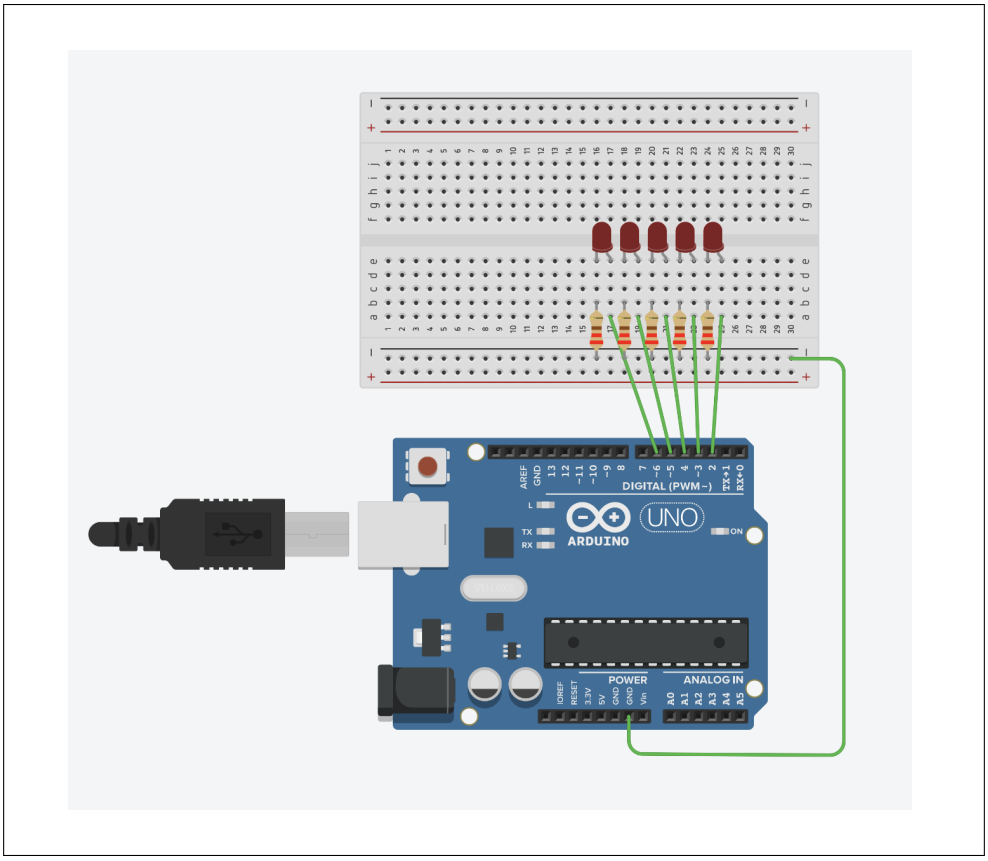
Required Components

Find and place the following components on your TinkerCad workspace:

- 1 Arduino Uno board.
- 1 Small breadboard.
- 5 LEDs (we suggest using varied colors).
- 5 $220\,\Omega$ resistors (it is essential to place one for each LED).
- Virtual jumper wires.

Breadboard Assembly Diagram

We will connect the 5 LEDs to 5 consecutive digital pins on the Arduino (from pin 2 to pin 6). This will greatly facilitate the logical design of our numeric loop.



Base Code: Initialization and Linear Sweep

Open the code editor in TinkerCad. Write the following program to observe how one `for` loop handles configuring all the pins and another performs a continuous turn-on sweep.

```
// Activity 6: Sequential sweep of 5 LEDs optimized with for loops

void setup() {
  // Configures pins 2 to 6 as outputs using a single loop
  // The variable 'i' will take the values: 2, 3, 4, 5, and 6
  for (int i = 2; i <= 6; i++) {
    pinMode(i, OUTPUT);
  }
}

void loop() {
  // Sweep from left to right
  for (int i = 2; i <= 6; i++) {
    digitalWrite(i, HIGH); // Turns on the LED connected to the current pin 'i'
    delay(150);            // Brief pause of 150 milliseconds
    digitalWrite(i, LOW);  // Turns off the current LED before moving to the
    next one
  }
}
```

Listing 1: Code to perform a linear sweep across 5 LEDs.

How does the code work?

The `for` loop is a control structure that repeats a block of code an exact number of times. Its header consists of three key parts separated by semicolons:

- **Initialization (`int`):** Creates a temporary variable called a counter ( ) and assigns it the value of the first physical pin we will use (2).
- **Stop condition ( ):** Before each iteration, Arduino checks if this mathematical condition is true. If is less than or equal to 6, it executes the code inside the curly braces. As soon as reaches 7, the loop ends.
- **Update or increment ( ):** At the end of each iteration, we add 1 to the current value of the variable (is equivalent to writing $i = i + 1$).

During each step of the loop, the `digitalWrite` `HIGH` command substitutes the variable with its corresponding numerical value at that instant, achieving the physical movement of the light.

Activity 6 Challenge

Step up to the next learning tier by solving the following logical challenges in your simulator:

1. **Bidirectional Sweep (Knight Rider Effect)**: Add a second `for` loop inside the `loop()` function to make the sequence perform a return journey (from right to left). *Hint: You must start the counter at 5, check that it is greater than or equal to 3 using `>=`, and subtract one unit at each step with `i--`.*
2. **Cumulative Effect**: Modify the code so that the LEDs turn on one by one sequentially but **do not turn off immediately**; instead, they should all stay on and then turn off all at once at the end of the round.
3. *Question for reflection*: If in the future we decide to expand our assembly by adding three more LEDs on pins 7, 8, and 9, what specific values in the header of our `for` loops would we need to modify in the code?