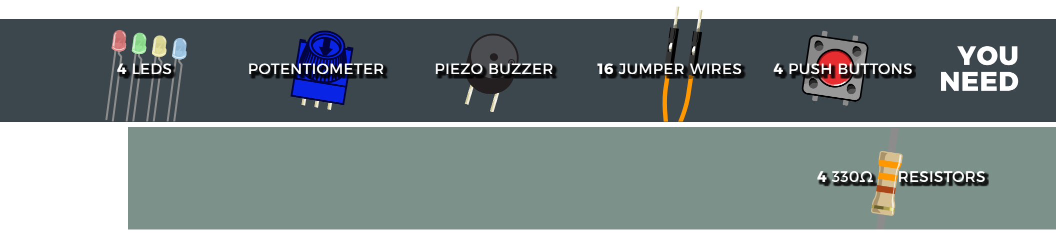
Circuit 2C: “Simon Says” Game

The “Simon Says” game uses LEDs to flash a pattern, which the player must remember and repeat using four buttons. This simple electronic game has been a classic since the late 1970s. Now you can build your own!



NEW CONCEPTS

FOR LOOPS: A **for loop** repeats a section of code a set number of times. The loop works by using a counter (usually programmers use the letter “i” for this variable) that increases each loop until it reaches a stop value. Here’s an example of a simple **for loop**:

```
for (int i = 0; i < 5; i++){
Serial.print(i);
}
```

The **for loop** takes three parameters in the brackets, separated by semicolons. The first parameter is the start value. In this case, integer `i` starts at 0. The second value is the stop condition. In this case, we stop the loop when `i` is no longer less than 5 (i < 5 is no longer true). The final parameter is an increment value. `i++` is shorthand for increase i by 1 each time, but you could also increase i by different amounts. This loop would repeat five times. Each time it would run the code in between the brackets, which prints the value of `i` to the Serial Monitor.

MEASURING DURATIONS OF TIME WITH MILLIS(): The RedBoard has a built-in clock that keeps accurate time. You can use the `millis()` command to see how many milliseconds have passed since the RedBoard was last powered. By storing the time when an event happens and then subtracting the current time, you can measure the number of milliseconds (and thus seconds) that have passed. This sketch uses this function to set a time limit for repeating the pattern.

CUSTOM FUNCTIONS: This sketch uses several user-defined functions. These functions perform operations that are needed many times in the program (for example, reading which button is currently pressed or turning all of the LEDs off). Functions are essential to make more complex programs readable and compact.