Arrays are an essential part of the Arduino programmer’s toolbox. They are sometimes such a necessity that we have already thrown a few in our sketches and projects here and there. Arrays are essentially lists of variables that can contain multiple elements of the same data type. They can be used to store a list of pin numbers, sensor values, character strings, and even bitmaps for animations. While we could have introduced arrays earlier when we discussed variables and data types, the topic truly requires its own chapter to best understand how to use them. Because arrays can consume a large amount of the available memory on the Arduino microcontroller, we should also look at the types of memory space on the microcontroller chip and methods for how to access them, in addition to discussing how to declare arrays, accessing and using arrays, as well as using character and multidimensional arrays. But before tackling arrays in detail, let’s jump in with our eighth project, Decision Machine, so that when we get to discussing arrays in more depth you’ll have already seen them in action.

What’s needed for this chapter:

- Arduino Uno
- 16 × 2 character liquid crystal display HD44780 compatible
- ADXL335 accelerometer (SparkFun breakout)
- 5mm LED of any color
- 1× 220 ohm and 1× 2.2 kilohm ¼ watt resistor or similar
- Hookup wires
- Solderless breadboard

**Project 8: Decision Machine**

Holding our completed project in your hands, you might ask it a simple question before giving it a gentle shake and at that moment a forecast will mystically appear from the murky depths of its display. If that sounds familiar, it’s because we loosely found inspiration for this project in a classic, sometimes irreverent, fortune-telling icon. Our prototype design uses an accelerometer to recreate the familiar rotate or shake to activate the device, and an LCD to provide us with the short but erudite answers to our imagined yes-or-no questions. To make it interesting, we will use arrays throughout our project to give us something to talk about in this chapter and to demonstrate how arrays work in our code.
Hooking It Up

With only few components to hook up, this project is not overly complex, although you will need lots of hookup wires. To display our fortune, we are using a 16 x 2 backlit character liquid crystal display or LCD. Ours has a black background with bright white text for effect and can display two rows of 16 characters each. This display uses the venerable HD44780 interface controller that has been around for ages, making it super easy to display text using the Arduino. Our circuit is fairly standard, using six digital pins to interface with the LCD with two little differences. First, we need to connect a 2.2 kilohm resistor from the contrast pin marked V0 in the schematic to ground. This pin controls the contrast for the LCD and is usually connected to a potentiometer to allow for manually adjusting the contrast of the screen. By using a single resistor, we keep things a little simpler. Secondly, to add theatricality, we are connecting the positive or anode pin of the LED backlight to PWM pin 11. With the `analogWrite()` function, we can fade the answer in and out of existence to simulate the smarmy answers emerging from the murky depths of the display. As usual, we could substitute many other versions of this LCD in different colors or even sizes, as long as it uses the same interface chip.

To detect the customary shake or rotation of the device after a question has been asked, we will use a 3-axis analog accelerometer, which fittingly enough is a device used to measure acceleration or the change in speed or movement along multiple axes. Our particular version is the ADXL335 from Analog Devices that provides an analog voltage corresponding to acceleration in the X, Y, or Z-axis. Because the chip is so small and not very breadboard friendly, we are using the breakout board available from SparkFun Electronics. This breakout board is the first device that needs pin headers or wires soldered to the device to work with our breadboards. For more on this topic, refer to the section on soldering in Chapter 12 With these in place, the connection is simple enough with an output from each of the three axes to three analog in pins marked A0, A1, and A2. Because this device runs on +3.3 volts we need to connect its positive power pin marked VCC to the Arduino interface board's 3.3V output pin. A number of other analog accelerometers would work just as well, but the code might need to be modified for fewer axes or if the accelerometer used some other communication protocol.

One last thing to be aware of is that to simplify wiring a little bit, we are making use of the ground bus on the side of the breadboard, usually marked by a blue line and a “-” sign. The neat thing about this row of pins is that they connect horizontally along the length of the board unlike the other pins that we have used so far. Not all breadboards have these, so you might need to adjust your wiring appropriately. Remember to take your time with the wiring to make sure the wires go to the correct pins. Figures 8-1 and 8-2 show how to hook up this project.