**Loops**

Review: Arrays

* An *array* is like a variable (we call it an *object*), but it can store more than one value.
* An array has an *index* that lets us access a particular *element* (storage spot) in the array.
* One way to think of an array is as a table where each row is an element:

Table 1: myArray

|  |  |
| --- | --- |
| *index* | *value* |
| 0 | 22.3 |
| 1 | 42 |
| 2 | 93.82 |
| 3 | 101 |
| 4 | 0.5 |

Loops­

* There are three control structures in programming: *sequence*, *selection*, and *repetition*. *Repetition* is also called *looping*.
* Loops execute a block of code multiple times.
* There are several types of loops in JavaScript:
  + for
  + for in
  + while
  + do while

*For loop*

* The *for* loop is used when you know how many times you want to *iterate* (go around the loop).
* Structure:  
  initial value condition increment  
   | | |  
  ( for i = 1; i <= 10; i++)  
  {  
   document.writeln(i); --- Body  
  }
* Examples:
  + // Display powers of a number  
    var base = 2;  
    // how many times will the loop execute? Final value of i?  
    for (i = 0; i < 5; i++)  
    {  
     result = Math.Pow(2, i);  
     // the + operator is used for concatenation here  
     answer = base + “ raised to the “ + i + “ = “ result;  
     alert( answer);  
    }  
    alert(“done”);
  + // Use a loop to display the contents of an array  
    // Number of iterations? Final value of i?  
    // i is for “index”  
    for (i = 0; i < length; i++)  
    {  
     alert(myArray[i]);  
    }  
    alert(“done”);
* The *while* loop iterates as long as a condition is true  
  Example:
  + // Enter values in an array until the user enters “done”  
    var name = “”;  
    var names = [];  
    while (name != done)  
    {  
     name = prompt(“Enter a name”);  
     names.push(name);  
    }