What if we wanted to do a two times table?

Well, we *could* write code to call twoTimer 10 times...

```javascript
function timesTable()
{
    var toPrint="";
    toPrint+=(0+" times two is "+twoTimer(0)+"\n");
    toPrint+=(1+" times two is "+twoTimer(1)+"\n");
    toPrint+=(2+" times two is "+twoTimer(2)+"\n");
    :;
    toPrint+=(9+" times two is "+twoTimer(9)+"\n");
    alert(toPrint);
}
```
But what if we wanted to do 100 calls? 1000?

- We need loops.
- Loops allow us to repeat code easily.
- There are many kinds of loops – one that appears in most programming languages is the **for loop**
for Loop Syntax

for ( <initialization>; <continuation>; <next iteration> ) {
    <statement list>
}

The **statement** sequence to be repeated is in the **<statement list>**
for Loop Syntax

```plaintext
for ( <initialization>; <continuation>; <next iteration> ) {
    < statement list>
}

<initialization> sets the iteration variable’s value for the first (if any) iteration of the loop
```
for Loop Syntax

```c
for ( <initialization>; <continuation>; <next iteration> ) {
< statement list>
}
```

<continuation> has the same form as the predicate in a conditional statement

If the <continuation> test is false outcome, the loop terminates and <statement list> is skipped

If <continuation> has a true outcome, the <statement list> is performed
for Loop Syntax

for ( <initialization>; <continuation>; <next iteration> ) {
    <statement list>
}

<next iteration> defines what happens at the end of the loop to start the next round
A loopy two times table

```javascript
function timesTable()
{
    var toMultiply;
    var doubled;
    var toPrint="";
    for (toMultiply = 0; toMultiply < 10; toMultiply++){
        doubled = twoTimer(toMultiply);
        toPrint+=(toMultiply + " times two is " + doubled + "\n");
    }
    alert(toPrint);
}
```
Let’s pretend that I wanted to keep track of all 117 grades from the midterm in variables and then find the average of all grades. I could do

```javascript
var mark0 = 78;
var mark1 = 67;
;
var mark116 = 83;

var avg = (mark0 + mark1 + ... + mark116)/117;
```

That’s kind of messy. It’s perfect data for an array
Arrays are a means to store tabular data

```javascript
var colours = new Array("red", "green", "blue");

0 1 2
"red" "green" "blue"
```

Equivalent to:

```javascript
var colours = new Array();
colours[0] = "red";
colours[1] = "green";
colours[2] = "blue";
```
Think of arrays as being a row of post office boxes.

As with any variable, each box can fit exactly one thing.
Arrays – a common mistake

The first element of an array has index 0 not 1!!!

index of "red" is 0; index of "green" is 1!

Note:
Normal folks count 1, 2, 3,…
Computer scientists count 0, 1, 2
Array indices

We use indices to access array values

```
colours[2] = "purple";
```

After this assignment statement, the value stored in slot *numbered* 2 of the array changes to "purple". Note because we start counting at 0, this is the *third* slot of the array.
Arrays & Loops

Now, I can deal with your grades more easily using an array:

```javascript
var marks = new Array(83, 79, 58, 97);
var total = 0;
for (var i = 0; i < marks.length; i++){
    total = total + marks[i];
}
var average = total / marks.length;
```
We can have arrays of arrays: 2 dimensional arrays

```javascript
var phoneBook = new Array();
phoneBook[0] = new Array("Ava", "604-123-4567");
```
```javascript
var phoneBook = new Array();
phoneBook[0] = new Array("Ava", "604-123-4567");
```

What is `phoneBook[2][1]`?

A. "Da"
B. "778-123-4567"
C. "Jenny"
D. "867-5309"
E. None of the above
Learning Goals [revisited]

- recognize and explain the five essential properties of an algorithm: input specified, output specified, definiteness, effectiveness, and finiteness
- explain how variables are used to ease data management and to describe actions on data, and be able to use variables for these purposes in familiar contexts
- recognize and explain the concept of sequences of instructions, variables, loops, functions, conditional statements, and arrays in short programs specified in a programming language such as JavaScript, or in other clearly expressed processes
- Be able to write short JavaScript programs using the above concepts