function initializeArrays()
{
  var n1 = new Array( 5 );   // allocate 5-element Array
  var n2 = new Array();      // allocate empty Array

  for ( var i = 0; i < n1.length; ++i )
    n1[i] = i;

  for ( i = 0; i < 5; ++i )
    n2[i] = i;

  outputArray( "Array n1 contains", n1 );
  outputArray( "Array n2 contains", n2 );
}

function outputArray( header, theArray )
{
  document.writeln( "<h2>" + header + "</h2> <p>" );
  for ( var ii in theArray )
    document.write( theArray[ii] + "<br/>" );
  document.writeln( "</p>" );
}

initializeArrays();
...but were afraid to ask.

Scope – Revisited

```javascript
function mystery( x, y )
{
    for ( var ii = 0; ii < x.length; ++ii )
        x[ii] = x[ii] * y;
    y = 7;
    document.writeln("<br/> x: ",x);
    document.writeln("<br/> y: ",y);
}

var myArray = [3, 4, 5];
var factor  = 2;

document.writeln ("<br/> myArray: ", myArray);
mystery(myArray, factor);

document.writeln ("<br/> myArray: ", myArray);
document.writeln ("<br/> factor : ", factor);
```

Arguments are passed ______________,
so original argument values in caller are ______________
BUT array/object arguments are a “reference”, so contents may be _____________
Exercise #1
a.) Write a function “sumArray” as follows:
   Input: an array
   Output: the sum of that array
b.) Write test code to create an array and call “sumArray” on it.

Exercise #2 – What’s the output?

```javascript
function printme( z ) {
    document.writeln("<br/> z is ",z);
}

var array1 = [17, 21, 42];
var array2 = [14, 19];
var x = 1;

printme (array1);
printme (array2[1]);
printme (x);
array1[x] = 57;
printme (array1);
```
function changeMe1( z ) {
    z[0] = 75;
}
function changeMe2( a, b) {
    a = b;
}
var array1 = [17, 21, 42];
var array2 = [14, 19];
var array3 = [7, 8, 9];
var x = 63;
changeMe1 (array1);
document.writeln("<br/> array1: ", array1);
changeMe1 (x);
document.writeln("<br/> x: ", x);
array1 = array2;
document.writeln("<br/> array1: ", array1);
changeMe2 (array1, array3);
document.writeln("<br/> array1: ", array1);

Exercise #4

- Write a function perfect(N) that returns an array of size N containing the first N perfect squares. So perfect(4) would return [0, 1, 4, 9].
Exercise #5

Write a function dotProduct(x, y) that takes two arrays of size n and returns the sum:

\[ x[0]*y[0] + x[1]*y[1] + \ldots + x[n-1]*y[n-1] \]

Functions as Arguments

```javascript
function start() {
    var a = [ 10, 1, 9, 2, 8, 3, 7, 4, 6, 5 ];

    document.writeln("<h1>Sorting an Array</h1>");
    document.writeln("Data items in original order: ", a);
    a.sort(compareIntegers); // sort the array
    document.writeln("Data items in ascending order: ", a);
}

// comparison function for use with sort
function compareIntegers( value1, value2 ) {
    return parseInt(value1) - parseInt(value2);
}
```
11.7 document Object

<table>
<thead>
<tr>
<th>Method or property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>getElementById(id)</code></td>
<td>Returns the DOM node representing the XHTML element whose <code>id</code> attribute matches <code>id</code>.</td>
</tr>
<tr>
<td><code>write(string)</code></td>
<td>Writes the string to the XHTML document as XHTML code.</td>
</tr>
<tr>
<td><code>writeln(string)</code></td>
<td>Writes the string to the XHTML document as XHTML code and adds a newline character at the end.</td>
</tr>
<tr>
<td><code>cookie</code></td>
<td>A string containing the values of all the cookies stored on the user’s computer for the current document. See Section 11.9, Using Cookies.</td>
</tr>
<tr>
<td><code>lastModified</code></td>
<td>The date and time that this document was last modified.</td>
</tr>
</tbody>
</table>
How can we use `getElementById`?

```xml
<?xml version = "1.0" encoding=utf-8 ?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.1//EN"
 "http://www.w3.org/TR/xhtml11/DTD/xhtml11.dtd">
<html xmlns = "http://www.w3.org/1999/xhtml">
<head>
<title>DHTML</title>
<script type = "text/javascript">
function changeInput(){
    var myEl = document.getElementById("ammount");
    myEl.value = 10;
}
</script>
</head>
<body>
<form action="">
<h1>Testing assigning values at runtime</h1>
<input type = "text" name = "amount" id = "ammount" />
<input type = "button" value = "Make it 10!" onclick = "changeInput()" />
</form>
</body>
</html>
```
### 11.8 window Object

<table>
<thead>
<tr>
<th>Method or property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>open(url, name, options)</code></td>
<td>Creates a new window with the URL of the window set to <code>url</code>, the name set to <code>name</code> to refer to it in the script, and the visible features set by the string passed in as <code>options</code>.</td>
</tr>
<tr>
<td><code>prompt(prompt, default)</code></td>
<td>Displays a dialog box asking the user for input. The text of the dialog is <code>prompt</code>, and the default value is set to <code>default</code>.</td>
</tr>
<tr>
<td><code>close()</code></td>
<td>Closes the current window and deletes its object from memory.</td>
</tr>
<tr>
<td><code>focus()</code></td>
<td>This method gives focus to the window (i.e., puts the window in the foreground, on top of any other open browser windows).</td>
</tr>
<tr>
<td><code>blur()</code></td>
<td>This method takes focus away from the window (i.e., puts the window in the background).</td>
</tr>
<tr>
<td><code>window.document</code></td>
<td>This property contains the <code>document</code> object representing the document currently inside the window.</td>
</tr>
<tr>
<td><code>window.closed</code></td>
<td>This property contains a boolean value that is set to true if the window is closed, and false if it is not.</td>
</tr>
<tr>
<td><code>window.opener</code></td>
<td>This property contains the <code>window</code> object of the window that opened the current window, if such a window exists.</td>
</tr>
</tbody>
</table>