

	<u>Regular variable</u>	<u>Array variable</u>
Declaration:	<code>int a; // variable name is a</code>	<code>int a[3]; // array name is a // size is 3 "elements"</code>
Initialization:	<code>a = 11;</code>	<code>a[0] = 83; a[1] = 12; cin >> a[2];</code>
Access:	<code>By name: cout << a;</code>	<code>By index: int i = 1; cout << a[i]; // valid indices range from // 0 to (array size - 1)</code>
Function parameter:	<code>void foo(int a);</code>	<code>void foo(int a[]);</code>
Function call:	<code>foo(a);</code> <---- ! ----> <code>foo(a);</code>	
Gotchas:	<p>The compiler does <u>not</u> check for use of a valid array index: <code>double A[8]; // valid indices: 0 to 7 A[8] = 100.0; // no such array element, A[8]!</code></p> <p>To pass an entire array to a function, use only the array name: <code>foo(a); // <u>not</u>: foo(a[]) <u>not</u>: foo(a[0])</code></p>	
	<p>A function that has an array parameter usually also has an <code>int</code> parameter that indicates the size of the array (number of elements). This is good programming practice:</p> <pre>void foo(double x[], int n) { for(int i = 0; i < n; i = i + 1) { // do something with every array element, x[i] } }</pre>	