**Searching for the Median**

The median of a single sorted array is trivial to find and is *O*(1) constant time. For example, a sorted array A = [5, 7, 9, 11, 15], which has an odd number of elements, has a unique median element 9 at index 2. Elements [5, 7] to the left are smaller than or equal to the median. Elements [11, 15] to the right are bigger than or equal to the median. The array has 5 elements with indexes in the range of 0 to 4, and the median element is at index (5-1)/2 = 2. In general, the median is at index (n-1)/2 if the number of elements in an array (n) is odd.

For a sorted array with an even number of elements, two elements in the middle are medians. For example, A = [5, 7, 9, 11] has medians of 7 and 9 at indexes of 1 and 2. In general, the medians are at index floor((n-1)/2) and at n/2. We could just pick the lower median, which leads to a single index computation no matter whether the array has an odd or an even number of elements, to wit, floor((n-1)/2).

Finding a median of two sorted arrays is more difficult and is no longer constant time. For example, two sorted arrays A = [5, 7, 9, 11, 15] and B = [1, 8] could be merged into a single sorted array in *O*(N) time to produce C = [1, 5, 7, 8, 9, 11, 15]. This resulting array has 7 elements, with a median of 8 at index floor((7-1)/2) = 3. It's possible to do better than *O*(N), by using a modified binary search leading to *O*(lgN) performance.

**Modified Binary Search**

In the case above, the median of value 8 came from the smaller of the two arrays. If we use different array values, A = [1, 2, 3, 4, 7] and B = [0, 5, 6, 9], then the combined sorted array C = [0, 1, 2, 3, 4, 5, 6, 7, 9] has a median of 4 at index 4. In this case, the median came from the larger array. This shows that the median can come from either array, no matter the size of either. Even if one of the arrays has a single element, that element could be the overall median. Thus, when we search for the median, both arrays must be involved in the search, otherwise we could potentially miss the true overall median.

Because both arrays are sorted, binary search can be used to search each of the arrays quickly, in *O*(lgN) time. The trick is to develop a test to determine quickly whether the selected element is the median or not — ideally, in constant time. Using a binary search, the mid-element of the sorted array A is selected, which is 3 at index 2. Is this element an overall median? What kind of a test is necessary?