

# Binary Search – Iterative and Recursive

The binary search algorithm has Big-O value of  $O(\log n)$ . The binary search method below uses an iterative approach to search an array.

```
//Iterative binary search of an ascending array
private static int search (int[] a, int searchValue) {
    int left = 0;
    int right = a.length-1;
    while (left <= right) {
        int midpoint = (left+right)/2;
        if (a[midpoint] == searchValue)
            return midpoint;
        else if (a[midpoint] < searchValue)
            left = midpoint + 1;
        else
            right = midpoint - 1;
    }
    return -1;
}
```

The binary search method listed below using a recursive approach to search an array.

```
//Recursive binary search of an ascending array
private static int search (int[] a, int target, int left, int right) {
    if (left > right)
        return -1;
    else {
        int midpoint = (left + right) / 2;
        if (a[midpoint] == target)
            return midpoint;
        else if (a[midpoint] < target)
            return search (a, target, midpoint + 1, right);
        else
            return search (a, target, left, midpoint - 1);
    }
}
```

Method calls for the iterative and recursive binary searches.

```
int[] a = {15,36,87,95,100,110,194,205,297,301,314,358,451,467,486};
int x = 320;
int location;
```

```
location = search (a, x);
```

← Iterative version

```
location = search (a, x, 0, a.length - 1);
```

← Recursive version

Examples:

- 1) Do a linear search of the following array. Indicate the number of comparisons needed before the indicated value is found.

Array a

12	19	21	25	26	29	31	35	36	39	40	42	47	50	52	56	58	65	70	78
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19

- a) searchValue = 12 → Number of comparisons = \_\_\_\_
- b) searchValue = 78 → Number of comparisons = \_\_\_\_
- c) searchValue = 39 → Number of comparisons = \_\_\_\_
- d) searchValue = 70 → Number of comparisons = \_\_\_\_
- 2) Do a binary search of the same array. Indicate the number of comparisons needed before the indicated value is found. Also indicate the value of the midpoint and the intervals for each comparison.
- e) searchValue = 12 → Number of comparisons = \_\_\_\_  
Midpoints = \_\_\_\_\_  
Search intervals = \_\_\_\_\_
- f) searchValue = 78 → Number of comparisons = \_\_\_\_  
Midpoints = \_\_\_\_\_  
Search intervals = \_\_\_\_\_
- g) searchValue = 39 → Number of comparisons = \_\_\_\_  
Midpoints = \_\_\_\_\_  
Search intervals = \_\_\_\_\_
- h) searchValue = 70 → Number of comparisons = \_\_\_\_  
Midpoints = \_\_\_\_\_  
Search intervals = \_\_\_\_\_