Problem 2: Search in a Rotated Sorted Array

In this problem, we use a variant of binary search. First, we will store the starting element of the given array. After that we declare two variables first and last to keep track of subarray. We find the middle element by average of first and last and compare the middle element with the starting element of given array as well as the number to search. This will create 6 different possibilities accordingly the first and last pointers should be updated. We repeat this procedure until first and last are same.

Time complexity:

O(log(n)): Since we are using binary search variant.

Space complexity:

O(1): Only 4 values are stored at a time.