```swift
// Copyright © 2020 AlgoExpert, LLC. All rights reserved.

class Program {
    // O(log(n)) time | O(1) space
    func searchForRange(_ array: [Int], _ target: Int) -> [Int] {
        var finalRange = [-1, -1]

        var leftPointer = 0
        var rightPointer = array.count - 1

        alteredBinarySearch(array, target, &leftPointer, &rightPointer, &finalRange, true)

        leftPointer = 0
        rightPointer = array.count - 1
        alteredBinarySearch(array, target, &leftPointer, &rightPointer, &finalRange, false)
        return finalRange
    }

    func alteredBinarySearch(_ array: [Int], _ target: Int, _ leftPointer: inout Int, _ rightPointer: inout Int, _ finalRange: inout [Int], _ goLeft: Bool) {
        while leftPointer <= rightPointer {
            let middle = (leftPointer + rightPointer) / 2

            if array[middle] > target {
                rightPointer = middle - 1
            } else if array[middle] < target {
                leftPointer = middle + 1
            } else {
                if goLeft {
                    if middle == 0 || array[middle] != array[middle - 1] {
                        finalRange[0] = middle
                        return
                    } else {
                        rightPointer = middle - 1
                    }
                } else {
                    if middle == array.count - 1 || array[middle] != array[middle + 1] {
                        finalRange[1] = middle
                        return
                    } else {
                        leftPointer = middle + 1
                    }
                }
            }
        }
    }
}
```