# 2024-03-30 - Handout – Leetcode 100 (Binary Search)

### Q1. Search Insert Position

Link: https://leetcode.com/problems/search-insert-position/description/

Given a sorted array of distinct integers and a target value, return the index if the target is found. If not, return the index where it would be if it were inserted in order.

You must write an algorithm with O(log n) runtime complexity.

**Example 1:**
Input: nums = [1,3,5,6], target = 5

Output: 2

**Example 2:**
Input: nums = [1,3,5,6], target = 2

Output: 1

### Q2. Find Minimum in Rotated Sorted Array

Link: https://leetcode.com/problems/find-minimum-in-rotated-sorted-array/

Suppose an array of length n sorted in ascending order is rotated between 1 and n times. Given the sorted rotated array nums of unique elements, return the minimum element of this array. For example, the array nums = [0,1,2,4,5,6,7] might become:

- [4,5,6,7,0,1,2] if it was rotated 4 times.
- [0,1,2,4,5,6,7] if it was rotated 7 times.

Notice that rotating an array [a[0], a[1], a[2], ..., a[n-1]] 1 time results in the array [a[n-1], a[0], a[1], a[2], ..., a[n-2]]. You must write an algorithm that runs in O(log n) time.

**Example 1:**
Input: nums = [3,4,5,1,2]

Output: 1

Explanation: The original array was [1,2,3,4,5] rotated 3 times.

**Example 2:**
Input: nums = [11,13,15,17]

Output: 11

Explanation: The original array was [11,13,15,17] and it was rotated 4 times.

**Followup: If nums may contain duplicates, would this affect the runtime complexity? How and why? How would you solve this while keeping the overall operation steps as low as possible?