1.

nums = (1, 3, 5, 4, 7)

max length = 1
current length = 1

process:

I =1: nums[1] = 3, and 3 > 1, so subsequence will continue. current length = 2

I = 2: nums[2] = 5, and 5 > 3, so subsequence will continue. current length = 3

I = 3: nums[3] = 4, but 4 <= 5, so subsequence breaks. current length = 1 again, but max length = 3

I = 4: nums[4] = 7, so 7 > 4, subsequence continues. current length = 2

at the end:

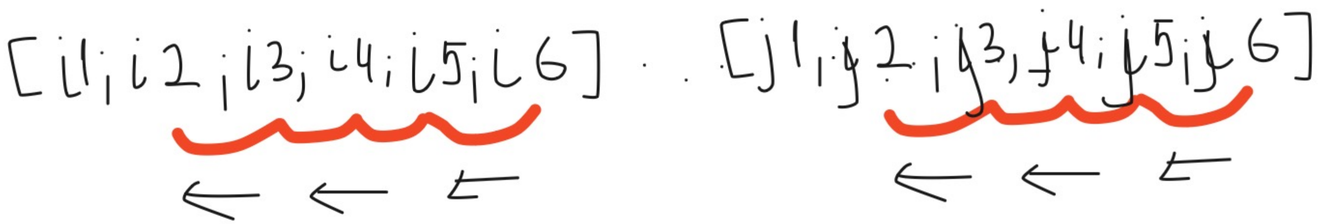
max length = compares the 2 and 3, 3 > 2, so the answer is 3

2

nums1 = [1, 2, 3, 0, 0, 0], m = 3
nums2 = [2, 5, 6], n = 3

p1: Points to the last valid element in nums1 (i.e., nums1[m-1]).
p2: Points to the last element in nums2 (i.e., nums2[n-1]).
p: Points to the last position in nums1 (i.e., nums1[m+n-1]), where merged elements will be placed.

start filling nums1 from the end (p), moving backwards, we compare nums1[p1] and nums2[p2]

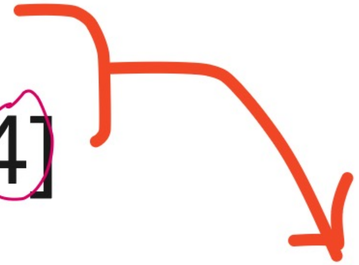


nums1[p1] > nums2[p2] , we put nums1[p1] to the nums1[p] and so on.

3.

[4, 9, 5]

[9, 4, 9, 8, 4]



intersection

4, 9