Question: https://leetcode.com/problems/maximum-length-of-repeated-subarray/

The approach is pretty simple maintain a 2D matrix of nums1.length+1 and nums2.length+1 dimension. Where at the value of [i][j] cell is zero if nums1[i-1] not equal to nums2[j-1] and if equal then its value will be value of [i-1][j-1] +1.

To handle edge cases like where i-1 position does not exist in nums1, we have kept first row and coloumn value zero.

Code:  
class Solution {

public int findLength(int[] nums1, int[] nums2) {

int dp[][] = new int[nums1.length+1][nums2.length+1], max = Integer.MIN\_VALUE;

for(int i=1; i<=nums1.length; i++){

for(int j=1; j<=nums2.length; j++){

if(nums1[i-1]==nums2[j-1]){

dp[i][j]=dp[i-1][j-1]+1;

max = Math.max(max, dp[i][j]);

}

}

}

return max==Integer.MIN\_VALUE?0:max;

}

}

Github Link :<https://lnkd.in/ecwtJeaz>