Question: <https://leetcode.com/problems/merge-sorted-array/>

(Simple Merge Sort)

Constraints: ****Merge**** nums1 and nums2 into a single array sorted in ****non-decreasing order, and it should be present in**** nums1, and nums1 and nums2 are sorted.  
   
This problem is one of the classic algorithms which we learn at the beginning of DSA that is Merge Sort. So what we are doing is we are taking two pointers i and j where I points which is the value from nums1 array to be selected next and j points the similar item from nums2.

Now we check if ith item from nums1 is smaller than jth item from nums2 then we store it in the merged array else we store the jth element from nums2. This step continues till one among 2 arrays reach its end.

Then we check if the other array has some elements left to be yet stored or not, if left then we store them in merged array.  
   
Solution:  
class Solution {

public void merge(int[] nums1, int m, int[] nums2, int n) {

int[] res=new int[m+n];

int i=0, j=0, k=0;

while(i<m&&j<n){

if(j==n&&i<m){

res[k]=nums1[i];

i++;

k++;

continue;

}

if(j<n&&i==m){

res[k]=nums1[j];

j++;

k++;

continue;

}

if(nums1[i]<nums2[j]){

res[k]=nums1[i];

i++;

k++;

}else{

res[k]=nums2[j];

j++;

k++;

}

}

while(i<m){

res[k]=nums1[i];

i++;

k++;

}

while(j<n){

res[k]=nums2[j];

j++;

k++;

}

for(int p=0; p<k; p++){

nums1[p]=res[p];

}

}

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