2. Merge Algorithm for two sorted Arrays

Algorithm merge (arrA, arrB)

index A < 0
index B < 0
index C < 0
arr < int [arr A. size() + arr B. size()]

while index A < arr A. size() and index B < arr B. size() do

if arr A [index A] < arr B [index B] then

arr C [index C] < arr A [index A]

index A = index A + L

else

arctindexc] < arrBCinlexB]
indexB < indexB+1

indexc < index c+1

if indexA < arrA. size () then

for i < indexA to arrA. size ()-1 do

arrC [indexC] < arrA[i]

indexC < indexC+1

{ increment counter i }

else if indexB < arrB.size() then

for i < indexB to arrB.size()-1 do

arrc [indexC] < arrB[i]

indexC = indexC +1

{increment counter i}

return arc

2. Merge Algorithm for two sorted Arrays Algorithm merge (and, and) X outrisel) > NE OUR HIME) index A < 0 index B < 0 index c < 0 anc - int [and size() + and size()] while indexA < arra-rise() and indexB < omB. size() do if arrational amo > [raxebui] Arma fi OMC [index c] + OMA [index A] indexA = indexA+L anctindex c] < amBlindexB] else indexB < indexB+1 indexc < indexc+1 indexA < arrAsize () then for i < indexA to arra-size()-1 do arra [index c] < arra [i] inlex.c. e index.c+1 { increment counter i } if index B < arrs. size() then for i < indexs to arrs. size()-1 do arc CintexC] < arrs[i] index c = index c +1 fincrement counter i } return arc : Running time is O(m+n)