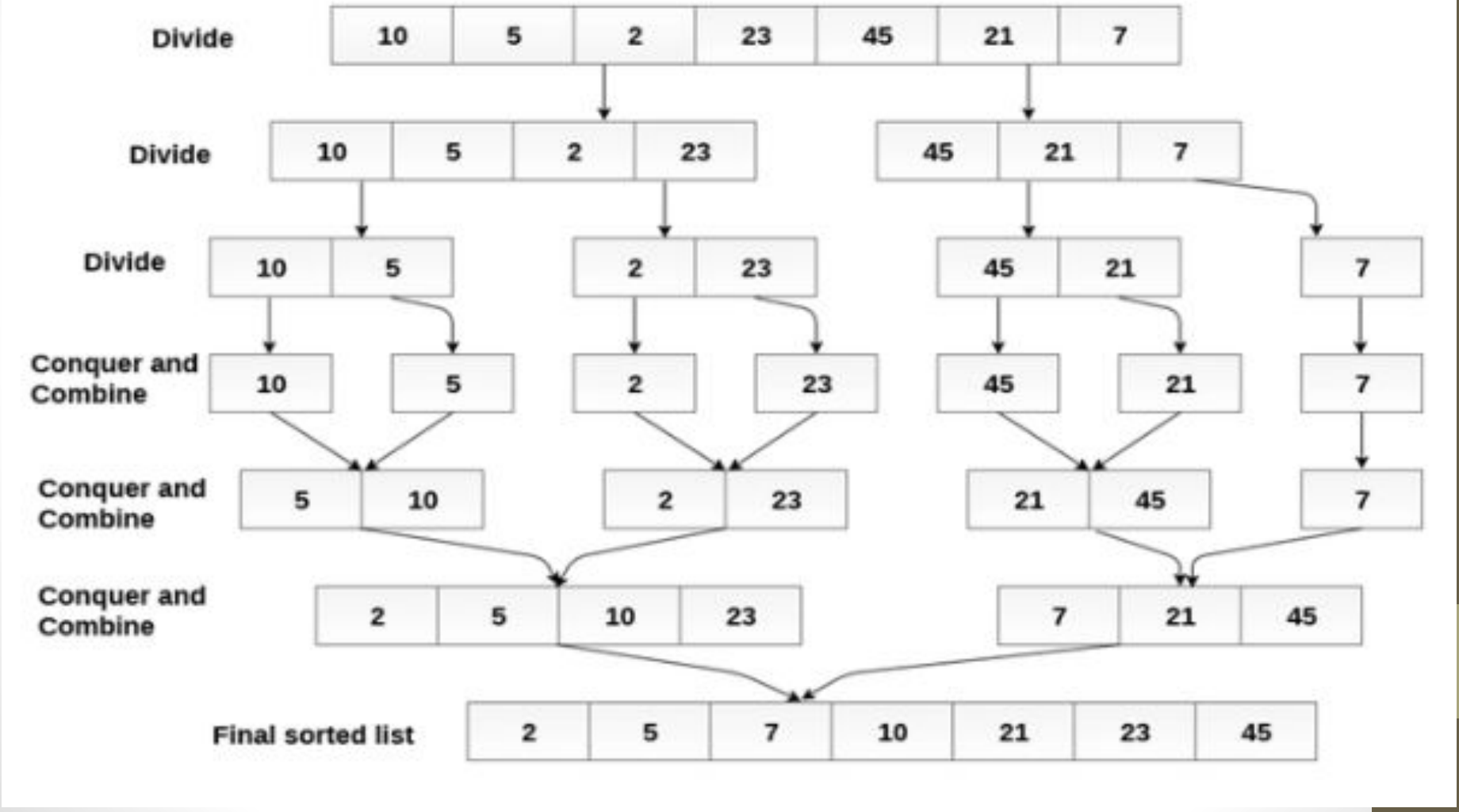
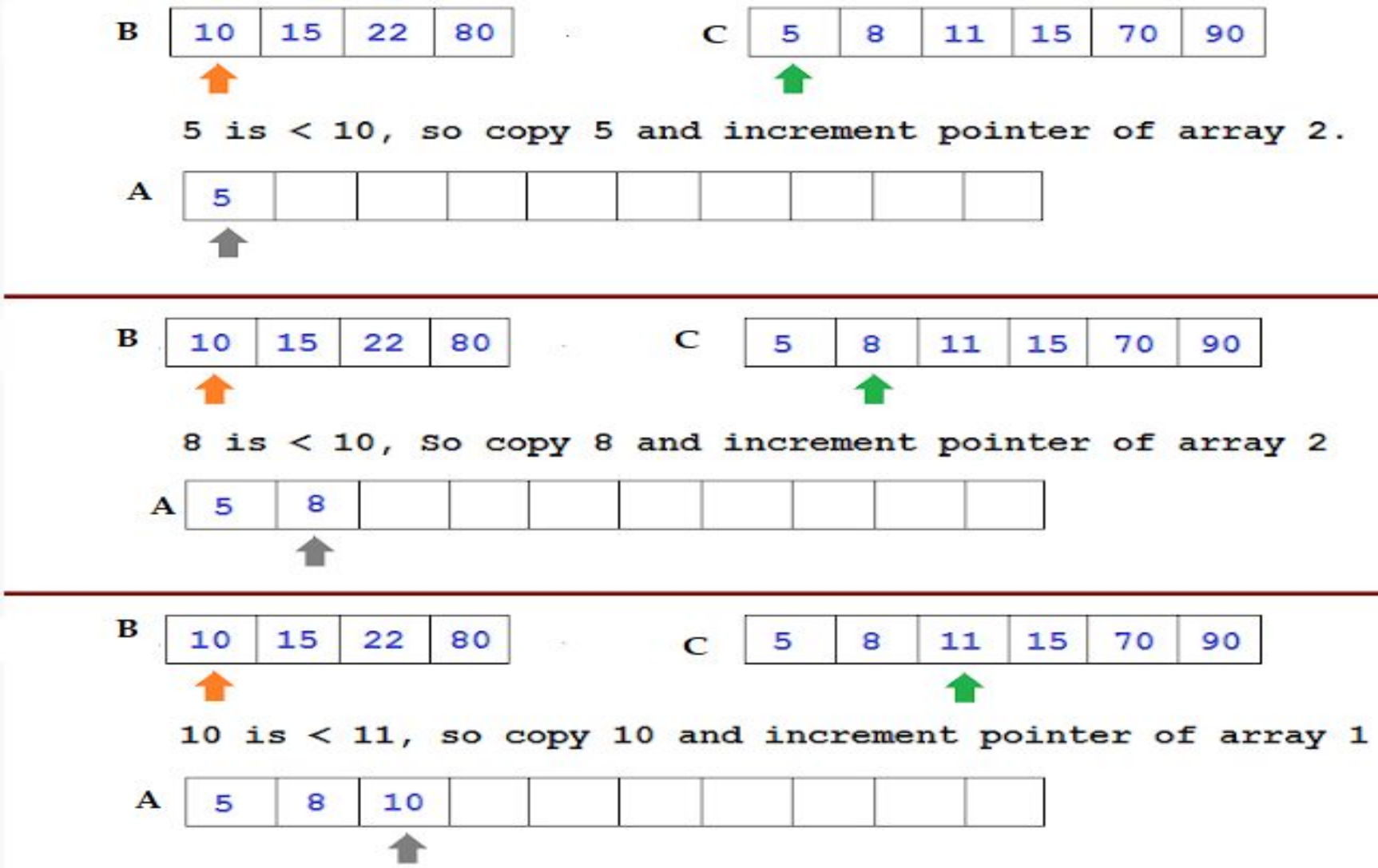
Merge-Sort

- Merge-sort is based on divide-and-conquer technique.
- It sorts a given array *A[0..n−1] as follows*
  - ***Divide*** *the array into* two halves *A[0…. n/2] and A[(n/2)+1 …. n-1]*
  - **Sort** each of them recursively by calling merge sort algorithm.
  - **Merge** the sorted arrays into a single sorted one.
- Merge sort breaking down a list into several sub-lists until each sublist consists of a single element and merging those sublists in a manner that results into a sorted list.

Merge Sort


Divide
10 5 2 23 45 21 7
Divide
10 5 2 23
45 21 7
Divide
10 5
2 23
45 21
7
Conquer and Combine
10
5
2
23
45
21
7
Conquer and Combine
5 10
2 23
21 45
7
Conquer and Combine
2 5 10 23
7 21 45
Final sorted list
2 5 7 10 21 23 45

Merging


B
10 15 22 80
C
5 8 11 15 70 90
5 is < 10, so copy 5 and increment pointer of array 2.
A
5
B
10 15 22 80
C
5 8 11 15 70 90
8 is < 10, So copy 8 and increment pointer of array 2
A
5 8
B
10 15 22 80
C
5 8 11 15 70 90
10 is < 11, so copy 10 and increment pointer of array 1
A
5 8 10

Merge Sort - Algorithm

ALGORITHM ***Mergesort(A[0..n − 1])***

//Sorts array *A[0..n − 1] by recursive merge sort*

//Input: An array *A[0..n − 1] of orderable elements*

//Output: Array *A[0..n − 1] sorted in non-decreasing order*

if n>1

Copy A[0…(n/2)] to B[0…(n/2)]

Copy A[(n/2+1) …n-1] to C[0…(n-1)/2]

Merge sort (B[0..(n/2)]

Merge sort (C[0..(n-1)/2]

Merge (B,C,A)

//Note all divisions above are floor divisions

Merge - algorithm

ALGORITHM ***Merge(B[0..p − 1], C[0..q − 1], A[0..p + q − 1])***

//Merges two sorted arrays into one sorted array

//Input: Arrays *B[0..p − 1] and C[0..q − 1] both sorted*

//Output: Sorted array *A[0..p + q − 1] of the elements of B and C*

i ←0; j ←0; k←0

while *i <p and j <q do*

  if *B[i]≤ C[j]*

    A[k]←B[i]; i ←i + 1

  else

    A[k]←C[j]; j ←j + 1

  k←k + 1

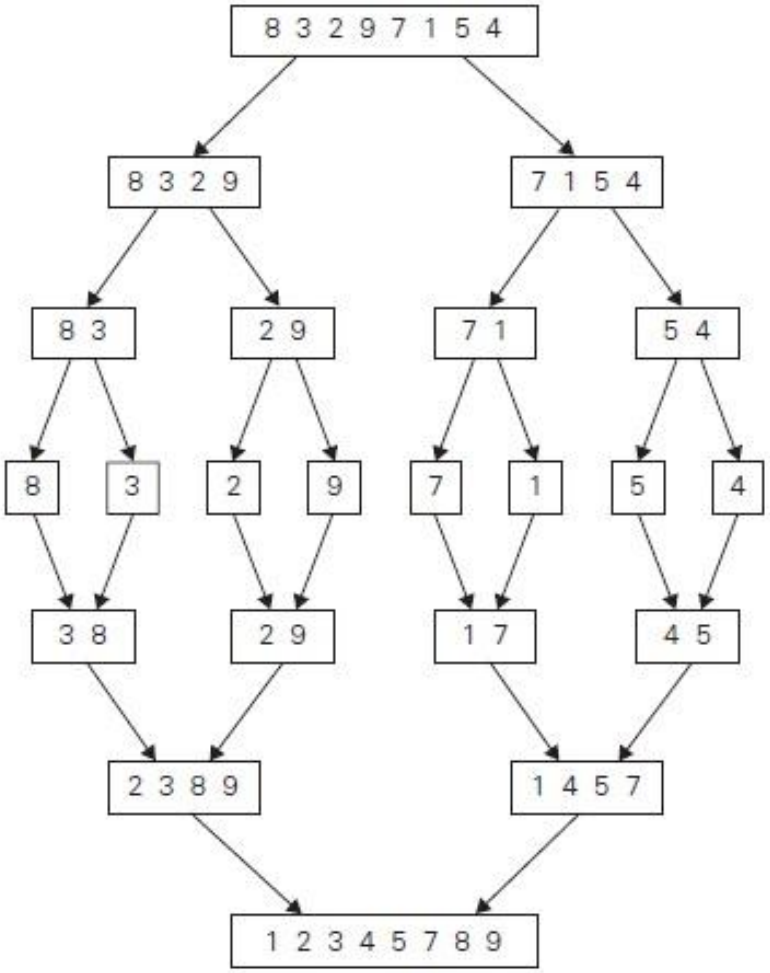
if *i = p*

  copy *C[j..q − 1] to A[k..p + q − 1]*

else

  copy *B[i..p − 1] to A[k..p + q − 1]*

Merge Sort



Ravindrababu - Merge Sort