Newton School of Technology

12

# Merge Sort

by Gladden Rumao

CSA221 : Data Structures and Algorithms

Newton School of Technology

# MergeSort

# Solving a Jigsaw Puzzle :

**DIVIDE**

**Identify key areas:** like the four corners, the edges & the middle section.

**Subdivide the sections i**nto smaller sections further if needed

**Conquer**

- **Focus on one section at a time**
- **Use trial and error** to complete the individual section
- **Utilize reference image :** If available

**Combine**

- **Connect sections**
- **Fill in gaps**

# Merge Sort :

- It is reliable sorting algorithm that can handle **large datasets with minimal complexity.**

- It uses the **divide ,conquer and combine** approach to break the problem into smaller subproblems, sort them individually, and then merge them in a sorted manner.

# Example

| Index | 0 | 1 | 2 | 3 | 4 | 5 | 6 |
|---|---|---|---|---|---|---|---|
| Data Item | 30 | 25 | 11 | 15 | 10 | 12 | 22 |

Newton School of Technology

# Example

| Index | 0 | 1 | 2 | 3 | 4 | 5 | 6 |
|---|---|---|---|---|---|---|---|
| Data Item | 30 | 25 | 11 | 15 | 10 | 12 | 22 |

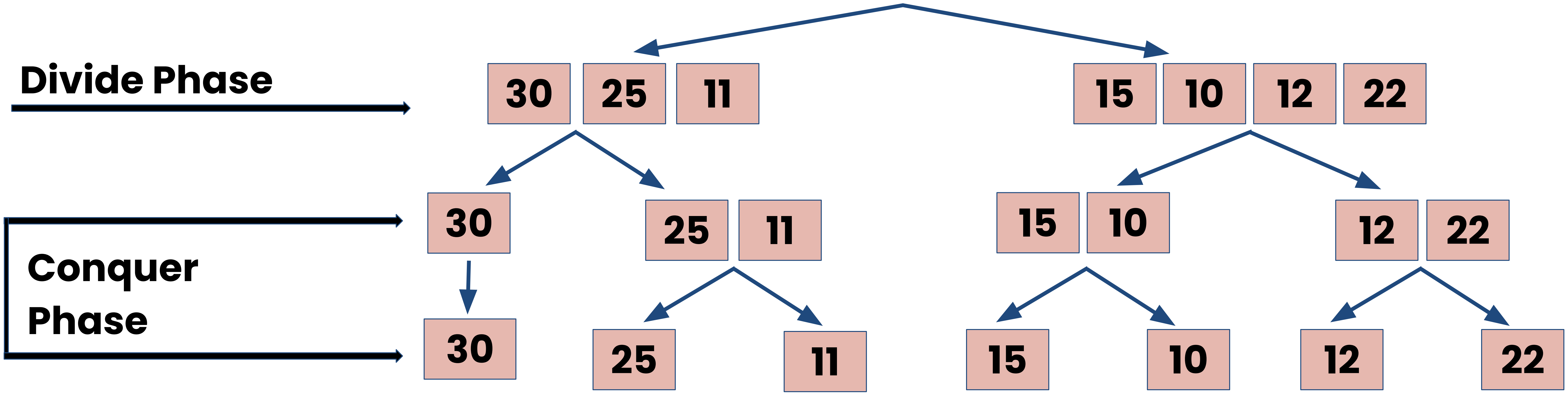
**Divide Phase**

| | | |
|---|---|---|
| 30 | 25 | 11 |

| | | | |
|---|---|---|---|
| 15 | 10 | 12 | 22 |

# Example

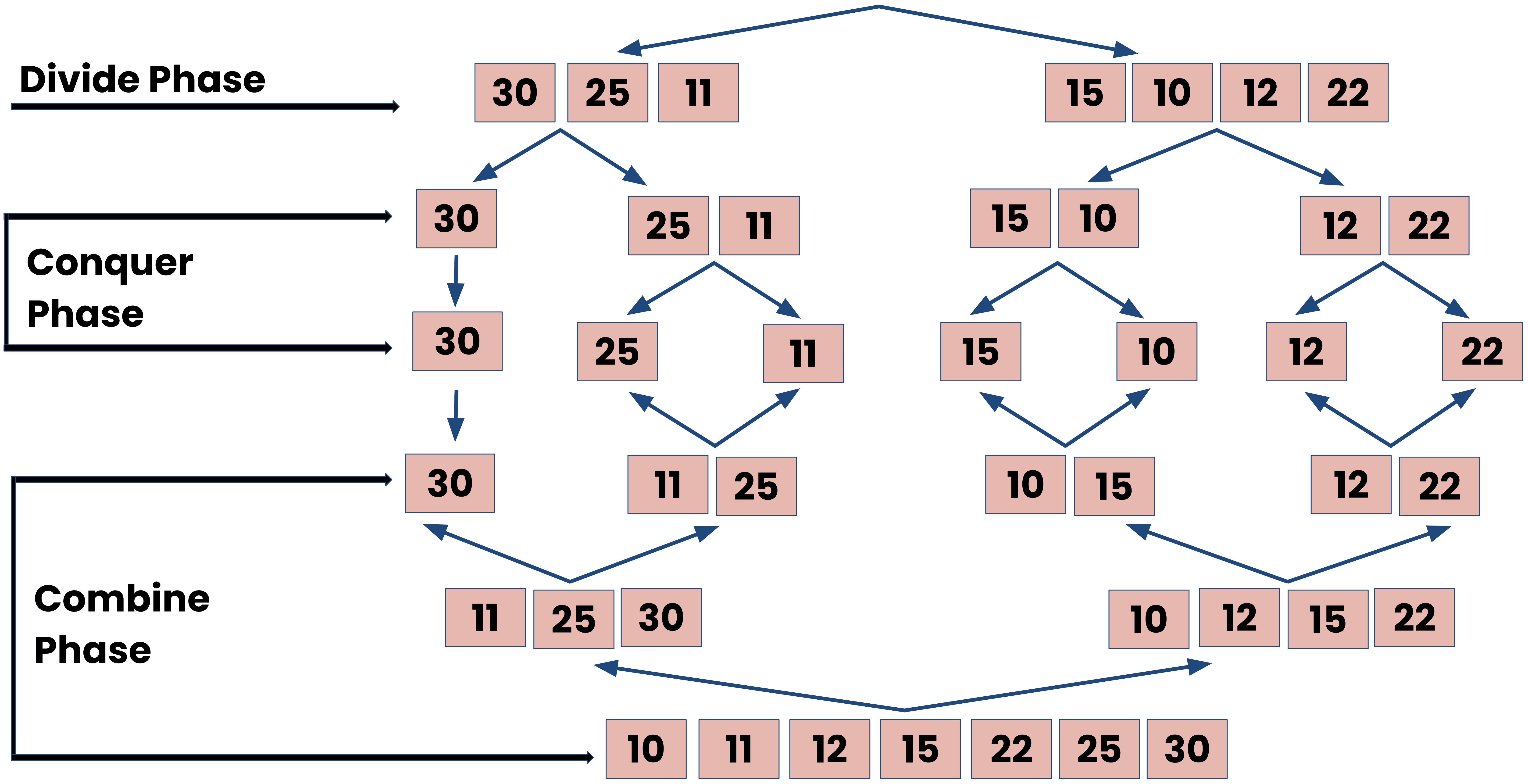
| Index | 0 | 1 | 2 | 3 | 4 | 5 | 6 |
|---|---|---|---|---|---|---|---|
| Data Item | 30 | 25 | 11 | 15 | 10 | 12 | 22 |



# Example

| Index | 0 | 1 | 2 | 3 | 4 | 5 | 6 |
|---|---|---|---|---|---|---|---|
| Data Item | 30 | 25 | 11 | 15 | 10 | 12 | 22 |

Newton School of Technology



# Advantages :

Newton School of Technology

1. **Stability**: Maintains the relative order of equal elements.
2. **Efficiency**: Performs well with **O(n log n)** time complexity for all cases.
3. **Scalability**: Handles large datasets effectively, especially when implemented for external sorting using disk storage.
4. **Predictable Performance**: Unlike quicksort, it doesn't degrade to $O(n^2)$ in the worst case.

**Merge Sort is particularly suited for applications where consistent performance and large-scale data handling are essential.**

# Divide-Conquer and Combine approach

**Divide** the subarray **A[ p : r ]** to be sorted into two adjacent subarrays, each of half the size. To do so, compute the **midpoint q** of A[ p : r ] (taking the average of p and r), and divide A[ p : r ] into subarrays **A[ p : q ]** and **A[ q+1 : r ].**

**Conquer** by sorting each of the two subarrays A[ p : q ] & A[ q+1 : r ] **recursively** using merge sort.

**Combine** by **merging** the two sorted subarrays **A[ p : q ] & A[ q+1 : r ]** back into A[ p : r ], producing the sorted answer.

# Suppose we have list of 16 elements

| | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 |
| a[] | 70 | 35 | 25 | 60 | 40 | 18 | 28 | 50 | 30 | 25 | 20 | 15 | 10 | 12 | 22 | 17 |

# Suppose we have list of 16 elements

| | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 |
| a[] | 70 | 35 | 25 | 60 | 40 | 18 | 28 | 50 | 30 | 25 | 20 | 15 | 10 | 12 | 22 | 17 |

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
| 70 | 35 | 25 | 60 | 40 | 18 | 28 | 50 |

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 |
| 30 | 25 | 20 | 15 | 10 | 12 | 22 | 17 |

# Suppose we have list of 16 elements

a[]

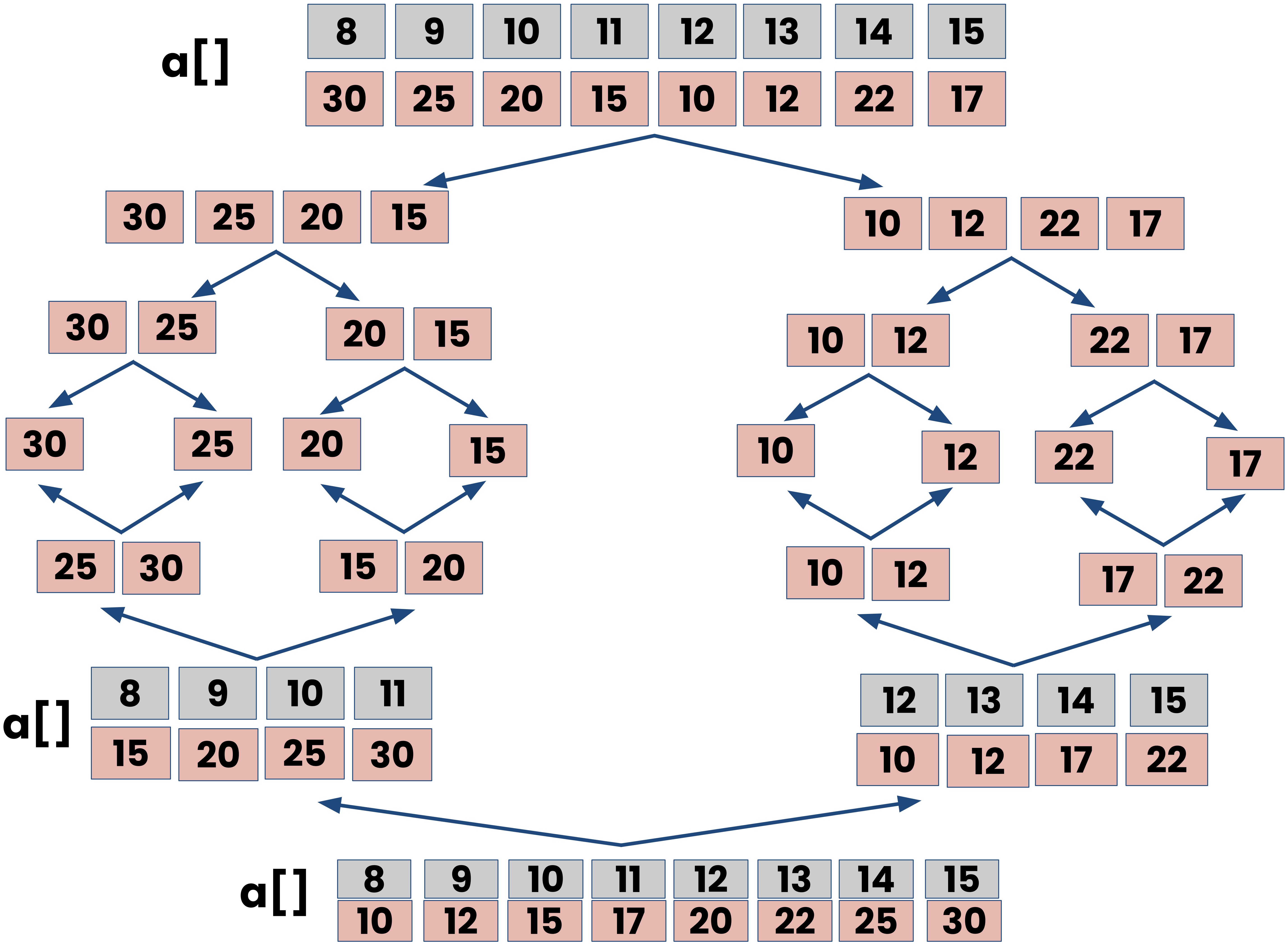
| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 70 | 35 | 25 | 60 | 40 | 18 | 28 | 50 | 30 | 25 | 20 | 15 | 10 | 12 | 22 | 17 |

| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
|---|---|---|---|---|---|---|---|
| 70 | 35 | 25 | 60 | 40 | 18 | 28 | 50 |

| 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 |
|---|---|---|---|---|---|---|---|
| 30 | 25 | 20 | 15 | 10 | 12 | 22 | 17 |

## Consider the indexes from 8 to 15 to understand the merging process

| 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 |
|---|---|---|---|---|---|---|---|
| 30 | 25 | 20 | 15 | 10 | 12 | 22 | 17 |


a[]
8 9 10 11 12 13 14 15
30 25 20 15 10 12 22 17
30 25 20 15
10 12 22 17
30 25
20 15
10 12
22 17
30
25
20
15
10
12
22
17
25 30
15 20
10 12
17 22
a[]
8 9 10 11
15 20 25 30
12 13 14 15
10 12 17 22
a[]
8 9 10 11 12 13 14 15
10 12 15 17 20 22 25 30

# Merging Process:

List L[]

| 0 | 1 | 2 | 3 |
|---|---|---|---|
| 15 | 20 | 25 | 30 |

List R[]

| 0 | 1 | 2 | 3 |
|---|---|---|---|
| 10 | 12 | 17 | 22 |

# Merging Process:

List L[]

| 0 | 1 | 2 | 3 |
|---|---|---|---|
| 15 | 20 | 25 | 30 |

i=0

List R[]

| 0 | 1 | 2 | 3 |
|---|---|---|---|
| 10 | 12 | 17 | 22 |

j=0

# Merging Process:

List L[]

| 0 | 1 | 2 | 3 |
|---|---|---|---|
| 15 | 20 | 25 | 30 |

i=0

List R[]

| 0 | 1 | 2 | 3 |
|---|---|---|---|
| 10 | 12 | 17 | 22 |

j=0

List a[]

| 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 |
|---|---|---|---|---|---|---|---|
| | | | | | | | |

k=8

# Merging Process:

**List L[]**

| 0 | 1 | 2 | 3 |
|---|---|---|---|
| 15 | 20 | 25 | 30 |

i=0

**List R[]**

| 0 | 1 | 2 | 3 |
|---|---|---|---|
| 10 | 12 | 17 | 22 |

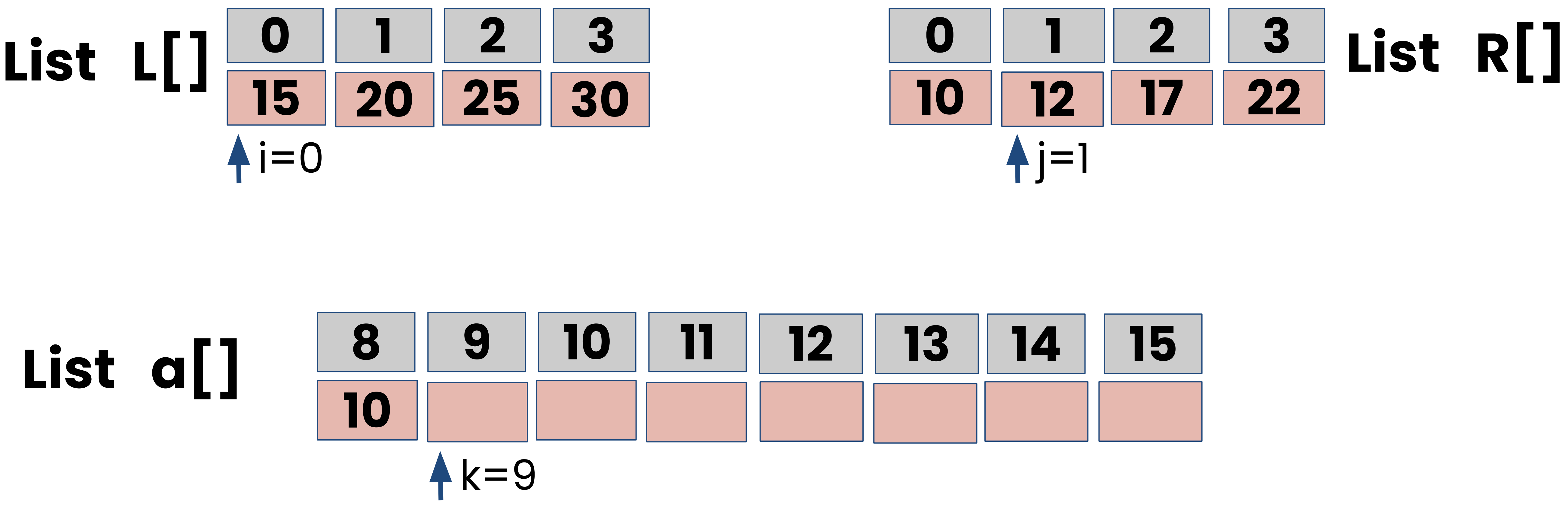
j=0

**List a[]**

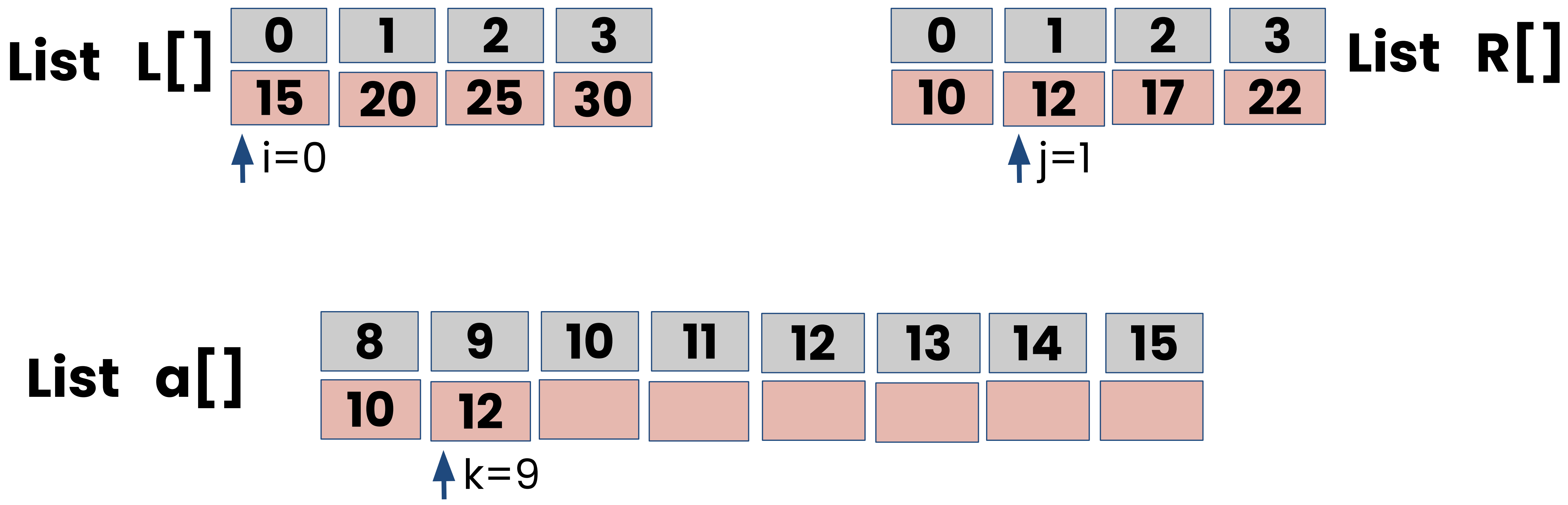
| 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 |
|---|---|---|---|---|---|---|---|
| 10 | | | | | | | |

k=8

# Merging Process:



# Merging Process:



# Merging Process:

List L[]

| 0 | 1 | 2 | 3 |
|---|---|---|---|
| 15 | 20 | 25 | 30 |

i=0

List R[]

| 0 | 1 | 2 | 3 |
|---|---|---|---|
| 10 | 12 | 17 | 22 |

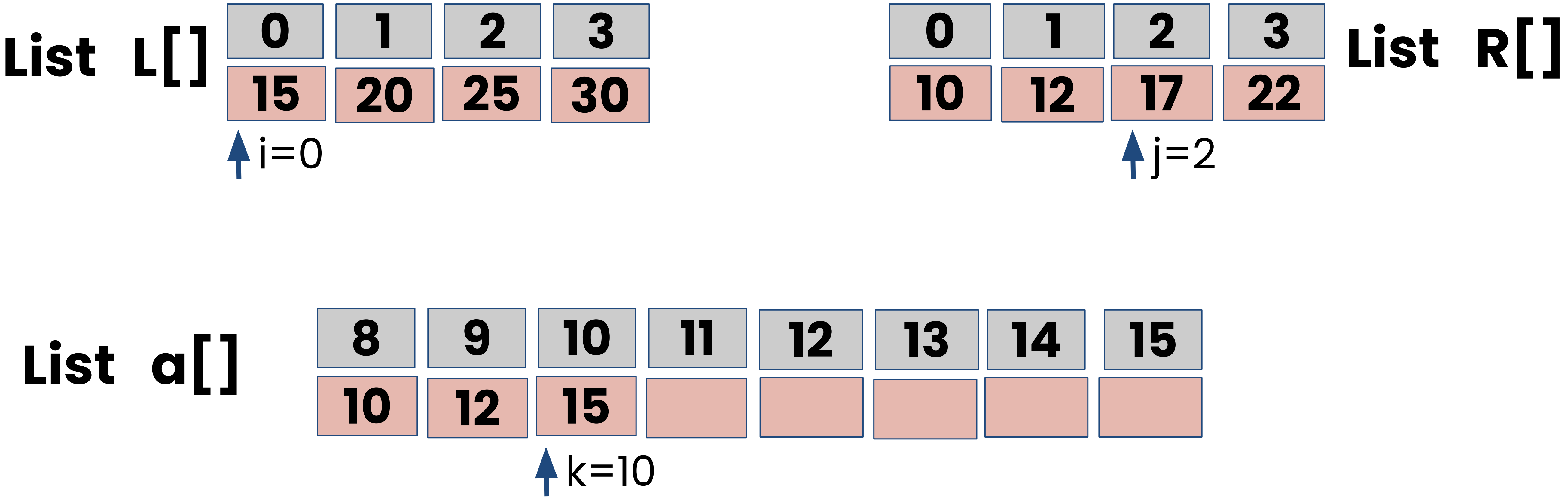
j=2

List a[]

| 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 |
|---|---|---|---|---|---|---|---|
| 10 | 12 | | | | | | |

k=10

# Merging Process:



# Merging Process:

**List L[]**

| 0 | 1 | 2 | 3 |
|---|---|---|---|
| 15 | 20 | 25 | 30 |

i=1

**List R[]**

| 0 | 1 | 2 | 3 |
|---|---|---|---|
| 10 | 12 | 17 | 22 |

j=2

**List a[]**

| 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 |
|---|---|---|---|---|---|---|---|
| 10 | 12 | 15 | | | | | |

k=11

# Merging Process:

**List L[]**

| 0 | 1 | 2 | 3 |
|---|---|---|---|
| 15 | 20 | 25 | 30 |
| | i=1 | | |

**List R[]**

| 0 | 1 | 2 | 3 |
|---|---|---|---|
| 10 | 12 | 17 | 22 |
| | | j=2 | |

**List a[]**

| 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 |
|---|---|---|---|---|---|---|---|
| 10 | 12 | 15 | 17 | | | | |
| | | | k=11 | | | | |

# Merging Process:

List L[]

| 0 | 1 | 2 | 3 |
|---|---|---|---|
| 15 | 20 | 25 | 30 |

i=1

List R[]

| 0 | 1 | 2 | 3 |
|---|---|---|---|
| 10 | 12 | 17 | 22 |

j=3

List a[]

| 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 |
|---|---|---|---|---|---|---|---|
| 10 | 12 | 15 | 17 | | | | |

k=12

# Merging Process:

**List L[]**

| 0 | 1 | 2 | 3 |
|---|---|---|---|
| 15 | 20 | 25 | 30 |

i=1

**List R[]**

| 0 | 1 | 2 | 3 |
|---|---|---|---|
| 10 | 12 | 17 | 22 |

j=3

**List a[]**

| 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 |
|---|---|---|---|---|---|---|---|
| 10 | 12 | 15 | 17 | 20 | | | |

k=12

# Merging Process:

**List L[]**

| 0 | 1 | 2 | 3 |
|---|---|---|---|
| 15 | 20 | 25 | 30 |

i=2

**List R[]**

| 0 | 1 | 2 | 3 |
|---|---|---|---|
| 10 | 12 | 17 | 22 |

j=3

**List a[]**

| 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 |
|---|---|---|---|---|---|---|---|
| 10 | 12 | 15 | 17 | 20 | | | |

k=13

# Merging Process:

List L[]

| 0 | 1 | 2 | 3 |
|---|---|---|---|
| 15 | 20 | 25 | 30 |

i=2

List R[]

| 0 | 1 | 2 | 3 |
|---|---|---|---|
| 10 | 12 | 17 | 22 |

j=3

List a[]

| 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 |
|---|---|---|---|---|---|---|---|
| 10 | 12 | 15 | 17 | 20 | 22 | | |

k=13

# Merging Process:

List L[]

| 0 | 1 | 2 | 3 |
|---|---|---|---|
| 15 | 20 | 25 | 30 |

i=2

List R[]

| 0 | 1 | 2 | 3 |
|---|---|---|---|
| 10 | 12 | 17 | 22 |

j=4

List a[]

| 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 |
|---|---|---|---|---|---|---|---|
| 10 | 12 | 15 | 17 | 20 | 22 | | |

k=14

# Merging Process:

**List L[]**

| 0 | 1 | 2 | 3 |
|---|---|---|---|
| 15 | 20 | 25 | 30 |

i=2

| 0 | 1 | 2 | 3 |
|---|---|---|---|
| 10 | 12 | 17 | 22 |

**List R[]**

j=4

**as j==4** , means List R[] has been completely scanned ,

now **copy** the remaining elements of List L[] to List a[]

**List a[]**

| 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 |
|---|---|---|---|---|---|---|---|
| 10 | 12 | 15 | 17 | 20 | 22 | | |

k=14

# Merging Process:

**List L[]**

| 0 | 1 | 2 | 3 |
|---|---|---|---|
| 15 | 20 | 25 | 30 |

i=2

**List R[]**

| 0 | 1 | 2 | 3 |
|---|---|---|---|
| 10 | 12 | 17 | 22 |

j=4

**copy L[2] to a[14]**

**List a[]**

| 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 |
|---|---|---|---|---|---|---|---|
| 10 | 12 | 15 | 17 | 20 | 22 | | |

k=14

# Merging Process:

List L[]

| 0 | 1 | 2 | 3 |
|---|---|---|---|
| 15 | 20 | 25 | 30 |

i=2

| 0 | 1 | 2 | 3 |
|---|---|---|---|
| 10 | 12 | 17 | 22 |

List R[]

j=4

copy L[2] to a[14]

List a[]

| 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 |
|---|---|---|---|---|---|---|---|
| 10 | 12 | 15 | 17 | 20 | 22 | 25 | |

k=14

# Merging Process:

List L[]

| 0 | 1 | 2 | 3 |
|---|---|---|---|
| 15 | 20 | 25 | 30 |

i=2

| 0 | 1 | 2 | 3 |
|---|---|---|---|
| 10 | 12 | 17 | 22 |

List R[]

j=4

**Increment pointer i & k**

List a[]

| 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 |
|---|---|---|---|---|---|---|---|
| 10 | 12 | 15 | 17 | 20 | 22 | 25 | |

k=14

# Merging Process:

List L[]

| 0 | 1 | 2 | 3 |
|---|---|---|---|
| 15 | 20 | 25 | 30 |

i=3

List R[]

| 0 | 1 | 2 | 3 |
|---|---|---|---|
| 10 | 12 | 17 | 22 |

j=4

**Increment pointer i & k**

List a[]

| 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 |
|---|---|---|---|---|---|---|---|
| 10 | 12 | 15 | 17 | 20 | 22 | 25 | |

k=15

# Merging Process:

**List L[]**

| 0 | 1 | 2 | 3 |
|---|---|---|---|
| 15 | 20 | 25 | 30 |

i=3

**List R[]**

| 0 | 1 | 2 | 3 |
|---|---|---|---|
| 10 | 12 | 17 | 22 |

j=4

**copy L[3] to a[15]**

**List a[]**

| 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 |
|---|---|---|---|---|---|---|---|
| 10 | 12 | 15 | 17 | 20 | 22 | 25 | |

k=15

# Merging Process:

List L[]

| 0 | 1 | 2 | 3 |
|---|---|---|---|
| 15 | 20 | 25 | 30 |

i=3

List R[]

| 0 | 1 | 2 | 3 |
|---|---|---|---|
| 10 | 12 | 17 | 22 |

j=4

**copy L[3] to a[15]**

List a[]

| 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 |
|---|---|---|---|---|---|---|---|
| 10 | 12 | 15 | 17 | 20 | 22 | 25 | 30 |

k=15

## Merging Process:

List L[]

| 0 | 1 | 2 | 3 |
|---|---|---|---|
| 15 | 20 | 25 | 30 |

i=4

List R[]

| 0 | 1 | 2 | 3 |
|---|---|---|---|
| 10 | 12 | 17 | 22 |

j=4

**Increment pointer i & k**

List a[]

| 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 |
|---|---|---|---|---|---|---|---|
| 10 | 12 | 15 | 17 | 20 | 22 | 25 | 30 |

k=16

# Merging Process:

List L[]

| 0 | 1 | 2 | 3 |
|---|---|---|---|
| 15 | 20 | 25 | 30 |

i=4

List R[]

| 0 | 1 | 2 | 3 |
|---|---|---|---|
| 10 | 12 | 17 | 22 |

j=4

**as i==4** , means List L[ ] has also been completely scanned .

List a[]

| 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 |
|---|---|---|---|---|---|---|---|
| 10 | 12 | 15 | 17 | 20 | 22 | 25 | 30 |

k=16

# Performance Analysis

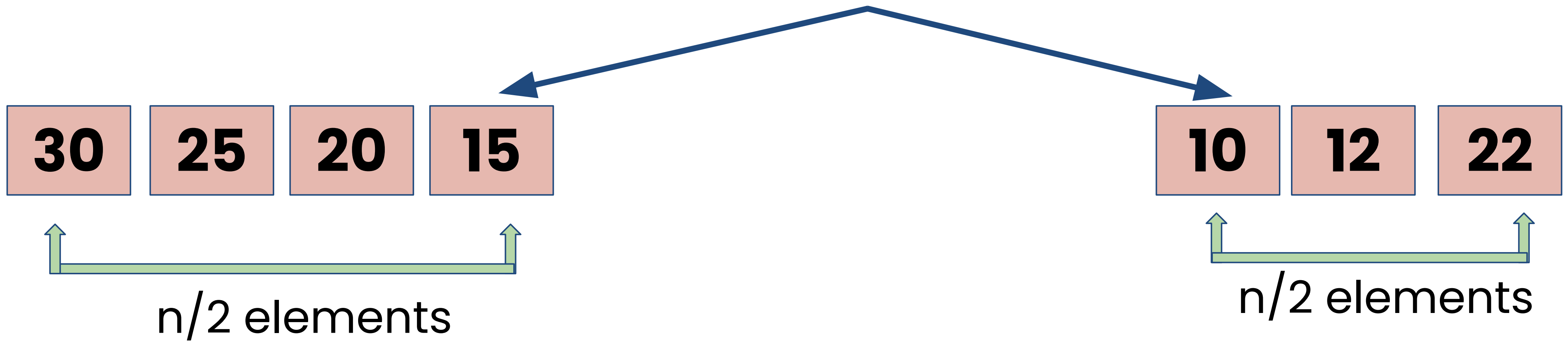
| Index | 0 | 1 | 2 | 3 | 4 | 5 | 6 |
|---|---|---|---|---|---|---|---|
| Data Item | 30 | 25 | 20 | 15 | 10 | 12 | 22 |

30 25 20 15

10 12 22

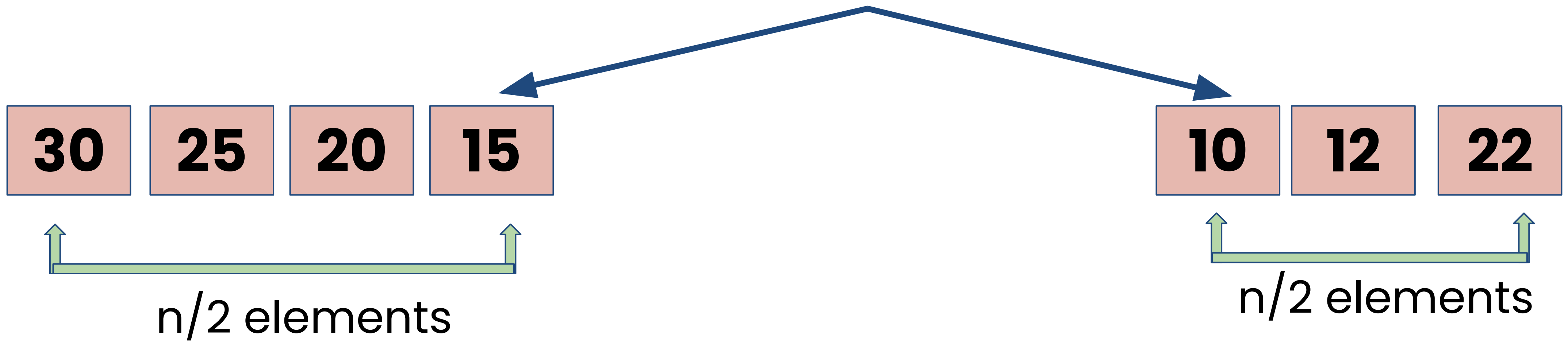
# Performance Analysis

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| **Index** | 0 | 1 | 2 | 3 | 4 | 5 | 6 |
| **Data Item** | 30 | 25 | 20 | 15 | 10 | 12 | 22 |



# Performance Analysis

| Index | 0 | 1 | 2 | 3 | 4 | 5 | 6 |
|---|---|---|---|---|---|---|---|
| Data Item | 30 | 25 | 20 | 15 | 10 | 12 | 22 |




**Recurrence Relation**

$T(n) = T(n/2) + T(n/2) + n$

$T(n) = 2T(n/2) + n$

$T(n) = \mathbf{O(n\log n)}$

# END