Tinker Academy

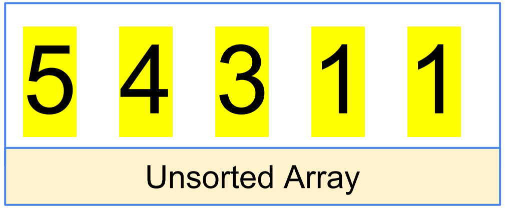
Programming Using Java
(Analysis Insertion Sort)

Spring 2015

Worst Case

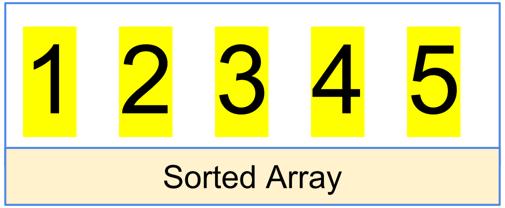
Insertion Sort

Simple Sorting Algorithm



5 4 3 1 1

Unsorted Array



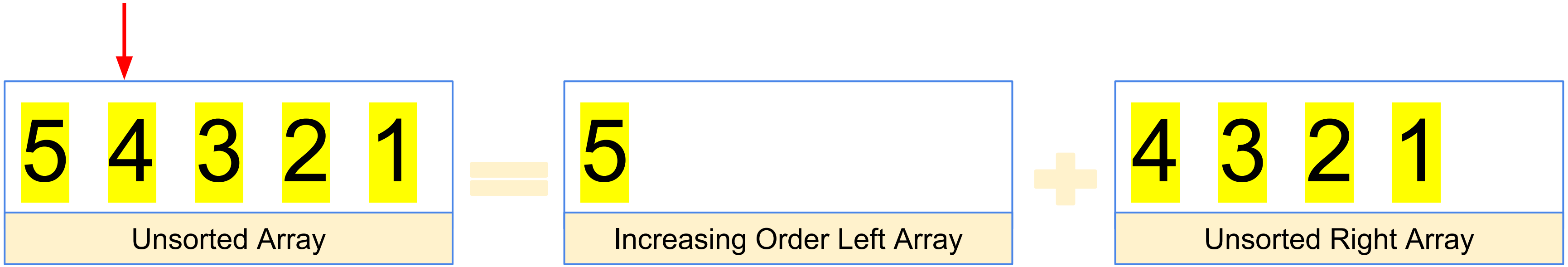
1 2 3 4 5

Sorted Array

Sorting is "slow", which means it takes more cpu time to complete sort

Insertion Sort

Uses a Pivot just like Bubble Sort. Pivot Starts at index 1



5 4 3 2 1

Unsorted Array

5

Increasing Order Left Array

4 3 2 1

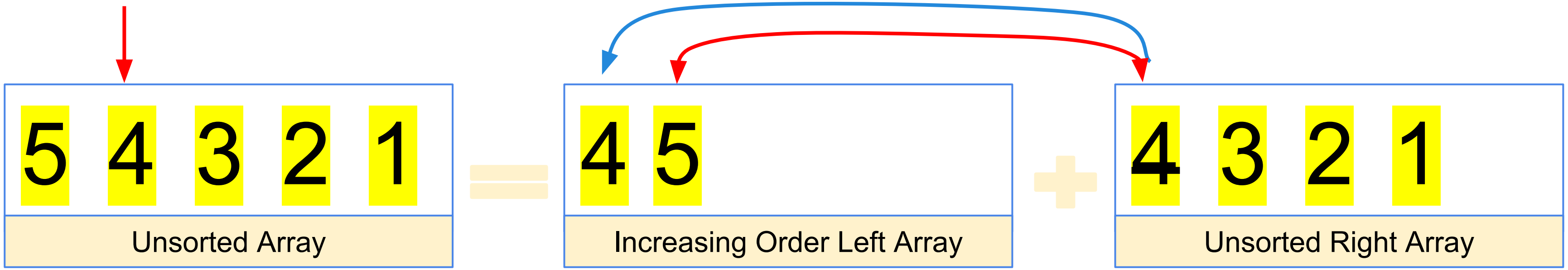
Unsorted Right Array

Pivot divides the array into 2, a left subarray in increasing order

Pivot divides the array into 2, an unsorted right subarray

Insertion Sort

Insertion Sort inserts the pivot item into left subarray but maintains order



5 4 3 2 1

Unsorted Array

4 5

Increasing Order Left Array

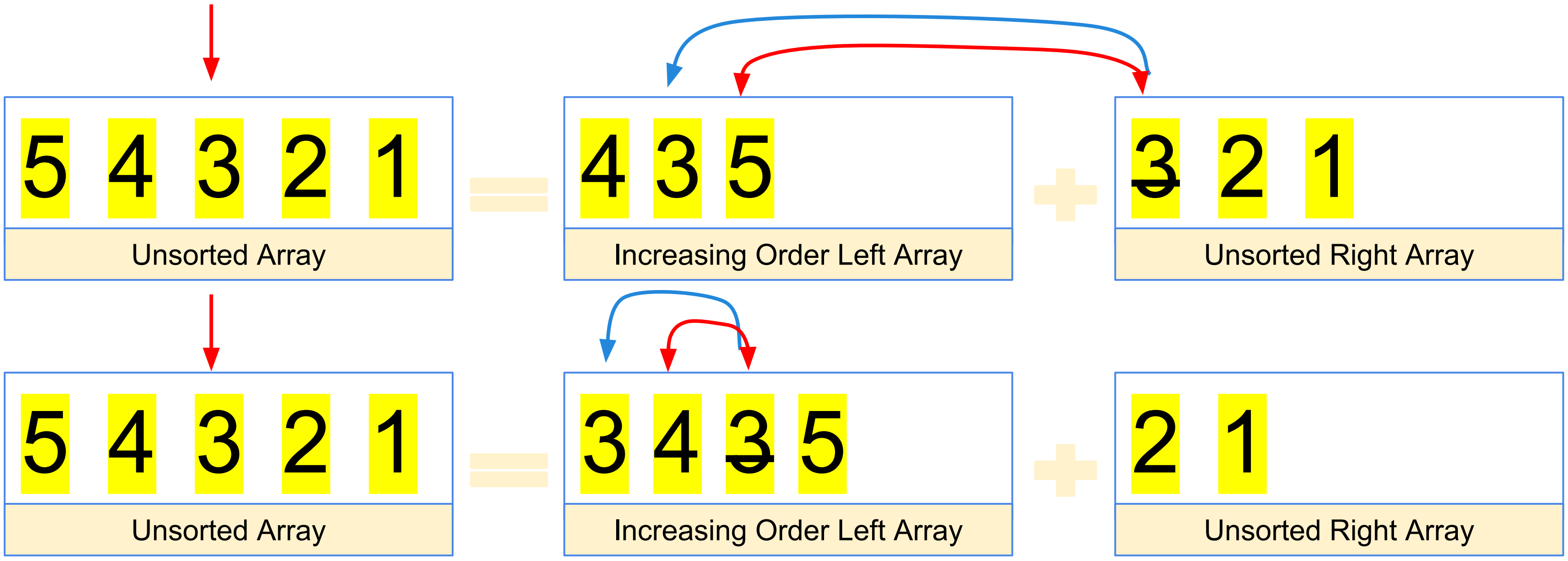
4 3 2 1

Unsorted Right Array

# Comparisons = 1

Insertion Sort

Insertion Sort inserts the pivot item into left subarray but maintains order



5 4 3 2 1

Unsorted Array

4 3 5

Increasing Order Left Array

~~3~~ 2 1

Unsorted Right Array

5 4 3 2 1

Unsorted Array

3 4 ~~3~~ 5

Increasing Order Left Array

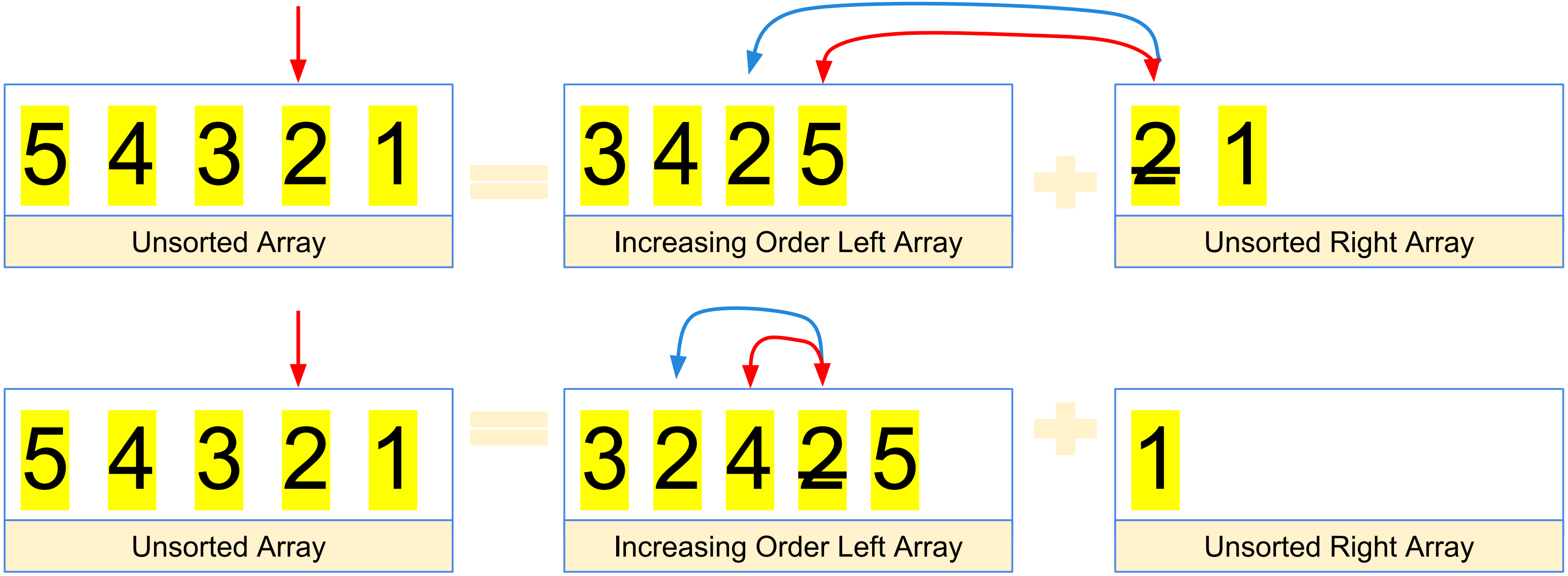
2 1

Unsorted Right Array

# Comparisons = 2

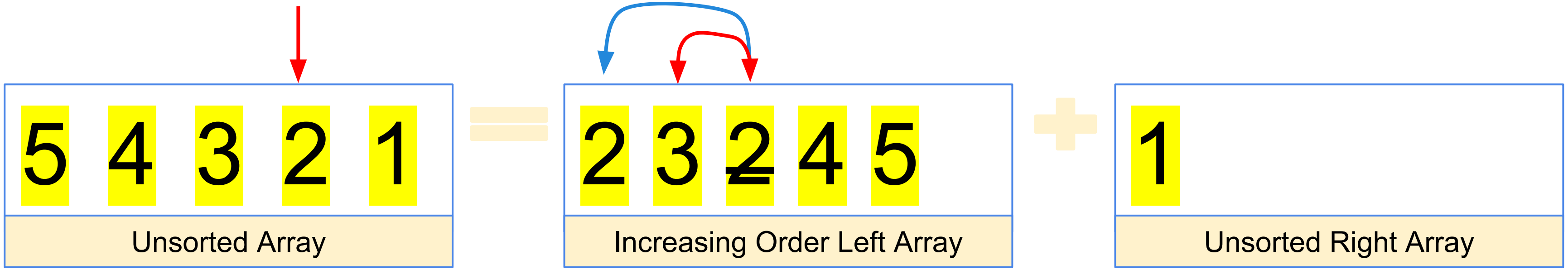
Insertion Sort

Insertion Sort inserts the pivot item into left subarray but maintains order



Insertion Sort

Insertion Sort inserts the pivot item into left subarray but maintains order



5 4 3 2 1

Unsorted Array

=

2 3 2 4 5

Increasing Order Left Array

+

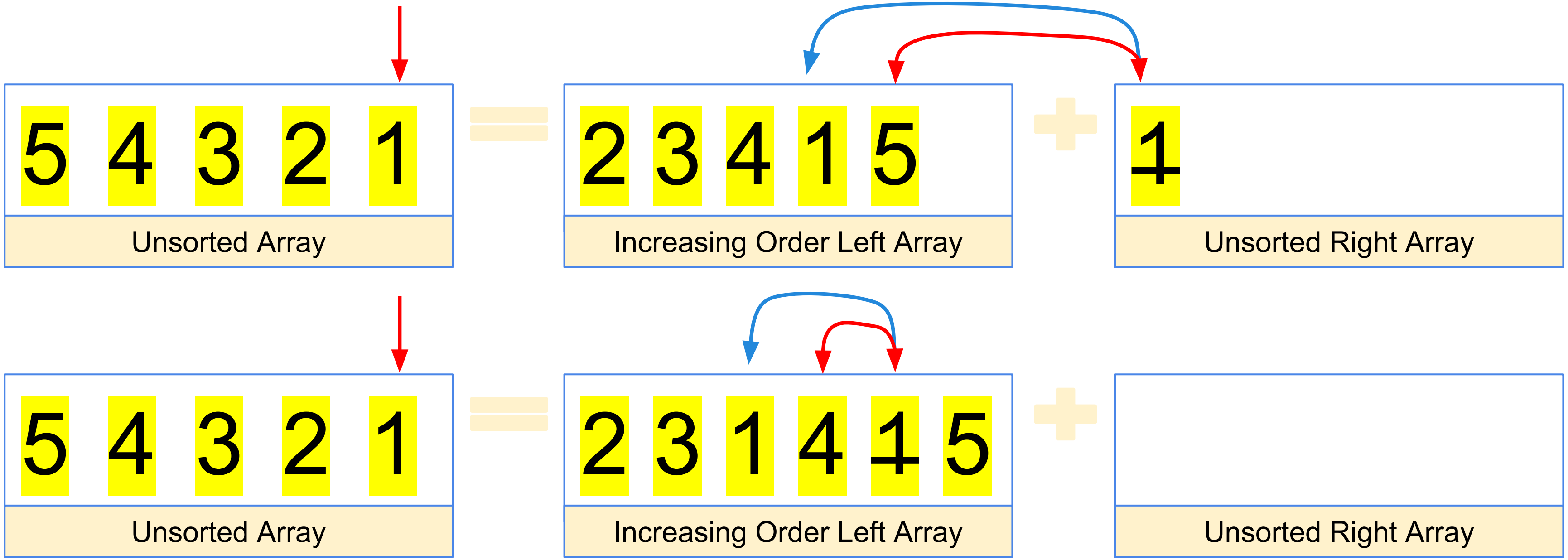
1

Unsorted Right Array

# Comparisons = 3

Insertion Sort

Insertion Sort inserts the pivot item into left subarray but maintains order



5 4 3 2 1

Unsorted Array

=

2 3 4 1 5

Increasing Order Left Array

+

~~1~~

Unsorted Right Array

5 4 3 2 1

Unsorted Array

=

2 3 1 4 ~~1~~ 5

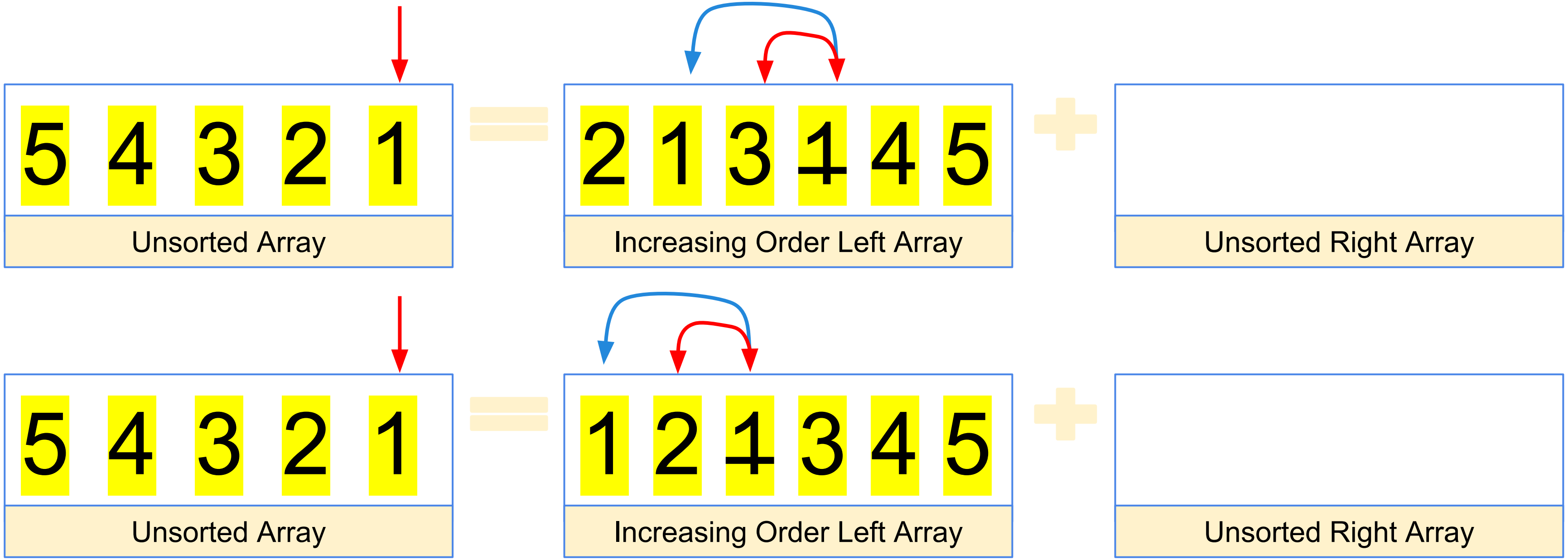
Increasing Order Left Array

+

Unsorted Right Array

Insertion Sort

Insertion Sort inserts the pivot item into left subarray but maintains order



5 4 3 2 1

Unsorted Array

2 1 3 1 4 5

Increasing Order Left Array

Unsorted Right Array

5 4 3 2 1

Unsorted Array

1 2 1 3 4 5

Increasing Order Left Array

Unsorted Right Array

# Comparisons = 4

Insertion Sort

Insertion Sort inserts the pivot item into left subarray but maintains order

Worst Case Array of Size 5

# Comparisons = 1 + 2 + 3 + 4

Worst Case Array of Size N

# Comparisons = 1 + 2 + 3 + ... + (N-1) = N(N-1)/2 ~= N*N

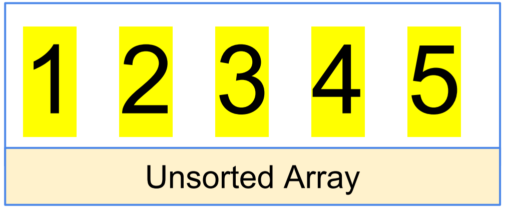
Worst Case Array of Size 1000000 (1 million)

# Comparisons = 1 + 2 + 3 + ... + (N-1) ~= 1 Trillion!

Best Case

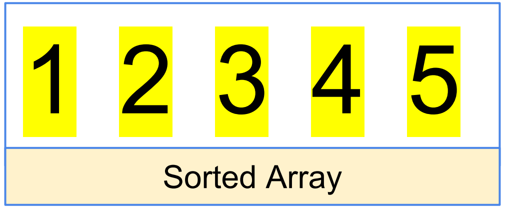
Insertion Sort

Simple Sorting Algorithm



1 2 3 4 5

Unsorted Array



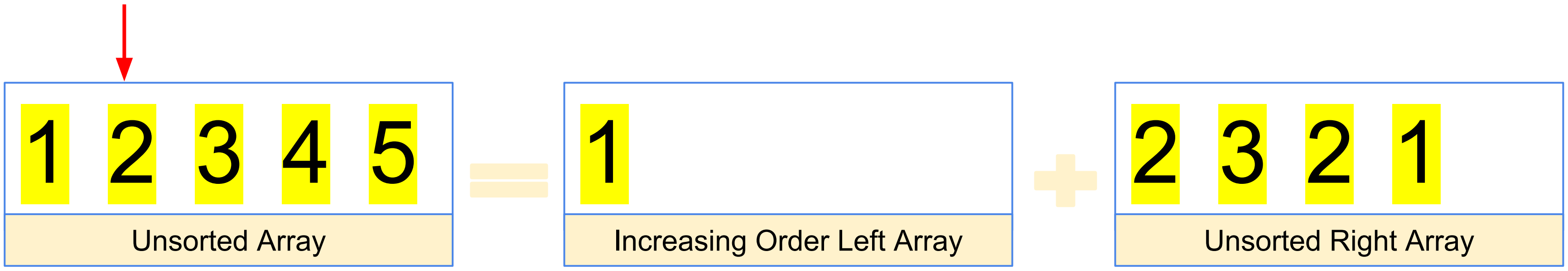
1 2 3 4 5

Sorted Array

Sorting is "slow", which means it takes more cpu time to complete sort

Insertion Sort

Uses a Pivot just like Bubble Sort. Pivot Starts at index 1



1 2 3 4 5

Unsorted Array

1

Increasing Order Left Array

2 3 2 1

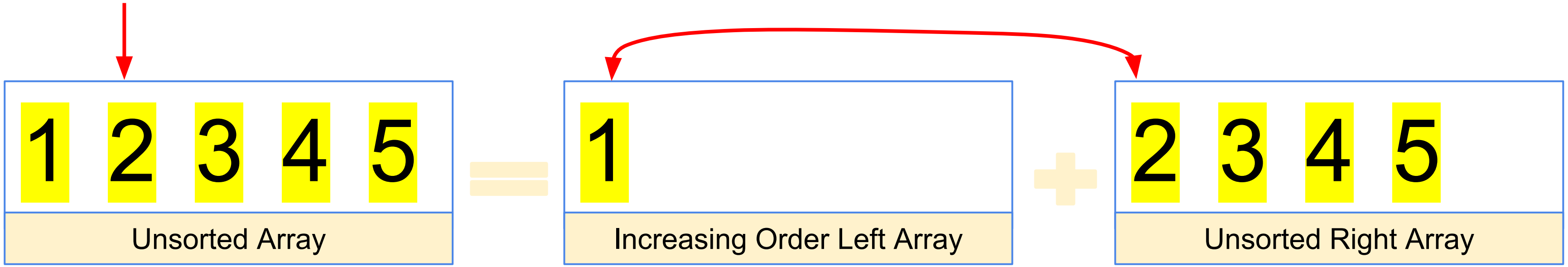
Unsorted Right Array

Pivot divides the array into 2, a left subarray in increasing order

Pivot divides the array into 2, an unsorted right subarray

Insertion Sort

Insertion Sort inserts the pivot item into left subarray but maintains order



1 2 3 4 5

Unsorted Array

1

Increasing Order Left Array

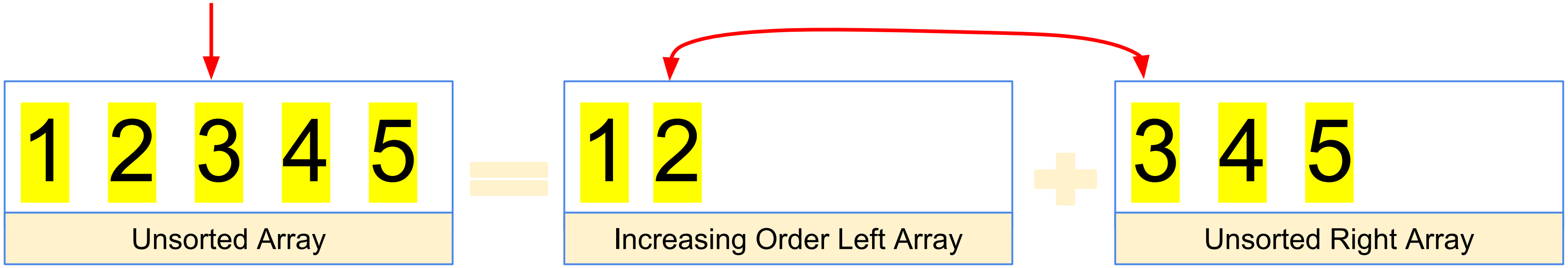
2 3 4 5

Unsorted Right Array

# Comparisons = 1

Insertion Sort

Insertion Sort inserts the pivot item into left subarray but maintains order



1 2 3 4 5

Unsorted Array

1 2

Increasing Order Left Array

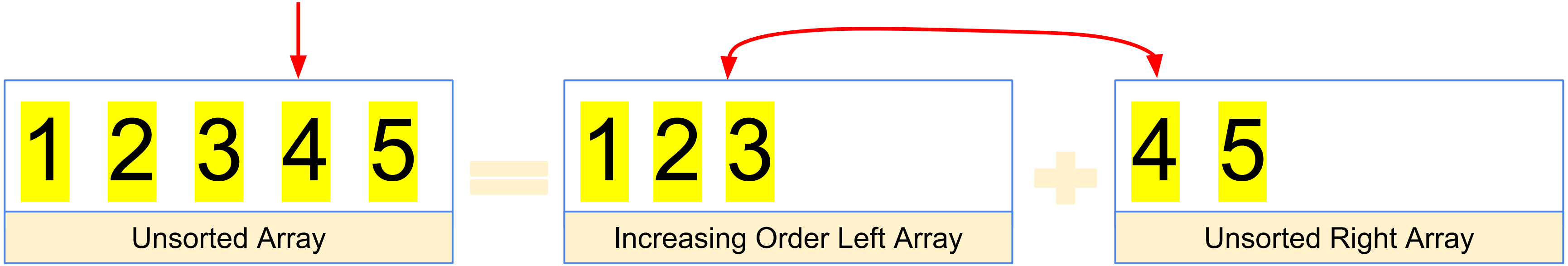
3 4 5

Unsorted Right Array

# Comparisons = 1

Insertion Sort

Insertion Sort inserts the pivot item into left subarray but maintains order



1 2 3 4 5

Unsorted Array

1 2 3

Increasing Order Left Array

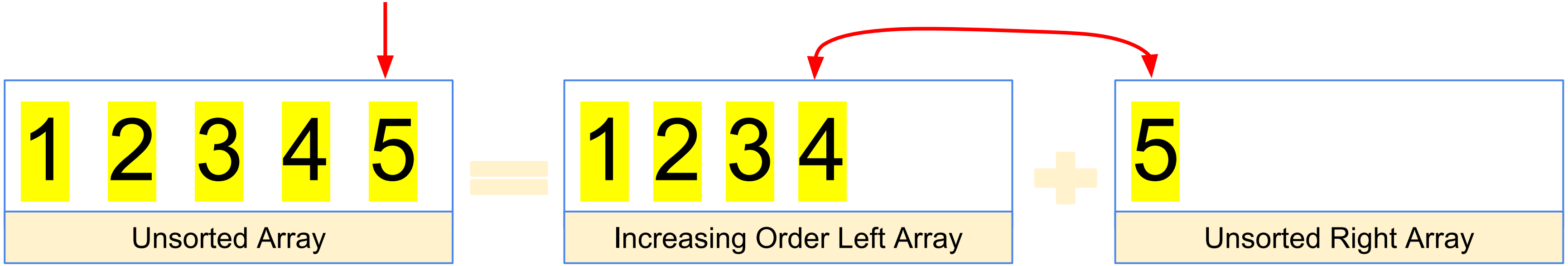
4 5

Unsorted Right Array

# Comparisons = 1

Insertion Sort

Insertion Sort inserts the pivot item into left subarray but maintains order



1 2 3 4 5

Unsorted Array

1 2 3 4

Increasing Order Left Array

5

Unsorted Right Array

# Comparisons = 1

Insertion Sort

Insertion Sort inserts the pivot item into left subarray but maintains order

Best Case Array of Size 5

# Comparisons = 1 + 1 + 1 + 1

Best Case Array of Size N

# Comparisons = 1 + 1 + 1 + ... + (1) = N-1

Best Case Array of Size 1000000 (1 million)

# Comparisons = 1 + 2 + 3 + ... + (N-1) ~= 1 million

Average Case

Insertion Sort

Insertion Sort inserts the pivot item into left subarray but maintains order

Average Case Array of Size N

# Comparisons ~= N*N

Proof requires understanding probability (indicator random variables)