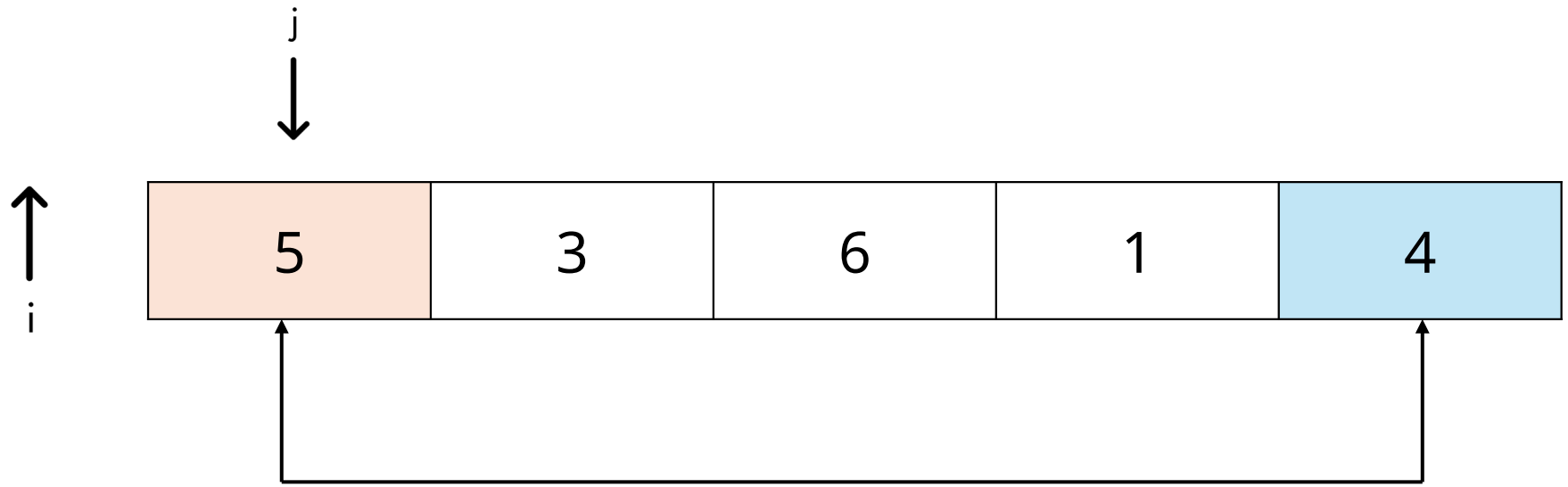
Step 1

Initial pointers: i = -1, j = 0, low = 0, high = 4

Since, arr[0] == pivot
Swap arr[j] with arr[high]

Pivot

5



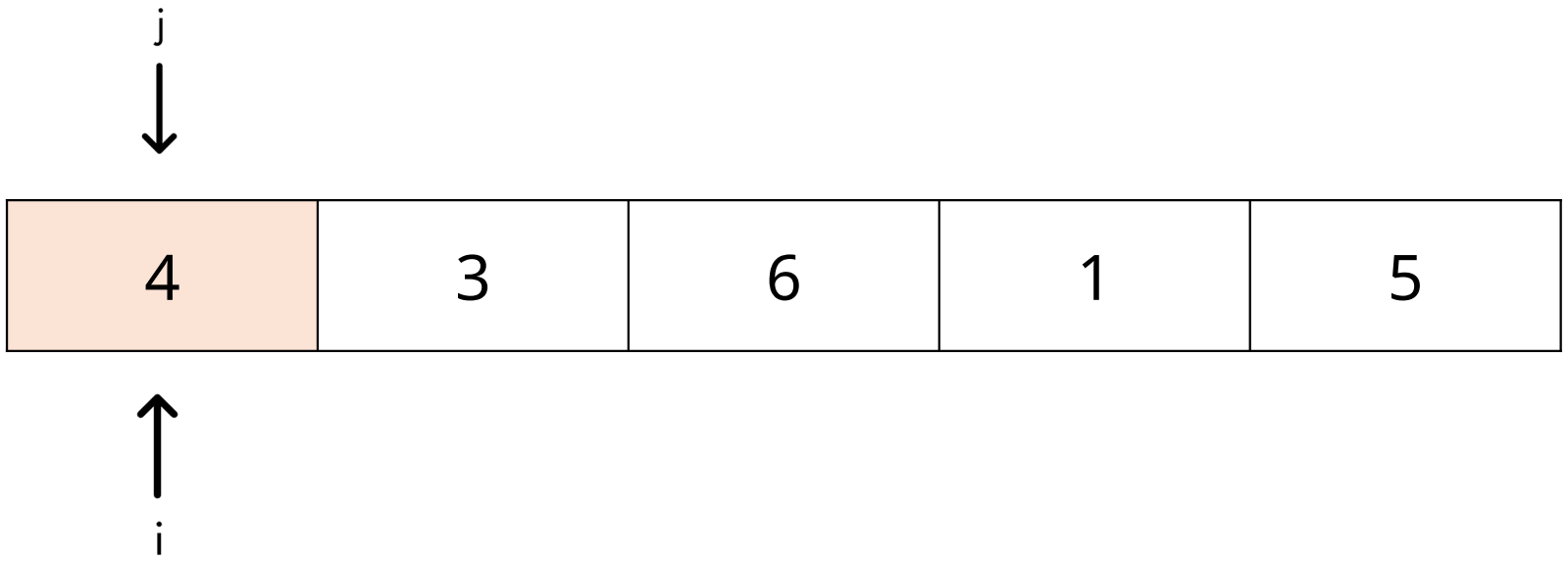
Step 1

Pointers: i = -1, j = 0, low = 0, high = 4

Since arr[0] < pivot
Increase i, i = 0
Since i == j, Skip swapping
Increase j

Pivot

5



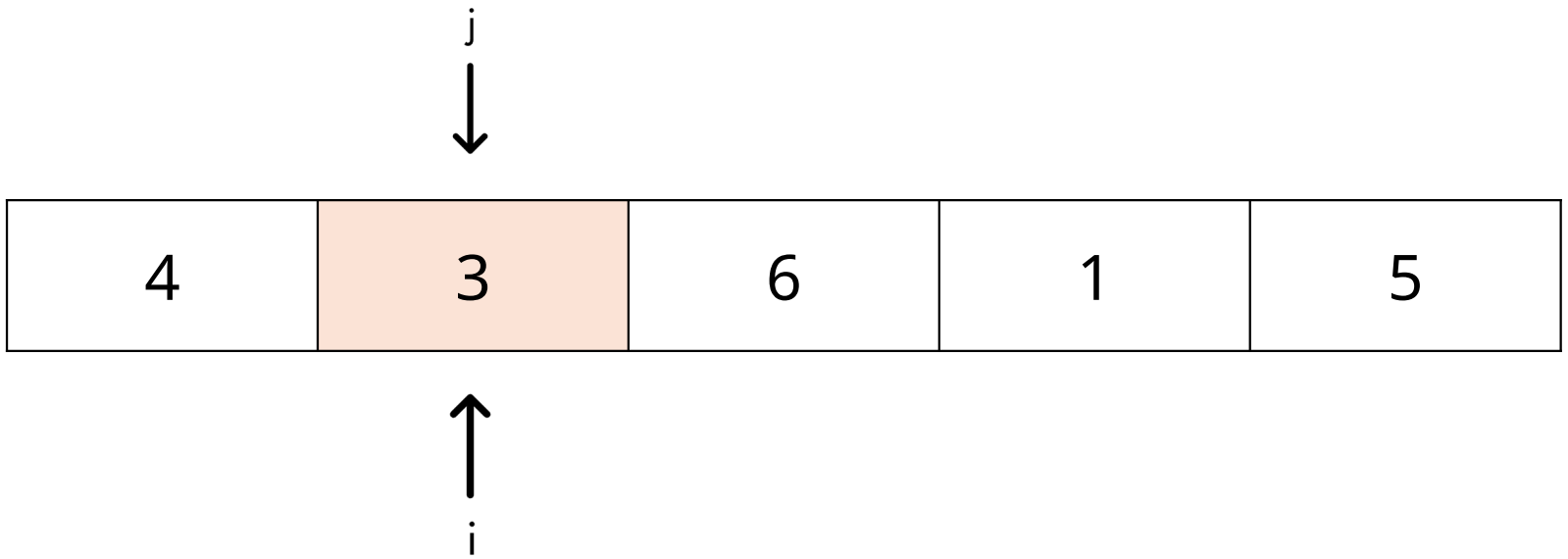
Step 2

Pointers: i = 0, j = 1, low = 0, high = 4

Since arr[1] < pivot
Increase i, i = 1
Since i == j, Skip swapping
Increase j

Pivot

5



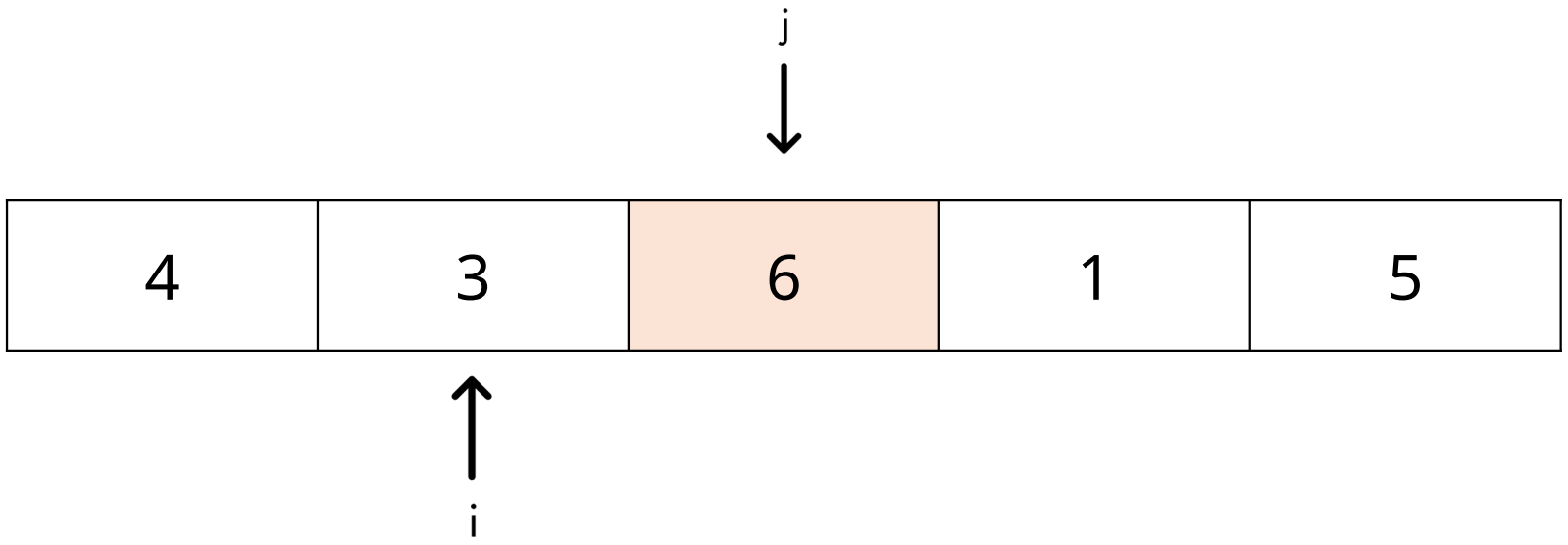
Step 3

Pointers: i = 1, j = 2, low = 0, high = 4

Since arr[2] > pivot
Skip swapping
Increase j

Pivot

5

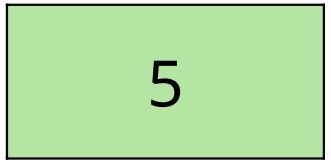


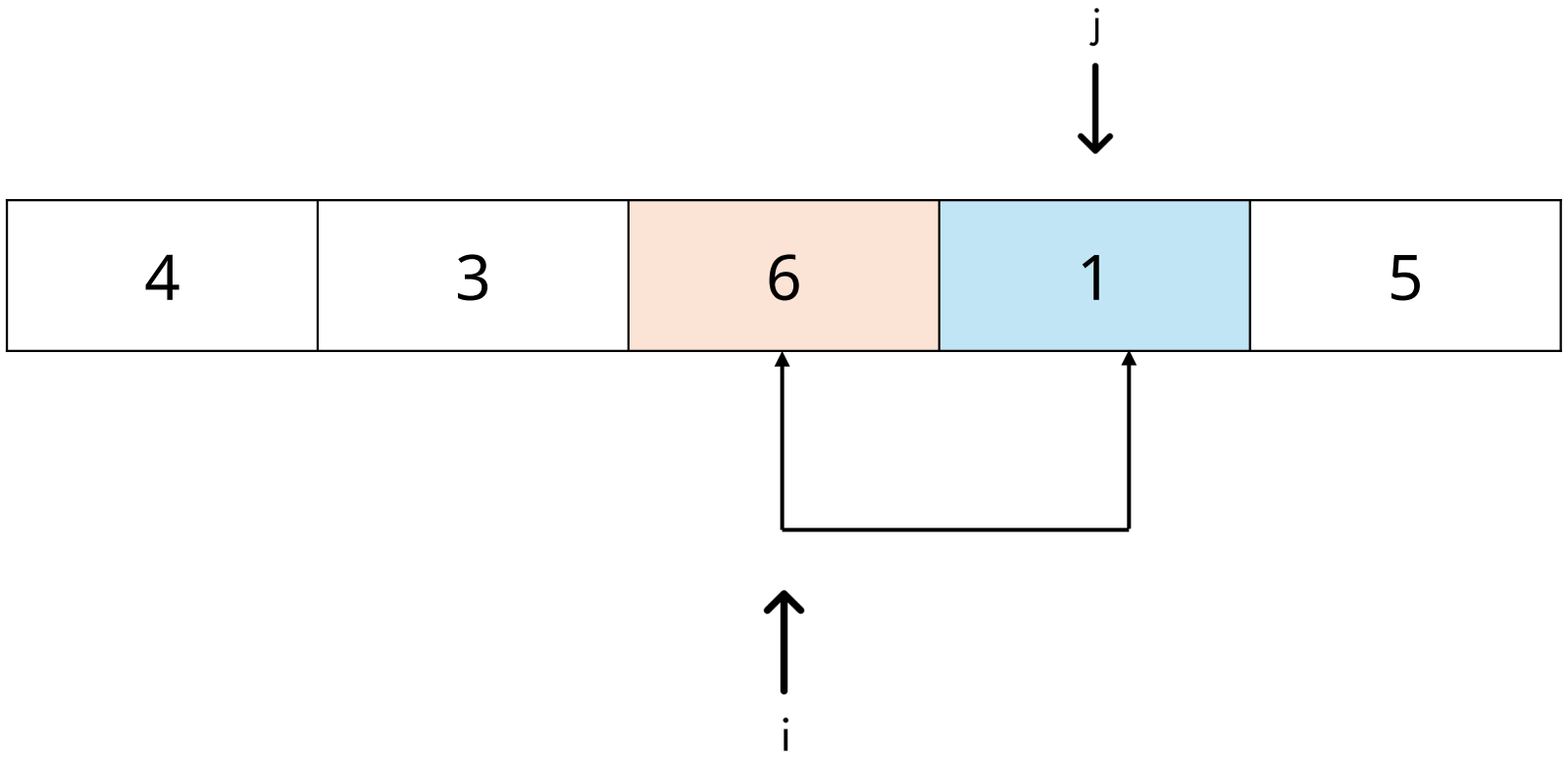
Step 4

Pointers: i = 1, j = 3, low = 0, high = 4

Since arr[3] < pivot
Increase i, i = 2
Since i != j,
Swap arr[2] and arr[3]
Increase j

Pivot





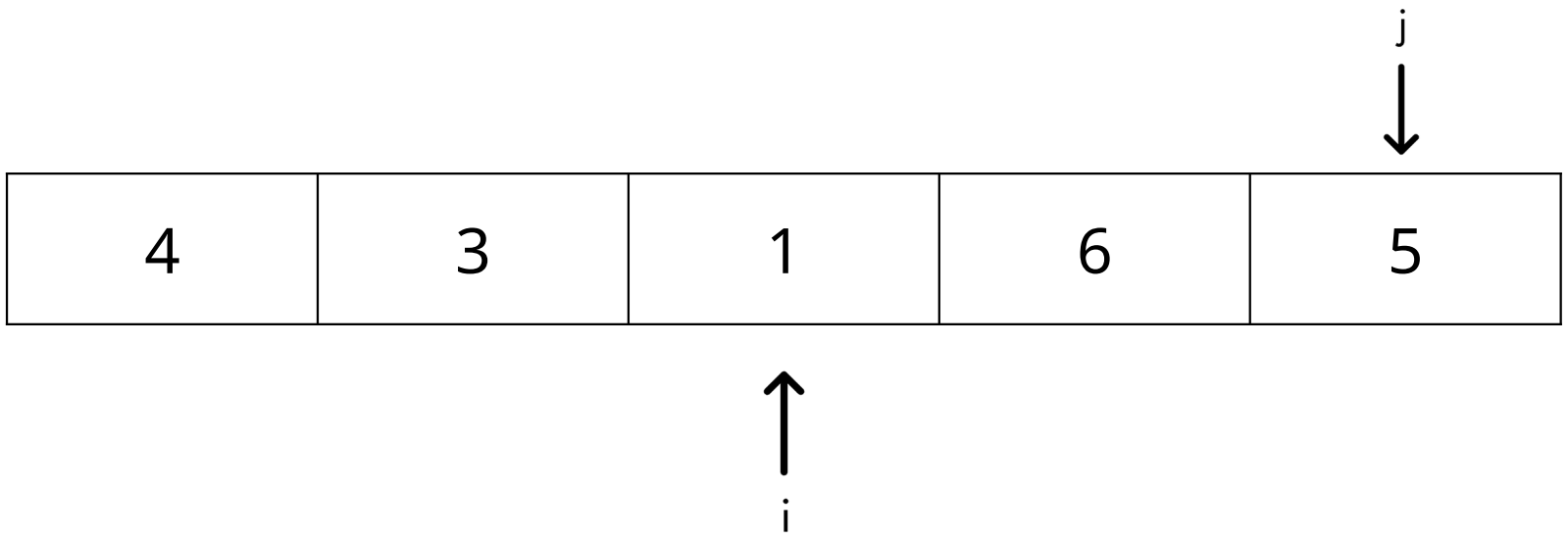
Step 5

Pointers: i = 2, j = 4, low = 0, high = 4

Since j < high is false,
Exit loop

Pivot

5

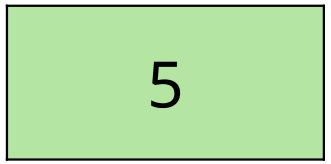


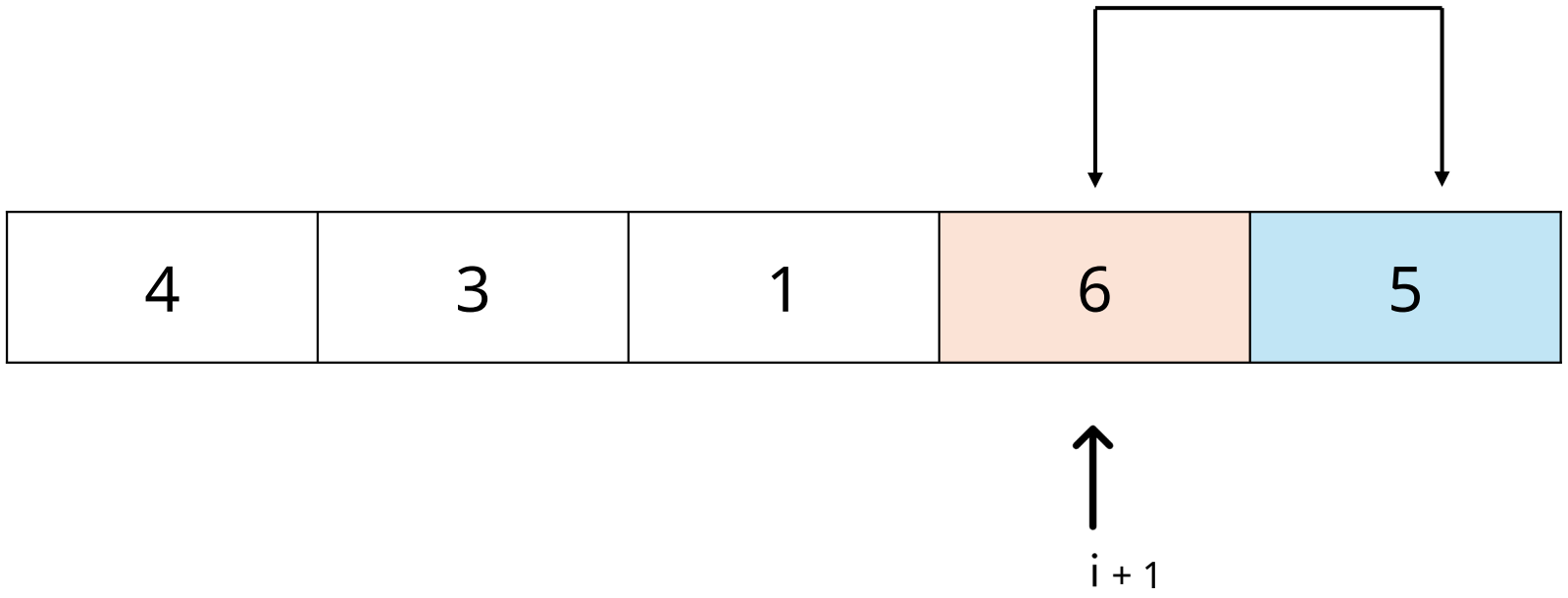
Step 5

Pointers: i = 2, j = 4, low = 0, high = 4

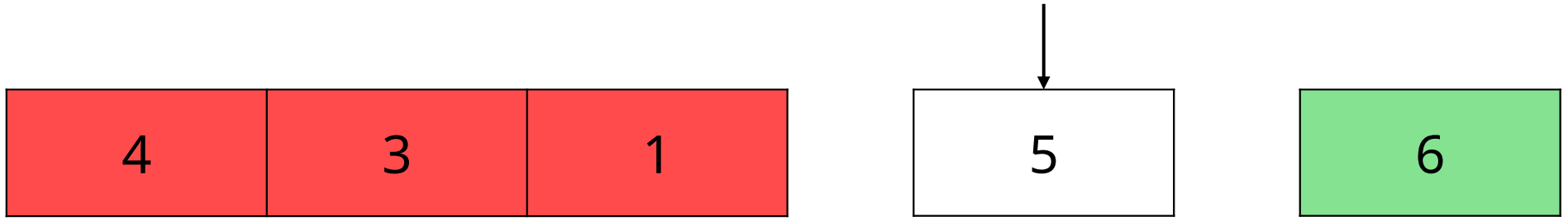
Since, i + 1 != high,
Swap arr[i + 1] and arr[high]

Pivot





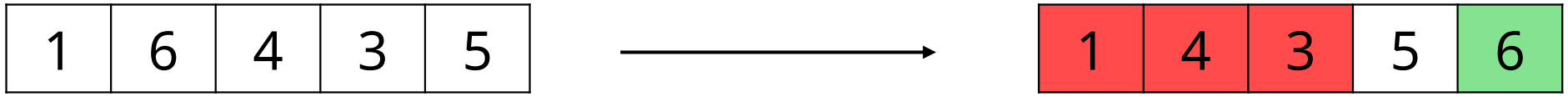
Continue ⏪ Now array is divided to two parts



1 - Return index of the pivot, which is 3

2 – Get array[3] as pivot, which is 5 again.

3 – Divide reflection array with pivot



Do same steps with calling function again

For left side use range: [low, pivotIndex – 1]

For right side use range: [pivotIndex + 1, high]

Recursion ◀◀ Steps of recursion