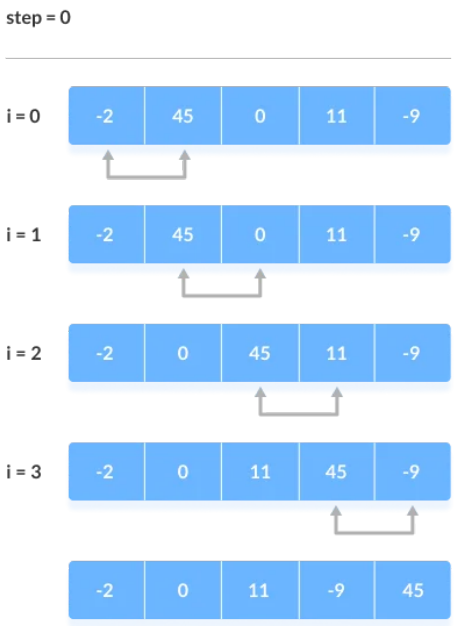
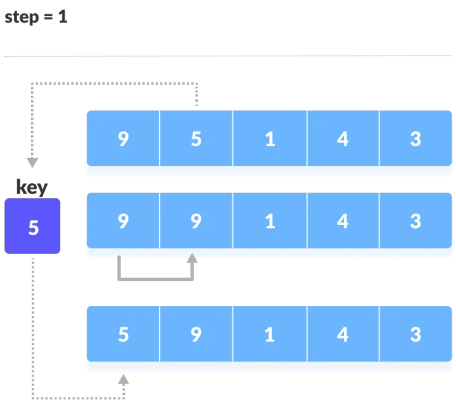
Sorting algorithms

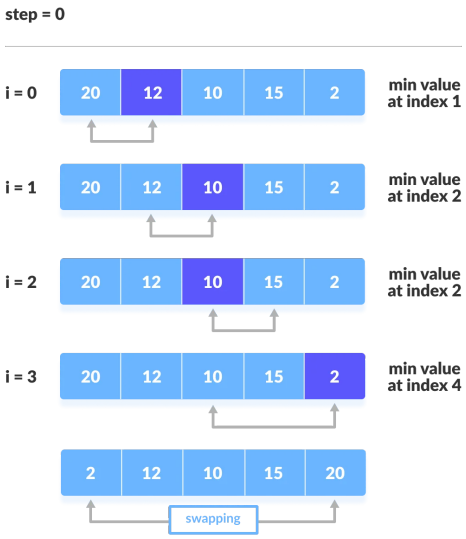
	Bubble sort
Data input	An array of n elements
Sorting variations	Ascending and descending
First loop	For loop, go through the whole array
Second loop	For logo, comparing array elements
Conditions	Swap only if greater element is on left side
Data output	A sorted array
Ascending	Go through the array and send greatest element to the last
Descending	Only change the comparison in the if condition
Algorithmic complexity	$O(n^2)$
Special cases	Think of it as bubbles rising to the top because they have small values



	Insertion sort
Data input	Array of n elements
Sorting variations	Ascending and descending
First loop	Go through the array and define the key element
Second loop	With the key element, check for smaller elements
Conditions	When smaller element is found, insert key element
Data output	Sorted array
Ascending	
Descending	Only change $key < array[j]$ to $key > array[j]$
Algorithmic complexity	$O(n^2)$
Special cases	



Selection sort
Data input
Sorting variations
First loop
Second loop
Conditions
Data output
Ascending
Descending
Algorithmic complexity
Special cases



	Merge sort
Data input	
Sorting variations	
First loop	
Second loop	
Conditions	
Data output	
Ascending	
Descending	
Algorithmic complexity	
Special cases	

Heap sort
Data input
Sorting variations
First loop
Second loop
Conditions
Data output
Ascending
Descending
Algorithmic complexity
Special cases

Quick sort
Data input
Sorting variations
First loop
Second loop
Conditions
Data output
Ascending
Descending
Algorithmic complexity
Special cases