Bubble Sort Algorithm

In this tutorial, you will learn how bubble sort works. Also, you will find working examples of bubble sort in C, C++, Java and Python.

Bubble sort is an algorithm that compares the adjacent elements and swaps their positions if they are not in the intended order. The order can be ascending or descending.

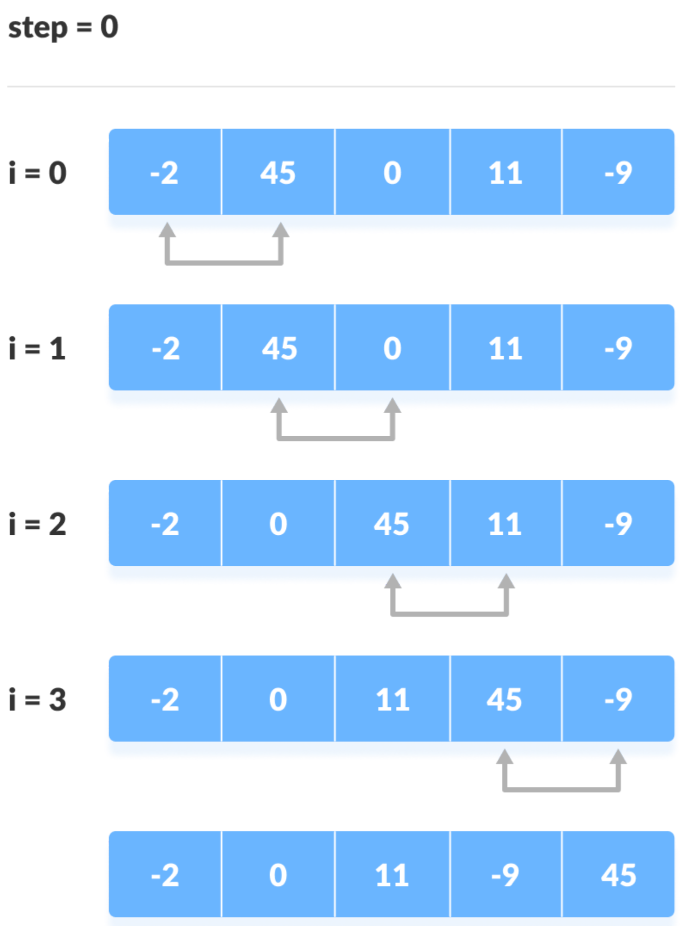
How Bubble Sort Works?

1. Starting from the first index, compare the first and the second elements.If the first element is greater than the second element, they are swapped.

   Now, compare the second and the third elements. Swap them if they are not in order.

Contents

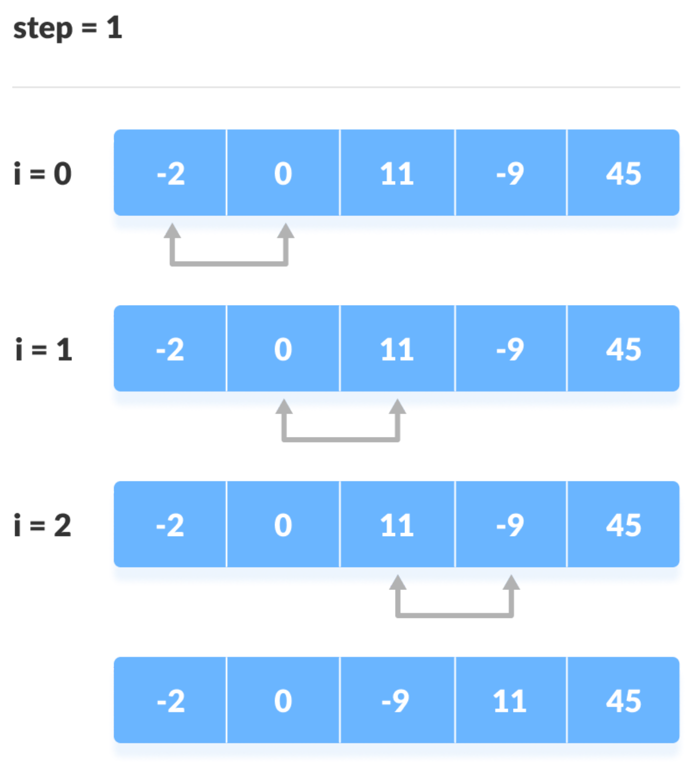
The above process goes on until the last element.



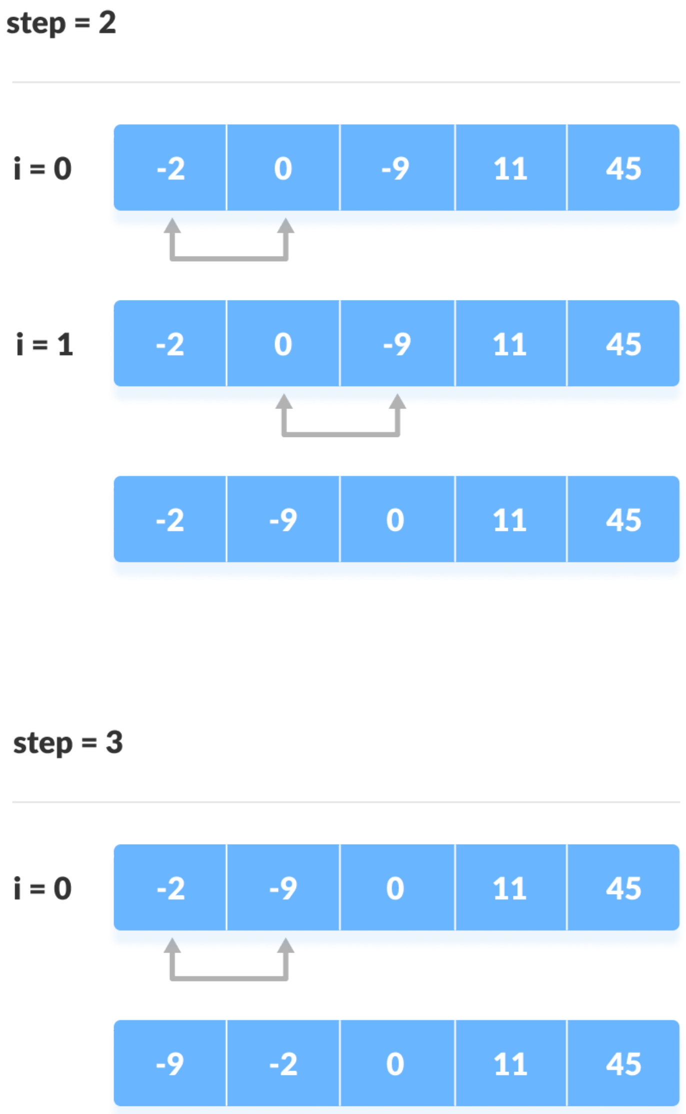
2. The same process goes on for the remaining iterations. After each iteration, the largest element among the unsorted elements is placed at the end.

   In each iteration, the comparison takes place up to the last unsorted element.

   The array is sorted when all the unsorted eleme' Contents ' their correct positions.



Contents



Bubble Sort Algorithm

Contents

```
bubbleSort(array)
  for i <- 1 to indexOfLastUnsortedElement-1
    if leftElement > rightElement
      swap leftElement and rightElement
end bubbleSort
```

Python, Java and C/C++ Examples

Python Java C C+

```
# Bubble sort in Python

def bubbleSort(array):
    for i in range(len(array)):
        for j in range(0, len(array) - i - 1):

            # To sort in descending order, change > to < in this line.

            if array[j] > array[j + 1]:
                (array[j], array[j + 1]) = (array[j + 1], array[j])


data = [-2, 45, 0, 11, -9]
bubbleSort(data)
print('Sorted Array in Ascending Order:')
print(data)
```

Optimized Bubble Sort

In the above code, all possible comparisons are mac Contents array is already sorted. It increases the execution time.

The code can be optimized by introducing an extra variable `swapped`. After each iteration, if there is no swapping taking place then, there is no need for performing further loops.

In such case, variable `swapped` is set false. Thus, we can prevent further iterations.

Algorithm for optimized bubble sort is

```
bubbleSort(array)
  swapped <- false
  for i <- 1 to indexOfLastUnsortedElement-1
    if leftElement > rightElement
      swap leftElement and rightElement
      swapped <- true
end bubbleSort
```

Optimized Bubble Sort Examples

Python Java C C+

```
# Python Program To Sort data in ascending order using bubble sort.

def bubbleSort(array):
    for i in range(len(array)):
        swapped = True
        for j in range(0, len(array) - i - 1):

            # To sort in descending order, change > to < in this line.

            if array[j] > array[j + 1]:
                (array[j], array[j + 1]) = (array[j + 1], array[j])
                swapped = False
                if swapped == True:
                    break
```

Contents

```
data = [-2, 45, 0, 11, -9]
bubbleSort(data)
print('Sorted Array in Ascending Order:')
print(data)
```

Complexity

Bubble Sort is one of the simplest sorting algorithms. Two loops are implemented in the algorithm.

Cycle	Number of Comparisons
1st	(n-1)
2nd	(n-2)
3rd	(n-3)
....	......
last	1

Number of comparisons: `(n-1) + (n-2) + (n-3) +.....+ 1 = n(n-1)/2` nearly equals to n^2

Complexity: $O(n^2)$

Contents

Also, we can analyze the complexity by simply observing the number of loops. There are 2 loops so the complexity is `n*n = n`^{2}

Time Complexities:

- **Worst Case Complexity:** `O(n`^{2}`)`

  If we want to sort in ascending order and the array is in descending order then, the worst case occurs.

- **Best Case Complexity:** `O(n)`

  If the array is already sorted, then there is no need for sorting.

- **Average Case Complexity:** `O(n`^{2}`)`

  It occurs when the elements of the array are in jumbled order (neither ascending nor descending).

Space Complexity:

Space complexity is `O(1)` because an extra variable `temp` is used for swapping.

In the optimized algorithm, the variable `swapped` adds to the space complexity thus, making it `O(2)` .

Bubble Sort Applications

Bubble sort is used in the following cases where

1. the complexity of the code does not matter.
2. a short code is preferred.

Contents

Data Structure & Algorithms

Contents

Contents

- Quicksort Algorithm
- Counting Sort Algorithm
- Radix Sort Algorithm
- Shell Sort Algorithm
- Bucket Sort Algorithm
- Hash Table
- Asymptotic Analysis
- Heap Data Structure
- Priority Queue
- Red-Black Tree
- Insertion in a Red-Black Tree
- Deletion in a Red-Black Tree
- AVL Tree

Contents

Get Latest Updates on Programiz

Enter Your Email

Subscribe

TUTORIALS

Python Tutorials

C Tutorials

Java Tutorials

Kotlin Tutorials

C++ Tutorials

Swift Tutorials

R Tutorials

DSA

EXAMPLES

Python Examples

C Examples

Java Examples

Kotlin Examples

C++ Examples

R Examples

COMPANY

About

Contents

Advertising

Contact

LEGAL

Privacy Policy

Terms And Conditions

App's Privacy Policy

App's Terms And Conditions

Contents