Bubble Sort

Dr. Sumit Srivastava
Dept. of CSE, BIT Mesra Ranchi
Email:- sumit@bitmesra.ac.in

Bubble Sort

- Bubble Sort is the simplest sorting algorithm that works by repeatedly swapping the adjacent elements if they are in the wrong order.
- This algorithm is not suitable for large data sets as its average and worst-case time complexity is quite high.

Bubble Sort Algorithm

- In Bubble Sort algorithm,
  - traverse from left and compare adjacent elements and the higher one is placed at right side.
  - In this way, the largest element is moved to the rightmost end at first.
  - This process is then continued to find the second largest and place it and so on until the data is sorted.

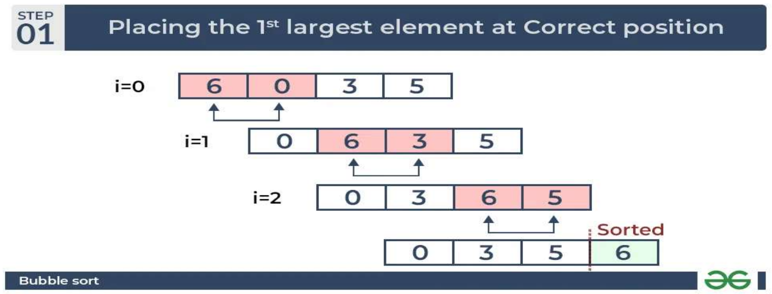
How does Bubble Sort Work?

- Let us understand the working of bubble sort with the help of the following illustration:

Input: arr[] = {6, 3, 0, 5}

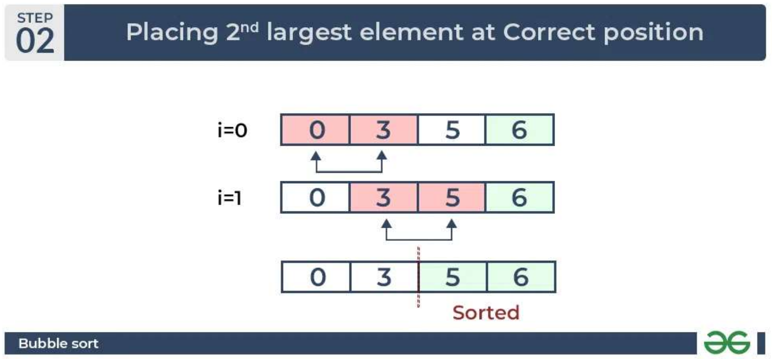
How does Bubble Sort Work?

- First Pass: The largest element is placed in its correct position, i.e., the end of the array.



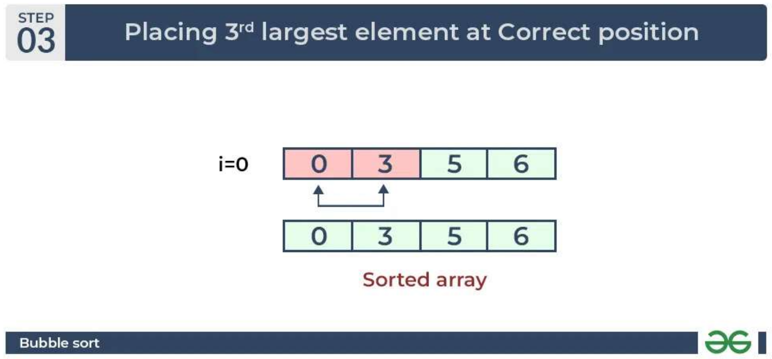
How does Bubble Sort Work?

- Second Pass: Place the second largest element at correct position.



How does Bubble Sort Work?

- Third Pass: Place the remaining two elements at their correct positions.



How does Bubble Sort Work?

- Total no. of passes: n-1
- Total no. of comparisons: n*(n-1)/2

Bubble Sort Function

```
void bubble_sort(int arr[], int n) {
 int i, j;
 for (i = 0; i < n - 1; i++) {
  for (j = 0; j < n - i - 1; j++) {
   if (arr[j] > arr[j + 1]) {
    int temp = arr[j];
    arr[j] = arr[j + 1];
    arr[j + 1] = temp;
   }  }   }   }
```

The function has two loops. The outer loop runs from the first element to the second-last element of the array. The inner loop runs from the first element to the second-last element of the unsorted part of the array. The condition of the inner loop is n - i - 1 because the last i elements of the array are already sorted.

In each iteration of the inner loop, we compare adjacent elements. If the left element is greater than the right element, we swap them. After the inner loop completes, the largest element is guaranteed to be at the end of the unsorted part of the array.

Bubble Sort Program

```
#include <stdio.h>

void bubble_sort(int arr[], int n) {
 int i, j;
 for (i = 0; i < n - 1; i++) {
  for (j = 0; j < n - i - 1; j++) {
   if (arr[j] > arr[j + 1]) {
    int temp = arr[j];
    arr[j] = arr[j + 1];
    arr[j + 1] = temp;
   }
  }
 }
}
```

```
int main() {
 int arr[] = {64, 34, 25, 12, 22, 11, 90};
 int n = sizeof(arr) / sizeof(arr[0]);
 bubble_sort(arr, n);
 printf("Sorted array: ");
 for (int i = 0; i < n; i++) {
  printf("%d ", arr[i]);
 }
 return 0;
}
```

Bubble Sort

Characteristics:

- Bubble sort is a simple sorting algorithm.
- It works by repeatedly swapping adjacent elements if they are in the wrong order.
- The algorithm sorts the array in ascending or descending order.
- It has a time complexity of O(n2) in the worst case, where n is the size of the array.

Bubble Sort

Usage:

- Bubble sort is useful for educational purposes and small data sets.
- It is not suitable for large data sets because of its time complexity.

Advantages:

- Bubble sort is easy to understand and implement.
- It requires minimal additional memory space to perform the sorting.

Disadvantages:

- It is not efficient for large data sets because of its time complexity.
- It has poor performance compared to other sorting algorithms, such as quicksort and mergesort.