



Insertion Sort

Insertion sort is a simple sorting algorithm that works the way we sort playing cards in our hands.

2.1

Algorithm

```
// Sort an arr[] of size n
```

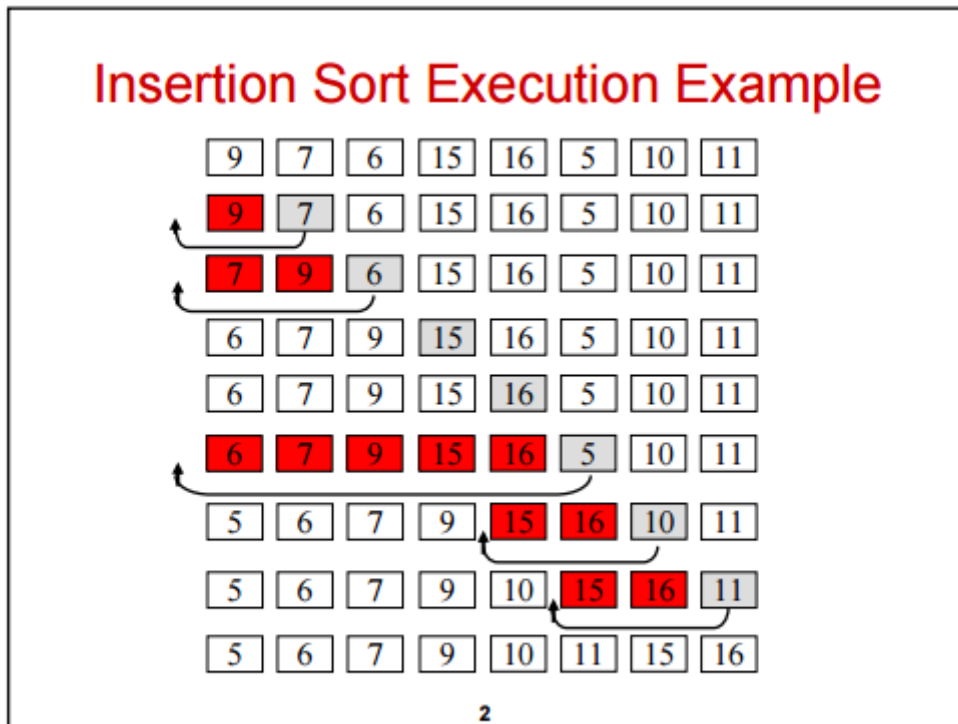
```
insertionSort(arr, n)
```

```
Loop from i = 1 to n-1.
```

```
.....a) Pick element arr[i] and insert it into sorted sequence arr[0...i-1]
```

Example:



**Another Example:**

12, 11, 13, 5, 6

Let us loop for $i = 1$ (second element of the array) to 5 (Size of input array)

$i = 1$. Since 11 is smaller than 12, move 12 and insert 11 before 12

11, 12, 13, 5, 6

$i = 2$. 13 will remain at its position as all elements in $A[0..i-1]$ are smaller than 13

11, 12, 13, 5, 6

$i = 3$. 5 will move to the beginning and all other elements from 11 to 13 will move one position ahead of their current position.

5, 11, 12, 13, 6

$i = 4$. 6 will move to position after 5, and elements from 11 to 13 will move one position ahead of their current position.

5, 6, 11, 12, 13

Recommended: Please solve it on “PRACTICE” first, before moving on to the solution.

C/C++

```
// C program for insertion sort
#include <stdio.h>
#include <math.h>
```



```

/* Function to sort an array using insertion sort*/
void insertionSort(int arr[], int n)
{
    int i, key, j;
    for (i = 1; i < n; i++)
    {
        key = arr[i];
        j = i-1;

        /* Move elements of arr[0..i-1], that are
           greater than key, to one position ahead
           of their current position */
        while (j >= 0 && arr[j] > key)
        {
            arr[j+1] = arr[j];
            j = j-1;
        }
        arr[j+1] = key;
    }
}

```

```

// A utility function to print an array of size n
void printArray(int arr[], int n)
{
    int i;
    for (i=0; i < n; i++)
        printf("%d ", arr[i]);
    printf("\n");
}

```

```

/* Driver program to test insertion sort */
int main()
{
    int arr[] = {12, 11, 13, 5, 6};
    int n = sizeof(arr)/sizeof(arr[0]);

    insertionSort(arr, n);
    printArray(arr, n);

    return 0;
}

```

[Run on IDE](#)

Python

Python program for implementation of Insertion Sort

```

# Function to do insertion sort
def insertionSort(arr):

    # Traverse through 1 to len(arr)
    for i in range(1, len(arr)):

        key = arr[i]

        # Move elements of arr[0..i-1], that are
        # greater than key, to one position ahead
        # of their current position
        j = i-1
        while j >=0 and key < arr[j] :
            arr[j+1] = arr[j]
            j -= 1
        arr[j+1] = key

```

```

# Driver code to test above
arr = [12, 11, 13, 5, 6]
insertionSort(arr)

```



```
print ("Sorted array is:")
for i in range(len(arr)):
    print ("%d" %arr[i])
```

This code is contributed by Mohit Kumra

[Run on IDE](#)

// Java program for implementation of Insertion Sort

```
class InsertionSort
{
    /*Function to sort array using insertion sort*/
    void sort(int arr[])
    {
        int n = arr.length;
        for (int i=1; i<n; ++i)
        {
            int key = arr[i];
            int j = i-1;

            /* Move elements of arr[0..i-1], that are
            greater than key, to one position ahead
            of their current position */
            while (j>=0 && arr[j] > key)
            {
                arr[j+1] = arr[j];
                j = j-1;
            }
            arr[j+1] = key;
        }
    }
}
```

```
/* A utility function to print array of size n*/
static void printArray(int arr[])
{
    int n = arr.length;
    for (int i=0; i<n; ++i)
        System.out.print(arr[i] + " ");

    System.out.println();
}

// Driver method
public static void main(String args[])
{
    int arr[] = {12, 11, 13, 5, 6};

    InsertionSort ob = new InsertionSort();
    ob.sort(arr);

    printArray(arr);
}
/* This code is contributed by Rajat Mishra. */
```

[Run on IDE](#)

Output:

```
5 6 11 12 13
```

Time Complexity: $O(n^2)$

Auxiliary Space: $O(1)$



Boundary Cases: Insertion sort takes maximum time to sort if elements are sorted in reverse order. And it takes minimum time (Order of n) when elements are already sorted.

Algorithmic Paradigm: Incremental Approach

Sorting In Place: Yes

Stable: Yes

Online: Yes

Uses: Insertion sort is used when number of elements is small. It can also be useful when input array is almost sorted, only few elements are misplaced in complete big array.

What is Binary Insertion Sort?

We can use binary search to reduce the number of comparisons in normal insertion sort. Binary Insertion Sort find use binary search to find the proper location to insert the selected item at each iteration. In normal insertion, sort it takes $O(i)$ (at i th iteration) in worst case. we can reduce it to $O(\log i)$ by using binary search. The algorithm as a whole still has a running worst case running time of $O(n^2)$ because of the series of swaps required for each insertion. Refer [this](#) for implementation.

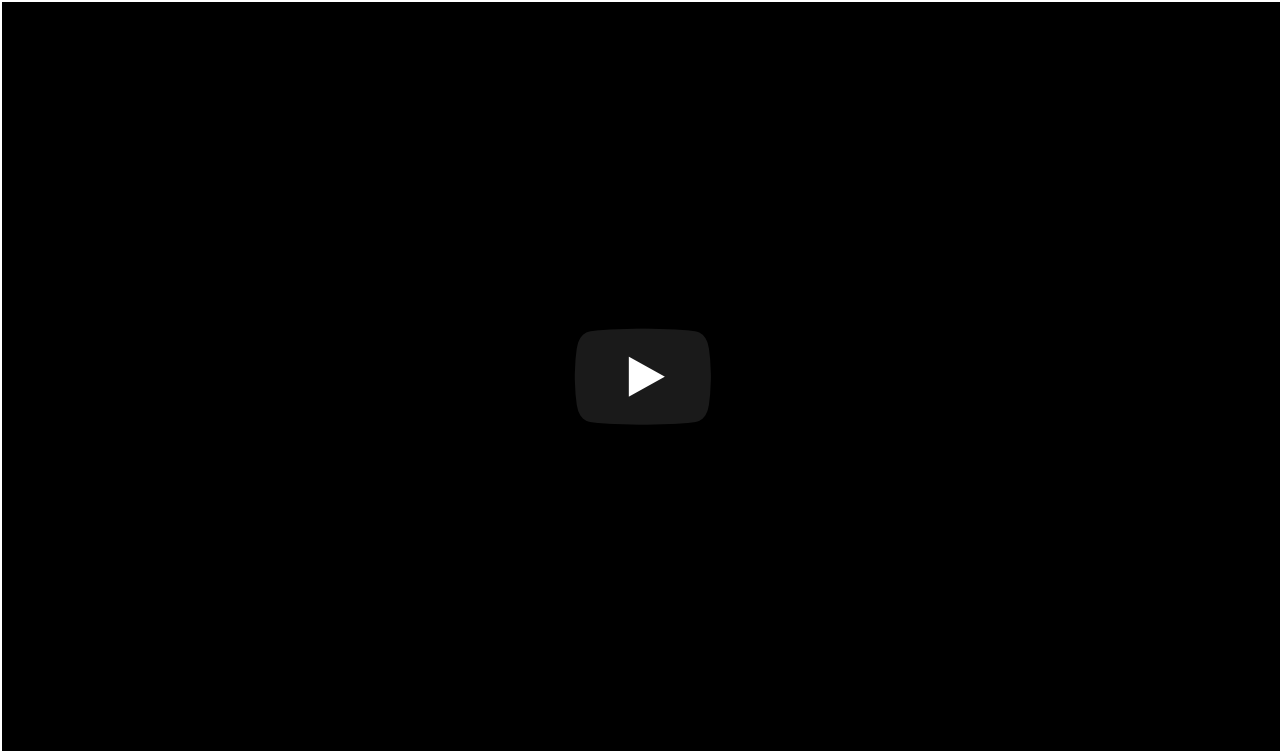
How to implement Insertion Sort for Linked List?

Below is simple insertion sort algorithm for linked list.

- 1) Create an empty sorted (or result) list
- 2) Traverse the given list, do following for every node.
.....a) Insert current node in sorted way in sorted or result list.
- 3) Change head of given linked list to head of sorted (or result) list.

Refer [this](#) for implementation.





Snapshots:

SORTED ARRAY

UNSORTED ARRAY

7

8

5

2

4

6

3

SELECT THE FIRST UNSORTED ELEMENT

SWAP OTHER ELEMENTS TO THE RIGHT TO CREATE THE CORRECT POSITION AND SHIFT THE UNSORTED ELEMENT.

ADVANCE THE MARKER TO THE RIGHT ONE ELEMENT

SORTED ARRAY

UNSORTED ARRAY

5

7

8

2

4

6

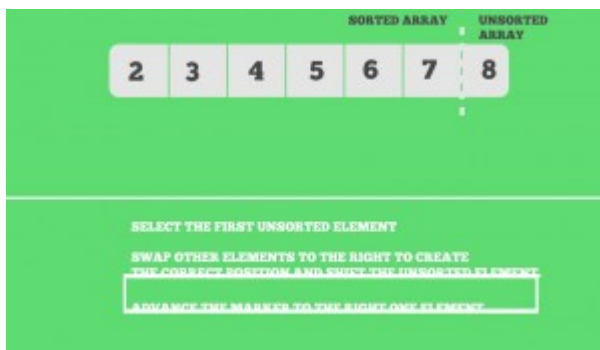
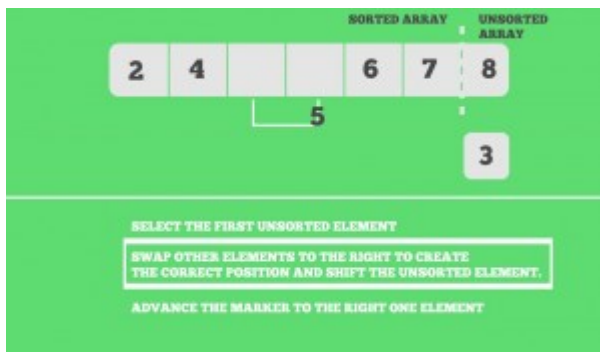
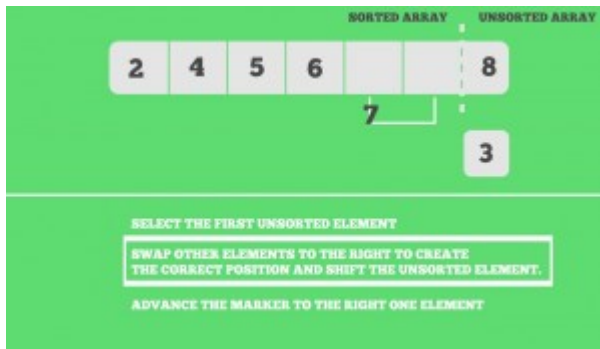
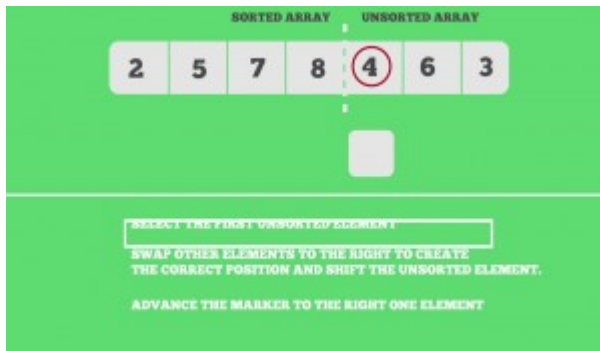
3

SELECT THE FIRST UNSORTED ELEMENT

SWAP OTHER ELEMENTS TO THE RIGHT TO CREATE THE CORRECT POSITION AND SHIFT THE UNSORTED ELEMENT.

ADVANCE THE MARKER TO THE RIGHT ONE ELEMENT





Quiz on Insertion Sort

Other Sorting Algorithms on GeeksforGeeks/GeeksQuiz

Selection Sort, Bubble Sort, Insertion Sort, Merge Sort, Heap Sort, QuickSort, Radix Sort, Counting Sort, Bucket Sort, ShellSort, Comb Sort,

Coding practice for sorting.



Image

Source:

http://www.just.edu.jo/~basel/algorithms/Algo%20Slides/algo_ch2_getting_started.pdf

Please write comments if you find anything incorrect, or you want to share more information about the topic discussed above.

GATE CS Corner Company Wise Coding Practice

Sorting

[Login to Improve this Article](#)

Please write to us at contribute@geeksforgeeks.org to report any issue with the above content.

Recommended Posts:

[Merge Sort](#)[Selection Sort](#)[QuickSort](#)[Bubble Sort](#)[Recursive Insertion Sort](#)[Sum of Manhattan distances between all pairs of points](#)[Maximum triplet sum in array](#)[Rearrange array such that even positioned are greater than odd](#)[Maximum sum of pairwise product in an array with negative allowed](#)[Sort the words in lexicographical order in Python](#)

(Login to Rate)

2.1

Average Difficulty : **2.1/5.0**
Based on **53** vote(s)

Basic

Easy

Medium

Hard

Expert



Add to TODO List



Mark as DONE

Writing code in comment? Please use ide.geeksforgeeks.org, generate link and share the link here.

Load Comments

Share this post!

@geeksforgeeks, Some rights reserved

[Contact Us!](#)[About Us!](#)[Careers!](#)[Privacy](#)[Policy](#)