**Bubble Sort Algorithm**

Bubble Sort is the simplest sorting algorithm that works by repeatedly swapping the adjacent elements if they are in wrong order.



 **// An optimized version of Bubble Sort**

**static void bubbleSort(int arr[], int n)     {**

**int i, j, temp;**

**boolean swapped;**

**for (i = 0; i < n - 1; i++){**

**swapped = false;**

**for (j = 0; j < n - i - 1; j++){**

**if (arr[j] > arr[j + 1]){**

**// swap arr[j] and arr[j+1]**

**temp = arr[j];**

**arr[j] = arr[j + 1];**

**arr[j + 1] = temp;**

**swapped = true;**

**}**

**}**

**// IF no two elements were**

**// swapped by inner loop, then break**

**if (swapped == false)**

**break;**

**}**

**}**

**Selection Sort**

The selection sort algorithm sorts an array by repeatedly finding the minimum element (considering ascending order) from unsorted part and putting it at the beginning.



**void selectionSort(int arr[])     {**

**int n = arr.length;**

**// One by one move boundary of unsorted subarray**

**for (int i = 0; i < n-1; i++) {**

**// Find the minimum element in unsorted array**

**int min\_idx = i;**

**for (int j = i+1; j < n; j++)**

**if (arr[j] < arr[min\_idx])**

**min\_idx = j;**

**// Swap the found minimum element with the first**

**// element**

**int temp = arr[min\_idx];**

**arr[min\_idx] = arr[i];**

**arr[i] = temp;**

**}**

**}**

**Insertion sort**

Insertion sort is based on the idea that one element from the input elements is consumed in each iteration to find its correct position i.e, the position to which it belongs in a sorted array.



**/\*Function to sort array using insertion sort\*/**

**void insertionSort(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;**

**}**

**}**