1.  
class selectionsort{  
 void sort(int arr[])   
 {   
 int n = arr.length;   
 for (int i = 0; i < n-1; i++)   
 {   
 int min\_idx = i;   
 for (int j = i+1; j < n; j++)   
 if (arr[j] < arr[min\_idx])   
 min\_idx = j;   
   
  
 int temp = arr[min\_idx];   
 arr[min\_idx] = arr[i];   
 arr[i] = temp;   
 }   
 }   
}  
  
2)  
void insertionSort(int arr[], int n)   
{   
 int i, key, j;   
 for (i = 1; i < n; i++) {   
 key = arr[i];   
 j = i - 1;   
  
 while (j >= 0 && arr[j] > key) {   
 arr[j + 1] = arr[j];   
 j = j - 1;   
 }   
 arr[j + 1] = key;   
 }   
}   
3.  
class BubbleSort   
{   
 void bubbleSort(int arr[])   
 {   
 int n = arr.length;   
 for (int i = 0; i < n-1; i++)   
 for (int j = 0; j < n-i-1; j++)   
 if (arr[j] > arr[j+1])   
 {   
 // swap arr[j+1] and arr[j]   
 int temp = arr[j];   
 arr[j] = arr[j+1];   
 arr[j+1] = temp;   
 }   
 }   
}  
4.  
class MergeSort {   
 void merge(int arr[], int l, int m, int r)   
 {   
 int n1 = m - l + 1;   
 int n2 = r - m;   
  
 int L[] = new int[n1];   
 int R[] = new int[n2];   
   
 for (int i = 0; i < n1; ++i)   
 L[i] = arr[l + i];   
 for (int j = 0; j < n2; ++j)   
 R[j] = arr[m + 1 + j];   
  
 int i = 0, j = 0;   
  
 int k = l;   
 while (i < n1 && j < n2) {   
 if (L[i] <= R[j]) {   
 arr[k] = L[i];   
 i++;   
 }   
 else {   
 arr[k] = R[j];   
 j++;   
 }   
 k++;   
 }   
  
 while (i < n1) {   
 arr[k] = L[i];   
 i++;   
 k++;   
 }   
  
 while (j < n2) {   
 arr[k] = R[j];   
 j++;   
 k++;   
 }   
 }   
 void sort(int arr[], int l, int r)   
 {   
 if (l < r) {   
 int m = (l + r) / 2;   
  
 sort(arr, l, m);   
 sort(arr, m + 1, r);   
   
 merge(arr, l, m, r);   
 }   
 }   
  
class QuickSort   
{   
 int partition(int arr[], int low, int high)   
 {   
 int pivot = arr[high];   
 int i = (low-1);  
 for (int j=low; j<high; j++)   
 {   
 if (arr[j] < pivot)   
 {   
 i++;   
   
 // swap arr[i] and arr[j]   
 int temp = arr[i];   
 arr[i] = arr[j];   
 arr[j] = temp;   
 }   
 }   
  
 int temp = arr[i+1];   
 arr[i+1] = arr[high];   
 arr[high] = temp;   
   
 return i+1;   
 }   
  
 void sort(int arr[], int low, int high)   
 {   
 if (low < high)   
 {   
 int pi = partition(arr, low, high);   
   
 sort(arr, low, pi-1);   
 sort(arr, pi+1, high);   
 }   
 }