```java
/*
 * Written by JJ Shepherd
 */
public class SortingAlgorithms {

    public static void main(String[] args)
    {
        int[] array = {10,8,7,6,12,5,11,9};
        quickSort(array);
        for(int i : array)
            System.out.println(i);

    }
    public static void mergeSort(int[] a)
    {
        int size = a.length;
        if(size < 2)
            return;
        int mid = size / 2;
        int leftSize = mid;
        int rightSize = size - mid;
        int[] left = new int[leftSize];
        int[] right = new int[rightSize];
        for(int i=0;i<mid;i++)
            left[i] = a[i];
        for(int i=mid;i<size;i++)
            right[i-mid] = a[i];
        mergeSort(left);
        mergeSort(right);
        merge(left,right,a);
    }
    public static void merge(int[] left, int[] right, int[] a)
    {
        int leftSize = left.length;
        int rightSize = right.length;
        int i = 0;//Left array's index
        int j = 0;//Right array's index
        int k = 0;//Merged Array's (a) index
        while(i<leftSize && j<rightSize)
        {
            if(left[i] <= right[j])
            {
                a[k] = left[i];
                i++;
                k++;
            }
            else
            {
                a[k] = right[j];
                j++;
                k++;
            }
        }
        while(i<leftSize)
        {
            a[k] = left[i];
            i++;
```

```
58              k++;
59          }
60          while(j<rightSize)
61          {
62              a[k] = right[j];
63              j++;
64              k++;
65          }
66      }
67      public static void quickSort(int[] a)
68      {
69          quickSort(a,0,a.length-1);
70      }
71      public static void quickSort(int[] a, int start, int end)
72      {
73          if(start >= end)
74              return;
75          int pivot = partition(a,start,end);
76          quickSort(a,start,pivot-1);
77          quickSort(a,pivot+1,end);
78      }
79      public static int partition(int[] a, int start, int end)
80      {
81          int pivot = a[end];
82          int i = start;
83          for(int j=start;j<=end;j++)
84          {
85              if(a[j] < pivot)
86              {
87                  int temp = a[i];
88                  a[i] = a[j];
89                  a[j] = temp;
90                  i++;
91              }
92          }
93          int temp = a[i];
94          a[i] = a[end];
95          a[end] = temp;
96          return i;
97      }
98  }
99
100
```