```java
/*
 * 归并排序可视化
 */

import java.util.ArrayList;

public class MergeSortAnimation {
    public static void main(String[] args) {
        Canvas c = new Canvas();
        Formats f = new Formats();
        HistogramData d = new HistogramData();

        ArrayList<double[]> state = new ArrayList<>();
        double nums[] = generateRandomNums(100);
        state.add(nums.clone());
        MergeSort(nums, state);
        showNums(nums);
        d.state = state;
        Histogram h = new Histogram(c, f, d);
        h.animate();
    }

    public static double[] generateRandomNums(int n) {
        double[] nums = new double[n];
        for (int i = 0; i < n; i++) {
            nums[i] = Math.random()*100;
        }
        return nums;
    }

    public static void showNums(double nums[]) {
        for (double item:nums) {
            System.out.println(item);
        }
    }

    private static void MergeSort(double[] a, ArrayList<double[]>
state) {
        Sort(a, 0, a.length - 1, state);
    }
```

```
40
41    private static void Sort(double[] a, int left, int right,
   ArrayList<double[]> state) {
42        // 归并排序的关键代码
43        if(left>=right)
44            return;
45
46        int mid = (left + right) / 2;
47        Sort(a, left, mid, state);
48        Sort(a, mid + 1, right, state);
49        merge(a, left, mid, right, state);
50    }
51
52
53    private static void merge(double[] a, int left, int mid, int
   right, ArrayList<double[]> state) {
54
55        double[] tmp = new double[a.length];
56        int r1 = mid + 1;
57        int tIndex = left;
58        int cIndex=left;
59
60        while(left <=mid && r1 <= right) {
61            if (a[left] <= a[r1])
62                tmp[tIndex++] = a[left++];
63            else
64                tmp[tIndex++] = a[r1++];
65        }
66
67        while (left <=mid) {
68            tmp[tIndex++] = a[left++];
69        }
70        state.add(a.clone()); // 记录左半部分归并状态
71
72        while ( r1 <= right ) {
73            tmp[tIndex++] = a[r1++];
74        }
75
76        while(cIndex<=right){
77            a[cIndex]=tmp[cIndex];
```

```
78              cIndex++;
79          }
80          state.add(a.clone()); // 记录右半部分归并状态
81      }
82
83  }
84
```