```
1    // Java                                                Name -
2    // Sorting Algorithms
3
4    public class sorting_algorithms_handout
5    {
6        public static void main(String[] args)
7        {
8            int[] nums1 = {3, 6, 5, 1, 9, 2, 4, 0, 8, 7};
9            display(nums1);
10           selectionSort(nums1);
11           display(nums1);
12           System.out.println();
13
14           int[] nums2 = {3, 6, 5, 1, 9, 2, 4, 0, 8, 7};
15           display(nums2);
16           insertionSort(nums2);
17           display(nums2);
18           System.out.println();
19
20           int[] nums3 = {3, 6, 5, 1, 9, 2, 4, 0, 8, 7};
21           display(nums3);
22           selectionSort(nums3);
23           display(nums3);
24       }
25
26       public static void display(int[] a)
27       {
28           for (int x : a)
29           {
30               System.out.print(" " + x);
31           }
32
33           System.out.println();
34       }
35
36   // ***************** SELECTION ***********************************
37
38       public static void selectionSort(int[] arr)
39       {
40           for (int i = 0; i < arr.length - 1; i++)
41           {
42               int minIndex = i;
43               int min = arr[minIndex];
44
45
46               for ( int j = i + 1; j < arr.length; j++)
47               {
48
49                   if ( arr[j] < min)
50                   {
51                       minIndex = j;
52                       min = arr[minIndex];
53                   }
54
55               }
56
57               int temp = arr[minIndex];       // swap
58               arr[minIndex] = arr[i];
59               arr[i] = temp;
60           }
61
62       }
63
64   // ******************* INSERTION ***********************************
65
66       public static void insertionSort(int[] arr)
67       {
```

```
68            int j = 0;
69
70            for (int i = 1; i < arr.length; i++)
71            {
72                int value = arr[i];
73                j = i;
74
75                while (j > 0 && arr[j - 1] >= value)
76                {
77                    arr[j] = arr[j - 1];    // shift
78                    j--;
79                }
80
81                arr[j] = value;     // insert
82            }
83
84        }
85
86
87
88      // ******************* MERGE ****************************************
89
90       public static void sort(int arr[])
91       {
92            if (arr.length > 1)
93                mergeSort(arr, 0, arr.length);
94       }
95
96       public static void mergeSort(int arr[], int start, int segLength)
97       {
98            // Divide the array in half and sort and merge the halves
99            int half = segLength / 2;
100
101           // Sort each of the two parts of the arrays
102           if (half > 1)
103           {
104               mergeSort (arr, start, half);
105           }
106
107           if (segLength - half > 1)
108           {
109               mergeSort (arr, start + half, segLength - half);
110           }
111
112           // Copy the two parts of the arrays & merge
113           int t1[] = new int[half];
114           int t2[] = new int[segLength - half];
115           int i = start;
116
117           for (int j = 0; j < half;)
118           {
119               t1[j++] = arr[i++];
120           }
121
122           for ( int j = 0; j < segLength - half;)
123           {
124               t2[j++] = arr[i++];
125           }
126
127           merge (arr, start, t1, t2);
128       }
129
130       public static void merge (int arr[], int start, int t1[], int t2[])
131       {
132           int index = start;
133           int end = start + t1.length + t2.length;
134           int t1index = 0;
```

```
135             int t2index = 0;
136 
137             while (t1index < t1.length && t2index < t2.length)
138             {
139                 if (t1[t1index] <= t2[t2index])
140                     arr[index++] = t1[t1index++];
141                 else
142                     arr[index++] = t2[t2index++];
143             }
144 
145             while (t1index < t1.length)
146             {
147                 arr[index++] = t1[t1index++];
148             }
149 
150             while (t2index < t2.length)
151             {
152                 arr[index++] = t2[t2index++];
153             }
154         }
155     }
```