view source

print?

```
01 void selectionSortArray(int Array[], int SIZE, long PerformanceDataArray[])
02 {
03     //Variable declarations for performance testing
04     long swapCounter = 0;//Increment this counter whenever a swap takes place
05     long comparisonCounter=0;//Increment this counter whenever a comparison takes place
06 
07     int TempVal = 0;
08     int IndexMinVal = 0;
09     int CompResult;
10 
11     // Now the Selection Sort begins //
12     for(int i = 0; i < SIZE -1; i++)
13     {
14         IndexMinVal = i;
15 
16         for(int j = i + 1; j < SIZE; j++)
17         {
18             CompResult = compare(Array[j], Array[IndexMinVal]);
19             comparisonCounter++;
20 
21             if(CompResult==-1)
22             {
23                 IndexMinVal = j;
24             }
25         }
26 
27         swap(Array[i], Array[IndexMinVal]);
28         swapCounter++;
29     }
30 
31     PerformanceDataArray[0] = comparisonCounter;
32     PerformanceDataArray[1] = swapCounter;
33 }//End of selectionSortArray()
34 
35 
36 void insertionSortArray(int Array[], int SIZE,  long PerformanceDataArray[])
37 {
38     //Variable declarations for performance testing
39     long swapCounter = 0;//Increment this counter whenever a swap takes place
40     long comparisonCounter =0;//Increment this counter whenever a comparison takes place
41 
42     int comparisonResult;
43     for(int i = 1; i<SIZE; i++)
```

```
44     {
45         for(int j = i; j>0; j--)
46         {
47             comparisonResult = compare(Array[j], Array[j-1]);
48             comparisonCounter++;
49 
50             if(comparisonResult == -1)
51             {
52                 swap(Array[j], Array[j-1]);
53             }
54                         swapCounter++;
55         }
56     }
57 
58     PerformanceDataArray[0] = comparisonCounter;
59     PerformanceDataArray[1] = swapCounter;
60 }//End of insertionSortArray()
61 
62 
63 void bubbleSort(int Array[], int SIZE,  long PerformanceDataArray[])
64 {
65     //Variable declarations for performance testing
66     long swapCounter = 0;//Increment this counter whenever a swap takes place
67     long comparisonCounter=0;//Increment this counter whenever a comparison takes place
68 
69     for(int x = 1; x<SIZE; x++)
70     {
71         for(int y = SIZE-1; y>=x; y--)
72         {
73             if(Array[y-1] > Array[y])
74             {
75                                 comparisonCounter++;
76                 swap(Array[y], Array[y-1]);
77                             swapCounter++;
78 
79             }
80         }
81     }
82 
83     PerformanceDataArray[0] = comparisonCounter;
84     PerformanceDataArray[1] = swapCounter;
85 }//End of bubbleSort()
```