**BubbleSort**

**1)** Looking at the diagrams below, we can see how the array is sorted. The algorithm looks at the first 2 elements of the array. It compares them and if the second element is smaller than the first, then they switch. From there, the algorithm compares the bigger element to the next element in the array. The process of examining two elements, switching, and comparing the bigger to the next continues until the last element of the array is reached ( the bigger of the 7th and 8th element). Because there are 8 elements in the array (size), these processes will repeat for the second run-through, third, fourth and so on until it finishes the 8th run. The same processes are repeated in each run through to eventually arrive at a properly sorted array of least to greatest.

In the particular diagrams, the numbers compared and switched/remain are highlighted in blocks. The colors don't have any greater significance other than showing the association between being compared and then the step that follows (to switch or not). The values (6.1, 4.2, 3.3, 7.4, 5.5, 2.6, 8.7, and 1.8) were used in the diagram.

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 1st Run | 6.1 | 4.2 | 3.3 | 7.4 | 5.5 | 2.6 | 8.7 | 1.8 |
|  | 4.2 | 6.1 | 3.3 | 7.4 | 5.5 | 2.6 | 8.7 | 1.8 |
|  | 4.2 | 6.1 | 3.3 | 7.4 | 5.5 | 2.6 | 8.7 | 1.8 |
|  | 4.2 | 3.3 | 6.1 | 7.4 | 5.5 | 2.6 | 8.7 | 1.8 |
|  | 4.2 | 3.3 | 6.1 | 7.4 | 5.5 | 2.6 | 8.7 | 1.8 |
|  | 4.2 | 3.3 | 6.1 | 7.4 | 5.5 | 2.6 | 8.7 | 1.8 |
|  | 4.2 | 3.3 | 6.1 | 7.4 | 5.5 | 2.6 | 8.7 | 1.8 |
|  | 4.2 | 3.3 | 6.1 | 5.5 | 7.4 | 2.6 | 8.7 | 1.8 |
|  | 4.2 | 3.3 | 6.1 | 5.5 | 7.4 | 2.6 | 8.7 | 1.8 |
|  | 4.2 | 3.3 | 6.1 | 5.5 | 2.6 | 7.4 | 8.7 | 1.8 |
|  | 4.2 | 3.3 | 6.1 | 5.5 | 2.6 | 7.4 | 8.7 | 1.8 |
|  | 4.2 | 3.3 | 6.1 | 5.5 | 2.6 | 7.4 | 8.7 | 1.8 |
|  | 4.2 | 3.3 | 6.1 | 5.5 | 2.6 | 7.4 | 8.7 | 1.8 |
|  | 4.2 | 3.3 | 6.1 | 5.5 | 2.6 | 7.4 | 1.8 | 8.7 |
|  |  |  |  |  |  |  |  |  |
| 2nd Run | 4.2 | 3.3 | 6.1 | 5.5 | 2.6 | 7.4 | 1.8 | 8.7 |
|  | 3.3 | 4.2 | 6.1 | 5.5 | 2.6 | 7.4 | 1.8 | 8.7 |
|  | 3.3 | 4.2 | 6.1 | 5.5 | 2.6 | 7.4 | 1.8 | 8.7 |
|  | 3.3 | 4.2 | 6.1 | 5.5 | 2.6 | 7.4 | 1.8 | 8.7 |
|  | 3.3 | 4.2 | 6.1 | 5.5 | 2.6 | 7.4 | 1.8 | 8.7 |
|  | 3.3 | 4.2 | 5.5 | 6.1 | 2.6 | 7.4 | 1.8 | 8.7 |
|  | 3.3 | 4.2 | 5.5 | 6.1 | 2.6 | 7.4 | 1.8 | 8.7 |
|  | 3.3 | 4.2 | 5.5 | 2.6 | 6.1 | 7.4 | 1.8 | 8.7 |
|  | 3.3 | 4.2 | 5.5 | 2.6 | 6.1 | 7.4 | 1.8 | 8.7 |
|  | 3.3 | 4.2 | 5.5 | 2.6 | 6.1 | 7.4 | 1.8 | 8.7 |
|  | 3.3 | 4.2 | 5.5 | 2.6 | 6.1 | 7.4 | 1.8 | 8.7 |
|  | 3.3 | 4.2 | 5.5 | 2.6 | 6.1 | 1.8 | 7.4 | 8.7 |
|  | 3.3 | 4.2 | 5.5 | 2.6 | 6.1 | 1.8 | 7.4 | 8.7 |
|  | 3.3 | 4.2 | 5.5 | 2.6 | 6.1 | 1.8 | 7.4 | 8.7 |
|  |  |  |  |  |  |  |  |  |
| 3rd Run | 3.3 | 4.2 | 5.5 | 2.6 | 6.1 | 1.8 | 7.4 | 8.7 |
|  | 3.3 | 4.2 | 5.5 | 2.6 | 6.1 | 1.8 | 7.4 | 8.7 |
|  | 3.3 | 4.2 | 5.5 | 2.6 | 6.1 | 1.8 | 7.4 | 8.7 |
|  | 3.3 | 4.2 | 5.5 | 2.6 | 6.1 | 1.8 | 7.4 | 8.7 |
|  | 3.3 | 4.2 | 5.5 | 2.6 | 6.1 | 1.8 | 7.4 | 8.7 |
|  | 3.3 | 4.2 | 2.6 | 5.5 | 6.1 | 1.8 | 7.4 | 8.7 |
|  | 3.3 | 4.2 | 2.6 | 5.5 | 6.1 | 1.8 | 7.4 | 8.7 |
|  | 3.3 | 4.2 | 2.6 | 5.5 | 6.1 | 1.8 | 7.4 | 8.7 |
|  | 3.3 | 4.2 | 2.6 | 5.5 | 6.1 | 1.8 | 7.4 | 8.7 |
|  | 3.3 | 4.2 | 2.6 | 5.5 | 1.8 | 6.1 | 7.4 | 8.7 |
|  | 3.3 | 4.2 | 2.6 | 5.5 | 1.8 | 6.1 | 7.4 | 8.7 |
|  | 3.3 | 4.2 | 2.6 | 5.5 | 1.8 | 6.1 | 7.4 | 8.7 |
|  | 3.3 | 4.2 | 2.6 | 5.5 | 1.8 | 6.1 | 7.4 | 8.7 |
|  | 3.3 | 4.2 | 2.6 | 5.5 | 1.8 | 6.1 | 7.4 | 8.7 |
|  |  |  |  |  |  |  |  |  |
| 4th Run | 3.3 | 4.2 | 2.6 | 5.5 | 1.8 | 6.1 | 7.4 | 8.7 |
|  | 3.3 | 4.2 | 2.6 | 5.5 | 1.8 | 6.1 | 7.4 | 8.7 |
|  | 3.3 | 4.2 | 2.6 | 5.5 | 1.8 | 6.1 | 7.4 | 8.7 |
|  | 3.3 | 2.6 | 4.2 | 5.5 | 1.8 | 6.1 | 7.4 | 8.7 |
|  | 3.3 | 2.6 | 4.2 | 5.5 | 1.8 | 6.1 | 7.4 | 8.7 |
|  | 3.3 | 2.6 | 4.2 | 5.5 | 1.8 | 6.1 | 7.4 | 8.7 |
|  | 3.3 | 2.6 | 4.2 | 5.5 | 1.8 | 6.1 | 7.4 | 8.7 |
|  | 3.3 | 2.6 | 4.2 | 1.8 | 5.5 | 6.1 | 7.4 | 8.7 |
|  | 3.3 | 2.6 | 4.2 | 1.8 | 5.5 | 6.1 | 7.4 | 8.7 |
|  | 3.3 | 2.6 | 4.2 | 1.8 | 5.5 | 6.1 | 7.4 | 8.7 |
|  | 3.3 | 2.6 | 4.2 | 1.8 | 5.5 | 6.1 | 7.4 | 8.7 |
|  | 3.3 | 2.6 | 4.2 | 1.8 | 5.5 | 6.1 | 7.4 | 8.7 |
|  | 3.3 | 2.6 | 4.2 | 1.8 | 5.5 | 6.1 | 7.4 | 8.7 |
|  | 3.3 | 2.6 | 4.2 | 1.8 | 5.5 | 6.1 | 7.4 | 8.7 |
|  |  |  |  |  |  |  |  |  |
| 5th Run | 3.3 | 2.6 | 4.2 | 1.8 | 5.5 | 6.1 | 7.4 | 8.7 |
|  | 2.6 | 3.3 | 4.2 | 1.8 | 5.5 | 6.1 | 7.4 | 8.7 |
|  | 2.6 | 3.3 | 4.2 | 1.8 | 5.5 | 6.1 | 7.4 | 8.7 |
|  | 2.6 | 3.3 | 4.2 | 1.8 | 5.5 | 6.1 | 7.4 | 8.7 |
|  | 2.6 | 3.3 | 4.2 | 1.8 | 5.5 | 6.1 | 7.4 | 8.7 |
|  | 2.6 | 3.3 | 1.8 | 4.2 | 5.5 | 6.1 | 7.4 | 8.7 |
|  | 2.6 | 3.3 | 1.8 | 4.2 | 5.5 | 6.1 | 7.4 | 8.7 |
|  | 2.6 | 3.3 | 1.8 | 4.2 | 5.5 | 6.1 | 7.4 | 8.7 |
|  | 2.6 | 3.3 | 1.8 | 4.2 | 5.5 | 6.1 | 7.4 | 8.7 |
|  | 2.6 | 3.3 | 1.8 | 4.2 | 5.5 | 6.1 | 7.4 | 8.7 |
|  | 2.6 | 3.3 | 1.8 | 4.2 | 5.5 | 6.1 | 7.4 | 8.7 |
|  | 2.6 | 3.3 | 1.8 | 4.2 | 5.5 | 6.1 | 7.4 | 8.7 |
|  | 2.6 | 3.3 | 1.8 | 4.2 | 5.5 | 6.1 | 7.4 | 8.7 |
|  | 2.6 | 3.3 | 1.8 | 4.2 | 5.5 | 6.1 | 7.4 | 8.7 |
|  |  |  |  |  |  |  |  |  |
| 6th Run | 2.6 | 3.3 | 1.8 | 4.2 | 5.5 | 6.1 | 7.4 | 8.7 |
|  | 2.6 | 3.3 | 1.8 | 4.2 | 5.5 | 6.1 | 7.4 | 8.7 |
|  | 2.6 | 3.3 | 1.8 | 4.2 | 5.5 | 6.1 | 7.4 | 8.7 |
|  | 2.6 | 1.8 | 3.3 | 4.2 | 5.5 | 6.1 | 7.4 | 8.7 |
|  | 2.6 | 1.8 | 3.3 | 4.2 | 5.5 | 6.1 | 7.4 | 8.7 |
|  | 2.6 | 1.8 | 3.3 | 4.2 | 5.5 | 6.1 | 7.4 | 8.7 |
|  | 2.6 | 1.8 | 3.3 | 4.2 | 5.5 | 6.1 | 7.4 | 8.7 |
|  | 2.6 | 1.8 | 3.3 | 4.2 | 5.5 | 6.1 | 7.4 | 8.7 |
|  | 2.6 | 1.8 | 3.3 | 4.2 | 5.5 | 6.1 | 7.4 | 8.7 |
|  | 2.6 | 1.8 | 3.3 | 4.2 | 5.5 | 6.1 | 7.4 | 8.7 |
|  | 2.6 | 1.8 | 3.3 | 4.2 | 5.5 | 6.1 | 7.4 | 8.7 |
|  | 2.6 | 1.8 | 3.3 | 4.2 | 5.5 | 6.1 | 7.4 | 8.7 |
|  | 2.6 | 1.8 | 3.3 | 4.2 | 5.5 | 6.1 | 7.4 | 8.7 |
|  | 2.6 | 1.8 | 3.3 | 4.2 | 5.5 | 6.1 | 7.4 | 8.7 |
|  |  |  |  |  |  |  |  |  |
| 7th Run | 2.6 | 1.8 | 3.3 | 4.2 | 5.5 | 6.1 | 7.4 | 8.7 |
|  | 1.8 | 2.6 | 3.3 | 4.2 | 5.5 | 6.1 | 7.4 | 8.7 |
|  | 1.8 | 2.6 | 3.3 | 4.2 | 5.5 | 6.1 | 7.4 | 8.7 |
|  | 1.8 | 2.6 | 3.3 | 4.2 | 5.5 | 6.1 | 7.4 | 8.7 |
|  | 1.8 | 2.6 | 3.3 | 4.2 | 5.5 | 6.1 | 7.4 | 8.7 |
|  | 1.8 | 2.6 | 3.3 | 4.2 | 5.5 | 6.1 | 7.4 | 8.7 |
|  | 1.8 | 2.6 | 3.3 | 4.2 | 5.5 | 6.1 | 7.4 | 8.7 |
|  | 1.8 | 2.6 | 3.3 | 4.2 | 5.5 | 6.1 | 7.4 | 8.7 |
|  | 1.8 | 2.6 | 3.3 | 4.2 | 5.5 | 6.1 | 7.4 | 8.7 |
|  | 1.8 | 2.6 | 3.3 | 4.2 | 5.5 | 6.1 | 7.4 | 8.7 |
|  | 1.8 | 2.6 | 3.3 | 4.2 | 5.5 | 6.1 | 7.4 | 8.7 |
|  | 1.8 | 2.6 | 3.3 | 4.2 | 5.5 | 6.1 | 7.4 | 8.7 |
|  | 1.8 | 2.6 | 3.3 | 4.2 | 5.5 | 6.1 | 7.4 | 8.7 |
|  | 1.8 | 2.6 | 3.3 | 4.2 | 5.5 | 6.1 | 7.4 | 8.7 |
|  |  |  |  |  |  |  |  |  |
| 8th Run | 1.8 | 2.6 | 3.3 | 4.2 | 5.5 | 6.1 | 7.4 | 8.7 |
|  | 1.8 | 2.6 | 3.3 | 4.2 | 5.5 | 6.1 | 7.4 | 8.7 |
|  | 1.8 | 2.6 | 3.3 | 4.2 | 5.5 | 6.1 | 7.4 | 8.7 |
|  | 1.8 | 2.6 | 3.3 | 4.2 | 5.5 | 6.1 | 7.4 | 8.7 |
|  | 1.8 | 2.6 | 3.3 | 4.2 | 5.5 | 6.1 | 7.4 | 8.7 |
|  | 1.8 | 2.6 | 3.3 | 4.2 | 5.5 | 6.1 | 7.4 | 8.7 |
|  | 1.8 | 2.6 | 3.3 | 4.2 | 5.5 | 6.1 | 7.4 | 8.7 |
|  | 1.8 | 2.6 | 3.3 | 4.2 | 5.5 | 6.1 | 7.4 | 8.7 |
|  | 1.8 | 2.6 | 3.3 | 4.2 | 5.5 | 6.1 | 7.4 | 8.7 |
|  | 1.8 | 2.6 | 3.3 | 4.2 | 5.5 | 6.1 | 7.4 | 8.7 |
|  | 1.8 | 2.6 | 3.3 | 4.2 | 5.5 | 6.1 | 7.4 | 8.7 |
|  | 1.8 | 2.6 | 3.3 | 4.2 | 5.5 | 6.1 | 7.4 | 8.7 |
|  | 1.8 | 2.6 | 3.3 | 4.2 | 5.5 | 6.1 | 7.4 | 8.7 |
|  | 1.8 | 2.6 | 3.3 | 4.2 | 5.5 | 6.1 | 7.4 | 8.7 |

2

B) Java code implement

|  |  |
| --- | --- |
| **SIZE** | **RUNTIME**  (millisecs) |
| 10,000 | 167 |
| 20,000 | 708 |
| 30,000 | 1691 |
| 40,000 | 3041 |
| 50,000 | 4738 |
| 60,000 | 6779 |

C)

**3) Iterations:** 2(n-1)

2(n-2)

2(n-3)

Total

2(n-1) + 2(n-2) +.... + 2(1)

= 2

= 2[ 1+2+...+ (n-1)]

= n(n-1)

=n2 - n

= O(n2)

The result of the algorithm agrees to the efficiency to the graph. From this we can say the graph uses the algorithm O(n2).

**4)** Of the three sorting algorithms, Merge Sort is the fastest and Bubble Sort is the slowest. Merge Sort is based on an algorithm of an efficiency O(nlogn), which is significantly smaller than O(n2) which Bubble and Selection are based on. Because of the difference in algorithm efficiency, Merge Sort will always run faster in comparison to Bubble and Selection when the array is very large. Comparing only selection and bubble, selection sort is faster when used on arrays that are typically larger while Bubble Sort is better suited for smaller scale arrays . Selection sort works to select individual elements and place them accordingly whereas Bubble Sort will compare 2 elements at a time. Because of these differences, it is better to use selection sort unless the array is significantly small; in which case Bubble Sort will be more efficient.