**Practical 01**

***Sorting Algorithms - Bubble Sort***

Bubble Sort is a notoriously inefficient sorting algorithm, there are a couple of ‘tweaks’ that can be applied to the algorithm to help make it more efficient, creating a type of Enhanced Bubble Sort. This practical will look at the Bubble Sort sorting algorithm and illustrate how different data structures impact its efficiency. Have a look at the videos in the Week 1 – Bubble Sort folder on the VLE, to get a better understanding of what is required to enhance the efficiency of the Bubble Sort algorithm.

**Consider the following array of Integer values.**

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **22** | **11** | **34** | **-5** | **3** | **40** | **9** | **16** | **6** |

1. Write the contents of the array for 3 passes of the outermost loop of bubble sort.

**Pass 1**

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **11** | **22** | **-5** | **3** | **34** | **9** | **16** | **6** | **40** |

**Pass 2**

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **11** | **-5** | **3** | **22** | **9** | **16** | **6** | **34** | **40** |

**Pass 3**

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **-5** | **3** | **11** | **9** | **16** | **6** | **22** | **34** | **40** |

1. Write code to implement the Bubble Sort sorting algorithm. **Complete in BubbleSort.java, Q2.java**
2. Implement 2 tweaks to the code that can make this algorithm more efficient. **Complete in BubbleSort.java, Q3.java**
3. How do these ‘tweaks’ make the algorithm more efficient?

**-They do the same number of swaps and do not check for the last rightmost integer in the array because that number will always be the highest one possible. They also check if the array has already been sorted.**

1. Write a piece of code that will propagate a static array of 10000 random elements. **Complete in Q5.java**
2. Write a piece of code that will propagate a static array of 10000 sorted elements. **Complete in Q6.java**
3. Write a piece of code that will propagate a static array of 10000 inversely sorted (sorted z – a rather than a - z) elements. **Complete in Q7.java**
4. Pass each of the arrays into the standard Bubble Sort algorithm and also into the Enhanced Bubble Sort, briefly explain the impact of the different data structures on the efficiency of each of the sorting algorithms.

**-Q2 Bubble Sorting Algorithm and Q3 Enhanced Bubble Sorting Algorithm both seem to do it in the same amount of swaps for array numList {22, 11, 34, -5, 3, 40, 9, 16, 6}, however the Enhanced BubbleSort Algorithm does it in way less comparisons.**

1. Implement an array list, populating it similar to the arrays in 5, 6 and 7 above. **Complete in Q9.java**
2. Rewrite the Bubble Sort algorithms to sort the array lists. **Complete in BubbleSort.java**
3. Does changing the data structure from a static array to an array list have an impact on the overall efficiency of the algorithm?

**-It does affect the efficiency of the algorithm by making it a little bit faster, however you need to change your code to suit an ArrayList.**

1. Write a print method that will validate your results in steps 1, 2 and 3 above. **Complete in Q2.java, Q3.java**
2. Write code that will record the number of comparisons both the Standard and Enhanced Bubble Sort algorithms will carry out on the datasets implemented in steps 5, 6, and 7 above. **Complete in BubbleSort.java, Q9.java**
3. Write code that will record the number of swap operations both the Standard and Enhanced Bubble Sort algorithms will carry out on the datasets implemented in steps 5, 6, and 7 above. **Complete in BubbleSort.java, Q9.java**