**ARRAY SORTING ALGORITHM PLAN**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **STUDENT NAMES: Daniel, Enfei and Ivan SORT NAME: Comb Sort** | | | | | **TIME**: |
| **RESOURCES:** *List ALL required resources*  Measuring tape or piece of string, items to sort (with numbers on them, or of different sizes) | | | | *Demo files*  CombSort.java |
| **Description of how the sort works:**  Comb sort improves on bubble sort. It was originally designed by Włodzimierz Dobosiewicz in 1980. Bubble sort compares two items with a gap of one, comb sort can have a gap of any size. This allows it eliminate *turtles*, small values near the end of the list. These slow the sorting down tremendously in bubble sort, since it must repeatedly go through the list many times to find their spot. Comb and Bubble sort do not have a problem with *Rabbits,* large values in the beginning  The gap shrinks in size with each iteration of the loop. The amount by which it shrinks is called the shrink factor. The gap starts out as the length of the list, and gets divided by the shrink factor with every iteration, typically 1.3, then modified bubble sort gets applied with that gap.  The modification: Two items are compared, at iteration, i and i + gap. They are then swapped to have the smaller number first (or vice versa for sorting largest smallest) until i + gap reaches the end of the list.  Once the gap is 1, the algorithm keeps sorting it with the same gap, effectively making it bubble sort. The difference is, at this point most turtles have been dealt with, which significantly speeds up the average time compared to Bubble sort.  1.3 was suggested for the shrink factor by the original authors after testing 200,000 random lists. Too small of a small value for shrink factor causes many unnecessary comparisons, and too large of a value does not deal with turtles effectively, since it will reach a value of 1 faster and face the same problem as Bubble sort. | | | | | |
| **WORKSHOP TIMELINE:** *Outline of all the steps included in your presentation including introduction, finished examples, and demonstration.* | | | | | |
| **#** | **Item** | **Student** | **Details** | | **Time** |
| **1** | Introduction | Enfei | * History * The Goal by using a gap (turtles and Rabbits and why it’s better than Bubble sort) | |  |
| **2** | Explanation | Daniel | * How bubble sort works * Explain what the gap does. * Gap shrinks each iteration * Once gap is 1, it uses regular bubble sort | |  |
| **3** | Physical example | Ivan | * Use measuring tape/string to demonstrate gap * Use blocks with numbers on them, or blocks with different heights to start sorting | |  |
| **4** | Code example | Ivan | * Explain specifically what each part of code does. | |  |
| **5** | Statistics + Tips | Daniel | * Why use 1.3 as a shrink factor * Speed vs Bubble sort. | |  |
|  |  |  |  | |  |
| **6** | Conclusion | Enfei | * Created by W.D. * Uses principles of bubble sort * Faster on average than bubble sort because it removes turtles (small values in the end) with the gap * 1.3 is the best shrink factor | |  |
| **TOTAL TIME** | | | | |  |
| **(if completed and submitted prior to presentation)** | | | | |  | **/10** |