ECE 368: DATA STRUCTURES AND ALGORITHMS
PROJECT 1, MILESTONE 1

Jain126@purdue.edu Kunwar Digraj Singh Jain HOMEWORK ID: **536358**

The current project we are working deals with **Shell Sort**, which is an improved form of the traditional **insertion sort**. We are supposed to develop an algorithm to use insertion sort to implement Shell Sort and also employ **Bubble Sort** to develop the same (shell sort). We are then supposed to compare the performance of both the algorithms, namely Shell Sort using insertion and Shell Sort using bubble. Basically, we need to see how shell sort improves insertion and bubble sorts.

To implement insertion sort, we are given the Pratt gap series which provides us with the gaps required to implement the shell sort using insertion sort. The 'gap' given by the series is used to break the array into smaller parts and sort them. This allows us to exchange elements that are very far apart. For example, if we are given an array: 2 4 1 7 4 9 5 3 9. Now if the initial gap is 3, the different arrays we see are: 2 7 5 and 4 4 3 and 1 9 9. We then apply insertion sort to all three sets of arrays and sort them separately. This allows to sort elements faster as far away elements are being compared easily. After one cycle, the array looks like:
2 3 1 5 4 9 7 4 9. In normal insertion sort, 3 would have taken a long time to get to its right place but with shell sort, the process was very quick. A similar strategy is employed to shell when using bubble sort as its fundamental sorting type. With every cycle completion, the gap is reduced to a smaller value and the arrays formed are sorted using bubble sort or insertion sort. In the last cycle, gap becomes 1 and it acts as a vanilla insertion or bubble sort.

Shell sort has improved efficiency over both insertion sort and bubble sort.