At the heart of computer science is Data Structure and sorting is an important issue in Data Structure that generates the sequence of the list of items. Just as sorting is important in everyday life is the same reason that it is important in programming- it is much easier to locate items that are sorted than unsorted [1]. Numerous computations and tasks become simple by properly sorting data in advance. It is worth nothing that an estimated 25% of the total active time of computers is devoted to sorting procedures [2], this makes it an area of high importance for further study and effort to improve the performance of the many electronic devices we now rely on so heavily. Much focus was dedicated on sorting collections of data that were too large for the computers of the day to store in memory [3]. In their article Paira, Chandra and Alam state that the effectiveness of sorting algorithms is to optimise the importance of other sorting algorithms and the optimality of these algorithms is judged when calculating their time and space complexities [4].

Time Complexity

Simplicity is an important characteristic of a good algorithm according to Chang,SK. A clear and well documented program is simple to read and easy to explain and maintenance can be made easily long after its conception either by the author or other programmers. Achieving a high efficiency requires some cost in the use of resources that the algorithm needs to reach such efficiency [5].

Desirable properties:

* Stability
* Time complexity- Good run-time- best, worst and average case time complexity
* In-place sorting- if memory is a concern
* Suitability- the properties of the sorting algorithm are well matched to the class of input instances which are expected- i.e. consider specific strengths and weaknesses when choosing a sorting algorithm

Insertion Sort

Merge Sort

Bucket Sort

An important aspect when selecting a sorting algorithm, the best-case, worst-case and average-case time complexity