ABSTRACT
The interest is to develop the fastest sorting algorithm and also efficient in all respect, has become one of the challenges of this century, resulting in many algorithm available to the individual, which needs to sort the list of different data. Presently, we have large number of data, we require some sorting techniques that can sort these data as quick as possible and also gives great efficiency with respect to space as well as time. In this paper, we will discuss some of the sorting algorithms and compare their time complexities for the list of data.

KEYWORDS: Sorting, insertion, merges, quick, bubble, selection.

INTRODUCTION
Sorting may be defined for the set of elements in the list by taking a permutations. The sorting algorithm is the best way to arrange the data items in a particular way. i.e. either ascending order or in descending order. Now, a days the data of any institution is in the large amount, that’s why we need to arrange them in a proper manner. To complete this task efficiently, we take an initiative towards it.

In this paper, we will focus on some sorting algorithms like insertion, selection, bubble, quick and merge on the basis of restrictions like what’s the approach that each of them follow, time complexity, space complexity, stability and many more. The process of every algorithm is distinct. We will also compare these algorithms on the basis of some parameter. The algorithm can also be helpful to arrange the data on the basis of certain requirements.

There are two types of sorting:
Internal Sorting:- Internal sorting is a process which takes place in primary memory of a computer system or takes place internally as the name suggest. It can be call when we have a small amount of data to be sorted. For a large set of data to be sorted, only a chunk of data is loaded in the memory at a time as the entire cannot exist in the main memory. The remaining data which is not loaded, kept or we can say stored in secondary memory.

External Sorting:- External sorting takes place externally and is used to sort the data items by using secondary storage devices such as floppy disk, hard disk we have two phases in external sorting technique i.e sorting phase and another one is merging phase. The data which can fit in the main memory is read, sorted and then written in a temporary file known as sorting phase. In merging phase the temporary files which contains sorted data is merged into a single large file.

WORKING PROCEDURE
Bubble sort:- Bubble sort sometimes called”sorting by exchange” as in order to find successive smallest element, the whole method relies heavily on the exchange of adjacent elements. This approach of sorting requires “n-1” passes, to sort the given list in an appropriate order.

Algorithm of bubble sort:-
Let us consider an Array A with N number of elements.
1. First step is to initialize i=0
2. Repeat steps 3 to 5 until i<N
3. Set j=0
4. Repeat step 5 until j<N-i-1
5. If A(j)>A(j+1) then
   Set temp=A[j]
   Set A[j+1]=temp
End if
6. Exit

Insertion Sort: In this type of sorting technique, the new element is inserted such that the preceding element follow some order. In insertion sort, the elements are considered one at a time. It is the algorithm which sorts the array by shifting its elements one by one. This sorting algorithm consists of two phases that is Searching and shifting. First the proper place of the element is searched after that shifting of elements takes place.

Algorithm for insertion sort:-
Let us consider an Array a with n number of elements
1. Set k=1.
2. For k=1 to (n-1)
   Set temp=a[k]
   Set j=k-1
3. Repeat step 4 until (temp<a[j]) and (j>=0)
4. Set a[j+1]=a[j]
5. Set temp=a[j+1]
6. End of for loop structure
7. Exit

Quick sort: This algorithm is based on “DIVIDE AND CONQUER”. It works by partitioning the array to be sorted and turn sorted recursively because we all know that it is much easier to sort small lists than long once.

This algorithm divides the list into three main parts-Pivot elements, Elements greater than the pivot elements, Elements less than the pivot elements.

Algorithm for Quick sort:
QUICK(A, p, r)
1. if p >=r then return
2. q = PARTITION(A, p, r)
3. QUICK(A, p, q - 1) Recursive call to Quick
4. QUICK(A, q +1, r)
5. Exit
PARTITION(A, p, r)
1. x = A[r]
2. i = p -1
3. for j = p to r - 1 do
4. if A[j] <=x then
5. i =i+1
END IF
Exchange A[i+1] and A[r]
return i+1
END FOR LOOP
Exit

Selection sort:- In selection sort technique, the successive elements are selected in order and placed into their proper position. It begins by finding the least element in the item list. A search is performed to locate the least element. If it is found the it is interchanged with the first element in the list. This sorting algorithm is simplest sorting algorithm.

Algorithm for selection sort:
Let A be an array with N number of elements
1. First of all initialize small with first array element i.e. small=A[L]
2. For I=L to U do{
3. small=A[I],pos=I
4. for J=I to U do{
5. { small=A[J]
6. pos=J
7. J=J+1}
8. temp=A[I]
10. A[pos]=temp
11. End

Merge Sort: Merging is the process of combining two or more sorted files into a third sorted file. Or in another way, this sort will work on the technique called divide and conquer. First it will divide the list into the two partitions or we can say i.e. parts, sort them and then merge the complete sorted list. T

```c
While(i<=M && j<=N)
{
    If(A[i]<B[j]){
        C[k]=A[i];
        K++
        i++
    }
    else{
        C[k]=B[j];
        K++
        j++
    }
}
While(i<=M)
{
    C[k++]=A[i++]
}
While(j<=N)
{
    C[k++]=B[j++]
}
```

<table>
<thead>
<tr>
<th>PARAMETER</th>
<th>INSERTION</th>
<th>SELECTION</th>
<th>BUBBLE</th>
<th>QUICK</th>
<th>MERGE</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sorting Approach</td>
<td>Insertion</td>
<td>Selection</td>
<td>Exchange</td>
<td>Partitioning</td>
<td>Merging</td>
</tr>
<tr>
<td>Time Complexity</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Best Case</td>
<td>O(N)</td>
<td>O(N^2)</td>
<td>O(N^2)</td>
<td>O(NlogN)</td>
<td>O(NlogN)</td>
</tr>
<tr>
<td>Worst Case</td>
<td>O(N^2)</td>
<td>O(N^2)</td>
<td>O(N^2)</td>
<td>O(N^2)</td>
<td>O(NlogN)</td>
</tr>
<tr>
<td>Average Case</td>
<td>O(N^2)</td>
<td>O(N^2)</td>
<td>O(N^2)</td>
<td>O(NlogN)</td>
<td>O(NlogN)</td>
</tr>
<tr>
<td>Sorting Type</td>
<td>Internal</td>
<td>Internal</td>
<td>Internal</td>
<td>Internal</td>
<td>Internal &amp; External</td>
</tr>
<tr>
<td>In Place</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>Algorithm Type</td>
<td>Incremental</td>
<td>Incremental</td>
<td>Incremental &amp; Exchange</td>
<td>Divide &amp; Conquer</td>
<td>Divide &amp; Conquer</td>
</tr>
<tr>
<td>Stability</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
<td>Typical In Place sort is not stable</td>
<td>Yes</td>
</tr>
</tbody>
</table>
TABLE 3.2: Advantages and Disadvantages of different sorting algorithms

<table>
<thead>
<tr>
<th>Sorting Type</th>
<th>Advantages</th>
<th>Disadvantages</th>
</tr>
</thead>
<tbody>
<tr>
<td>Insertion Sort</td>
<td>Simple and easy to implement. Faster than bubble sort.</td>
<td>The main disadvantage of insertion sort is that it is inefficient for large data list.</td>
</tr>
<tr>
<td>Bubble Sort</td>
<td>The main advantage of bubble sort is it is simple to use and easy to implement.</td>
<td>It is code inefficient.</td>
</tr>
<tr>
<td>Selection Sort</td>
<td>Simple and easy to implement</td>
<td>Inefficient for large lists, so similar to the more efficient insertion sort, the insertion sort should be used in its place.</td>
</tr>
<tr>
<td>Quick Sort</td>
<td>Fast and efficient. Very fast and it requires very less additional space.</td>
<td>Quick search is not a stable search. Show unpredictable results when list is already sorted.</td>
</tr>
<tr>
<td>Merge Sort</td>
<td>It can be applied to files of any size. Reading of the input during the run-creation step is sequential =&gt; Not much seeking. Reading through each run during merging and writing the sorted record is also sequential. The only seeking necessary is as e switch from run to run. If heap sort is used for the in-memory part of the merge, its operation can be overlapped with I/O Since I/O is largely sequential, tapes can be used.</td>
<td>If the recursion is used then it takes twice the space in memory as compared with quick sort algorithm</td>
</tr>
</tbody>
</table>

CONCLUSION
This paper discusses the different types of sorting techniques and compares the five different sorting algorithms. The insertion selection and bubble sorting algorithm will have almost the same complexity in worst and average case. But if we see the time complexity of insertion sort in comparison with bubble and selection then it is a better choice. The remaining algorithms are suitable for the small amount of data. The quick sort will work upon the large amount of data and all the data must be randomized.

REFERENCES
[2] Data structures using c and c++ by Yedidyah langsam,Aaron M. Tenenbaum second Indian printing (Prentice Hall of India private limited), New Delhi-110001