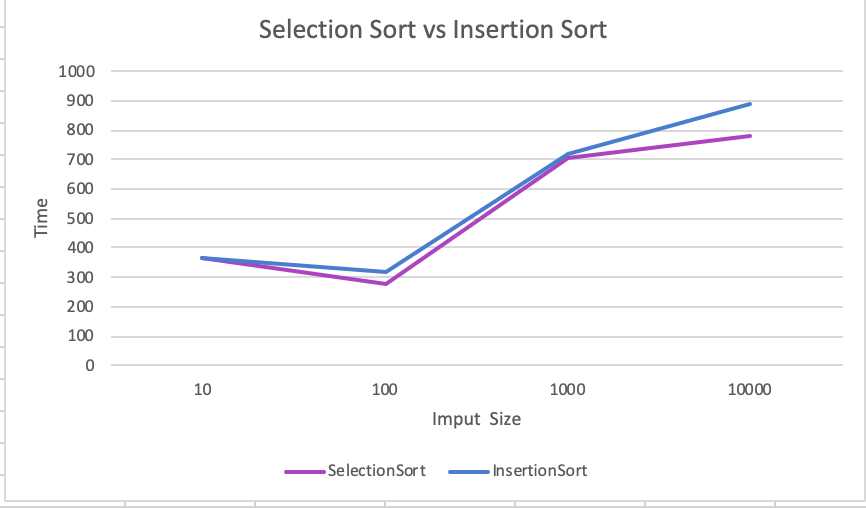
Practical 4:

**Quick Questions:**

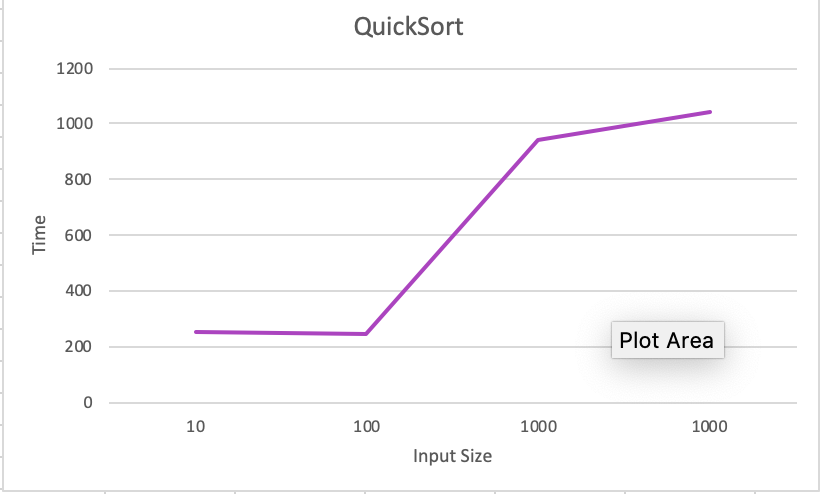
1. Linear
2. Stable sorting algorithms preserve existing relative order of elements when comparing  equal keys.
3. A
   1. Comparison vs. Non comparison
   2. Time complexity
   3. Space complexity
   4. Stability
   5. Internal vs. External
   6. Recursive vs. Non-recursive

|  |  |  |  |
| --- | --- | --- | --- |
| **Input size** | **Selection sort** | **Insert sort** | **Bogo sort** |
| 10 | 363 | 365 | 463 |
| 100 | 279 | 320 |  |
| 1000 | 702 | 719 |  |
| 10 000 | 781 | 890 |  |

|  |  |
| --- | --- |
| **Input size** | **Quick sort** |
| 10 | 252 |
| 100 | 284 |
| 1000 | 939 |
| 10 000 | 1040 |



We can see from the graph that insertion sort was slower compared to selection sort. This is due to the time complexity as it has which is o(n). this is the same time complexity for BogoSort which was very slow and shows that it is not a good sort to use as it is a very ineffective algorithm. It took very long to compile compared to the other sorts. The best sort was the selection sort, which has a time complexity of O(n^2).



The time complexity for quicksort is O(log n)

Time complexity for each sort:

* SelectionSort: O(n^2)
* InsertionSort: O(n)
* BogoSort: O(n)