Performance Analysis of Hash Table

# Time Execution Analysis:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Function** | **100**  (in microsecond) | **1000**  (in microsecond) | **10000**  (in microsecond) | **100000**  (in microsecond) |
| **Selection Sort** | 784 | 93258 | 5894048 | 521 |
| **Insertion Sort** | 14 | 203 | 238345 | 201 |
| **Built In Sort** | 198 | 243 | 25299 | 110 |

# Questions:

# 1:

The performance of these algorithms depends upon the number of unsorted items in the List. The number of elements inserted in linked list may be sorted for one sorting algorithm and may not be same for other algorithms. So this is the reason that there is so much difference in the values for all the three algorithms.

# 2:

The Reason behind this change is as the number of Nodes inserted in Linked List are random and they can be sorted for one algorithm and can be unsorted for another algorithm. So, when we are testing a algorithm then it behavior depends upon the number of unsorted numbers that can vary while testing the all algorithms. So this is the reason that there is no same scenario for all the algorithms

# 3:

Selection sort is best than built in algorithm when the number of nodes are small. While Insertion sot is best in all cases. Built in algorithm is giving showing average Behavior. So, insertion sort is best of all three.