|  |  |  |
| --- | --- | --- |
| **Name:** Nick Thompson |  | CSC 231-001 |
|  |  | Spring 2018 |
|  |  | Assignment 09, Extra Credit |

# Instructions

Complete the extra credit section from the assignment web page. Fill in the table below with your runtime data, and provide a short answer to subsequent questions.

For the runtime data table, be sure to include time units (s, ms, µs):

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | **list[]** | | **sorted list[] + binary search** | | **BST** | | | |
|  | *store* | *search* | *store* | *search* | *height* | *store* | *search* |
| users\_250000 | 231.2ms | 39ms | 1.1s | <1μs | 40 | 2.8s | <1μs |
| users\_1000000 | 960.9ms | 155.1ms | 5.2s | <1μs | 44 | 13s | <1μs |
| users\_5000000 | 4.8s | 773.7ms | 29.9s | <1μs | 55 | 79.6s | <1μs |

1. Your data should indicates that the simple list[] with a sequential search has the smallest store time but the worst search time. In general, in what circumstances would it be beneficial to suffer the overhead of using the sorted list[] + binary search or the BST?

Suffering the overhead is beneficial when you would be searching the list much more than you would be manipulating it.

1. The overhead of building the BST is the highest, but it has the same search time as the sorted list[] + binary search. So what is the benefit of using a BST for large data sets?

The benefit is that you avoid the data shifting (and duplicating) that comes with an array based list.