Name: Edgar Ontiveros

HYPOTHESIS**:**

I believe that the sequential/linear search algorithm O(n) will be more efficient on a smaller data set, while the binary search algorithm O(log(n) ) will be more efficient on a larger data set; I also believe that the JAVA api built-in binary search will be more efficient (faster) than a user-defined binary search. I expect JAVA's garbage collector CPU caching will influence the data results.

RESULTS**:**

|  |  |  |  |
| --- | --- | --- | --- |
| **ARRAY SIZE** | **Sequential** | **Binary** | **Built-in Binary** |
| **10** | **16938.4** | **10778.6** | **15957.4** |
| **100** | **18663** | **16445.6** | **18672.4** |
| **1000** | **29318.2** | **7760.8** | **17016.6** |
| **10000** | **153490.4** | **20449.2** | **21122.4** |
| **100000** | **742691** | **21496.2** | **18539.8** |
| **1000000** | **1383446** | **14844.2** | **23307** |
| **10000000** | **8168311.8** | **19093.8** | **19996.4** |

|  |  |  |  |
| --- | --- | --- | --- |
| **ARRAY SIZE** | **Sequential** | **Binary** | **Built-in Binary** |
| **10** | **15029** | **15583** | **15626.2** |
| **100** | **14720.8** | **12257** | **14633.4** |
| **1000** | **31043.2** | **9793.2** | **16685.8** |
| **10000** | **200362.8** | **19586.8** | **18010** |
| **100000** | **652211.2** | **31289.8** | **20658.4** |
| **1000000** | **1905818.2** | **14474** | **21784** |
| **10000000** | **13020012.6** | **20818.4** | **19003.2** |

CONCLUSION:

For most cases, not found cases took more time to search than found cases on all search algorithms (Sequential, User Binary, Built-in Binary).

For the found cases: User Binary was fastest and Sequential was slowest for 10 elements; User Binary was fastest and Built-in Binary was slowest for 100 elements; User Binary was fastest and Sequential was slowest for 1,000 elements; User Binary was fastest and Sequential was slowest for 10,000 elements; Built-in Binary was fastest and Sequential was slowest for 100,000 elements; User Binary was fastest and Sequential was slowest for 1,000,000 elements; User Binary was fastest and Sequential was slowest for 10,000,000 elements. Essentially, for the found cases, Sequential progressively became slower as the sample size increased (which is what was expected), User Binary occasionally became faster as the sample size increased, but stayed relatively consistent, and Built-in Binary was slightly slower than User Binary as the sample size increased, but stayed relatively consistent as well.

For the not found cases: Sequential was fastest and Built-in Binary was the slowest for 10 elements; User Binary was fastest and Sequential was slowest for 100 elements; User Binary was fastest and Sequential was slowest for 1,000 elements; Built-in Binary was fastest and Sequential was slowest for 10,000 elements; Built-in Binary was fastest and Sequential was slowest for 100,000 elements; User Binary was fastest and Sequential was slowest for 1,000,000 elements; Built-in Binary was fastest and Sequential was slowest for 10,000,000 elements. Built-in Binary was slightly faster in more cases than User Binary was, and the two stayed relatively consistent; Sequential, again, saw a huge performance loss as the sample size increased.