**Name:** Aras Soylu

**Student ID:** 22401732

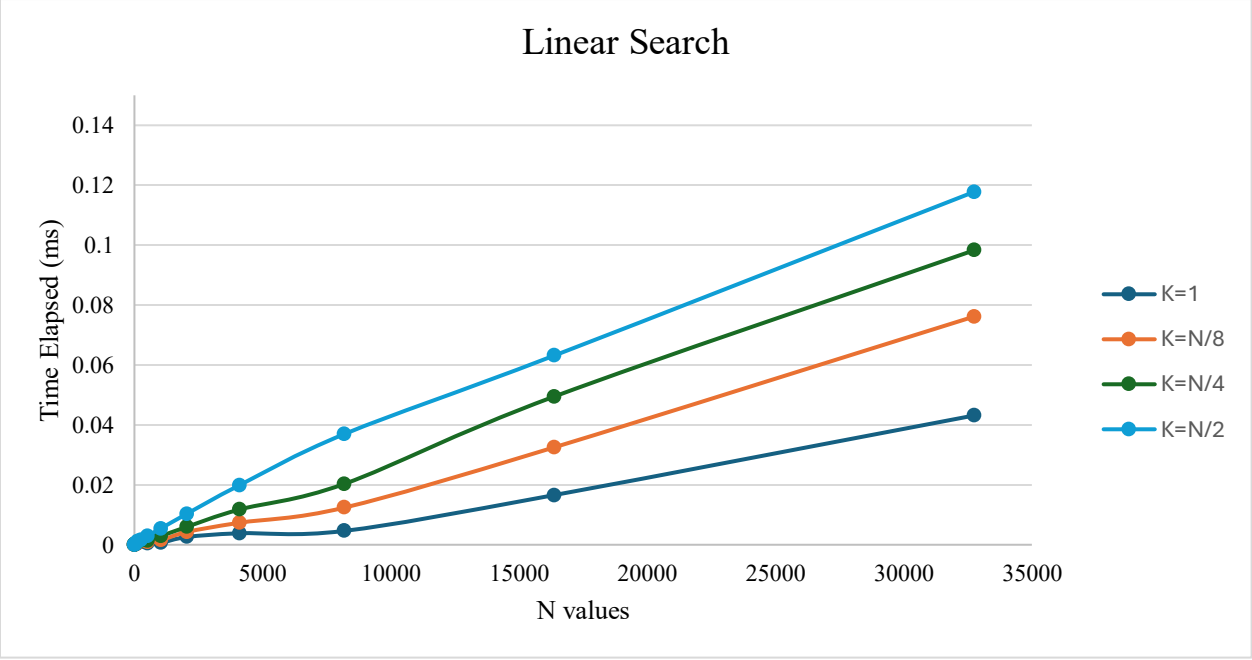
**Course/ Section:** CS201-2

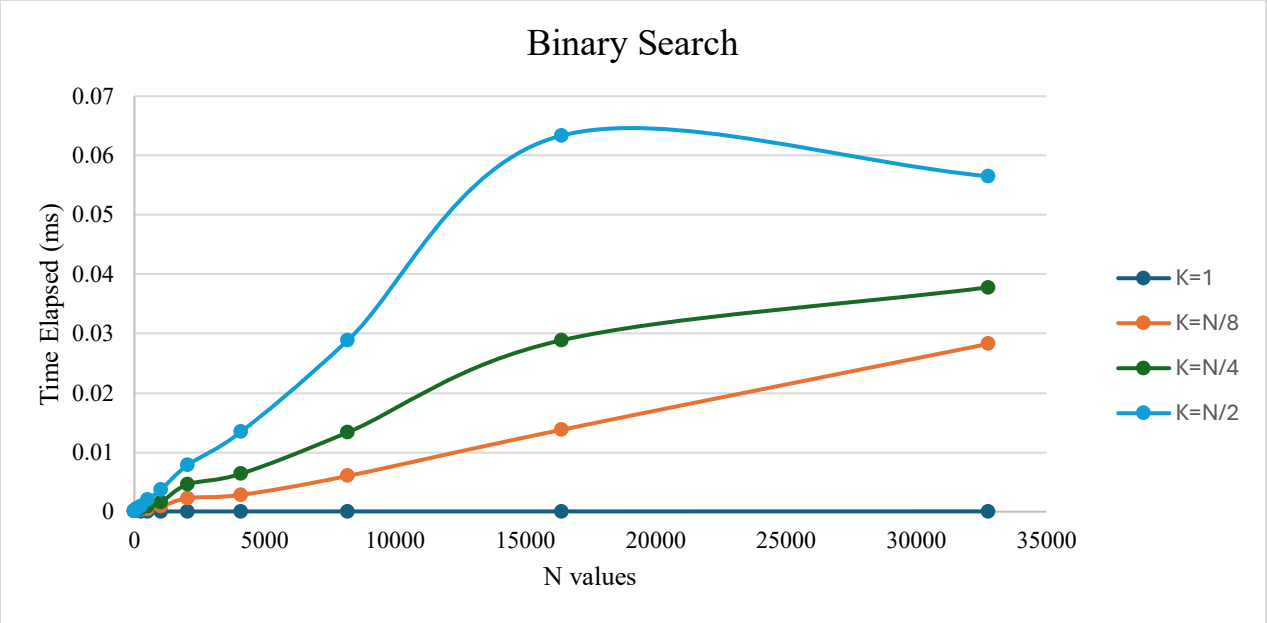
### 3.4 Table For the Results

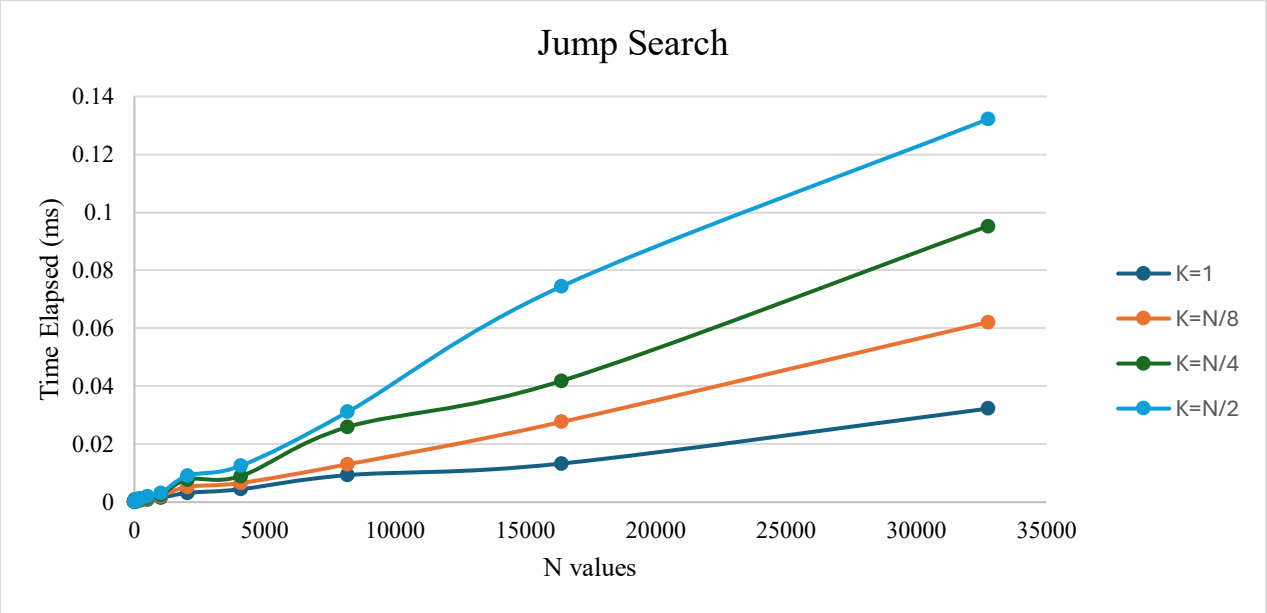
***Table 1: Results***

| N | Linear Search | | | | Binary Search | | | | Jump Search | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| K | K=1 | K=N/8 | K=N/4 | K=N/2 | K=1 | K=N/8 | K=N/4 | K=N/2 | K=1 | K=N/8 | K=N/4 | K=N/2 |
| 8 | 0,000048 | 0,000052 | 0,000054 | 0,00008 | 0,000078 | 0,000077 | 0,000101 | 0,000106 | 0,000087 | 0,000085 | 0,000089 | 0,000125 |
| 16 | 0,000063 | 0,000073 | 0,00009 | 0,000131 | 0,000078 | 0,000086 | 0,000107 | 0,000152 | 0,00093 | 0,000102 | 0,000118 | 0,000164 |
| 32 | 0,000137 | 0,00015 | 0,000166 | 0,000225 | 0,000079 | 0,000101 | 0,000121 | 0,000207 | 0,000153 | 0,000183 | 0,00021 | 0,000312 |
| 64 | 0,000224 | 0,000295 | 0,000384 | 0,000568 | 0,000073 | 0,000145 | 0,000233 | 0,000411 | 0,000228 | 0,000279 | 0,00037 | 0,000471 |
| 128 | 0,000497 | 0,000644 | 0,000842 | 0,001223 | 0,000081 | 0,000223 | 0,000336 | 0,00061 | 0,000412 | 0,000611 | 0,000787 | 0,001095 |
| 256 | 0,000847 | 0,00115 | 0,001352 | 0,001555 | 0,000061 | 0,00031 | 0,000558 | 0,000898 | 0,000594 | 0,000863 | 0,00112 | 0,00142 |
| 512 | 0,00045 | 0,000873 | 0,001344 | 0,003002 | 0,000061 | 0,000501 | 0,000849 | 0,002055 | 0,000909 | 0,001291 | 0,001623 | 0,002088 |
| 1024 | 0,000624 | 0,001478 | 0,00286 | 0,005404 | 0,000049 | 0,000855 | 0,001597 | 0,003738 | 0,001479 | 0,002073 | 0,002478 | 0,003103 |
| 2048 | 0,002588 | 0,004273 | 0,005984 | 0,010346 | 0,000043 | 0,002292 | 0,004601 | 0,007871 | 0,003189 | 0,005319 | 0,007864 | 0,009141 |
| 4096 | 0,003825 | 0,007341 | 0,011765 | 0,019855 | 0,000044 | 0,002838 | 0,0064 | 0,01347 | 0,00448 | 0,006645 | 0,009022 | 0,012594 |
| 8192 | 0,004627 | 0,012402 | 0,02033 | 0,036961 | 0,000047 | 0,006028 | 0,013376 | 0,028888 | 0,009328 | 0,013079 | 0,025962 | 0,031162 |
| 16384 | 0,016595 | 0,032549 | 0,049496 | 0,063181 | 0,000049 | 0,013794 | 0,028878 | 0,063366 | 0,013317 | 0,02766 | 0,041835 | 0,074412 |
| 32768 | 0,043172 | 0,07621 | 0,098331 | 0,117837 | 0,000052 | 0,028269 | 0,037798 | 0,05652 | 0,032284 | 0,062125 | 0,095292 | 0,13219 |

### 3.5 Plot For Each Algorithm






### 4.1 Specifications of Computer

Processor: 8-core CPU

RAM: 16 GB

Chip: Apple M2

Operating System: MacOS

Memory: 256 GB SSD

### 4.2 Algorithm Working With Unsorted Data

Linear Search: works with unsorted data as it iterates over each element.

Binary Search: does not work with unsorted data as it requires sorted data to recursively divide and search in half.

Jump Search: does not work with unsorted data as it requires sorted data for block selection.

### 4.3 Theoretical best / average / worst cases and effect of finding K closest values

#### 4.3.1 Theoretical Cases

##### 4.3.1.1 Linear Search

Best case: Θ(1)

Average case: Θ(N)

Worst case: Θ(N)

##### 4.3.1.2 Binary Search

Best case: Θ(1)

Average case: Θ(log N)

Worst case: Θ(log N)

##### 4.3.1.3 Jump Search

Best case: Θ(1)

Average case: Θ(N)

Worst case: Θ(N)

#### 4.3.2 Adapted Cases

##### 4.3.2.1 Linear Search

When we adapt the linear search algorithm to our example, we observe that the best case becomes Θ(K), the average case becomes Θ(N+K) and the worst case becomes Θ(N+K) as we take K neighbours of the target key.

##### 4.3.2.2 Binary Search

When we adapt the binary search algorithm to our example, we observe that the best case becomes Θ(N), the average case becomes Θ((log N)+K) and the worst case becomes Θ((log N)+K) as we take K neighbours of the target key.

#### 4.3.2.3 Jump Search

When we adapt the jump search algorithm to our example, we observe that the best case becomes Θ(K), the average case becomes Θ(N+K) and the worst case becomes Θ(N+K) as we take K neighbours of the target key.

## 4.4 Observed best / average / worst cases from the table and plots, and comparison

4.4.1 Observed results

From my experimental table and plots, it is seen that;

- For each algorithm the largest time elapses are recorded when N and K are the largest. Therefore, these values are the worst cases in my observed results.

- Likewise, for each algorithm, the smallest running times are recorded when N and K is the smallest, providing best case scenerios in the experiment.

- The intermediate values of N and K represent the average cases which we can understand from the plots for the observed results since the lines for K=N/8 and K=N/4 lie in between K=1 and K=N/2 for each algorithm.

### 4.4.2 Comparison between theoretical and observed

- For K=1, binary search is faster when compared to linear and jump search algorithms and grows logarithmically as N grows which maches with the theoretical time complexities (Θ(log N) versus Θ(N)).

- For other values of K, the elapsed time for all algorithms matches the theoritical time complexities as binary search grows with a smaller rate.

- Additionally, comparing linear and jump search with each other also supports the idea that there is a correlation between theoretical results and observed results as they both grow linearly with the increasing value of N with small non-consistencies.

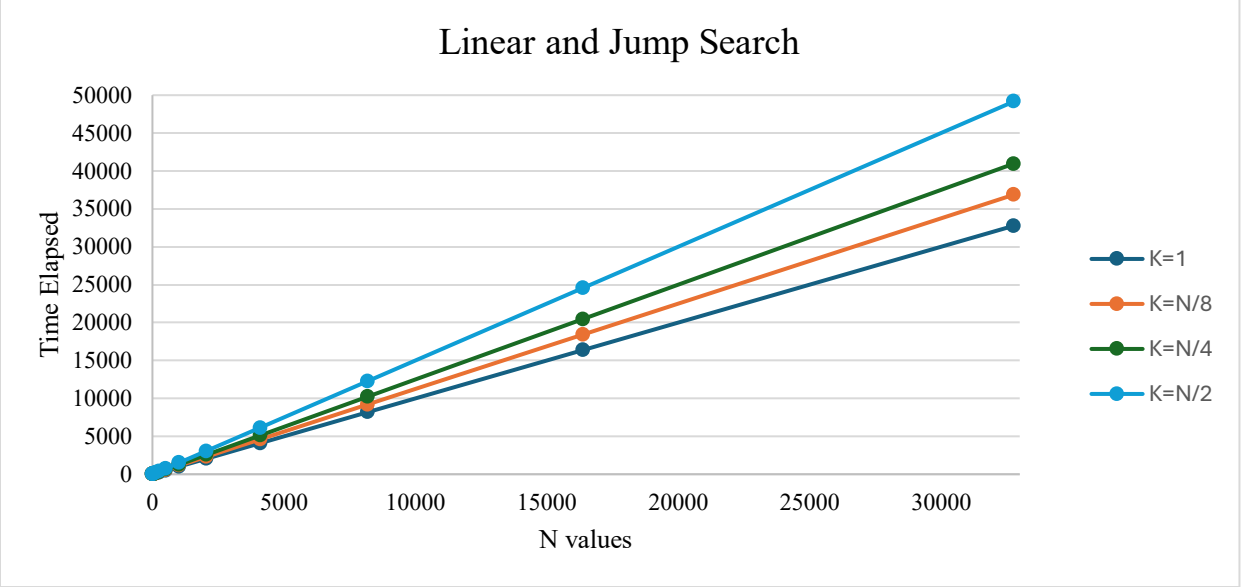
**- Possible reasons for inconsistencies:**

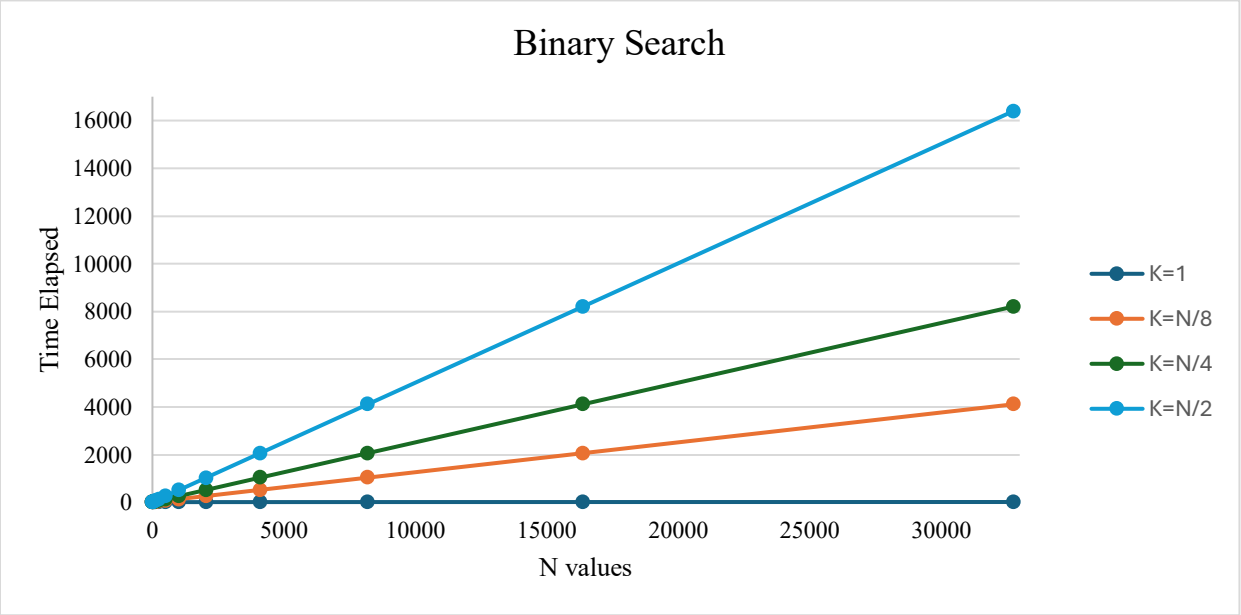
- implementation details which can show inconsistencies due to use of recursion and loops.

- background processes which can add small variations for small N values.

## 4.5 Theoretical Plots vs Experimental plots

### 4.5.1 Theoretical Plots




### 4.5.2 Comparison With Results

Both theoretical and observed plots for each search algorithm shows the same overall linear complexities. In the case of binary search, the observed value for K=N/2 at N=16384, there is an anomaly where the time elapsed exceeds the time at N=32768.