1. Running Environment
    a. Computer: MacBook Pro 2020
    b. Memory: 512gb
    c. Version: 18.0.1
    d. Processor: Apple M1 Chip
    e. OS: MacOS Ventura 13.2.1

2. Description
    a. For this experiment I coded three searching algorithms (linear, binary iterative, binary recursive), each of them goes through the randomly generated array of one million elements per epoch (1_000_000) amounts of time. We add the time it took for each of them and took the individual averages for each epoch.
    b. I reran the program 5 times to get a sense of performances.

Linear – A. Sizes (100_000, 1_000_000, 10_000_000)

1_000	0.000029	milliseconds
10_000	0.000030	milliseconds
100_000	0.000033	milliseconds

Binary Iterative – A. Sizes (100_000, 1_000_000, 10_000_000)

1_000	0.000032	milliseconds
10_000	0.000036	milliseconds
100_000	0.000042	milliseconds

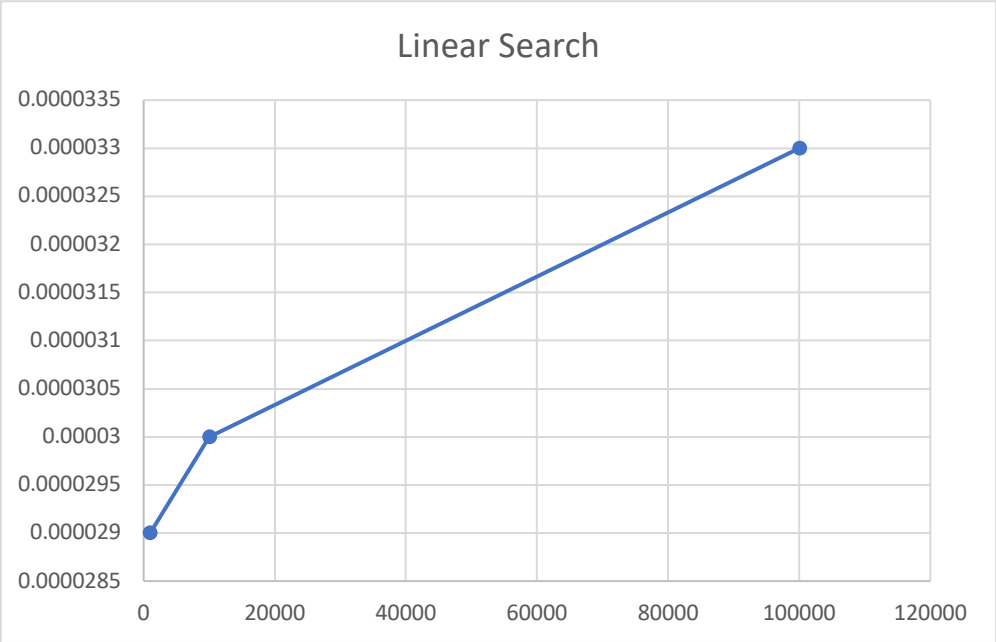
Binary Recursive – A. Sizes (100_000, 1_000_000, 10_000_000)

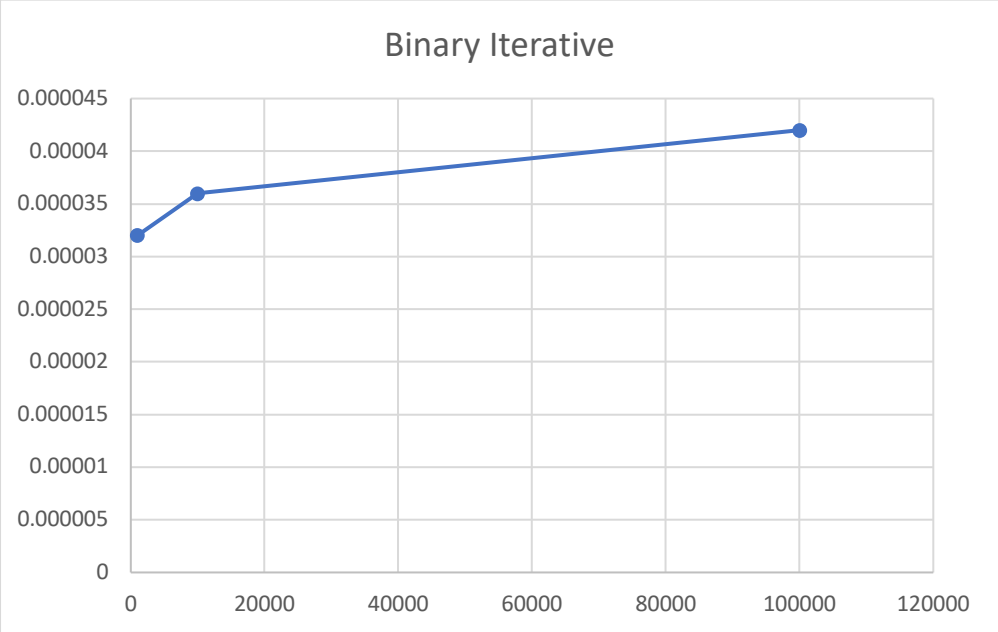
1_000	0.000038	milliseconds
10_000	0.000041	milliseconds
100_000	0.000055	milliseconds

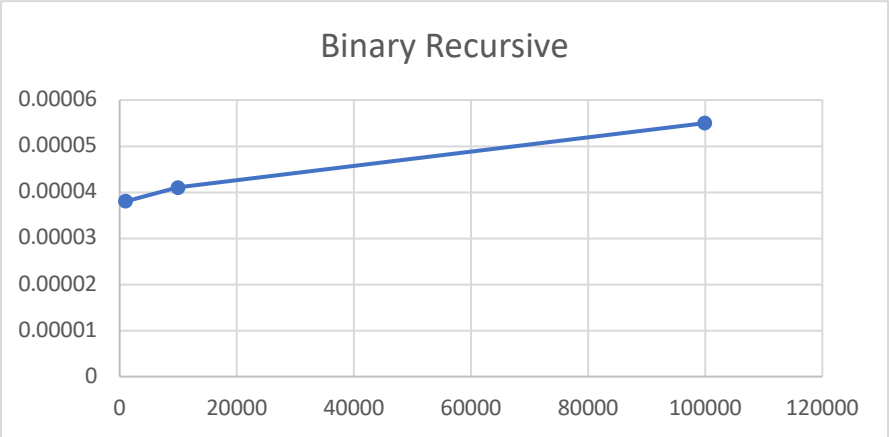
Array sort method

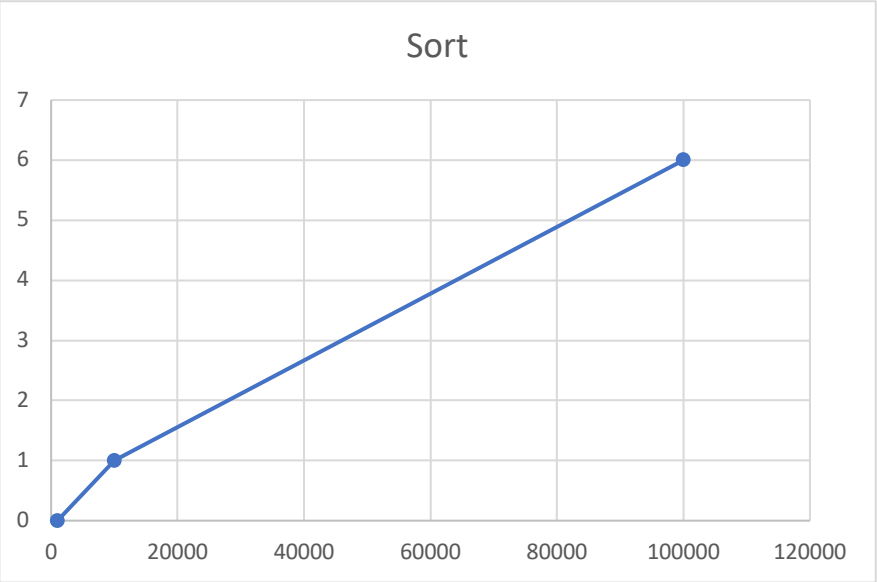
1_000	0.000000	milliseconds
10_000	1.000000	milliseconds
100_000	6.000000	milliseconds

3. Results


Linear Search
0.0000335
0.000033
0.0000325
0.000032
0.0000315
0.000031
0.0000305
0.00003
0.0000295
0.000029
0.0000285
0
20000
40000
60000
80000
100000
120000


Binary Iterative
0.000045
0.00004
0.000035
0.00003
0.000025
0.00002
0.000015
0.00001
0.000005
0
0
20000
40000
60000
80000
100000
120000





4. Factors
   Factors that may contribute to performance.
   - Code
   - Computer
   - Resources

5. Big-O time complexity
   a. Linear Search -> O(n)
   b. Binary Search -> O(log2(n))
6. Big-O time complexity estimation (sorting array)
   a. It seems to look like a linear function, thus O(n)

7. Runtime relations to big O
   a. The plots relate to complexity in terms of the linearity of the function, whether is linear, logarithmic, exponential, quadratic and so on.

Jose Alberto Robles Tostado
Cs 2401 – Lab 7