

Luke Broussard

1/28/2026

CS 320

Dr. Carl

Project 0

Q1. ThreeSum was significantly slower than ThreeSumFast. ThreeSum was getting 8x slower every time data was doubled, and Fast was 4x slower. So ThreeSum was $O(n^3)$ and TSFast was $O(n^2 \log n)$.

ThreeSum:

```
@Luke-broussard → /workspaces/dsa/src/projects (main) $ java -cp ../../algs4.jar DoublingRatio
250      0.0    2.8
500      0.0    3.2
1000     0.2    5.5
2000     1.5    7.8
4000     12.4   8.2
8000     24.2   2.0
16000    185.3  7.7
32000    1515.9 8.2
```

ThreeSumFast:

```
@Luke-broussard → /workspaces/dsa/src/projects (main) $ java -cp ../../algs4.jar DoublingRatio
250      0.0    1.5
500      0.0    8.7
1000     0.0    0.7
2000     0.1    2.9
4000     0.2    4.4
8000     1.0    4.4
16000    4.4    4.4
32000    17.8   4.0
```

Q2. Using a generic stack was anywhere from 40x to 100x times slower than using a stack made with normal numbers. Autoboxing is a really slow process that takes up a lot of runtime.

```
@luke-broussard → .../dsa/src/projects/project0 (main) $ java -cp ../../algs4.jar StackComparisons
N      Primitives  Generics  Difference
-----
1000000 0.022         0.050     2.27
2000000 0.001         0.068     68.00
4000000 0.001         0.156     156.00
8000000 0.006         0.247     41.17
16000000 0.007        0.608     86.86
32000000 0.009        1.117    124.11
```

Q3. The array based stack worked better than the linked list stack even though they basically do the same function. My guess is all the pointers in linked list cause the runtime to be slower than the array.

```
@luke-broussard → .../dsa/src/projects/project0 (main) $ java -cp ../../algs4.jar ListvsArray
Array Stack: 0.5520 s
Linked Stack: 1.0400 s
Ratio (List/Array): 1.88
This means that the linked list stack is 1.8840579710144927 times slower than the array stack for push operations.
```