Program Structures and Algorithms Spring 2023

(SEC - 1)

NAME: Pranav Kapoor

NUID: 002998253

Task:

Solve 3-SUM using the Quadrithmic, Quadratic, and (bonus point) quadraticWithCalipers approaches, as shown in skeleton code in the repository.

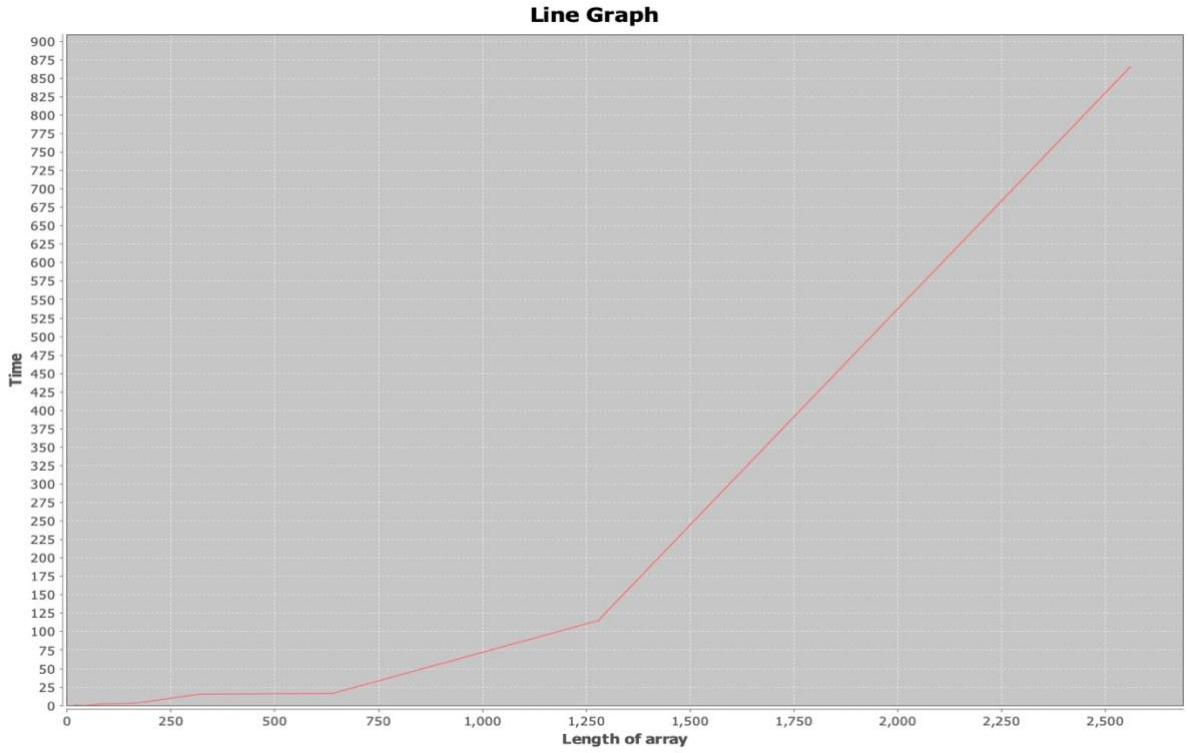
Relationship conclusion:

For 3 sum cubic,
The order of growth for is N^3.

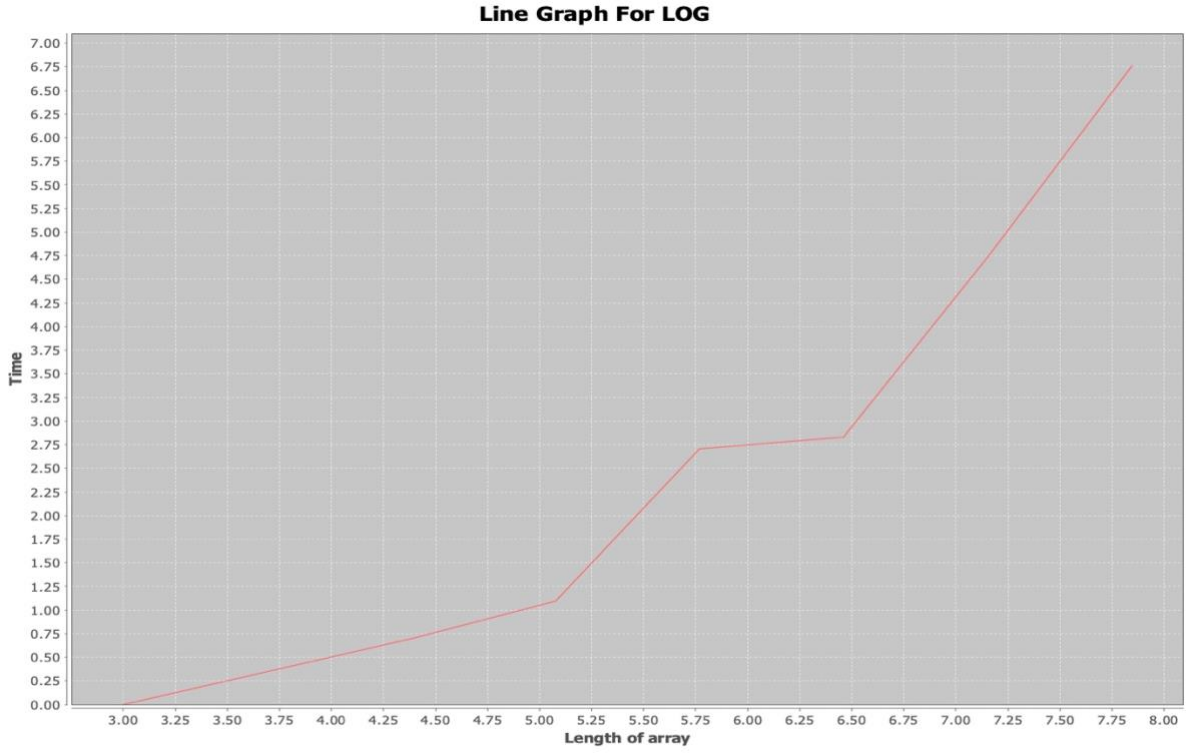
Values

```
Plot for Three Sum Cubic:
 x : 20 y : 1.0
 x : 40 y : 0.0
 x : 80 y : 2.0
 x : 160 y : 3.0
 x : 320 y : 15.0
 x : 640 y : 17.0
 x : 1280 y : 115.0
 x : 2560 y : 866.0

Three Sum Cubic Log:
 x : 2.995732273553991 y : 0.0
 x : 4.382026634673881 y : 0.6931471805599453
 x : 5.075173815233827 y : 1.0986122886681096
 x : 5.768320995793772 y : 2.70805020110221
 x : 6.461468176353717 y : 2.833213344056216
 x : 7.154615356913663 y : 4.74493212836325
 x : 7.847762537473608 y : 6.763884908562435
```

Plotting the graph of array length vs time:



Plotting the graph of ln(array length) vs ln(time):



Slope Calculation:

m = (y2 – y1) / (x2 – x1)
= (6.5 – 5) / (7.75 – 7.25) = 3

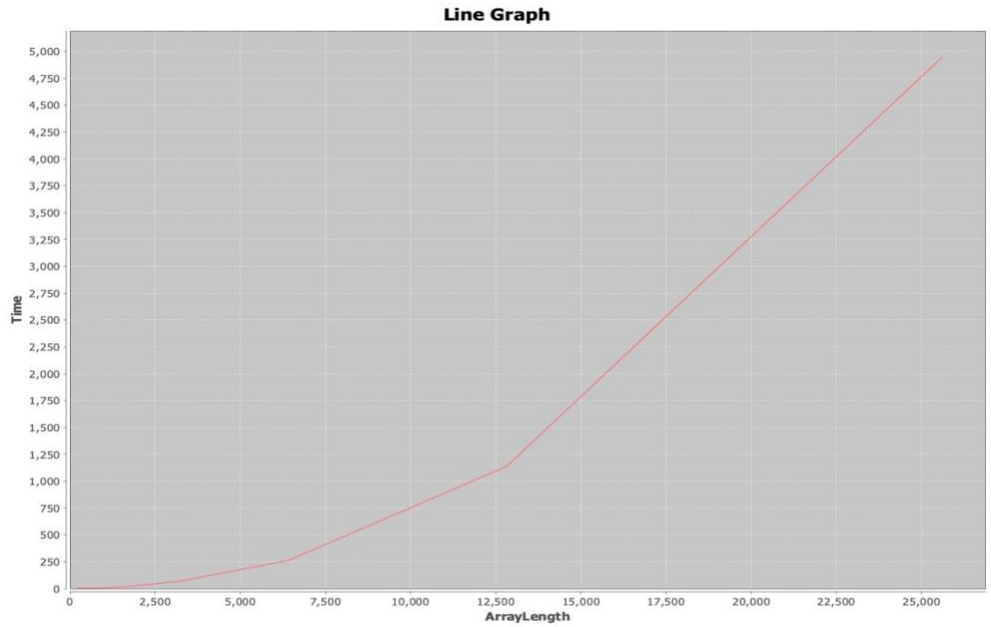
Equation is T = aN^3 hence proved

For 3 sum quadratics,
The order of growth for is N^2.

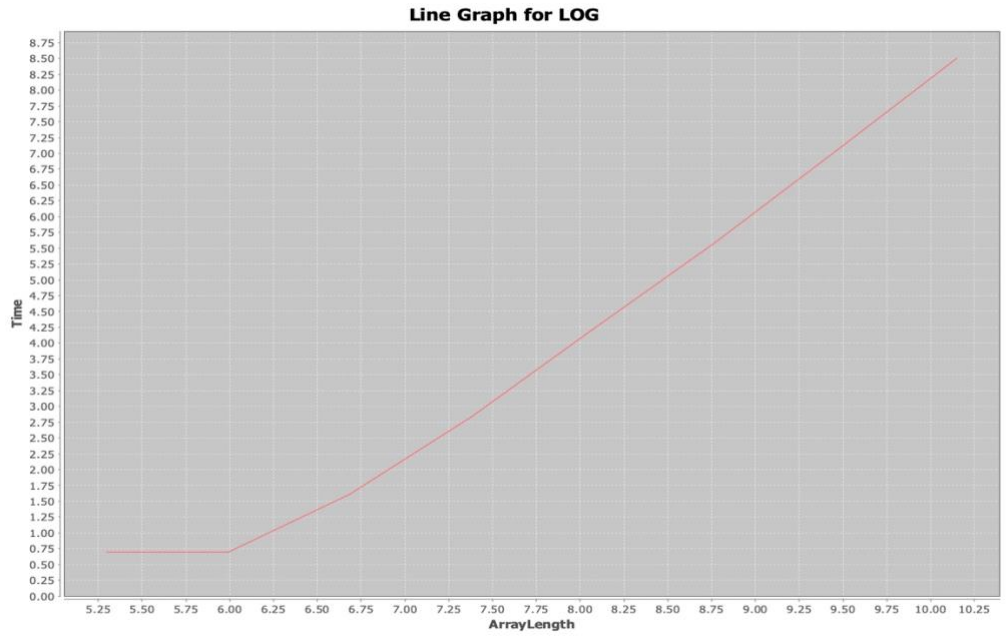
Values:

```
ThreeSumQuadratic [Java Application] /Users/Pranavkapoor1/Library

Three Sum Quadratics:
 x : 200 y : 2.0
 x : 400 y : 0.0
 x : 800 y : 2.0
 x : 1600 y : 8.0
 x : 3200 y : 17.0
 x : 6400 y : 13.0
 x : 12800 y : 48.0
 x : 25600 y : 192.0

Three Sum Quadratics:
 x : 5.298317366548036 y : 0.6931471805599453
 x : 6.684611727667927 y : 0.6931471805599453
 x : 7.3777589082278725 y : 2.0794415416798357
 x : 8.070906088787819 y : 2.833213344056216
 x : 8.764053269347762 y : 2.5649493574615367
 x : 9.457200449907708 y : 3.8712010109078907
 x : 10.150347630467653 y : 5.2574953720277815
```

Plotting the Graph of array length vs time:



Plotting the graph of ln(array length) vs ln(time):



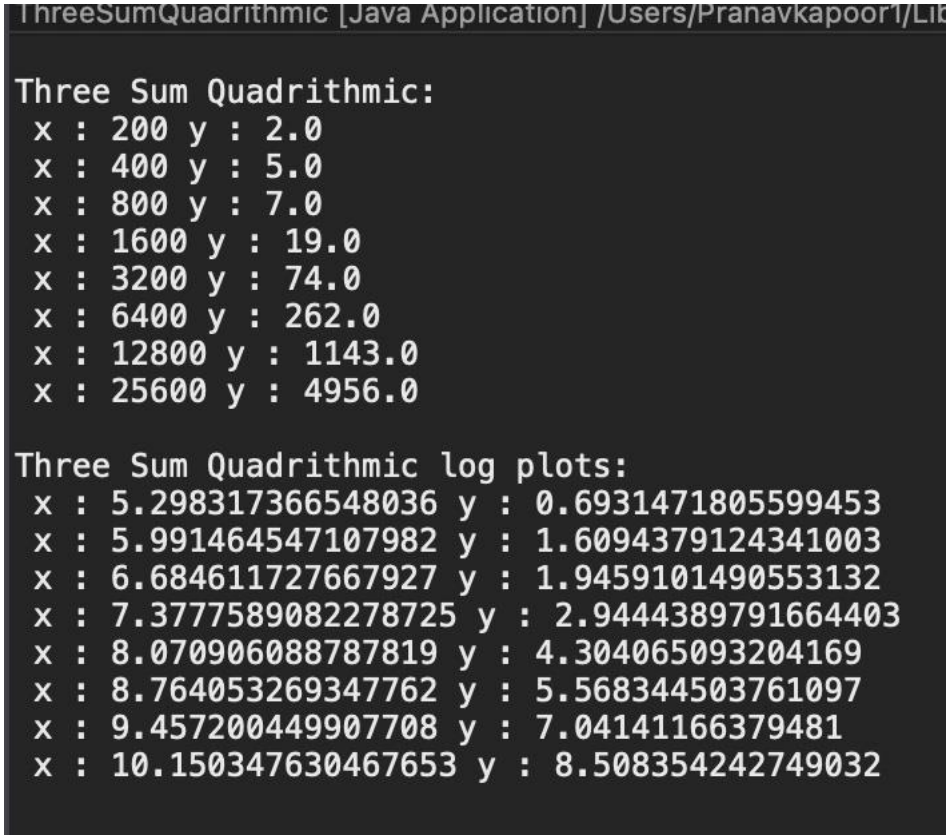
Slope Calculation:

m = (y2 – y1) / (x2 – x1)
= (8.25 – 5.5) / (10 – 8.75) = 2.7/1.25 = 2.16 ~ 2

Equation is T = aN^2 hence proved.

For 3 sum quadrathmic:

The order of growth for is N^2LogN.



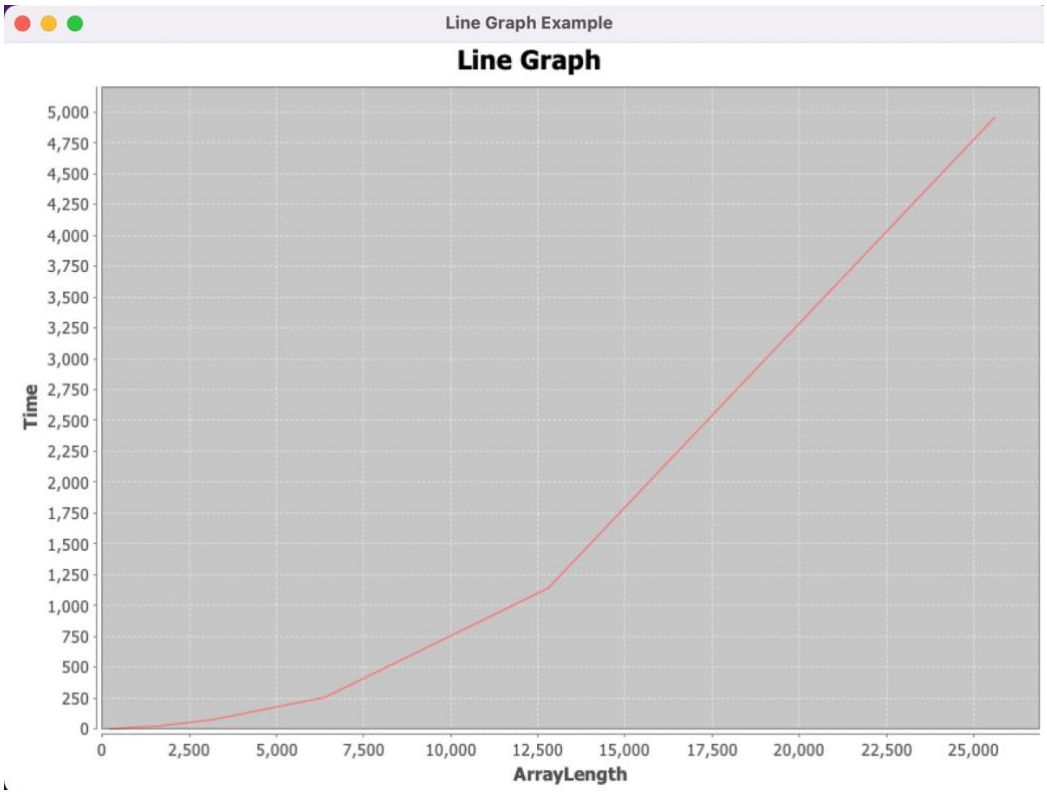
Slope Calculation:

m = (y2 – y1) / (x2 – x1)
= (8.2 – 6) / (10 – 9.0) = 2.2 > 2

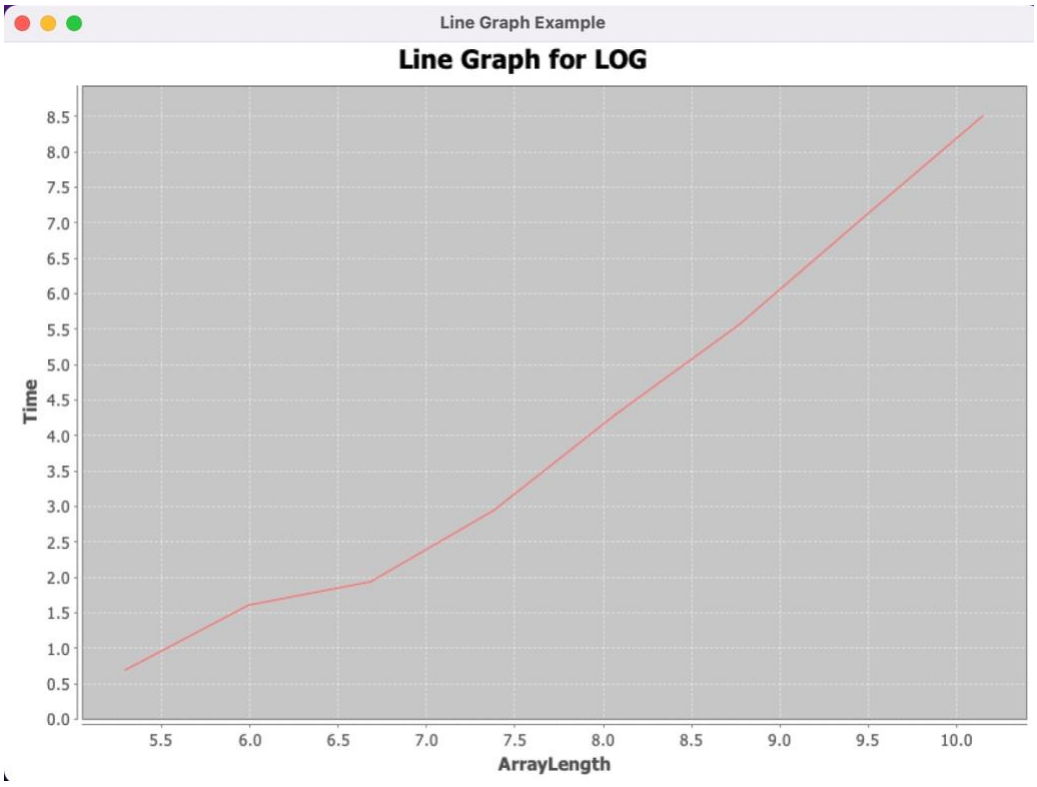
Slope is greater than N^2.

Hence proved

Plotting the graph of array length vs time:

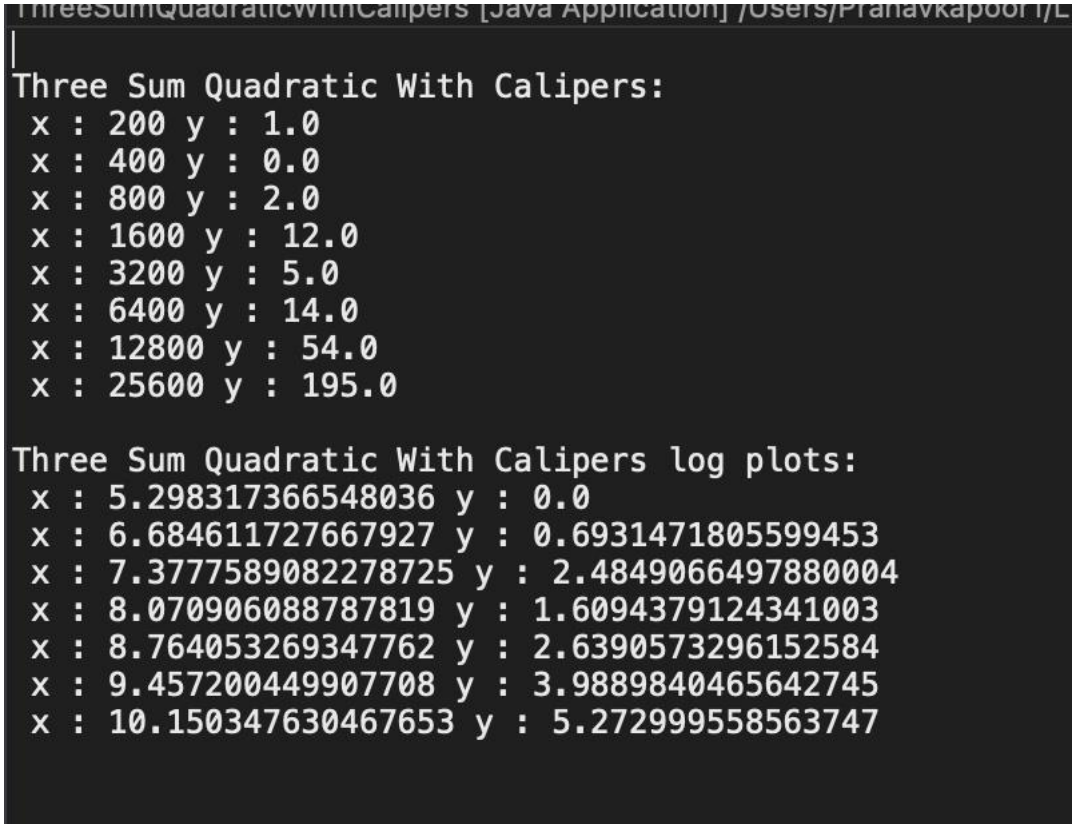


Plotting the graph of ln(array length) vs ln(time):



For 3 sum quadratics with calipers,

The order of growth is N^2.



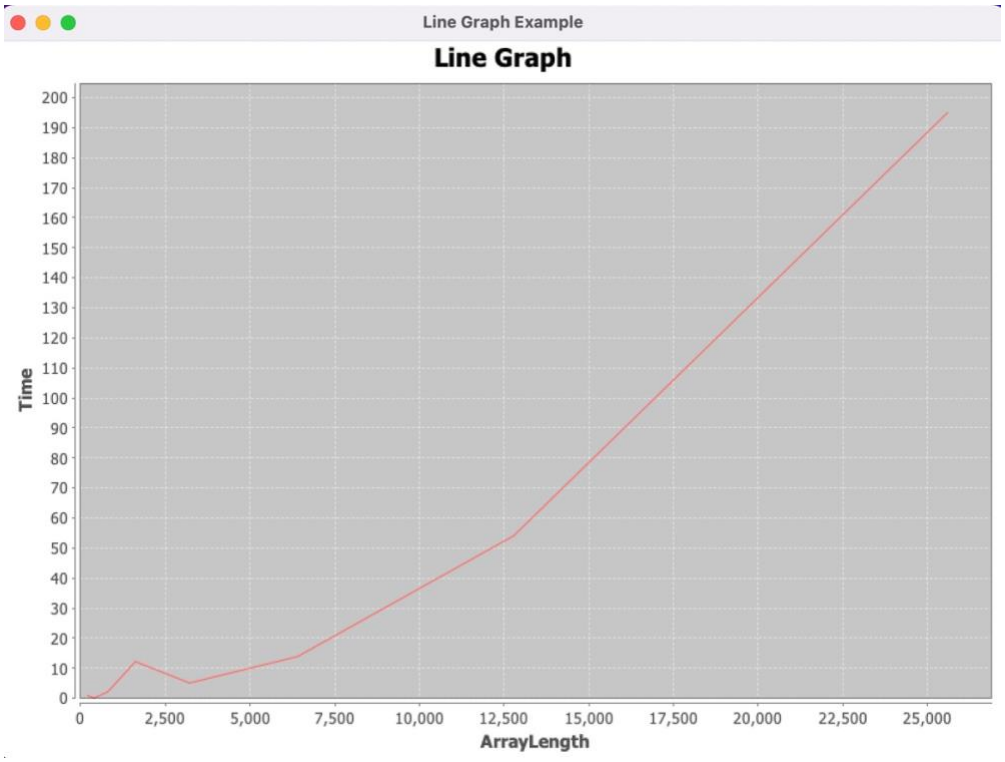
Slope Calculation:

m = (y2 – y1) / (x2 – x1)
= (5 – 4) / (10 – 9.5) = 1.00/0.5 = 2

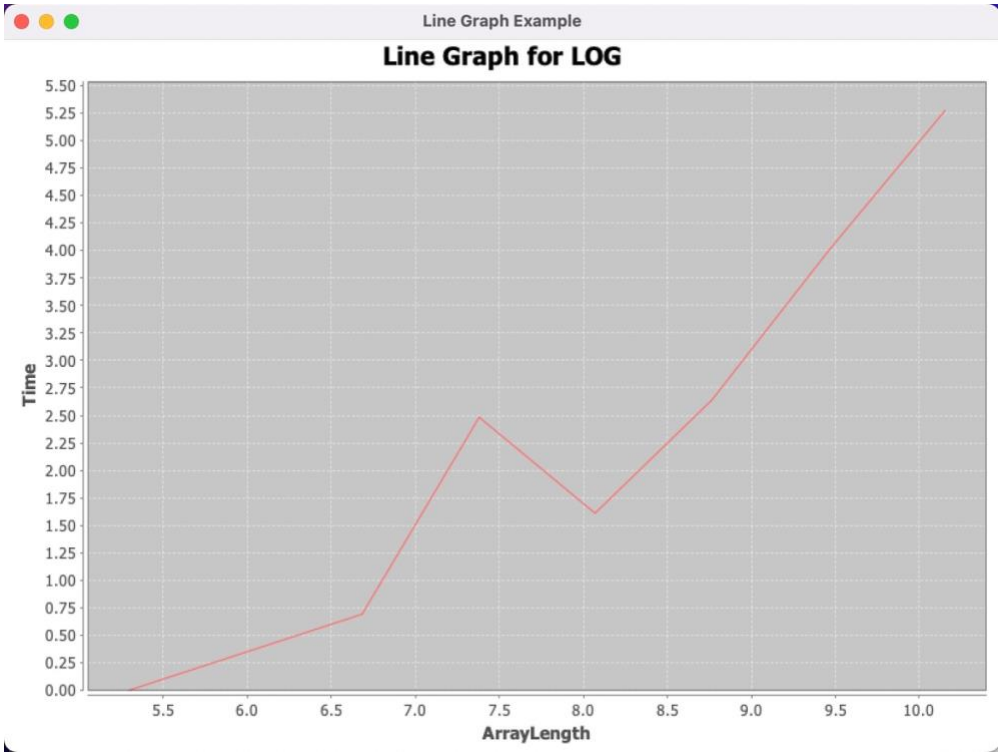
Equation is T = aN^2

Hence proved.

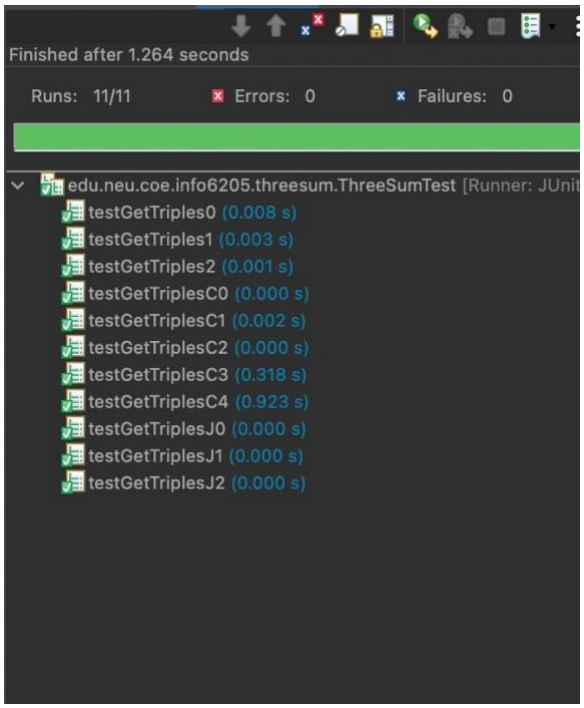
Plotting the graph and values of array length vs time:



Plotting the graph of ln(array length) vs ln(time):



Unit tests



Description brief explanation of how the quadratic method works.

1. Withoutcalipers

We are given the middle element of a triplet and use two pointers, one pointing to the element before the middle and one pointing to the element after. We then compare the sum of these three elements to the target value. If the sum is greater than the target, we move the right pointer to the left. If the sum is less than the target, we move the left pointer to the right. We repeat this process for each middle element. The overall complexity of this algorithm is O(N^2) as we have one loop to iterate through the middle elements and another loop to move the left and right pointers to find the triplets.

2. With calipers

We are provided with a starting element and use two pointers, one starting at the next element and another at the element before the last. We then compare the sum of these three elements to the target value. If the sum is less than the target, we move the left pointer to the right. If the sum is greater than the target, we move the right pointer to the left. We repeat this process for each starting element. The overall complexity of this algorithm is O(N^2) as we have one loop to iterate through the starting elements and another loop to move the left and right pointers to find the triplets.