

Program Structures and Algorithms

Assignment-2

Summer-2022

Dimpleben Kanjibhai Patel – 002965372

Problem: 3-SUM

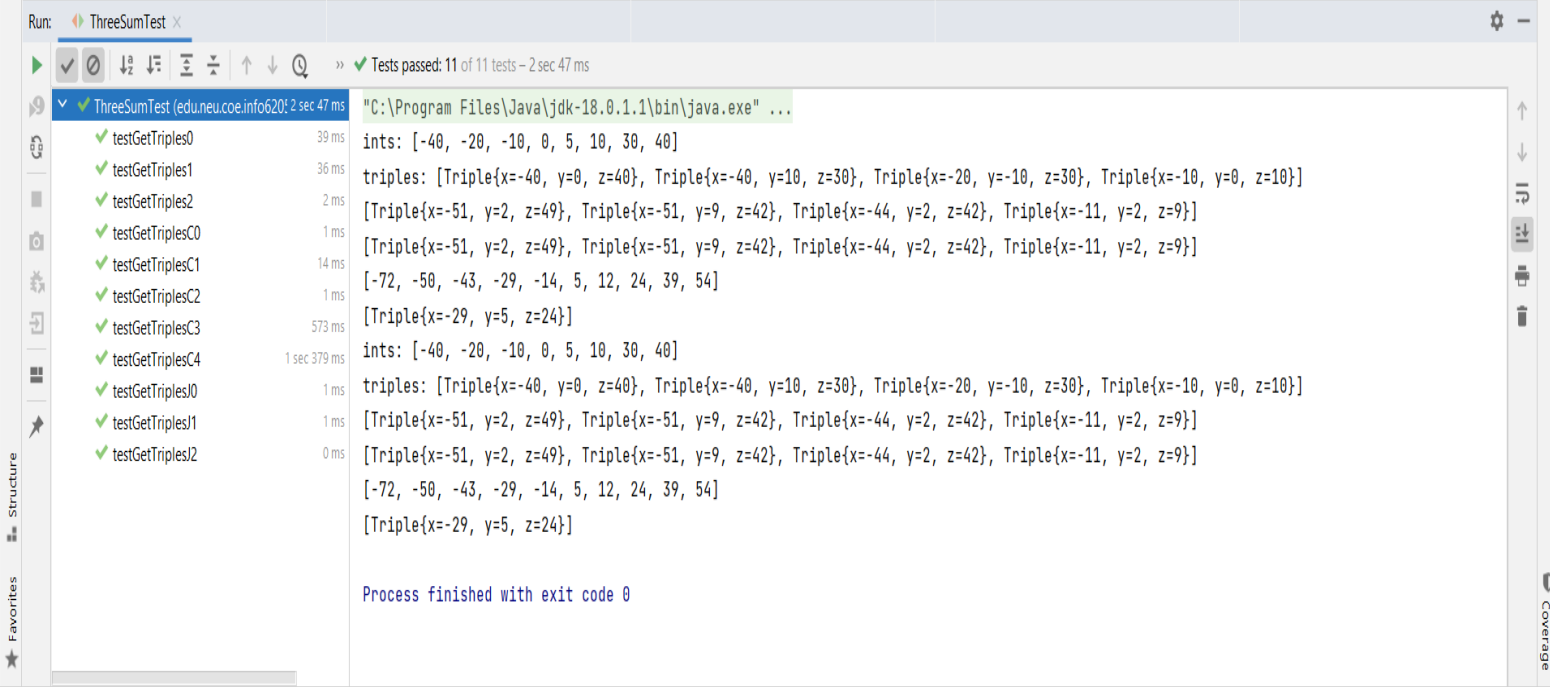
Task: Solve 3-SUM using the Quadrithmic, Quadratic and quadraticWithCalipers approaches.

Submit:

- Evidence of your unit tests running
- A spreadsheet showing your timing observations using the doubling method for at least five values of N- for each of the algorithms.
- Your brief explanation of why the quadratic methods works.

Output:

a. Evidence of all unit test passing:



```
Run: ThreeSumTest x
Tests passed: 11 of 11 tests - 2 sec 47 ms

ThreeSumTest (edu.neu.coe.info620) 2 sec 47 ms
  testGetTriples0 39 ms ints: [-40, -20, -10, 0, 5, 10, 30, 40]
  testGetTriples1 36 ms triples: [Triple{x=-40, y=0, z=40}, Triple{x=-40, y=10, z=30}, Triple{x=-20, y=-10, z=30}, Triple{x=-10, y=0, z=10}]
  testGetTriples2 2 ms [Triple{x=-51, y=2, z=49}, Triple{x=-51, y=9, z=42}, Triple{x=-44, y=2, z=42}, Triple{x=-11, y=2, z=9}]
  testGetTriplesC0 1 ms [Triple{x=-51, y=2, z=49}, Triple{x=-51, y=9, z=42}, Triple{x=-44, y=2, z=42}, Triple{x=-11, y=2, z=9}]
  testGetTriplesC1 14 ms [-72, -50, -43, -29, -14, 5, 12, 24, 39, 54]
  testGetTriplesC2 1 ms [Triple{x=-29, y=5, z=24}]
  testGetTriplesC3 573 ms ints: [-40, -20, -10, 0, 5, 10, 30, 40]
  testGetTriplesC4 1 sec 379 ms triples: [Triple{x=-40, y=0, z=40}, Triple{x=-40, y=10, z=30}, Triple{x=-20, y=-10, z=30}, Triple{x=-10, y=0, z=10}]
  testGetTriplesI0 1 ms [Triple{x=-51, y=2, z=49}, Triple{x=-51, y=9, z=42}, Triple{x=-44, y=2, z=42}, Triple{x=-11, y=2, z=9}]
  testGetTriplesI1 1 ms [Triple{x=-51, y=2, z=49}, Triple{x=-51, y=9, z=42}, Triple{x=-44, y=2, z=42}, Triple{x=-11, y=2, z=9}]
  testGetTriplesI2 0 ms [-72, -50, -43, -29, -14, 5, 12, 24, 39, 54]
  [Triple{x=-29, y=5, z=24}]

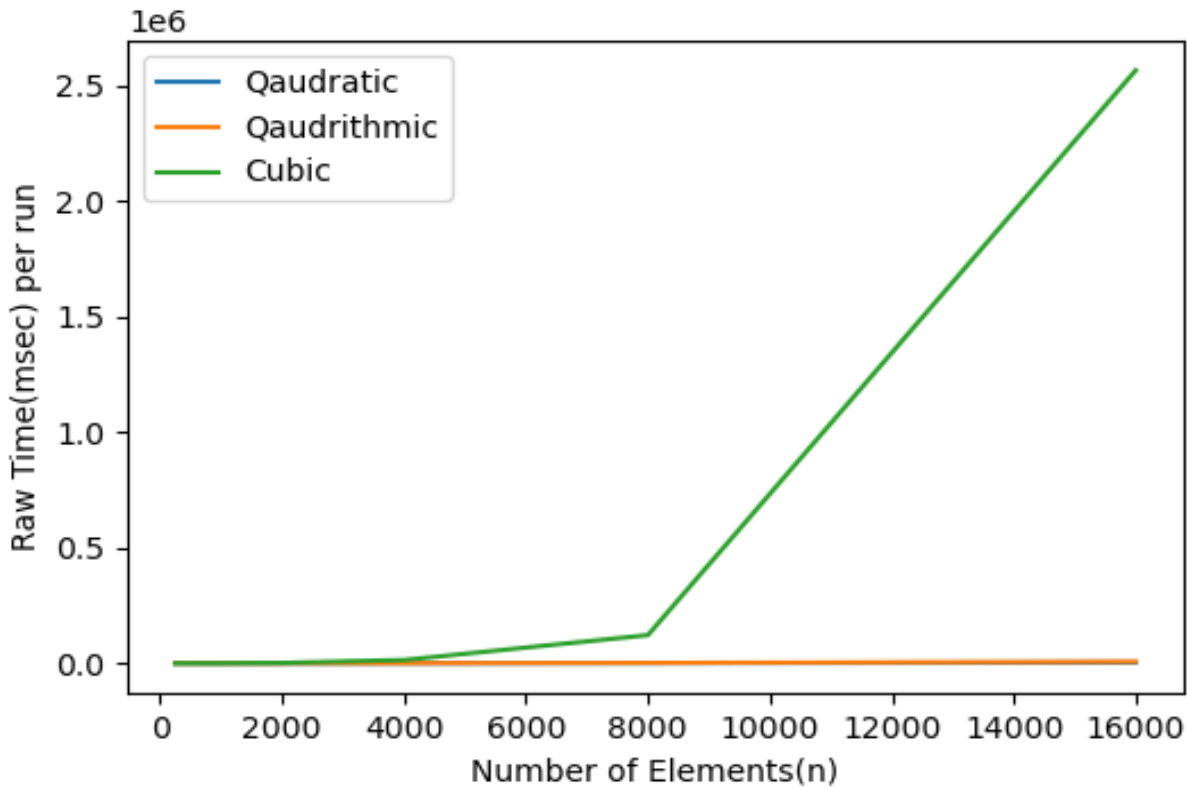
Process finished with exit code 0
```

b. Timing Observation:

Observation Table for Raw time per run:

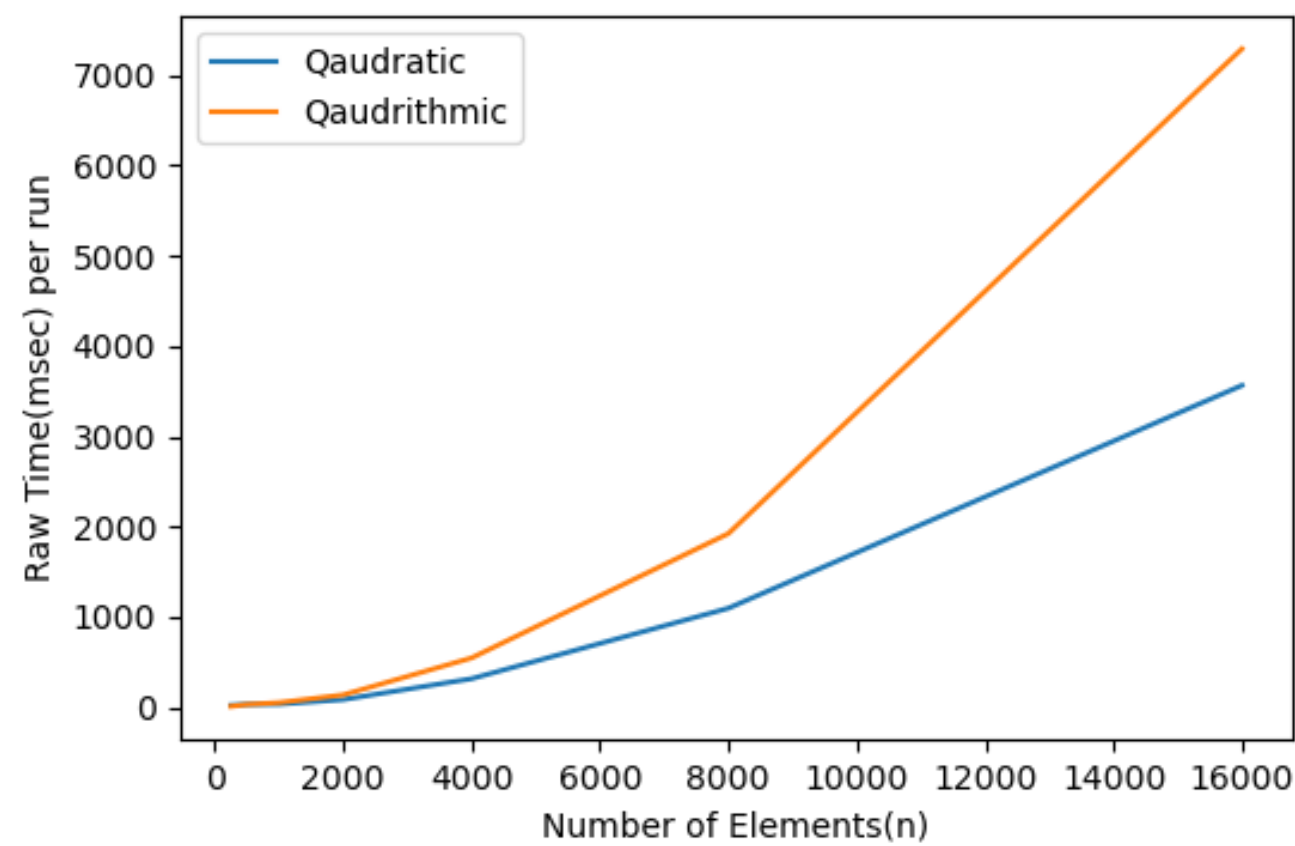
N	Qaudratic (n^2)(msec)	Quadrithmic ($n^2 \log(n)$)(msec)	Cubic(n^3)(msec)
250	24	12	25
500	34	33	69
1000	36	53	276
2000	87	139	1898
4000	318	548	14458
8000	1098	1924	122062
16000	3563	7287	2566687

Graph for Qaudratic Vs Qaudrithmic Vs Cubic



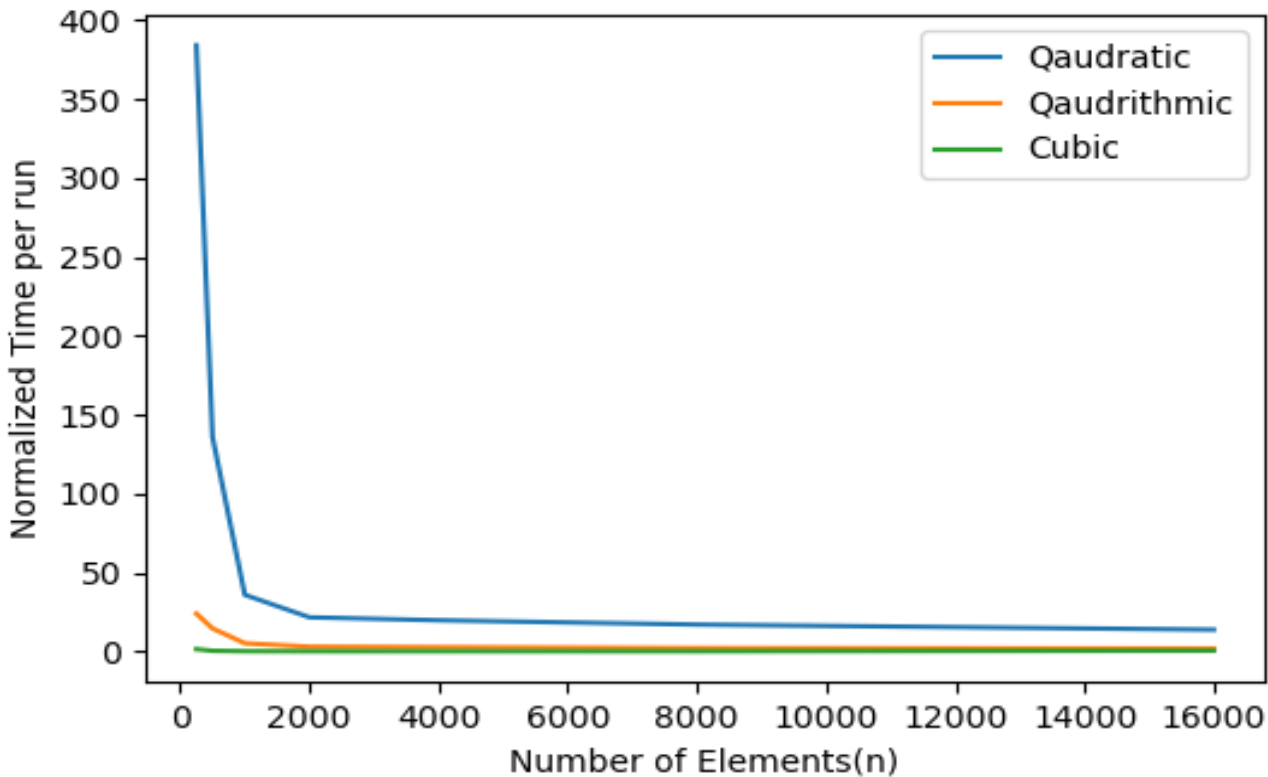
As from the above graph, we can observe for larger value of n , time is drastically increases. We can not see clear difference for Quadratic and Quadrithmic solutions, so we will plot separate graph for that.

Graph for Quadratic Vs Quadrithmic



Observation Table for Normalized time per run:

N	Qaudratic (n ²)	Quadrithmic (n ² log(n))	Cubic(n ³)
250	384	24.10	1.60
500	136	14.72	0.55
1000	36	5.32	0.28
2000	21.75	3.17	0.24
4000	19.88	2.86	0.23
8000	17.16	2.32	0.24
16000	13.92	2.04	0.63



c. Explanation:

Cubic solution is a brute force approach which takes $O(n^3)$ time complexity as there are three nested loops for traversing the array.

For Quadrithmic solution, first we sort the array, then we have two nested loops for the elements and third element we can search using binary search algorithm. Therefore it will take $O(n^2 \log(n))$ time complexity.

For Quadratic solution, First we will sort the array which will take $n \log(n)$ time complexity. After sorting an array, we will fix first element of the triplet. Then we will use two pointer method to find whether there is any pair whose sum is equal to $(0 - \text{first element})$. The two-pointer method will take linear time i.e. n and fixing each element will take n time so total time complexity will be $O(n^2)$.