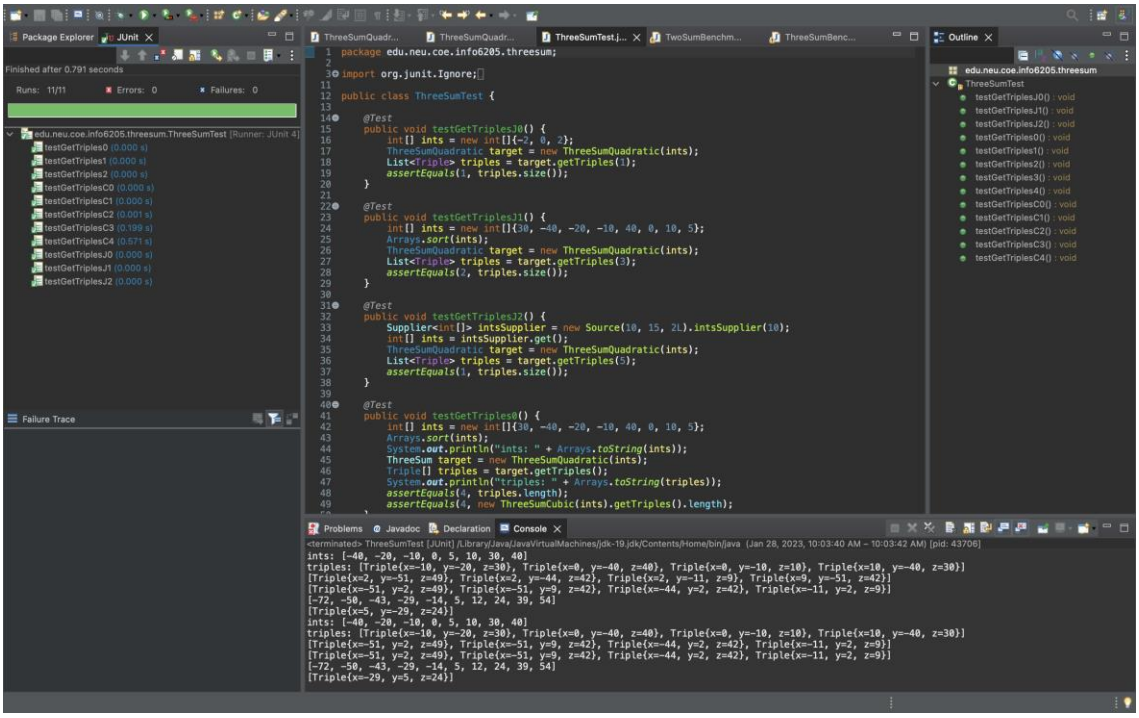
Assignment 2 (3-SUM)

(a) Evidence (screenshots) of unit tests running:



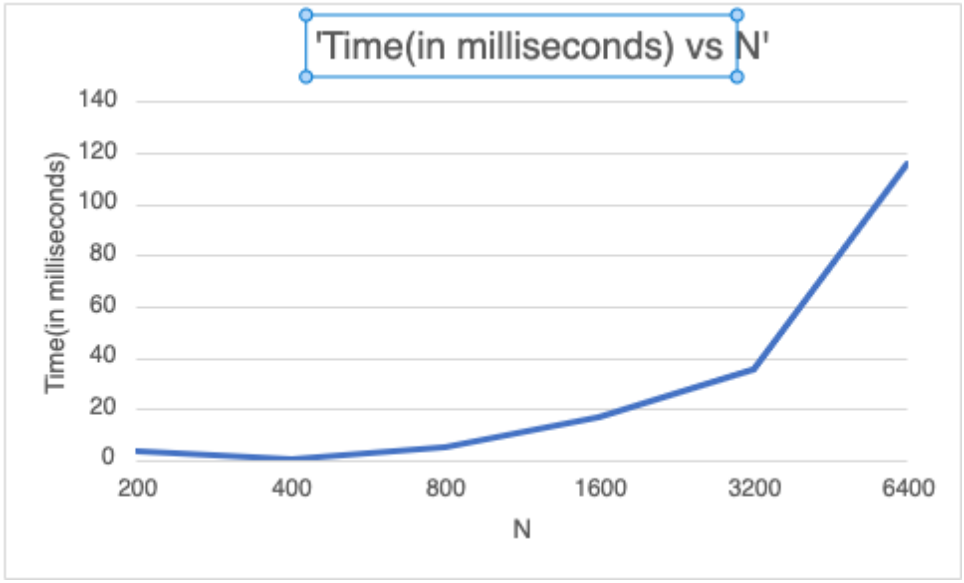
(b) A spreadsheet showing your timing observations:

ThreeSumQuadratic :

Table which shows the raw time as well as the normalized time:

N	Time(in milliseconds)
200	10
400	1
800	5
1600	17
3200	30
6400	121

Graph:



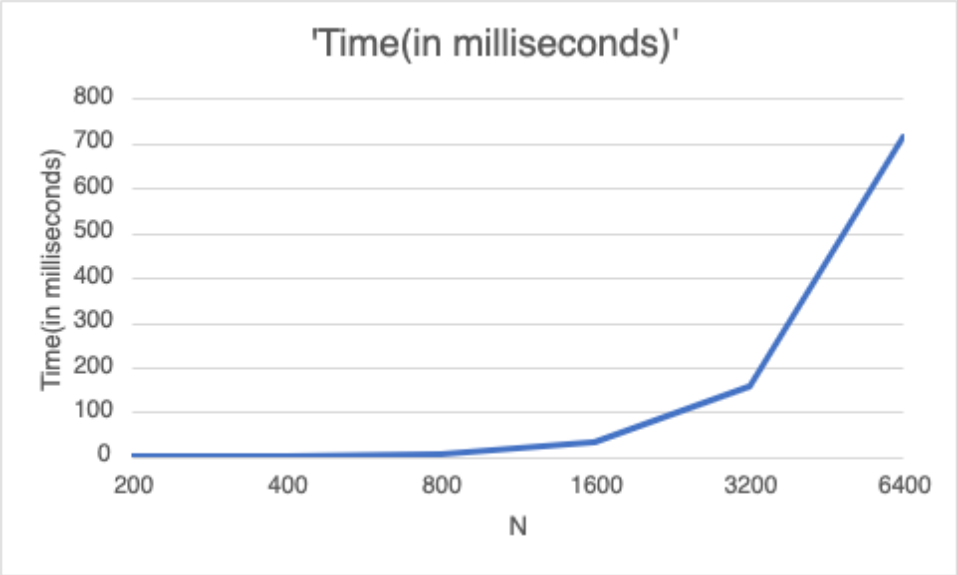
ThreeSumQuadrithmic :

Table which shows the raw time as well as the normalized time:

N	Time(in milliseconds)
200	2
400	4
800	8

1600	33
3200	158
6400	717

Graph:

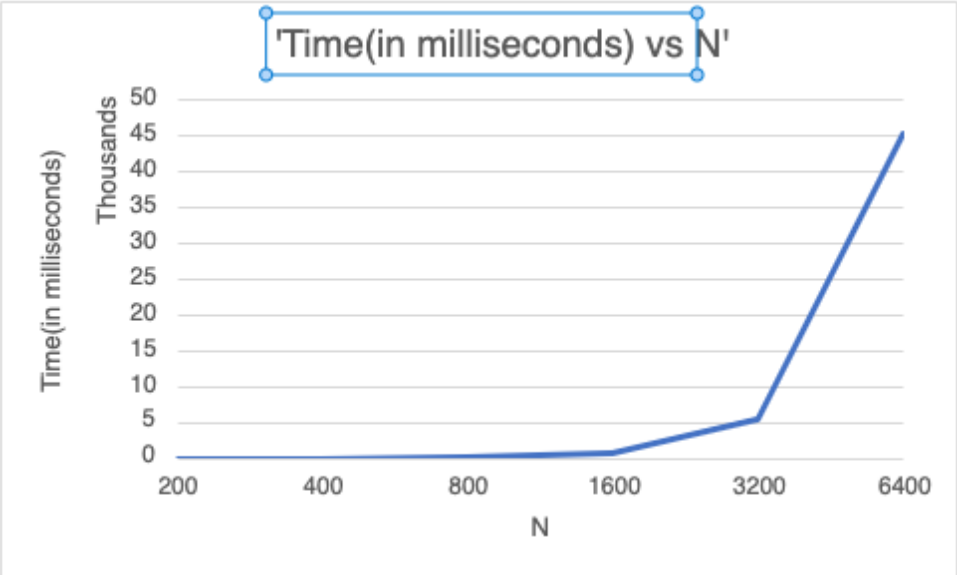


ThreeSumCubic

Table which shows the raw time as well as the normalized time:

N	**Time(in milliseconds)**
200	5
400	12
800	93
1600	713
3200	5650
6400	45273

Graph:



(c) your brief explanation of why the quadratic method(s) work:

The quadratic technique, commonly referred to as the "two-pointer approach," is an algorithm used to locate all triplets that are distinct and add up to a specific target within an array. The quadratic method's fundamental principle is to first sort the array before using two pointers—one starting at the array's first element and the other at its last element—to iteratively search through the array for any distinct triplets that add up to the desired sum.

The algorithm works by using the first pointer in the to fix one triplet element, while the second pointer is used to locate the other two triplet elements. As it moves toward the first pointer and the middle of the array, the second pointer begins at the end of the array. The first pointer is shifted to the right if the total of the three components is less than the desired total. The second pointer is pushed

to the left if the sum is higher than the intended sum. Until all distinct triplets that add up to the desired amount have been identified, this process is repeated.

This technique functions because it makes use of the array's sorted property. Any elements between the first and last elements of a sorted array cannot amount to the goal sum if the sum of the first and last elements is less than the target sum. It is also impossible for any items between the first and last element to total to the target sum if the sum of the first and last element is greater than the target sum. Therefore, the algorithm effectively reduces the number of potential solutions by shifting the first and last pointers toward one another.