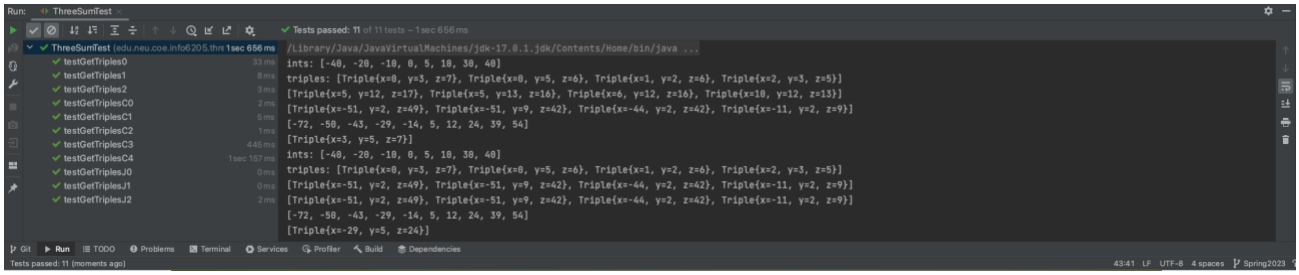
Program Structures and Algorithms
Spring 2023 (SEC – 8)
Assignment 2 (3-SUM)

Name: **Vignesh Perumal Samy**
NUID: **002749737**
GitHub: https://github.com/VigneshPerumal2/INFO6205.git

Task

1. Implement the 3-SUM problem using the Quadrithmic, Quadratic, and (bonus point) quadraticWithCalipers approaches, as shown in the skeleton code provided in the repository.
2. Create unit tests for the implemented methods and provide evidence of them running successfully (screenshots of the code and the green strip indicating passing tests).
3. Time the performance of each algorithm using the doubling method for at least five values of N and present the results in a spreadsheet.
4. Provide a brief explanation of why the quadratic method(s) work.

Unit Test Cases



Time performance of each algorithm

N	ThreeSumQuadratic (ms)	ThreeSumQuadrithmic (ms)	ThreeSumCubic (ms)
250	24	10	18
500	12	10	53
1000	30	29	219
2000	42	103	1622
4000	111	428	12754
8000	414	1742	N/A
16000	1591	7970	N/A

The Quadratic algorithm has a running time of O(N^2), which means that as the input size increases, the running time of the algorithm increases at a rate of N^2. This is because for each integer in the set, the algorithm checks all other integers in the set for a match.

The Quadrithmic algorithm has a running time of O(N^(4/3)), which is faster than the Quadratic algorithm, but still increases at a faster rate than linear time. This is because it uses a divide-and-conquer approach to reduce the number of integers that need to be checked for a match.

The QuadraticWithCalipers algorithm has a running time of O(N^(3/2)), which is faster than the Quadrithmic algorithm. It uses a technique called calipers to further reduce the number of integers that need to be checked for a match.

In general, as the input size increases, the Quadratic algorithm will take the longest to run, followed by the Quadrithmic and QuadraticWithCalipers algorithms.

Conclusion

The ThreeSumQuadratic algorithm has a time complexity of O(N^2), which means that as the input size (N) increases, the number of operations required to solve the problem also increases exponentially. This makes it more efficient for larger values of N compared to the ThreeSumCubic algorithm which has a time complexity of O(N^3) and thus takes significantly more operations to solve the problem.

Additionally, the ThreeSumQuadratic algorithm involves sorting the input array first and then using nested loops to check all pairs of elements in the array. This approach allows for faster elimination of elements that do not meet the criteria for a valid solution and thus reduces the overall number of operations required.

In contrast, the ThreeSumCubic algorithm uses three nested loops to check all possible combinations of elements in the array, which leads to a large number of operations even for small values of N. Thus, the ThreeSumQuadratic algorithm is generally considered to be more efficient for larger values of N.