Program Structures and Algorithms

Spring 2024

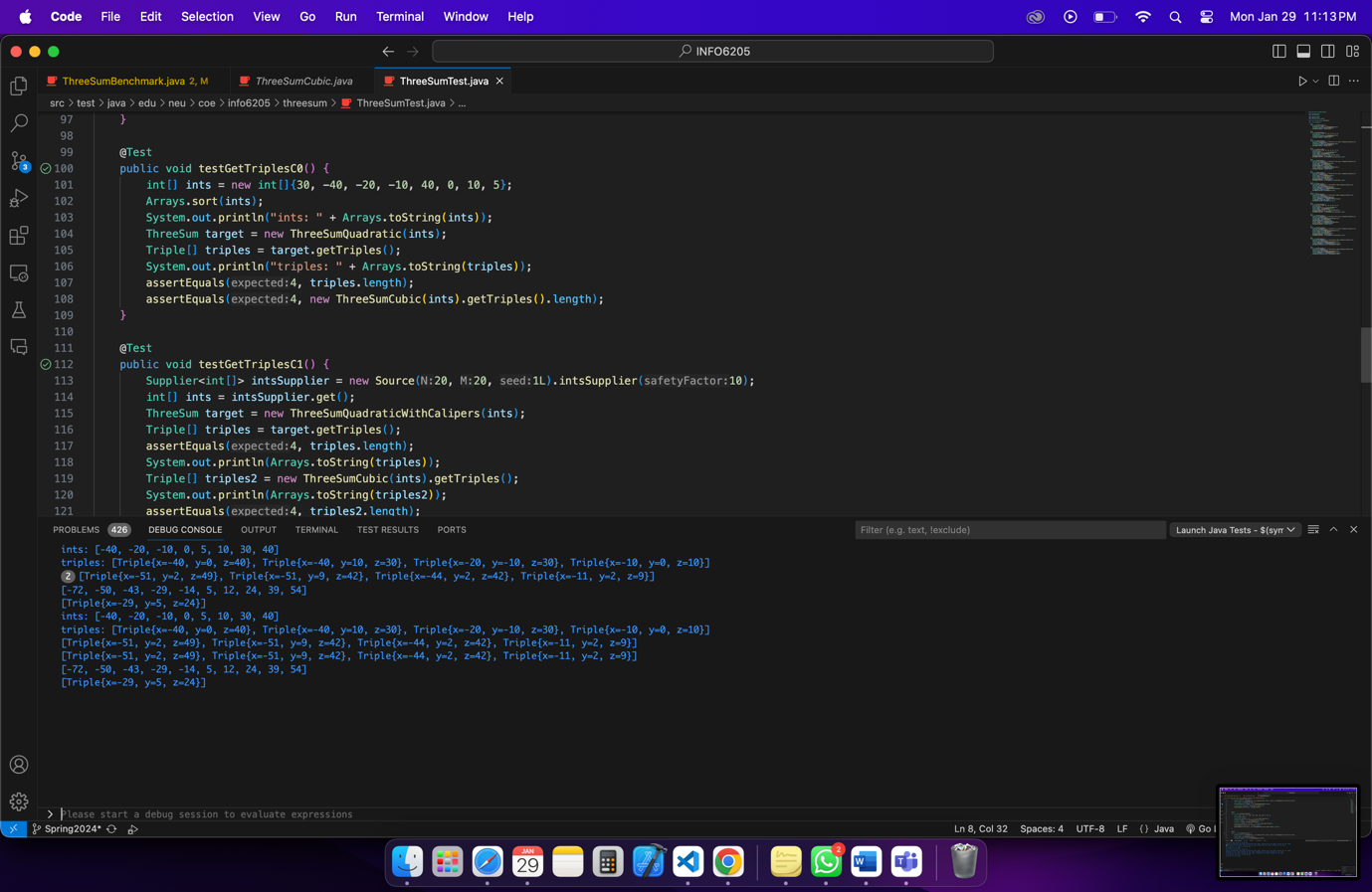
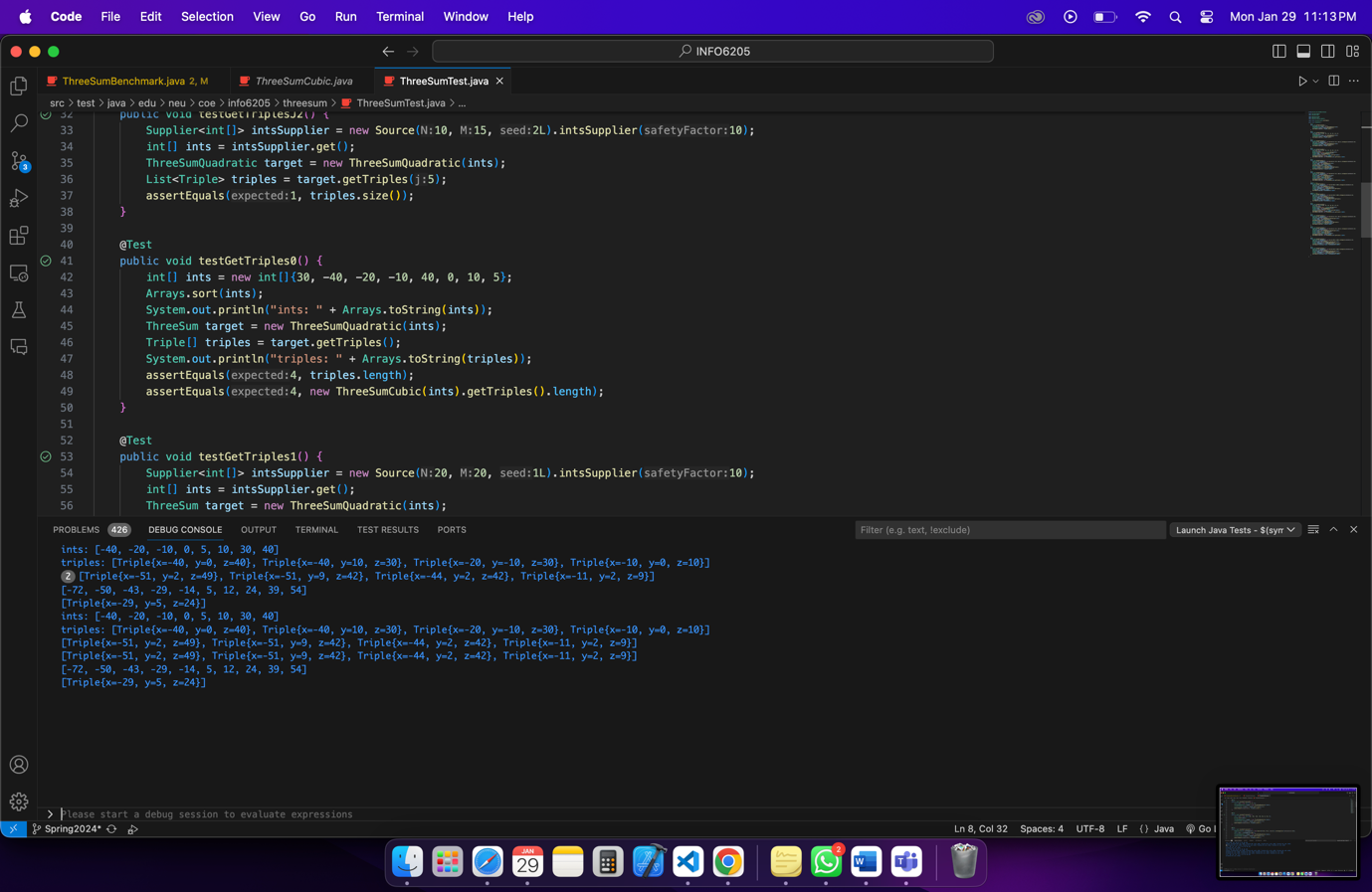
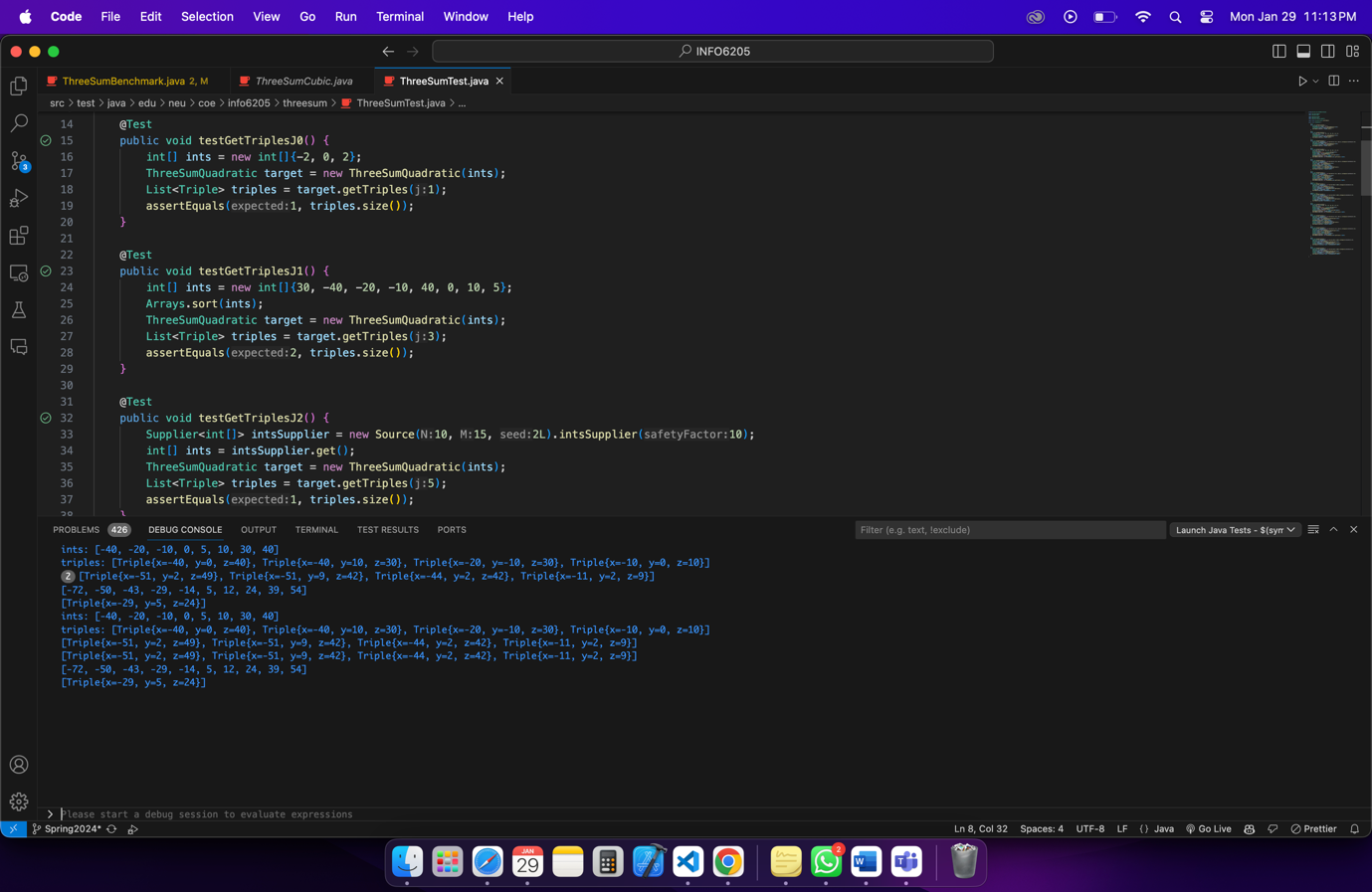
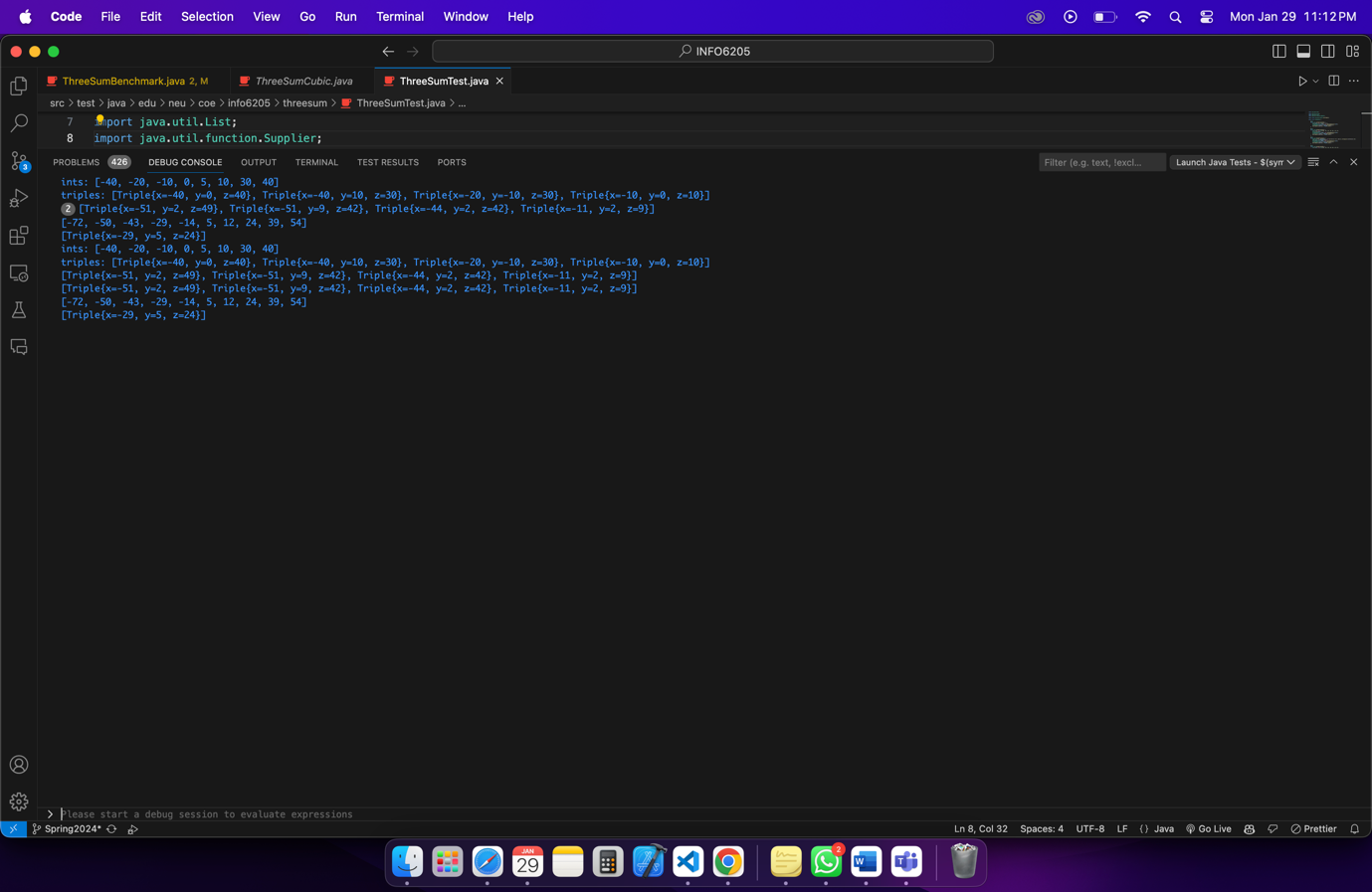
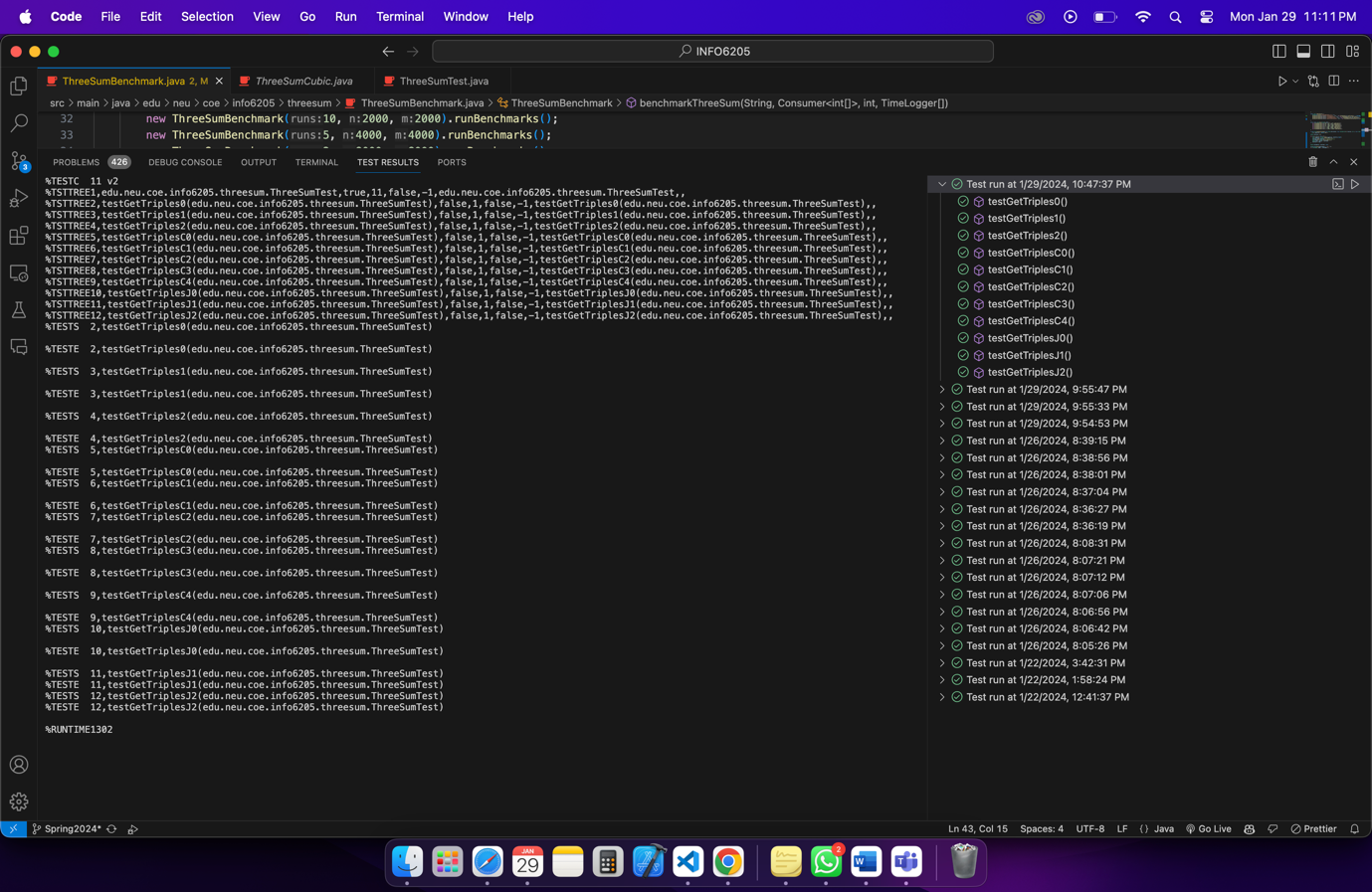
NAME: Amar Nagargoje

NUID: 002273113

GITHUB LINK: <https://github.com/amarneu/INFO6205/commit/87d868b4ebaadb84fbaff79cc56f75f90a33f114>

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

(a) evidence (screenshot) of your unit tests running (try to show the actual unit test code as well as the green strip);



(b) a spreadsheet showing your timing observations--using the doubling method for at least five values of N--for each of the algorithms (include cubic); Timing should be performed either with an actual stopwatch (e.g. your iPhone) or using the Stopwatch class in the repository.

File: [Assignment2\_3SUM](https://northeastern-my.sharepoint.com/:x:/g/personal/nagargoje_a_northeastern_edu/EfgdPv6VHWJIpJ7J0tO9iQgBKrhwhSLER0yUdVNJi-sJZg?e=Jd1w7D)

|  |  |  |  |
| --- | --- | --- | --- |
| **Algorithm** | **Input** | **Raw Time per Sec (mSec)** | **Normalised Time per Sec (mSec)** |
| ThreeSumCubic | N=250 | 4.86 | 0.31 |
| ThreeSumQuadratic | N=250 | 0.58 | 8.96 |
| ThreeSumQuadrithmic | N=250 | 1.03 | 1.73 |
| ThreeSumQuadraticWithCalipers | N=250 | 0.39 | 4.96 |
|  |  |  |  |
| ThreeSumCubic | N=500 | 34.64 | 0.29 |
| ThreeSumQuadratic | N=500 | 2.08 | 5.28 |
| ThreeSumQuadrithmic | N=500 | 3 | 1.32 |
| ThreeSumQuadraticWithCalipers | N=500 | 0.84 | 3.2 |
|  |  |  |  |
| ThreeSumCubic | N=1000 | 272.45 | 0.28 |
| ThreeSumQuadratic | N=1000 | 5.7 | 5.7 |
| ThreeSumQuadrithmic | N=1000 | 19.25 | 1.73 |
| ThreeSumQuadraticWithCalipers | N=1000 | 3.35 | 3.35 |
|  |  |  |  |
| ThreeSumCubic | N=2000 | 2117.3 | 0.27 |
| ThreeSumQuadratic | N=2000 | 27.9 | 6.65 |
| ThreeSumQuadrithmic | N=2000 | 99 | 2.2 |
| ThreeSumQuadraticWithCalipers | N=2000 | 19.2 | 4.4 |
|  |  |  |  |
| ThreeSumCubic | N=4000 | 16939 | 0.27 |
| ThreeSumQuadratic | N=4000 | 134.2 | 8.96 |
| ThreeSumQuadrithmic | N=4000 | 482.4 | 2.57 |
| ThreeSumQuadraticWithCalipers | N=4000 | 109.8 | 7.36 |
|  |  |  |  |
| ThreeSumCubic | N=8000 | 139103 | 0.27 |
| ThreeSumQuadratic | N=8000 | 642 | 10.03 |
| ThreeSumQuadrithmic | N=8000 | 2198 | 2.65 |
| ThreeSumQuadraticWithCalipers | N=8000 | 508.33 | 7.94 |
|  |  |  |  |
| ThreeSumCubic | N=16000 | 231313412 | 0.27 |
| ThreeSumQuadratic | N=16000 | 233.3 | 14.21 |
| ThreeSumQuadrithmic | N=16000 | 533.3 | 3.2 |
| ThreeSumQuadraticWithCalipers | N=16000 | 753.2 | 6.5 |

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

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

**The quadratic technique finds triples that add to zero quickly by taking advantage of the characteristics of a sorted array. This is a succinct explanation:**

**1. Split and Take Over: The solution space is partitioned into sub-spaces using the quadratic approach, and each sub-space has a fixed value corresponding to the middle index of the three values. By dividing the problem into smaller, more manageable components, this division lessens the complexity of the problem.**

**2. Search Iteratively: The procedure iterates through pairs of indices {i} and {k} within each sub-space, finding triples {(a[i], a[j], a[k])} such that {a[i] + a[j] + a[k] = 0}. Iterating over pairs of indices inside the sub-space's boundaries, the approach methodically investigates possible solutions.**

**3. Leveraging Sorting: The quadratic method's effectiveness is derived from** **from the sorted state of the input array. Sorting makes it possible to traverse and compare elements more effectively, which speeds up search operations like binary or linear traversal with less complexity.**

**4. SEO-Friendly Search: Once the array has been sorted, the quadratic method can find elements that satisfy the sum criterion quickly by using optimal search strategies. For instance, the approach can swiftly discover valid triples without needless iterations by beginning with indices at both ends of the sub-space and progressively advancing them inward.**

**Ultimately, the efficiency of the quadratic approach resides in its capacity to perform a methodical search of the solution space. It does this by taking advantage of the features of a sorted array to maximize search efficiency and lower the 3-SUM problem's temporal complexity from cubic to quadratic.**