Program Structures and Algorithms
Spring 2023(SEC –3)

NAME: Akshay Parab
NUID: 002766150

**Task:** Solve 3-SUM using the Quadrithmic, Quadratic, and quadraticWithCalipers approaches, as shown in skeleton code in the repository

**Relationship Conclusion:**

*Quadratic Solution Working:*

This approach for solving the 3-sum problem divides the problem space into N sub-spaces, where each sub-space corresponds to a fixed value for the middle index of the three values. This means that for each value in the array, it is considered as the middle value of the three values whose sum is zero.

To find the other two values, the approach uses two pointers, one starting from the left side of the middle value, and one starting from the right side of the middle value. These pointers then move outwards, checking if the sum of the values at the current indices is equal to zero. If the sum is zero, then a triplet is found, and the pointers continue moving outwards.

Since each sub-space can be solved in O(N) time, the overall complexity of the algorithm is O(N^2). It is important to note that the array provided in the constructor MUST be ordered for this approach to work correctly. This is because the algorithm relies on the array being in sorted order to determine which direction to move the pointers.

*QuadraticWithCalipers Solution Working:*

QuadraticWithCaliper approach for solving the ThreeSum problem is a modification of Quadratic approach explained above.

For each sub-space, the algorithm expands the scope of the other two indices by setting the second index to the immediate next of the first index and the third index to the last index of the array. This means that for each value of the first index, the algorithm will check all possible combinations of the second and third index that can be formed by using the immediate next index of the first index and the last index of the array. Similar to the Quadratic Solution since each sub-space can be solved in O(N) time, the overall complexity of the algorithm is O(N^2).

**Evidence to support that conclusion:**

**Graphical Representation:**

**Timing Observations**

*Note: All time mentioned below are in milliseconds*

| Sr. No. | Input Size(n) | Iterations | QuadraticWithCalipers | Quadratic | Quadrithmic | Cubic |
|---|---|---|---|---|---|---|
| 1 | 250 | 100 | 0.55 | 0.66 | 0.86 | 4.69 |
| 2 | 500 | 50 | 0.92 | 1.26 | 2.58 | 25.14 |
| 3 | 1000 | 20 | 3.2 | 4.55 | 14.4 | 199.5 |
| 4 | 2000 | 10 | 9.6 | 14.3 | 59.9 | 1590.0 |
| 5 | 4000 | 5 | 81.6 | 79.2 | 319.2 | 13400.4 |
| 6 | 8000 | 3 | 366.667 | 419.0 | 1491.0 | - |
| 7 | 16000 | 2 | 1476.5 | 2251.5 | 6248.0 | - |

## Unit Test Screenshots: