Program Structures and Algorithms
Spring 2023(SEC –3)

NAME: Kaushik Gnanasekar
NUID: 002766012

Task: Assignment 2 (3-SUM): Solve 3-SUM using the Quadrithmic, Quadratic, and quadraticWithCalipers approaches

Relationship Conclusion:

Quadratic Algorithm Explanation:
This implementation of the Three Sum Problem follows the approach of dividing the solution space into n subspaces where each subspace corresponds to a fixed value for the middle index of the three values. Each subspace is solved using pointers to expand the scope. A pointer to expand the scope from 0 index to the mid and another pointer to expand the scope from the end of the space to mid. Since each subspace can be solved in O(n) time, the overall complexity is O(n^2). Also the array should be sorted.

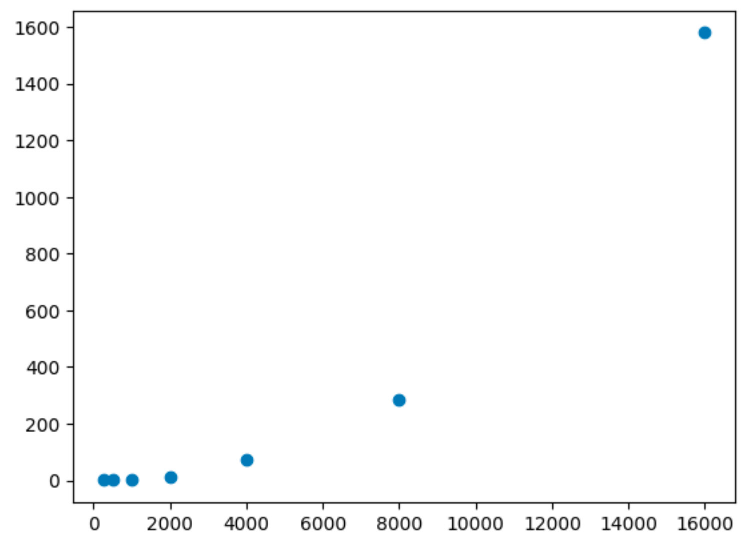
Quadratic(Calipers) Algorithm Explanation:
In this implementation of the problem, we sort the array and divide the solution space into n subspaces. We scope the subspace using two pointers, and expand the space by moving the pointers through the indices. Since each subspace can be solved in O(n), the overall complexity is O(n^2).

Evidence to support that conclusion:

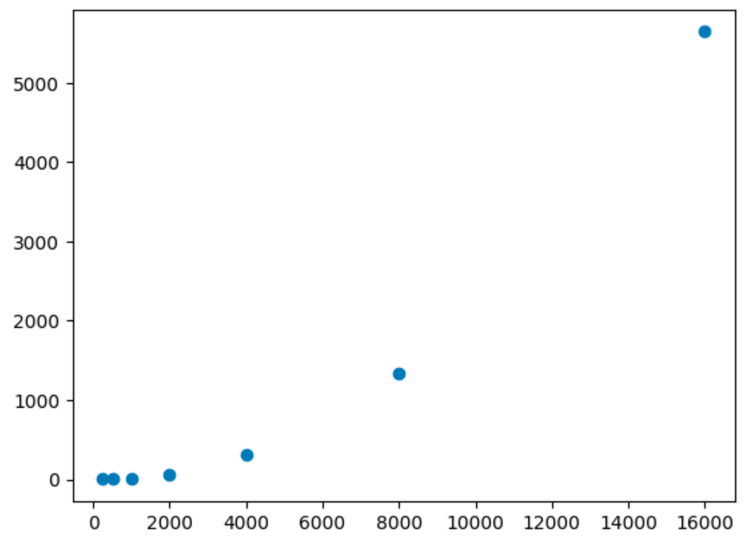
N		Quadratic (ms)	Quadratic-Calipers (ms)	Quadrithmic (ms)	Cubic (ms)
250	Raw Time	0.62	0.49	0.72	3.11
	Normalized	9.92	7.84	1.45	0.2
500	Raw Time	1.02	1.26	2.62	23.66
	Normalized	4.08	5.04	1.17	0.19
1000	Raw Time	2.9	3.75	12.35	182.4
	Normalized	2.9	3.75	1.24	0.18
2000	Raw Time	10.3	15.1	62	1484.7
	Normalized	2.57	3.77	1.41	0.19
4000	Raw Time	74.2	82.4	315.8	11157.6
	Normalized	4.64	5.15	1.65	0.17
8000	Raw Time	285.33	418	1334.33	
	Normalized	4.46	6.53	1.61	
16000	Raw Time	1579	1751.5	5643.5	
	Normalized	6.17	6.84	1.58	

Graphical Representation:

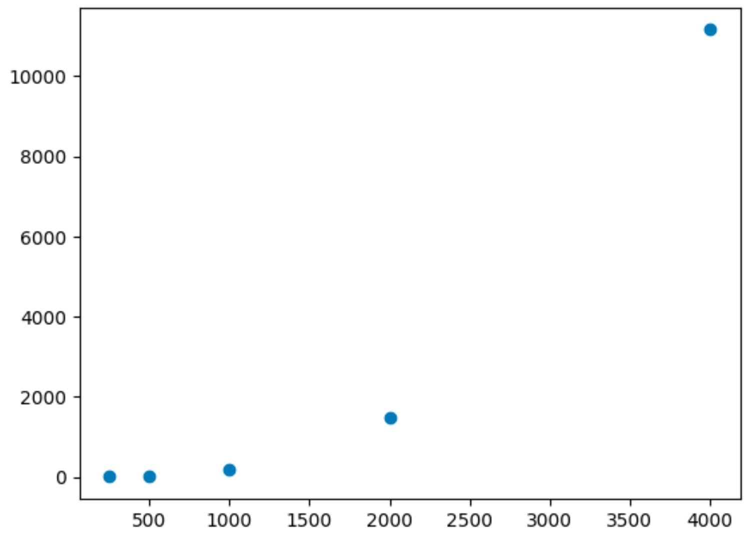
1. **Quadratic (n^2)**



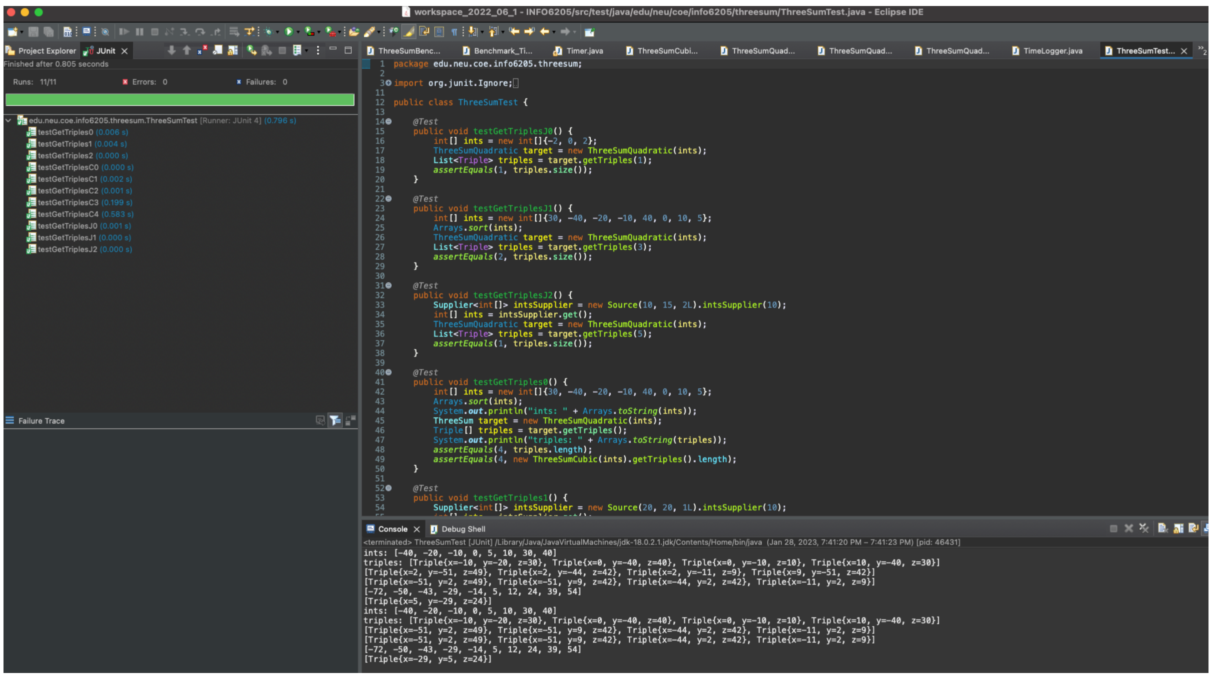
2. **Quadrithmic ($n^2 \log n$)**



3. Cubic (n^3)



Unit Test Screenshots:



workspace_2022_06_1 - INFO6205/src/test/java/edu/neu/coe/info6205/threesum/ThreeSumTest.java - Eclipse IDE

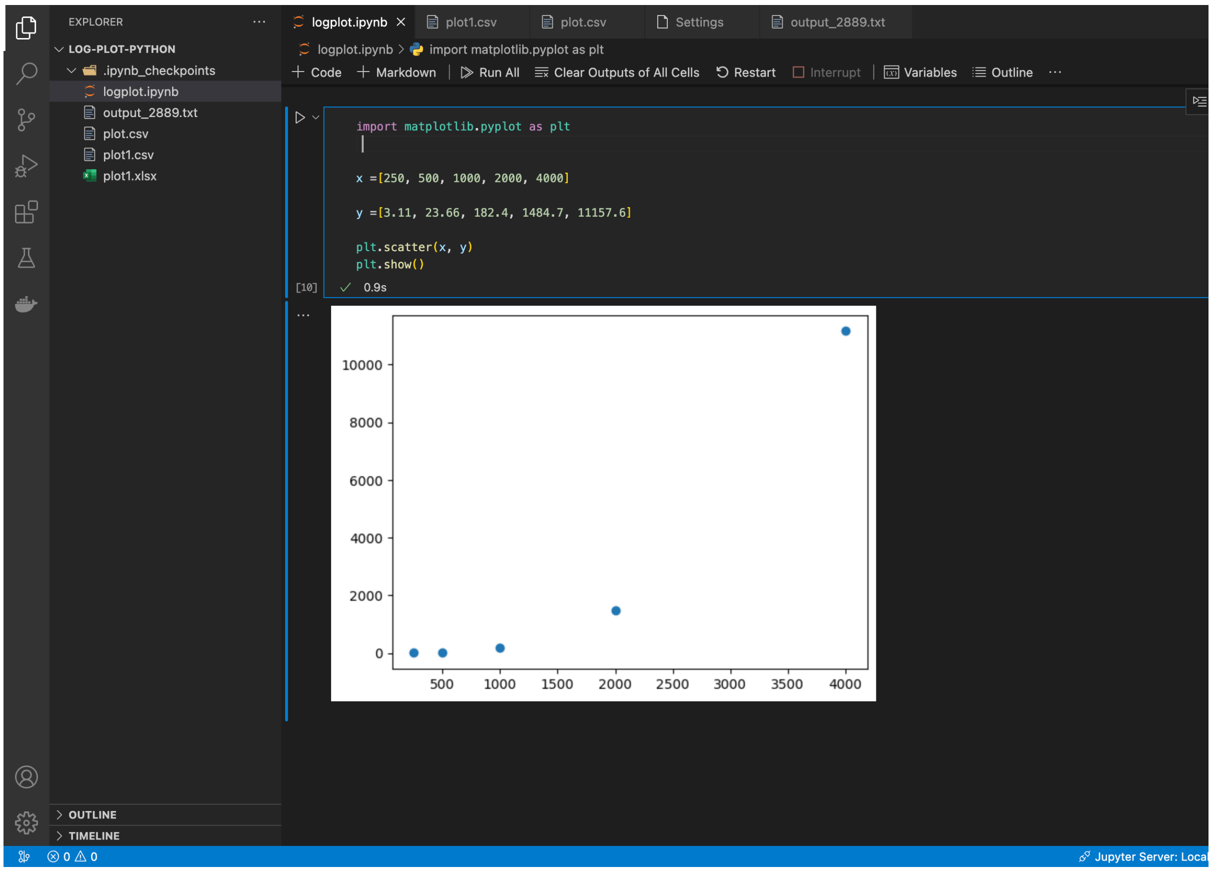
Finished after 0.805 seconds

Runs: 11/11 Errors: 0 Failures: 0

```
edu.neu.coe.info6205.threesum.ThreeSumTest [Runner: JUnit 4] (0.796 s)
    testGetTriples0 (0.006 s)
    testGetTriples1 (0.004 s)
    testGetTriples2 (0.000 s)
    testGetTriplesC0 (0.000 s)
    testGetTriplesC1 (0.002 s)
    testGetTriplesC2 (0.001 s)
    testGetTriplesC3 (0.199 s)
    testGetTriplesC4 (0.583 s)
    testGetTriplesJ0 (0.001 s)
    testGetTriplesJ1 (0.000 s)
    testGetTriplesJ2 (0.000 s)
```

```java
package edu.neu.coe.info6205.threesum;

import org.junit.Ignore;

public class ThreeSumTest {

    @Test
    public void testGetTriplesJ0() {
        int[] ints = new int[]{-2, 0, 2};
        ThreeSumQuadratic target = new ThreeSumQuadratic(ints);
        List<Triple> triples = target.getTriples(1);
        assertEquals(1, triples.size());
    }

    @Test
    public void testGetTriplesJ1() {
        int[] ints = new int[]{30, -40, -20, -10, 40, 0, 10, 5};
        Arrays.sort(ints);
        ThreeSumQuadratic target = new ThreeSumQuadratic(ints);
        List<Triple> triples = target.getTriples(3);
        assertEquals(2, triples.size());
    }

    @Test
    public void testGetTriplesJ2() {
        Supplier<int[]> intsSupplier = new Source(10, 15, 2L).intsSupplier(10);
        int[] ints = intsSupplier.get();
        ThreeSumQuadratic target = new ThreeSumQuadratic(ints);
        List<Triple> triples = target.getTriples(5);
        assertEquals(1, triples.size());
    }

    @Test
    public void testGetTriples0() {
        int[] ints = new int[]{30, -40, -20, -10, 40, 0, 10, 5};
        Arrays.sort(ints);
        System.out.println("ints: " + Arrays.toString(ints));
        ThreeSum target = new ThreeSumQuadratic(ints);
        Triple[] triples = target.getTriples();
        System.out.println("triples: " + Arrays.toString(triples));
        assertEquals(4, triples.length);
        assertEquals(4, new ThreeSumCubic(ints).getTriples().length);
    }

    @Test
    public void testGetTriples1() {
        Supplier<int[]> intsSupplier = new Source(20, 20, 1L).intsSupplier(10);
```

```
<terminated> ThreeSumTest [JUnit] /Library/Java/JavaVirtualMachines/jdk-18.0.2.1.jdk/Contents/Home/bin/java  (Jan 28, 2023, 7:41:20 PM – 7:41:23 PM) [pid: 46431]
ints: [-40, -20, -10, 0, 5, 10, 30, 40]
triples: [Triple{x=-10, y=-20, z=30}, Triple{x=0, y=-40, z=40}, Triple{x=0, y=-10, z=10}, Triple{x=10, y=-40, z=30}]
[Triple{x=2, y=-51, z=49}, Triple{x=2, y=-44, z=42}, Triple{x=2, y=-11, z=9}, Triple{x=9, y=-51, z=42}]
[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}]
[-72, -50, -43, -29, -14, 5, 12, 24, 39, 54]
[Triple{x=5, y=-29, z=24}]
ints: [-40, -20, -10, 0, 5, 10, 30, 40]
triples: [Triple{x=-10, y=-20, z=30}, Triple{x=0, y=-40, z=40}, Triple{x=0, y=-10, z=10}, Triple{x=10, y=-40, z=30}]
[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}]
[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}]
[-72, -50, -43, -29, -14, 5, 12, 24, 39, 54]
[Triple{x=-29, y=5, z=24}]
```



```
import matplotlib.pyplot as plt


x =[250, 500, 1000, 2000, 4000]

y =[3.11, 23.66, 182.4, 1484.7, 11157.6]

plt.scatter(x, y)
plt.show()
```

[10] ✓ 0.9s