

## PSA INFO6205 Assignment 1A

### AIM:

This is a mini-assignment for those of you who are keen to do some extra work. It won't be graded, (like assignment 0), but I think it's worth your time to think about it. Post your code and unit test runs as a zip file before the deadline.

Your task is to implement *ThreeSumQuadratic* (as discussed in Lesson 2.4).

### CODE:

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

import java.util.*;

/**
 * Implementation of ThreeSum which follows the approach of dividing the
 * solution-space into
 *   * N sub-spaces where each sub-space corresponds to a fixed value for the
 *     middle index of the three values.
 *   * Each sub-space is then solved by expanding the scope of the other two
 *     indices outwards from the starting point.
 *   * Since each sub-space can be solved in O(N) time, the overall complexity is
 *     O(N^2).
 *   * <p>
 *   * NOTE: The array provided in the constructor MUST be ordered.
 *   */
public class ThreeSumQuadratic implements ThreeSum {

    private final int[] a;
    private final int length;

    /**
     * Construct a ThreeSumQuadratic on a.
     *
     * @param a a sorted array.
     */
    public ThreeSumQuadratic(int[] a) {
        this.a = a;
        length = a.length;
    }

    public Triple[] getTriples() {
        List<Triple> triples = new ArrayList<>();
        for (int i = 0; i < length; i++) triples.addAll(getTriples(i));
        Collections.sort(triples);
        return triples.stream().distinct().toArray(Triple[]::new);
    }
}
```

```

/**
 * Get a list of Triples such that the middle index is the given value j.
 *
 * @param j the index of the middle value.
 * @return a Triple such that
 */
public List<Triple> getTriples(int j) {
    List<Triple> triples = new ArrayList<>();
    // Arrays.sort(a);
    // System.out.println(Arrays.toString(a));
    // System.out.println();
    // int cnt = 0;
    // int n = a.length;
    // HashMap<Integer, Integer> map = new HashMap<Integer, Integer>();
    // for (int i = 0; i < n; i++) {
    //     map.put(a[i], 1); //Dvaluekey, index/map fi
    // }
    // for (int i = 0; i < n - 1; i++) {
    //     for (int x = i + 1; x < n; x++) {
    //         int smallValue = a[i] + a[x];
    //         if (smallValue > length) //a[i]+a[j]>target7, BEERSRE
    //             break;
    //         int bigvalue = length - smallValue;
    //         Integer bigIndex = map.get(bigvalue);
    //         if (bigIndex != null && bigIndex > i && bigIndex > x) {
    //             // END
    //         }
    //     }
    // }
    // return triples;
    // }

    int low = 0, high = length - 1;
    while (low < j && high > j) {
        int sum = this.a[j] + this.a[low] + this.a[high];
        if (sum > 0) {
            high--;
        } else if (sum < 0) {
            low++;
        } else {
            triples.add(new Triple(this.a[j], this.a[low],
this.a[high]));
            low++;
            high--;
        }
    }
    return triples;
}
}

```

## UNIT TESTS:

The screenshot shows an IDE with the following components:

- Project Explorer:** Shows a project structure with folders like 'lab\_1', 'life', 'pq', 'randomwalk', 'runLengthEncoding', 'reduction', 'sort', 'symbolTable', 'threesum', 'union\_find', 'util', 'BinarySearch', 'CallByValue', 'ComparableTuple', 'Counter', and 'HuffmanCoding'. The 'threesum' folder is expanded, showing 'Source', 'ThreeSum', 'ThreeSumBenchmark', 'ThreeSumCubic', 'ThreeSumQuadratic', 'ThreeSumQuadraticWithCalipers', and 'Triple'.
- Code Editor:** Displays the 'ThreeSumTest.java' file. The code includes imports for 'Arrays', 'ThreeSumQuadraticWithCalipers', 'ThreeSumCubic', 'ThreeSumQuadratic', and 'Triple'. It defines a 'target' of 100 and tests the 'getTriples()' method. The test method 'testGetTriples()' is annotated with '@Ignore' and contains a loop that generates random integers and tests the 'getTriples()' method. The output shows the input array [-40, -20, -10, 0, 5, 10, 30, 40] and the expected output of 6 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}].
- Run Console:** Shows the test results. The test 'testGetTriples0' failed, while 'testGetTriples1' through 'testGetTriples12' passed. The output shows the input array [-40, -20, -10, 0, 5, 10, 30, 40] and the expected output of 6 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}].
- Bottom Bar:** Shows the status bar with 'Tests failed: 2, passed: 8 (moments ago)', '27 LF UTF-8 4 spaces', and '12:54 PM 2/3/2022'.