CSCI 421 Design and Analysis of Algorithms Spring 2019

Lecture 1 Activity 1

**Faster 3-Sum**

1. (50 points) As discussed in class, implement a comparison-based faster 3-Sum algorithm with running time O(n2). Run the program against the following input array (assuming it is already sorted). Attach your code and screenshots of the output here.

-25 -10 -7 -3 2 4 8 10

public class ThreeSum {

public static int count(int[] a){

int N = a.length;

int count = 0;

for (int i = 0; i < N; i++)

for (int j = i+1; j < N; j++)

for (int k = j+1; k < N; k++)

if (a[i] + a[j] + a[k] == 0){

System.out.println("Triplet Found: " + a[i] + " " + a[j] + " " + a[k] );

count++;

}

return count;

}

public static void main(String[] args){

int arr1[] = {-25, -10, -7, -3, 2, 4, 8, 10};

System.out.println("For the array: ");

for(int i = 0; i < arr1.length; i++){

System.out.print(arr1[i] + " ");

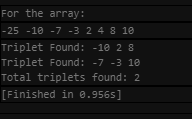
}

System.out.println();

System.out.println("Total triplets found: " + count(arr1));

}

}



1. (50 points) As discussed in class, implement a hash table based faster 3-Sum algorithm with running time O(n2). Run the program against the following input array (note: it does not have to be sorted). Attach your code and screenshots of the output here. Hint: put the array values into a hash table.

-10 -7 -25 2 -3 4 10 8