**Question 1)**

The algorithm is in *zehra\_mogulkoc\_hw3.java* folder. With checkSum function, every element in the array is scanned and with the binary search, the number is searched that makes the correct sum. Run time for first loop is n, and runtime of binary search is logn. Therefore, total running time complexity is Ѳ (nlogn). If the given array is not sorted, merge sort can be used before checkSum which has also nlogn time comlexty. The total result would not change.

**Question 2)** Induction: T(n) = nlogn

Base Case:

When n=2, it should hold.

From function: T(2)=2

2log2=2

Thus, T(2)=2log2

Induction:

Assume, for n=2k, k>1 it is true; show n=2k+1 holds

T(2k) = 2k log2k  is true

T(2k+1) = 2 \* T(2k+1/2) + 2k+1 By substitution,

= 2 \* T(2k) + 2k+1 By simplification,

= 2 \* (2k log2k  ) + 2k+1 By substitution of T(2k)= 2k log2k

= 2k+1(log2k +1) By factoring,

= 2k+1(log2k +log2) Since log2=1

= 2k+1 log2k+1 Since (log2k +log2) = log2k+1

Thus, T(2k+1) = 2k+1 log2k+1  is true

**Question 3)** Horner’s Rule

1. Because of the for loop from n to 0:

Ѳ(n)

1. Naive polynomial-evaluation:

NaivePoly(A,x):

y=0

**for** i=1 **to** A.length

p=1

**for** j=1 **to** i-1

p=p\*x

y = y+A[i]\*p

The runtime complexity is Ѳ(n2)

Horner’s rule is faster than Naïve algorithm.

1. Loop Invariant:

**Initialization:**

Before first loop iteration, there is no element to sum, therefore the sum y is 0. Which holds for algorithm.

**Maintenance:**

At the start of i-th iteration of the for loop, 0< i <n

Assume y=ai+x is correct.

After i-th iteration, i becomes i=i-1.

y’ = ai+x\*y

= ai+x

= ai+

= aix0+ ai+1x1+ ai+2x2+…+anxn-i

=ak+i xk

**Termination:**

The loop end when i=-1. Substitute i in equation:

Y=ak+i+1 xk

=akxk

Which is the equation that we are expecting.

1. According to the proof of loop invariant, Horner’s rule can calculate the polynomial correctly. With the given coefficients, it is a sum that equals a polynomial. Therefore, code fragment is correct for evaluating a polynomial.