ECGR 3090-R01 - Assignment 3

1. For each of the following 6 program fragments, give a Big-Oh analysis of the running time -

(1)

```
sum = 0;
for(i = 0; i < n; ++i)
        ++sum;
```

$O(n)$

(2)

```
sum = 0;
for(i = 0; i < n; ++i)
        for(j = 0; j < n; j++)
                ++sum;
```

$O(n^2)$

(3)

```
sum = 0;
for(i = 0; i < n; ++i)
        for(j = 0; j < n*n; ++j)
                ++sum;
```

$O(n^3)$

(4)

```
sum = 0;
for(i = 0; i < n; ++i)
        for(j = 0; j < i; ++j)
                ++sum;
```

$O(n^2)$

(5)

```
sum = 0;
for(i = 0; i < n; ++i)            n
        for(j = 0; j < i*i; ++j)      n^2
                for(k = 0; k < j; ++k)    n
                        ++sum;
```

$n\left(\frac{(n-1)^2}{2^2}\right)$

$O(n^4)$

(6)

```
sum = 0;
for(i = 1; i < n; ++i)    n  n^2
        for(j = 1; j < i*i; ++j)
```

$\left(\frac{(n-1)}{2}\right)^2$

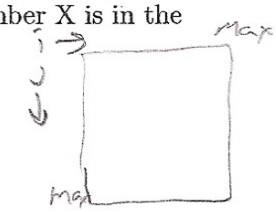
$O(n^4)$

```
if (j %i == 0) n
        for(k = 0; i < j; ++k) n
                ++sum;
```

2. Programs A and B are analyzed and found to have worst-case running times no greater than $150Nlog_2N$ and N^2, respectively. Answer the following questions -
a. Which program has the better guarantee on the running time for large values of N (N > 10,000)? Algorithm 1 ($150N\log_2 N$)
b. Which program has the better guarantee on the running time for small values of N (N < 100)? Algorithm 2 (N^2)
c. Which program will run faster on average for N = 1000?
Algorithm 2 (N^2)

3. Solve the following recurrence relations using the Master theorem -
a. $T(n) = 3T(\frac{n}{c}) + \frac{n}{2}$
b. $T(n) = 4T(\frac{n}{2}) + n^{2.5}$

4. The input is an N by N matrix of numbers that is already in memory. Each individual row is increasing from left to right. Each individual column is increasing from top to bottom. Give an *O(N)* worst-case algorithm that decides if a number X is in the matrix.




```
int i=0, j=N-1
while (i ≠ N-1 and j ≠ 0) {
    if (matrix[i][j] = ele) return true
    if (matrix[i][j] < ele) i++
    else j--
}
return false
```