Worksheet: Complexity Analysis

1. For each of the following code fragments, derive the Θ time complexity (tight bound) as a function of N. Remember, this means it must be big O and Ω of the function, bounded above and below for the worst-case scenario. You must show all work, including approximate operation counts. For partial credit, give the best O and Ω bounds you can figure out, even if they are not tight.

a)

```
sum = 0;
for ( i = 0; i < 2*N; i+=4)
    sum += i;
```

b)

```
sum = 0;
for ( i = 0; i < 3*N; i++)
    for ( j = 0; j < 5; j++)
        for ( k = i; k > 1; k--)
            sum++;
```

Consider reversing b and c order since b is mathematically more complex.

c)

```
sum = 0;
s = keyboard.nextLine(); // keyboard is an initialized Scanner
N = s.length();
v = "aeiou";
for ( i=0; i < v.length(); i++)
    if ( s.indexOf(v.charAt(i)) >= 0)
        sum++;
```

2. For each of the functions listed below, figure out its growth rate (simplify first!). Then rank the functions in growth rate order, starting with the slowest growth rate (ie, those resulting in a fast runtime), and ending with the fastest growth rate (worst runtime). If two functions have the same asymptotic complexity, then rank them based on the original expressions (including constants).

Function	big- Θ()	rank
$(\log N/4)^3$		
$(N^2 - 4) / (N + 2)$		
$(3N + \log N)^2$		
$2 \log N^2$		
$(2^N)^2 + \log N$		
$N^2 \log 16$		
$2 N \log \log N$		
$4 N^2 + N(10 + \log N)$		

New Question

2.5

For each of the asymptotic complexities for run time, come up with an example that would create this complexity as a function of an input of size "N."

Asymptotic Complexity	Your Example	Lecture Example
Linear time: O(N)		
Constant time: O(1)		
Linearithmic time: O(NlogN)		
Quadratic time: $O(N^2)$		
Sublinear time: O(logN)		

3. Suppose you have a quadratic time algorithm that takes 10ms to run when the input size is 25. How long do you think it will take to run for an input of size 100?

4. How could you raise some number N to the 64th power using only 6 multiplications.

5. Set A has n-1 unique integers from the inclusive range [1,n] - find the missing number.