Prime Factors

Here is a video walkthrough of the solutions.

Determine the best and worst case runtime of prime_factors in $\Theta(.)$ notation as a function of N.

```
int prime_factors(int N) {
    int factor = 2;
    int count = 0;
    while (factor * factor <= N) {
        while (N % factor == 0) {
            System.out.println(factor);
            count += 1;
            N = N / factor;
        }
        factor += 1;
    }
    return count;
}
```

Best Case: $\Theta(\quad)$, Worst Case: $\Theta(\quad)$

Solution:
Best Case: $\Theta(log(N))$, Worst Case: $\Theta(\sqrt{N})$
Explanation: In the best case, N is some power of 2. Then the inner while loop will halve N each time until it becomes 1. At this point, both the inner and outer while loop conditions will be false and the function will return. Halving N each time results in a $\Theta(\log N)$ runtime.
In the worst case, N will not be divisible by any value of factor. This means we increment factor by 1 each time until factor * factor > N. This is at most $\sqrt{N}$ loops.