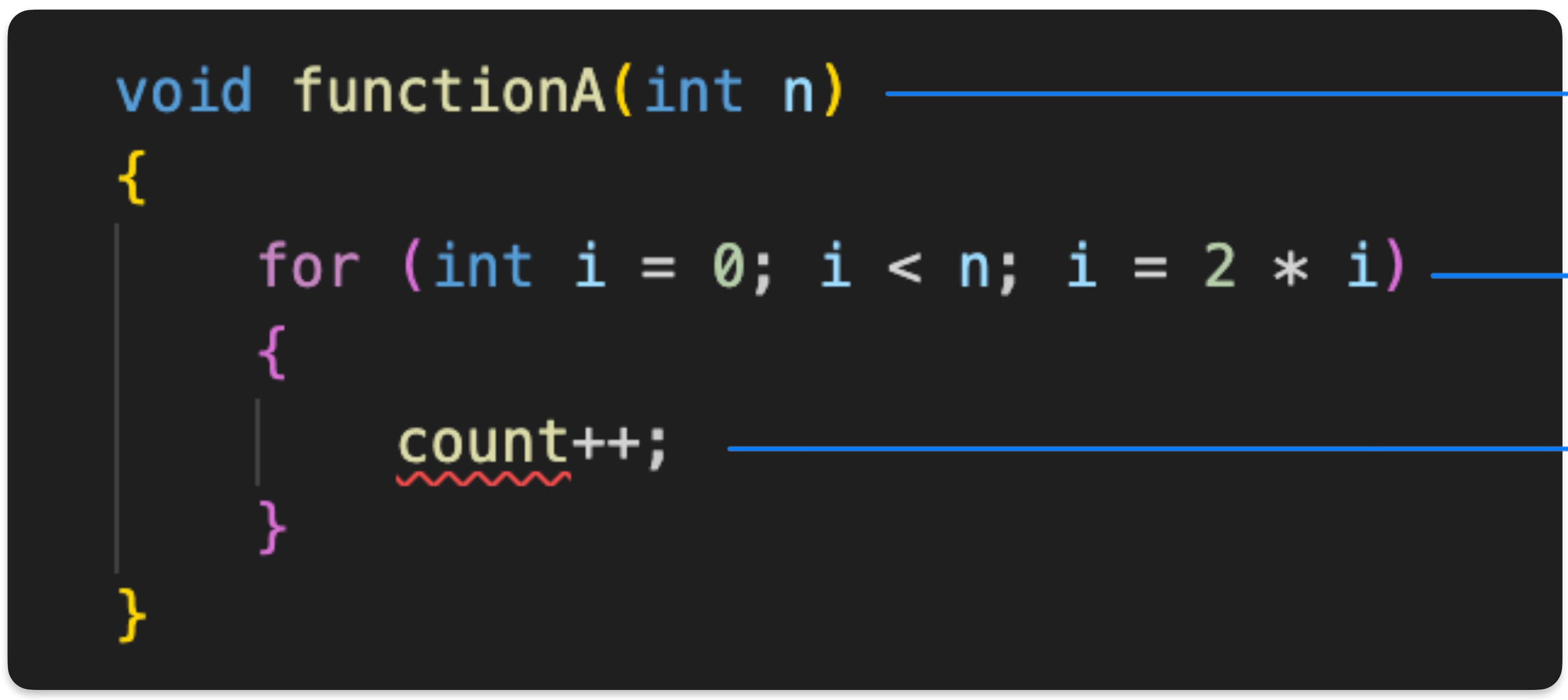
Question 1) Analyze Runtime of the following 2 functions



```
void functionA(int n)
{
    for (int i = 0; i < n; i = 2 * i)
    {
        count++;
    }
}
```

We take the biggest

$$\overbrace{O(1) + O(\log n)} = \boxed{O(\log n)}$$

$$O(\log n) * 1 = O(\log n)$$

$$O(1)$$

Solving Runtime

when to multiply: Loops! OR when cost depends on iterations

when to add: Operations happen sequentially

How to think of log

Chop the workload!

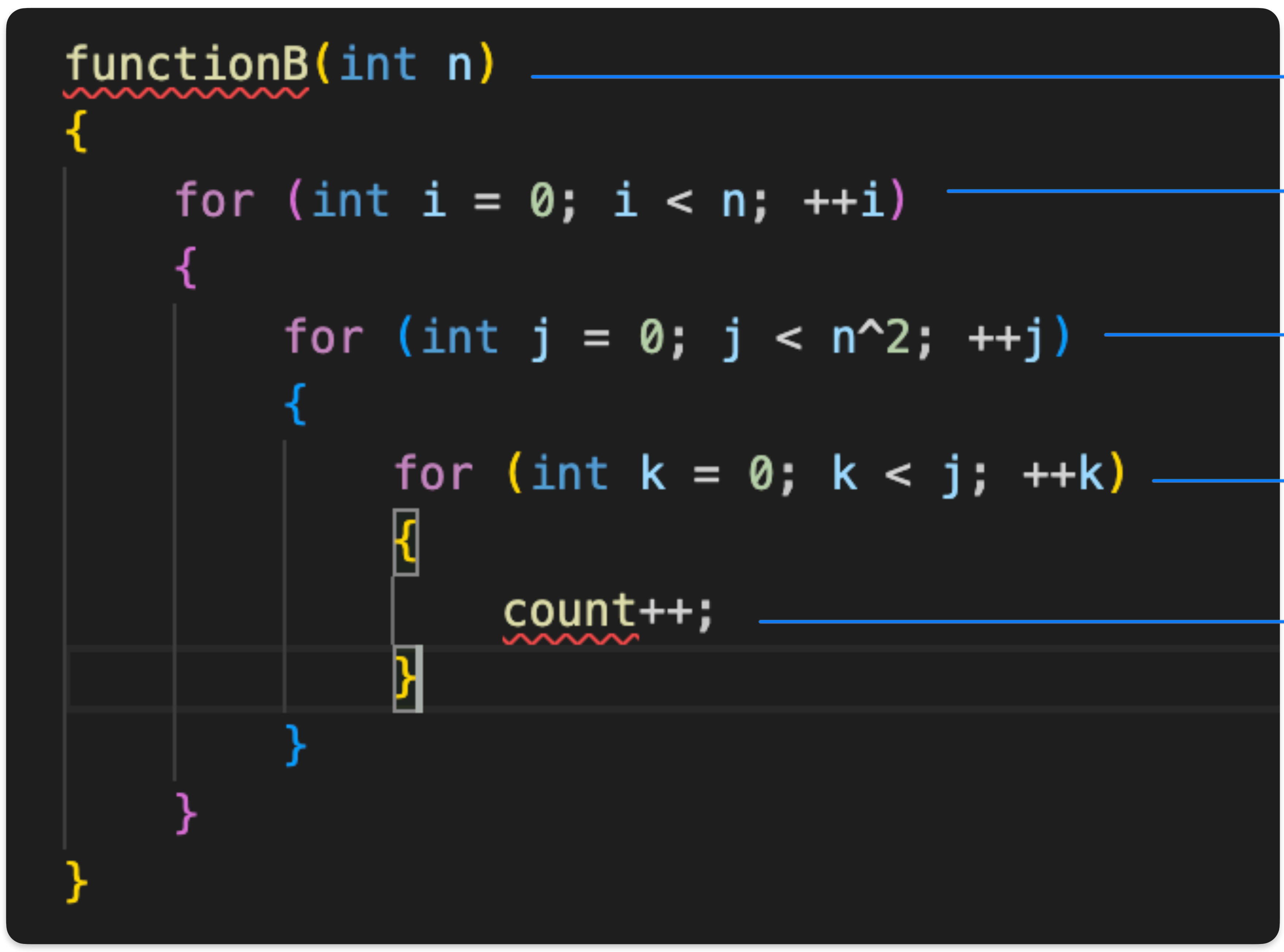
O(n) takes way more steps to finish

O (logn) takes less steps to finish

Ex) List of n items and $n=8$

i<n; ++i $O(n)$: 1, 2, 3, 4, 5, 6, 7, 8

i<n; i=2*i $O(\log n)$: 1, 2, 4, 8



```
functionB(int n)
{
    for (int i = 0; i < n; ++i)
    {
        for (int j = 0; j < n^2; ++j)
        {
            for (int k = 0; k < j; ++k)
            {
                count++;
            }
        }
    }
}
```

$$O(1) + O(n^5) = O(n^5)$$

$$O(n) * n^4 = O(n^5)$$

$$O(n^2) * n^2 = O(n^4)$$

$$O(n^2) * 1 = O(n^2)$$

$$O(1)$$

How to solve runtime for k < j

$k<j$ is also $k<n^2$ because

j can only run up to n^2 iterations