Key Points:

1. **Linear Function (c_1):** - The cost c_1 associated with the outer `for` loop is linear in n because the loop runs $n-1$ times. - This means the cost function for the outer loop is proportional to n, often expressed as $c_1 \cdot (n-1)$.

2. **Varying Inner Loop (Σ):** - For the inner loop, which runs a different number of times depending on the value of j, you can't simply multiply by n as you would with the outer loop. Instead, you need to sum up the costs across all iterations of the inner loop. - Summation notation Σ is used to represent the total cost over all iterations of the inner loop, which reflects the fact that the "window" (or the number of iterations) changes with each step of j.

For example, if the inner loop starts at j and runs until 0, the number of times the inner loop runs is j, and the summation Σ would account for all such j values from 2 to n.

Example in Insertion Sort:

- **Outer Loop (Linear):** - The outer loop runs $n-1$ times, so the cost is linear in n. - This is captured by $c_1 \cdot (n-1)$.

- **Inner Loop (Summation):** - The inner loop runs j times for each j, and you sum over all j values. - The total cost is represented as a summation: $\sum_{j=2}^{n} t_j$, where t_j depends on the specific operations within the inner loop.

Summation Notation:

- **Why Use Σ?** - When analyzing loops that do not run a fixed number of times (like the inner loop in insertion sort), summation notation accurately reflects the total number of operations. - It captures the idea that the inner loop's runtime is dependent on the current state of the outer loop, not just on n.

Summary:

- The cost c_1 for the outer loop is linear because it scales with n. - The inner loop's cost cannot be simply represented as a multiple of n; instead, it requires summation notation to account for the changing number of iterations as j progresses.