1)

- The initialization x = 1 is O(1).
- The nested loops iterate n × n = n^2 times.
- The statement x = x + 1; executes inside the inner loop, contributing O(1) per iteration.

Summation Representation

$$T(n) = 1 + \sum_{i=1}^{n} \sum_{j=1}^{n} 1$$

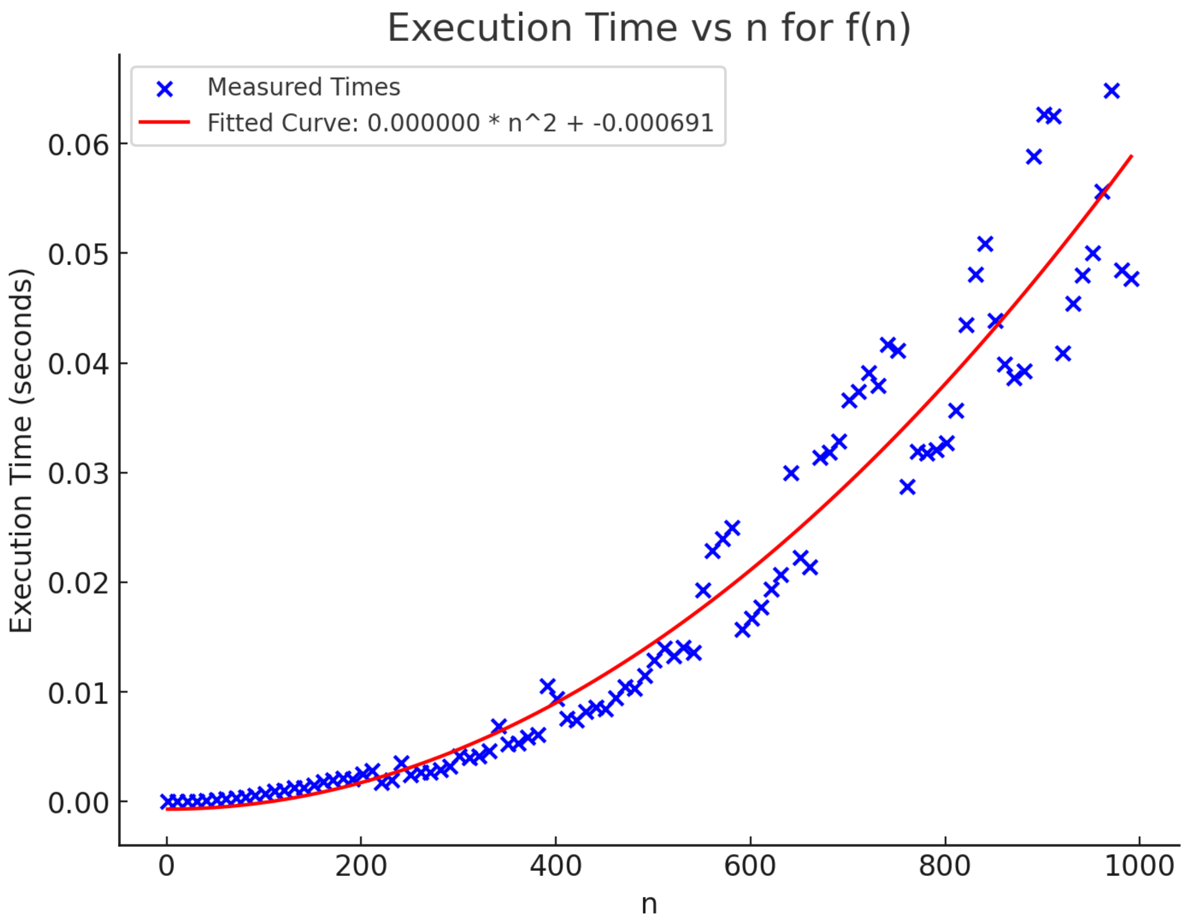
$$T(n) = 1 + \sum_{i=1}^{n} \sum_{j=1}^{n} O(1)$$

$$T(n) = 1 + O(n^2)$$

Thus, T(n) = $O(n^2)$.

---

2)



The plot shows the execution time of $f(n)$ as n increases. The red curve is a quadratic fit, confirming the theoretical $O(n^2)$ complexity.

---

3)

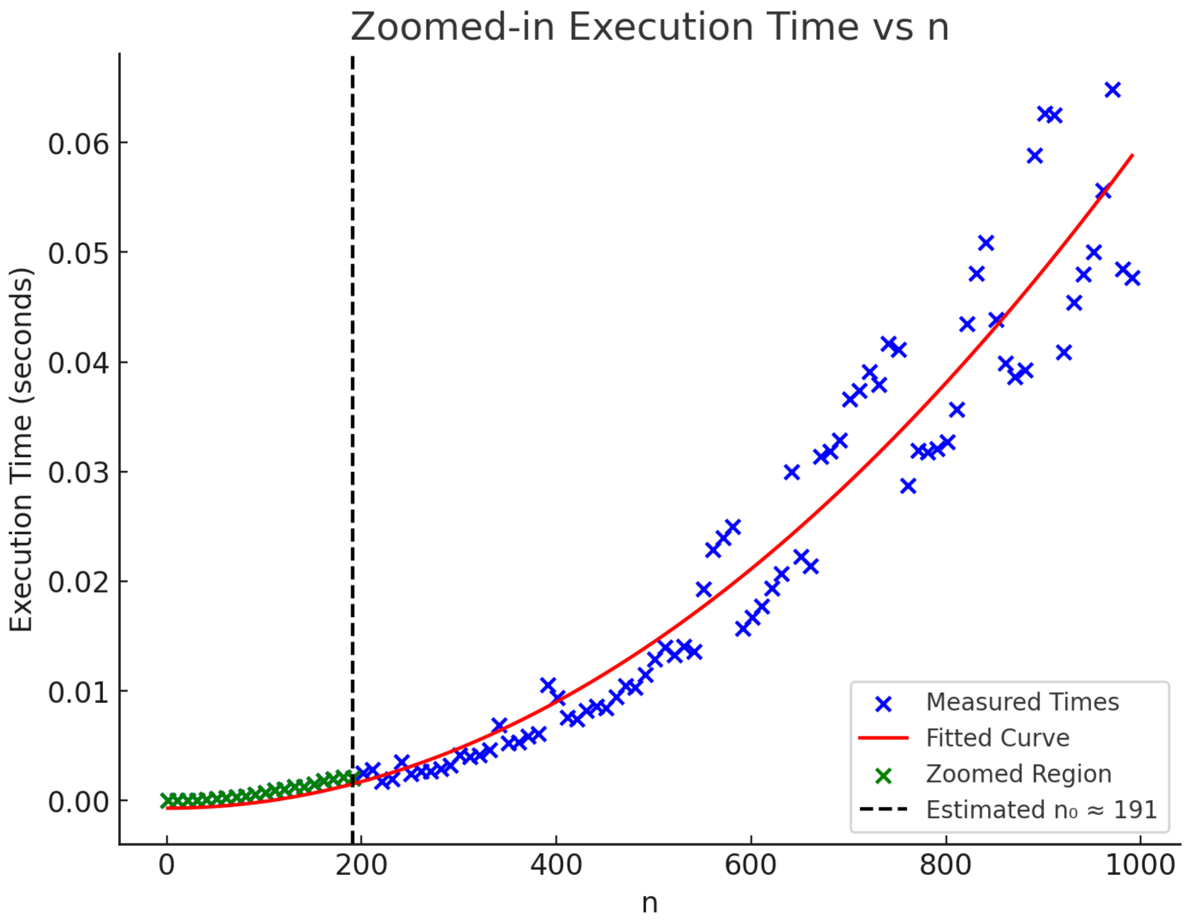
From our analysis:

- **Big-O (Upper Bound):** $O(n^2)$, since the worst-case growth is $O(n^2)$.
- **Big-Omega (Lower Bound):** $\Omega(n^2)$, as the function consistently executes n^2 operations.
- **Big-Theta:** Since the upper and lower bounds match, $\Theta(n^2)$.

Thus, $T(n) = \Theta(n^2)$.

---

4)



The zoomed-in plot highlights the region where the experimental data begins following the quadratic trend. The approximate location of n_0 is around **n ≈ 200**, where the execution times start aligning with the fitted curve.

---

5)

Will this increase runtime?

- Yes, but only slightly. The additional operation y = i + j; still executes in constant time O(1) per iteration.
- The number of iterations remains O(n^2).
- Adding an extra constant-time operation does not change the asymptotic complexity.

Effect on Summation:

- The runtime now includes another O(1) operation per iteration.
- The summation expands to:

$$T(n) = 1 + \sum_{i=1}^{n}\sum_{j=1}^{n}(1+1)$$

$$T(n) = 1 + 2O(n^2)$$

$$T(n) = O(n^2)$$

- **Big-O, Big-Omega, and Big-Theta remain the same** ($\Theta(n^2)$).