Insertion Sort

The previous example is an **insertion sort**, which is a simple sorting algorithm that requires $O(n^2)$ comparisons to sort an array of N elements. How do we know that T(n) is $O(n^2)$?

The algorithm starts with the pointer at position a_1 in the array, and makes 1 comparison (with a_0). Then the pointer moves to position a_2 and makes 2 comparisons, and so on to a_n. So the number of comparisons is:

$$\sum_{i=1}^{n-1} i = 1 + 2 + 3 + \ldots + n - 1$$

What is the *closed form* for this arithmetic series? Well, Gauss observed that you can add this series to itself, using the sum in reverse, and divide by 2 to get the closed form.

$$\begin{array}{r} 1 + 2 + 3 + \ldots + n - 1 \\ \underline{n - 1 + n - 2 + n - 3 + \ldots + n - n - 1} \\ n + n + n + \ldots + n \end{array}$$

Where there are n-1 terms. Since we added the same sum twice, we must divide this expression by 2, giving:

$$\sum_{i=1}^{n-1} i = \frac{n(n-1)}{2} = \frac{n^2 - n}{2} \cong O\left(\frac{1}{2}n^2 - \frac{1}{2}n\right) = O(n^2)$$