Recall:

**InsertionSort(A)**

1. for \( j = 2 \) to \( n \)
2. \( \text{temp} = A_j \)
3. \( i = j - 1 \)
4. while \( i \geq 0 \) and \( \text{temp} < A_j \)
5. \( A_{i+1} = A_i \)
6. \( i = i - 1 \) \( \text{Basic step} \)
7. \( A_{i+1} = \text{temp} \)

Picture:

\[
\begin{array}{cccccc}
A_1 & \rightarrow & \cdots & \rightarrow & A_{j-1} & A_j & A_{j+1} & \cdots & A_n \\
\text{sorted} & & & & & & & & \\
\end{array}
\]
notation:
\[ t_j = \# \text{ of executions of while loop test on } j^{th} \text{ iteration of outer for loop } (2 \leq j \leq n) \]

Let \( T(n) = \text{cost of } \text{I. S. on } A[1 \ldots n] \).

So
\[
T(n) = c_1 n + c_2 (n-1) + c_3 (n-1) + c_4 \sum_{j=2}^{n} t_j + c_5 \sum_{j=2}^{n} (t_j-1) + c_6 \sum_{j=2}^{n} (t_j-1) + c_7 \cdot (n-1)
\]

\[
= (c_4+c_5+c_6) \sum_{j=2}^{n} t_j + (c_1+c_2+c_3-c_5-c_6+c_7) n
\]
\[
+ (-c_2-c_3+c_5+c_6-c_7)
\]
look at: best, worst, average cases

- **Best case** : already sorted \( t_j = 1 \)

So
\[
\sum_{i=2}^{n} t_i = \sum_{i=2}^{n} 1 = n - 1
\]

after some algebra:

\[
\frac{1}{T(n)} = (c_1 + c_2 + c_5 + c_4 + c_7)n + (-c_2 - c_3 - c_4 - c_7)
\]

- **Worst case** : anti-sorted \( t_j = j \)

So
\[
\sum_{j=2}^{n} j = \sum_{j=2}^{n} j = \sum_{j=1}^{n} j - 1 = \frac{n(n+1)}{2} - 1
\]

after some algebra:

\[
\frac{1}{T(n)} = \left( \frac{1}{2} c_4 + \frac{1}{2} c_5 + \frac{1}{2} c_6 \right)n^2 + (c_1 + c_2 + c_5 + \frac{1}{2} c_4 - \frac{1}{2} c_5 - \frac{1}{2} c_6 + c_7)n + (-c_2 - c_3 - c_4 - c_7)
\]
Avg. Case: \( t_j = \frac{j}{2} \), so

\[
\sum_{j=2}^{n} t_j = \sum_{j=2}^{n} \frac{j}{2} = \frac{1}{2} \sum_{j=2}^{n} j = \frac{1}{2} \left( \frac{n(n+1)}{2} - 1 \right)
\]

Some algebra:

\[
T(n) = \left( \begin{array}{c}
\end{array}\right) \frac{n^2}{2} + \left( \begin{array}{c}
\end{array}\right) n + \left( \begin{array}{c}
\end{array}\right)
\]

exercise...

Results:

<table>
<thead>
<tr>
<th></th>
<th>( T(n) )</th>
<th>Asymptotic run time</th>
</tr>
</thead>
<tbody>
<tr>
<td>Best</td>
<td>( an + b )</td>
<td>( \Theta(n) )</td>
</tr>
<tr>
<td>Worst</td>
<td>( cn^2 + dn + e )</td>
<td>( \Theta(n^2) )</td>
</tr>
<tr>
<td>Avg.</td>
<td>( fn^3 + gn + h )</td>
<td>( \Theta(n^2) )</td>
</tr>
</tbody>
</table>
Ex. Suppose we have 4 algorithms A, B, C, D solving same problem.

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Asymptotic Runtime</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>$\Theta(n^2)$</td>
</tr>
<tr>
<td>B</td>
<td>$\Theta(n^4)$</td>
</tr>
<tr>
<td>C</td>
<td>$\Theta(n^4)$</td>
</tr>
<tr>
<td>D</td>
<td>$\Theta(n^2) + 100$</td>
</tr>
</tbody>
</table>

Note:
- D is superior for large, worse for small $n$.
- B, C are same since
  \[
  \frac{10n^2 + 2n + 100}{10n^2} = 1 + \frac{1}{5n} + \frac{10}{n^2}
  \]
  \[
  \downarrow \quad \downarrow
  \]
  \[
  0 \\ 0
  \]
  as $n \to \infty$
A ! B are same:
they can be equalized by running
B on a faster device.

Strategy for Algorithm Analysis:

- Choose some basic operation
  (barometer operation)
- Count # of executions of this
  op. in best, worst, avg. case.
- Determine this count as a function
  of input size.
- Determine asymptotic growth of
  this function.
Do this for insertion sort.

Basic op: Comparison of array elements.

Concentrate on worst case.

\[ j = 2 \ : \ 1 \ \text{comparison} \]

\[ j = 3 \ : \ 2 \ \text{comp}. \]

\[ \vdots \]

\[ i \ : \ j-1 \ \text{comp}. \]

\[ \vdots \]

\[ j = n \ : \ n-1 \ \text{comp} \]

\[ T(n) = 1 + 2 + \ldots + (n-1) = \frac{n(n-1)}{2} \]

\[ \therefore \ T(n) = \Theta(n^2) \]
Problem Example:
How to insert array indices into a list.

\[ A = \begin{pmatrix} a & b & c & d \end{pmatrix} \]

Want: \[ L = (1 \ 2 \ 0 \ 3) \]
Start: \[ L = ( ) \]

Insert 0: \[ L = (0) \]

Insert 1: \[ L = (0) \]
\[ L = (1 \ 0) \]

Insert 2: \[ L = (1 \ 0) \]
\[ L = (1 \ 2 \ 0) \]

Insert 3: \[ L = (1 \ 2 \ 0) \]
\[ L = (1 \ 2 \ 0) \]
\[ L = (1 \ 2 \ 0) \]
\[ L = (1 \ 2 \ 0 \ 3) \]