Comparing Algorithms
- compare running time
- we will see a "rough measure" that characterizes how fast the function grows (rate of growth)
- compare functions for LARGE values of n, that is, compare functions in the limit (ASYMPTOTICALLY)

Ex) choice of linear-time program and quadratic-time program

\[ f(n) = 100n \]
\[ g(n) = 2n^2 \]

Sorting Algorithm
- puts elements of a list in a certain order
  Ex. Insertion Sort
  - how most people sort a deck of cards
  - pick the next card (to be sorted)
make room for it by shifting sorted items
insert it into the correct location

Illustration: Input 5 2 4 6 1 3

```
PARTIALLY
  5  2  4  6  1  3
SORTED
  2  5  4  6  1  3
  2  4  5  6  1  3
  2  4  5  6  1  3
  1  2  4  5  6  3
  1  2  3  4  5  6
```

Algorithm
1) for i = 2 to n do
   2) next = A[i]
   3) j = i - 1
   4) while (j > 0 and A[j] > next)
      6) j = j - 1
   7) A[i + 1] = next

$\text{Time complexity: } \Theta(n^2)$
**Cost #times**

<table>
<thead>
<tr>
<th>Cost</th>
<th>#times</th>
</tr>
</thead>
<tbody>
<tr>
<td>C₁</td>
<td>n</td>
</tr>
<tr>
<td>C₂</td>
<td>n-1</td>
</tr>
<tr>
<td>C₃</td>
<td>n-1</td>
</tr>
<tr>
<td>C₄</td>
<td>1</td>
</tr>
<tr>
<td>C₅</td>
<td>0</td>
</tr>
<tr>
<td>C₆</td>
<td>0</td>
</tr>
<tr>
<td>C₇</td>
<td>n-1</td>
</tr>
</tbody>
</table>

**ti** = # of times while statement is executed at iteration *i*

<table>
<thead>
<tr>
<th>BEST CASE</th>
<th>WORST CASE</th>
</tr>
</thead>
<tbody>
<tr>
<td>ti</td>
<td>i=2 ( ti-1 )</td>
</tr>
<tr>
<td>n</td>
<td>( \sum_{i=2}^{n} ti-1 )</td>
</tr>
</tbody>
</table>

**Total Cost**

\[
f(n) = C₁n + C₂(n-1) + C₃(n-1) + C₄ \sum_{i=2}^{n} ti + C₅ \sum_{i=2}^{n} ti-1 + C₆ \sum_{i=2}^{n} ti-1 + C₇(n-1)
\]

\[
k = \frac{n(n+1)}{2}
\]

**Best Case Analysis**

- Array is already sorted
- \( A[j] \leq \text{next upon the first time the while loop is run.} \)
- Then \( ti = 1 \) for each \( i = 2, ..., n \)
\[ f(n) = (C_1 + C_2 + C_3 + C_4 + C_7)n - (C_2 + C_3 + C_4 + C_7) = k_1n + k_2 \]

\[ f(n) = -n \] rate of growth is

**Worst Case Analysis**

Array is in reverse order

\[ A[j] > \text{next in while loop} \]