Inheritance

• Provides a convenient mechanism for building groups of related abstractions
• Most of the time, this involves a hierarchy of related types—with object at the top
• Subclass inherits all of the attributes of its superclass
  – Can also add new attributes
  – Can also override inherited attributes
Efficiency

• Efficiency is mostly about choosing the right algorithms and data representations

• Inventing efficient algorithms is hard
  – Try and map your problem onto an existing algorithm

• Efficiency can be looked at in terms of time or space
  – Often possible to trade one for the other
Measured Efficiency

• Speed of machine
• Cleverness of Python information
• Input on which you test it
A More Abstract View of Complexity

• What is the number of “basic steps” as a function of the “size” of the input?

• A step is an operation that takes constant time, i.e., time independent of the size of the input.

• Random access machine (RAM)
  – Sequential
  – Constant time for memory access
Multi-level Memory

- Disk: 600MB/sec, terabytes
- Main memory: 10 GB/sec, gigabytes
- Shared cache: 100GB/sec, megabytes
- L1 data and instruction caches: 700GB/sec, 128Kb
Goals of Complexity Analysis

• Compare algorithms
• Understand impact of changing size of inputs
  – Right definition of size not always obvious
Complexity of What?

• Best case: minimum running time over all possible inputs of a given size
• Worst case: maximum running time over all possible inputs of a given size
• Average (expected) case: Mean running time overall all expected inputs of a given size
Asymptotic Complexity

• Rate of growth as a function of size of input
• \( f(x) \in O(n^2) \) means the execution time of the function \( f \) grows no faster than \( kn^2 \), where \( n \) is the size of the input. I.e., this an **upper bound**
# How Things Grow

<table>
<thead>
<tr>
<th></th>
<th>n=1</th>
<th>n = 100</th>
<th>n = 10,000</th>
</tr>
</thead>
<tbody>
<tr>
<td>$O(1)$</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>$O(\log n)$</td>
<td>0</td>
<td>4.6</td>
<td>9</td>
</tr>
<tr>
<td>$O(n)$</td>
<td>1</td>
<td>100</td>
<td>10,000</td>
</tr>
<tr>
<td>$O(n^2)$</td>
<td>1</td>
<td>10,000</td>
<td>100,000,000</td>
</tr>
<tr>
<td>$O(2^n)$</td>
<td>1</td>
<td>1,267,650,600</td>
<td>Really big!</td>
</tr>
</tbody>
</table>