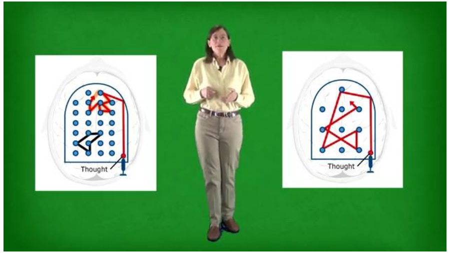
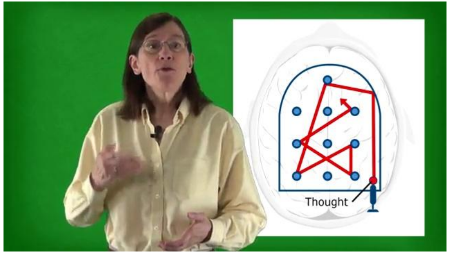
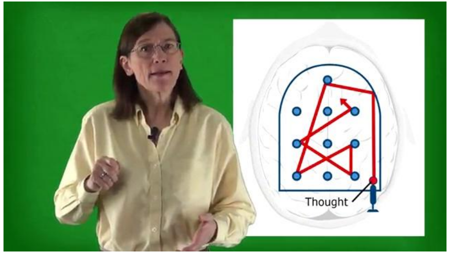
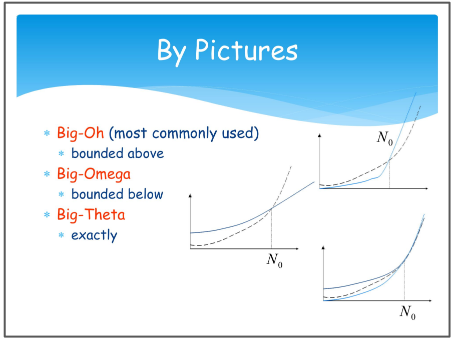


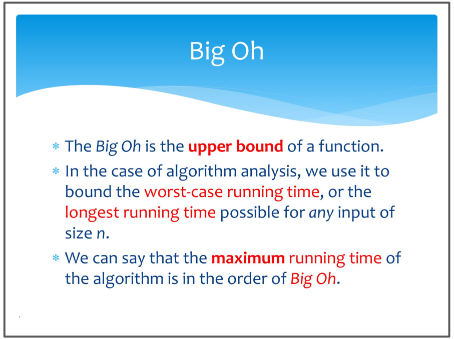






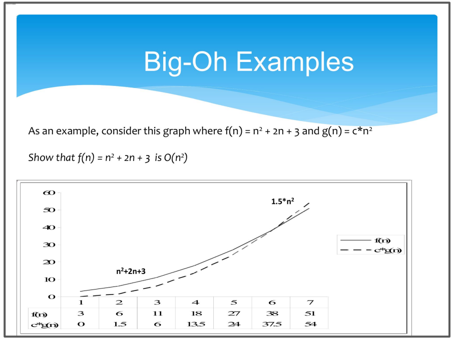
By Pictures

- Big-Oh (most commonly used)
  - bounded above
- Big-Omega
  - bounded below
- Big-Theta
  - exactly



Big Oh

- The *Big Oh* is the **upper bound** of a function.
- In the case of algorithm analysis, we use it to bound the worst-case running time, or the longest running time possible for *any* input of size *n*.
- We can say that the **maximum** running time of the algorithm is in the order of *Big Oh*.

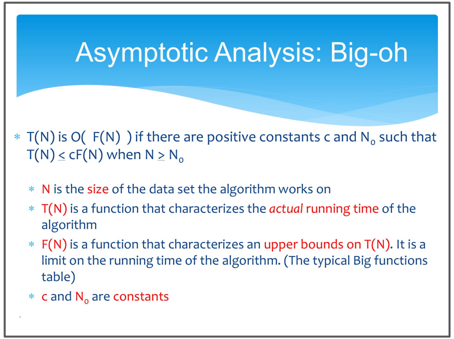


Big-Oh Examples

As an example, consider this graph where $f(n) = n^2 + 2n + 3$ and $g(n) = c*n^2$

Show that $f(n) = n^2 + 2n + 3$ *is* $O(n^2)$

	1	2	3	4	5	6	7
f(n)	3	6	11	18	27	38	51
c*g(n)	0	1.5	6	13.5	24	37.5	54



Asymptotic Analysis: Big-oh

- T(N) is O(F(N)) if there are positive constants c and N_0 such that $T(N) \leq cF(N)$ when $N \geq N_0$
  - N is the size of the data set the algorithm works on
  - T(N) is a function that characterizes the *actual* running time of the algorithm
  - F(N) is a function that characterizes an upper bounds on T(N). It is a limit on the running time of the algorithm. (The typical Big functions table)
  - c and N_0 are constants