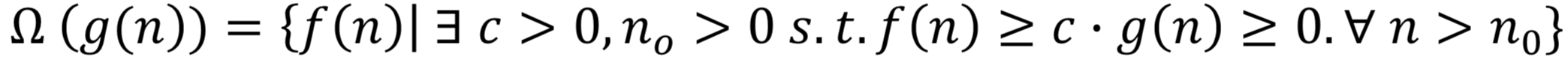
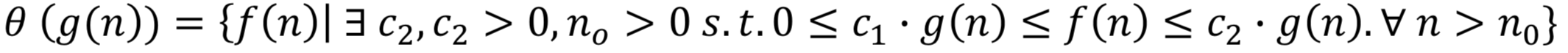
Algorithm:

A step-by-step procedure for solving a mathematical problem in a finite number of steps that frequently involves repetition of an operation.







Prove f(xxx) = O(yyy), find c, n0­, show that definition holds

Prove f(xxx) ≠ O(yyy), assume =, find equation about c and n by using definition, set n = …n0 …c s.t. n > n0 holes, then find contradiction

Prove O(xxx) = O(yyy), show O(xxx) ⊆ O(yyy) and O(yyy) ⊆ O(xxx)

Prove O(n2+3n) = O(n2-50n)

Show O(n2+3n) ⊆ O(n2-50n)

Let f(n) ∈O(n2+3n), f(n) ≤ c(n2+3n), n > n0

Want f(n) ∈O(n2-50n), f(n) ≤ c’(n2-50n)

Let c’ = c+1, c(n2+3n) ≤ (c+1)(n2-50n) => n ≥ 50+53c => n0’ = max(n0, 50+53c)

Show O(n2-50n) ⊆ O(n2+3n), vice versa

Prove O(xxx) ≠ O(yyy), find counter example

Greedy Algorithm:

The optimal local solution is taken at each time step

Proof:

1. At every step of an ideal optimal algorithm show that greedy does not do worse
2. Called optimal substructure
3. Proof by induction
   1. Basic step is by construction
   2. Inductive step shows by contradiction

Very efficient, no turning back

May not lead final optimal solution, but still ok

Can prove for some algorithm how far we can be form optimal solution

Master theorem:

Xxx