Scott Felch  
July 6, 2015  
CSCI 405  
HW #1 (late)

#1 Prove that f(n) = O(g(n)) if g(n) = Ω(f(n)).

Upper bound:  
O(g(n)) = { f(n): Ǝ positive constants C1 and n0 s.t. 0 ≤ f(n) ≤ c1g(n) ∀ n ≥ n0 }

Lower bound:  
Ω(f(n)) = { g(n) : Ǝ positive constants C2 and n0 s.t. 0 ≤ c2f(n) ≤ g(n) ∀ n ≥ n0 }

Assume f(n) and g(n) are same functions in each statement.

O(g(n)) =>

Assigning c1 = (1 / c2 ) proves O(g(n)) =

#2 Prove that 2lg n = n

by identity

by 1

#3 Prove that…

#4 Prove that…

#5

#6 Solve T(n) = T(n-1) + n3

T(n) = T(n – 1) + n3

T(n – 1) = T(n – 2) + n3 + (n – 1)3

T(n – 2) = T(n – 3) + n3 + (n – 1)3  + (n – 2)3

….

T(n – (n – 1)) = T(1) = 13

T(3) = T(2) + 33

T(2) = T(1) + 33 + 23

T(1) = T(0) + 33 + 23 + 13

Adding total of n terms when finished recursing

T(n – k)= kT(n – k) + n \* n3

T(n – k) = kT(n –k) + n4

O(n4) complexity