

CIS 435, Spring 2002, Joseph Leung
Homework #1

1.2-2 Consider linear search again. How many elements of the input sequence need to be checked on the average, assuming that the element being searched for is equally likely to be any element in the array? How about in the worst case? What are the average-case and worst-case running times of linear search in Θ notation? Justify your answers.

1.3-3 Use mathematical induction to show that the solution of the recurrence $T(n) = 2$ if $n = 2$, $T(n) = 2T(n/2) + n$ if $n = 2^k, k > 1$, is $T(n) = n \lg n$.

1.3-5 Referring back to the searching problem, observe that if the sequence A is sorted, we can check the midpoint of the sequence v and eliminate half of the sequence from further consideration. Binary search is an algorithm that repeats this procedure, halving the size of the remaining portion of the sequence each time. Write pseudocode, either iterative or recursive, for binary search. Argue that the worst-case running time of binary search is $\Theta(\lg n)$.

2-3 (first 3 columns) (a) Rank the following functions by order of growth; that is, find an arrangement g_1, g_2, \dots, g_{15} of the following functions satisfying $g_1 = \Omega(g_2), g_2 = \Omega(g_3), \dots, g_{14} = \Omega(g_{15})$. Partition your list into equivalence classes such that $f(n)$ and $g(n)$ are in the same class if and only if $f(n) = \Theta(g(n))$.

$\lg(\lg^* n)$	$2^{\lg^* n}$	$(\sqrt{2})^{\lg n}$
$(\frac{3}{2})^n$	n^3	$\lg^2 n$
$\ln \ln n$	$\lg^* n$	$n2^n$
$2^{\lg n}$	$(\lg n)^{\lg n}$	e^n
$\lg^*(\lg n)$	$2\sqrt{2\lg n}$	n

(b) Give an example of a single nonnegative function $f(n)$ such that for all functions $g(n)$ in part (a), $f(n)$ is neither $O(g(n))$ nor $\Omega(g(n))$.