COSC 302: Analysis of Algorithms — Spring 2018
Adam Pettway
Colgate University

Problem Set 2 — Order of Growth and Asymptotic Analysis

---

1. Pair-Finder(A, x)

```
mergeSort(A)
left = 0
right = A.length − 1
while left < right do
    if A[left] + A[right] = x then
        return True
        if A[left] + A[right] < x then
            left = left + 1
        if A[left] + A[right] > x then
            right = right − 1
return False
```

   Invariant: $left < right$ and $A[1...left-1]$ and $A[right...n]$ cannot be used to find the sum x. Maintance: If $A[left] + A[right] = x$ then the loop terminates. If x is smaller then it will decrease the right pointer and increase the left pointer if the x is smaller than the current sum. Termination: If there is a pair then the loop terminates and the function returns true. Otherwise we reach the point there $left > right$. Which means no such pair exists so the loop terminates and the False is returned.

2. Problem 3.1-1 in CLRS.
   We shall prove that that $max(f(n), g(n)) = \Theta(f(n) + g(n))$.

   Since $f(n)$ and $g(n)$ are nonnegative functions $f(n) \leq f(n) + g(n)$ and $g(n) \leq f(n) + g(n)$. Therefore, $max(f(n), g(n)) = O(f(n) + g(n))$.

   Note that $f(n) + g(n) \leq 2 * max(f(n), g(n))$ because $f(n) \leq max(f(n), g(n))$ and $g(n) \leq max(f(n), g(n))$ by definition.

   Also, the definition of Ω if there exists a positive constant k which $0 \leq cg(n) \leq f(n) \forall n \geq n_0$. In this case, $n_0 = 0 and k = 2$.

   Since we proved that $max(f(n), g(n)) = \Omega(f(n)+g(n))$ and $max(f(n), g(n)) = O(f(n)+g(n))$ then by definition $max(f(n), g(n)) = \Theta(f(n) + g(n))$.

3. Problem 3-2 (Big Oh, Big Omega, and Big Theta only) in CLRS.

A	B	O	Ω	Θ
$lg^k n$	n^ϵ	No	Yes	No
n^k	c^n	Yes	No	No
$\sqrt{n}$	$n^{\sin n}$	No	No	No
2^n	$2^{n/2}$	Yes	Yes	Yes
n^{lgc}	c^{lgn}	Yes	No	No
$lg(n!)$	$lg(n^n)$	Yes	No	No

4. Problem 3-4 (parts c,d,e,g) in CLRS.

   c $f(n) = O(g(n)) \implies lg(f(n)) = O(g(n))$. We will prove this directly. By defition of upper bound: $f(n) \leq g(n)$ and by definition of log $lg(f(n)) \leq lg(g(n))$ which means that $lg(f(n)) = O(lg(g(n)))$

   d Suppose that $f(n) = 2n$ and $g(n) = lgn$. Then by definition, $f(n) = O(g(n)$ when $c = 2$. However, $2^{2n} \neq O(2^n)$ since there is no c that works so that so that $f(n) \leq g(n)$ for all n. Therefore the statement is not always true.

   e Suppose that $f(n) = \frac{1}{n}$ thus $(f(n))^2 = \frac{1}{n^2}$. So the statement $f(n) = O((f(n))^2)$ is a contradiction since there exists no c that works so that for all n $f(n) \leq (f(n))^2$. Therefore the statement is not true.

   g Suppose that $f(n) = 2^n$ so we must show that $2^n \leq c * 2^{\frac{n}{2}}$ If we simplify this we get $n \leq c * \frac{n}{2}$. But there exists no $c > 1$ where $n \leq c * \frac{n}{2}$ for all n. Therefore the statement is not true.