

Due by the beginning of class, Sept. 8.

1. Prove: A binary tree with  $n$  nodes has depth at least  $\lfloor \log n \rfloor$  and at most  $n - 1$ . (Hint: Show that if a binary tree has depth  $d$  and has  $n$  nodes, then  $n \leq 2^{d+1} - 1$ .)
2. Prove that  $\log(n!) \in \Theta(n \log n)$ .
3. Prove by induction on  $k$ : (Remember these formulas, they may be used later)

$$(a) \sum_{i=1}^k i(i+1) = \frac{1}{3}k(k+1)(k+2)$$

$$(b) \sum_{i=0}^k i2^i = (k-1)2^{k+1} + 2$$

$$(c) \sum_{i=0}^k \frac{i}{2^i} = 2 - \frac{k+2}{2^k}$$

4. Place the following functions into increasing asymptotic order. If two or more of the functions are of the same asymptotic order, then indicate this. Prove the correctness of your ordering. (In other words, if you claim that  $g(n)$  is greater than  $f(n)$  then show that  $f(n) \in O(g(n))$  but  $f(n)$  is not in  $\Theta(g(n))$ ).

$$4n, \quad n^2, \quad n \log n, \quad n \ln n, \quad \lg n, \quad e^n$$

Note:  $\log$  means logarithm base 2, and  $\ln$  means logarithm base  $e$  (natural log).

5. Let  $G = (V, E, W)$  be a weighted graph such that no two different edges in  $E$  have the same weight ( $\forall e_1, e_2 \in E, e_1 \neq e_2 \Rightarrow W(e_1) \neq W(e_2)$ ). Prove that  $G$  has a unique unrooted minimal spanning tree (The set of edges participating in an MST for  $G$  is unique).
6. Suppose there is a C++ library on your machine which has the following function:

*HamC*( $G$ ):

**Input:** Graph  $G$

**Output:** Yes, if  $G$  has a simple cycle which traverses all nodes; No, otherwise.

Such a cycle is known as a *Hamiltonian Cycle*. Let *HamC*( $G$ ) implement the above routine. Suppose further, that *HamC*() runs in polynomial time, that is, it runs in time  $O(n^c)$  for some constant  $c$  on an  $n$  node graph.

A *Hamiltonian Path from  $u$  to  $v$*  is a simple path from  $u$  to  $v$  which traverses all the nodes of a graph. Your mission is to write an algorithm with the following input and output:

*HamP*( $G, u, v$ ):

**Input:** Graph  $G$ , and nodes  $u$  and  $v$  in  $G$ .

**Output:** Yes, if there is a Hamiltonian Path from  $u$  to  $v$  in  $G$ ; No otherwise.

Your algorithm should run in polynomial time. (Hint: You'll have a tough time doing this without using the *HamC*() subroutine.)