Sohan Gadiraju

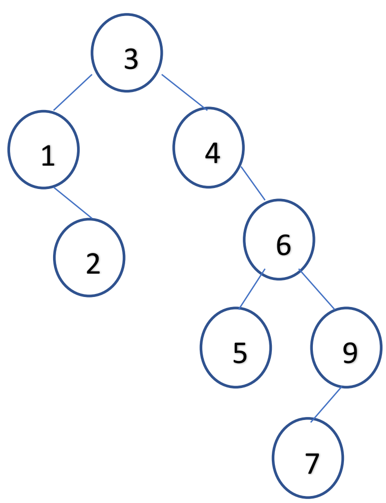
**CISC 220-080 – Honors: Data Structures**
**Fall 2019**

**Homework #5**

**1)** Since the parent node of a node cannot be null, and if a node contains one child it has to contain one null link, and if a node contains no child it has to have two null links. Therefore, if a Binary Tree has N nodes, it will have 2N links. Additionally, as every node has a unique parent (except the root), the number of links to the parents will be N-1. By subtracting the number of parent links from the total number of links, we can get the number of links that represent null, which would be 2N – (N – 1) = N + 1.

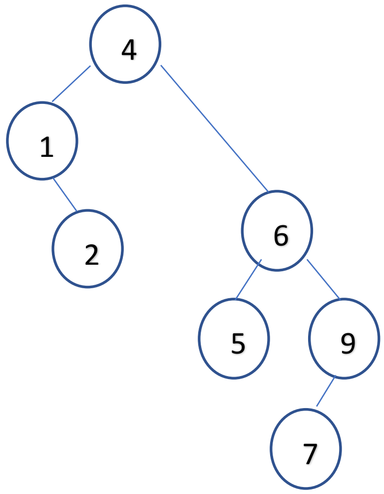
**2)** In order to prove that the maximum number of nodes in a binary tree of height $h$ is $2^{h+1} - 1$, we first need to prove the base case where $h$=0. When $h$=0, $2^{0+1} - 1 = 1$. This is correct because a binary tree of height 0 only has 1 node. Now, we do the induction step. We assume that the statement is true for h=k+1 where k >= 0. Now we need to show that $2^{k+1} - 1 = 2^{(k+1)+1} - 1$. Due to how Binary Trees work, by definition both sub-trees must have a height of h. Therefore, the number of nodes in the whole tree is 1 + ($2^{k+1} - 1$) + ($2^{k+1} - 1$) which simplifies to $2^{k+2} - 1$. Therefore, we have proved that the maximum number of nodes in any binary tree of height h is $2^{h+1} - 1$.

**3)** When 3 is inserted it becomes the root node, then 1 will be inserted as the left child and 4 will be inserted as the right child. Since 6 and 9 are larger, 6 becomes the right child of 4 and 9 becomes the right child of 6. Since 2 is less than 3 but bigger than 1, it becomes the right child of 1. Since 5 is greater than 3 and 4 but less than 6, 5 becomes the left child of 6. Lastly, 7 becomes the left child of 9. This gives us the following binary tree:



Sohan Gadiraju

Deleting the root would result in 4 becoming the root. So the tree would become




4) **Prefix:** - * * a b + c d e
   **Infix:** (a * b) * (c + d) – e
   **Postfix:** a b * c d + * e -