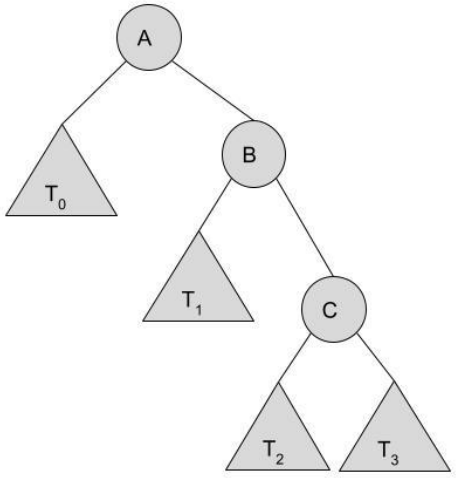
Assignment 5 (100 points)

Part 1(50 points): AVL Trees, Binary Heaps, Tries

1. (10 points) Complete the following leftRotate function with respect to the following binary tree:

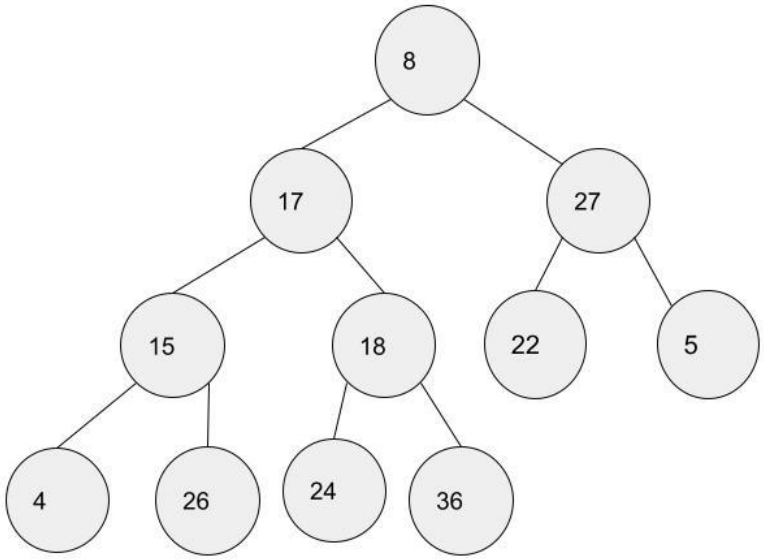



```
Node*  leftRotate(Node *A)
{




}
```

2. (20 points) Consider the following arbitrary complete binary tree which is stored in an array. The root of the tree is at index 1. The tree has currently 11 nodes.



a) If you want to build a max heap from this array using heapify operation, from what index you need to start the process of percolate down? Write the index number based on the above tree: (3 points)

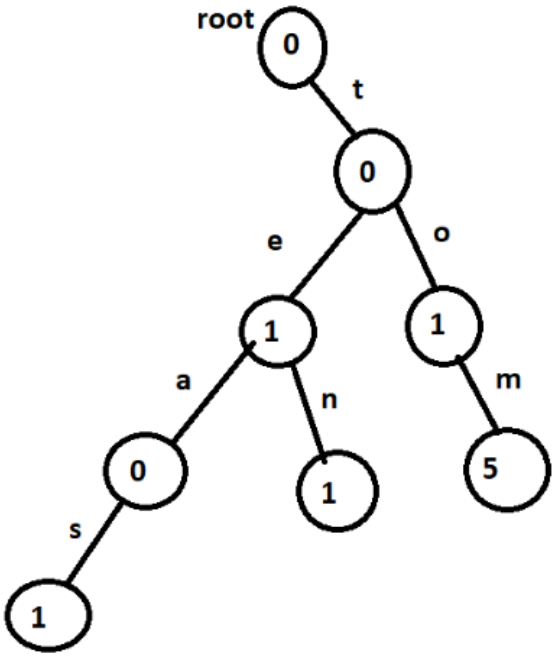
b) Show the heapify algorithm step by step. (12 points)

c) What is the run-time to build a heap from an arbitrary array with n numbers using heapify? (3 points)

d) What is the run-time to build a heap from an arbitrary array with n numbers **without** using heapify? (2 points)

3. (10 points) What is the run-time of heap sort? Briefly outline the steps of performing the heap sort. (Should not be more than 4-5 lines)

4. (10 points) Write the list of words and their count for the following trie.



Part 2 (50 points) : DFS Implementation: You need to upload a separate .cpp file Download the bfs.cpp file from Canvas and add a function to perform a depth-first search on a graph. You have to use stack implementation of DFS using STL (#include <stack>)