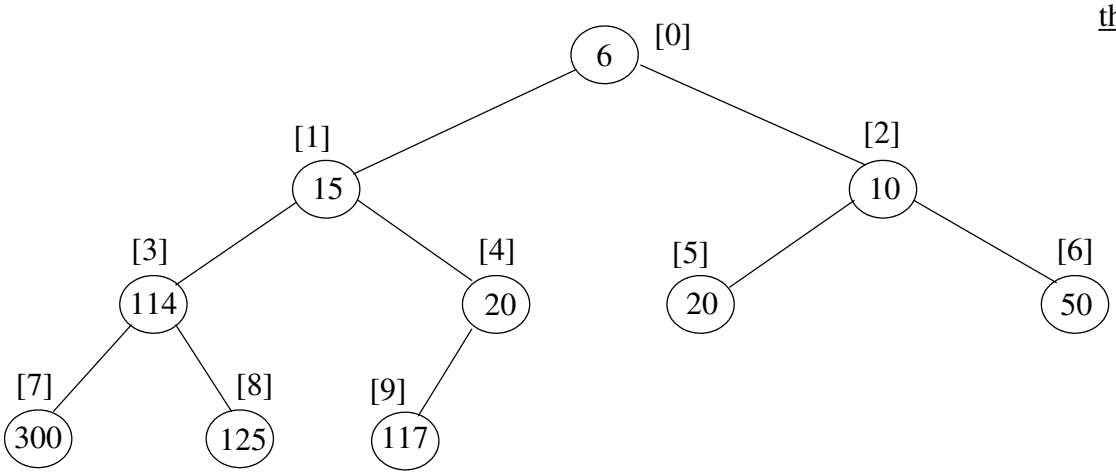
Data Structures Name:_____

1. A very "non-intuitive", but powerful array-based approach to implement an priority queue is called a *heap*. An array is used to store a *complete binary tree* (a full tree with any additional leaves as far left as possible) with the items being arranges by *heap-order property*, i.e., each node is less than or equal to either of its children. Below is an example of a heap "viewed" an a complete binary tree. The array indexes are indicated in [ ]'s.



# Nodes this level	Total Nodes from root
1	
	1
2	
	3

a) Complete the above table of node counts.

b) The *height* of a tree is its number of levels. The above tree has a height of 4. Assuming there are "n" items in the heap, what's the relationship between the n and the height of the complete binary tree?

c) What is the worst-case theta (Θ( )) notation for adding a new item in the heap by sifting it up the tree?

d) Write the algorithm for adding a new item to the heap. Use a siftUp(int currentPositon) function

```
template <class T>
class BinaryHeap {
private:
      int maxSize;
      int numItems;
      T * heap;
      ...
```

2. The item with the highest priority is at the root. Where is the item with the second highest priority?

b) The delete operation returns the root node and eliminates it from the tree by:

- copying the "last leaf item" in the tree (i.e., right most item in the array) to the root,
- "sifting this item down" the tree by repeatedly exchanging it with the smaller of its two children until it is in the correct spot.

What is the worst-case theta ($\Theta(\ )$) notation for deleting the smallest item in the heap?

c) Write the algorithm for deleting an item from the heap. Use a siftDown(int currentPositon) function

3. Rewrite the siftUp function used in the insert operation recursively.