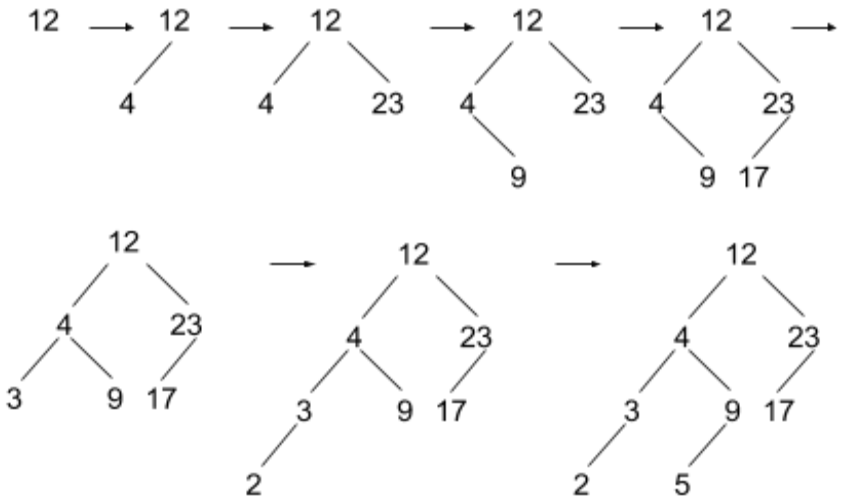
Shuni Li
Comp221 HW5
Professor Shilad Sen
Feb 25, 2016

1. (5 points) Show the sequence of steps for inserting the following values into a binary search tree: 12, 4, 23, 9, 17, 3, 2, 5. Show the order in which a pre-order, post-order, and in-order traversal would visit each node.



Pre-order: 12, 4, 3, 2, 9, 5, 23, 17
Post-order: 2, 3, 5, 9, 4, 17, 23, 12
In-order: 2, 3, 4, 5, 9, 12, 17, 23

2. (10 points) Design a decrease-by-half algorithm for computing $ceil(log_2 n)$ and determine its time efficiency.

```
Algorithm computeLog(n)
        If n < 2
                Return 1
        else
                return 1+ computeLog(n/2)
```

$$T(n) = T(floor[n/2]) + 1 = floor[log(n)] + 1 = ceil[log(n+1)] = \Theta(log(n))$$

3. (10 pts) In a max-heap, we typically only access the largest value. That value is always in the root position. Given a max-heap with n = 255 values in it:
    a. Which locations in the heap might contain the second largest value?
       The left or right child of the root. (position 1 or 2)
    b. Which locations in the heap might contain the third largest value?

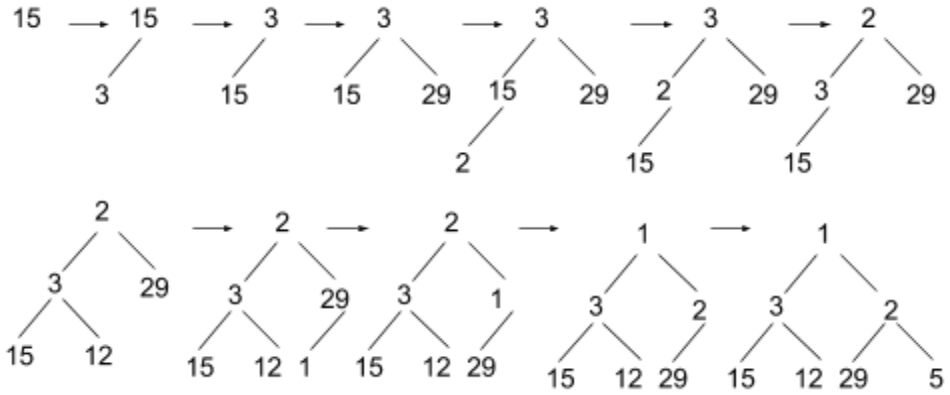
The left or right child of the root. (position 1 or 2)
Or, the left or right child of the second largest value. (position 3, 4, 5, 6)

c. Which locations in the heap might contain the fourth largest value?
Case 1: left-subtree is all greater than right-subtree. So the left or right child of the second largest value. Position: 3, 4
Case 2: left-subtree is all less than right-subtree. So the left or right child of the second largest value. Position: 5, 6
Case 3: more general case, cannot determine which node at level 2 is greater. Position: 3,4,5,6

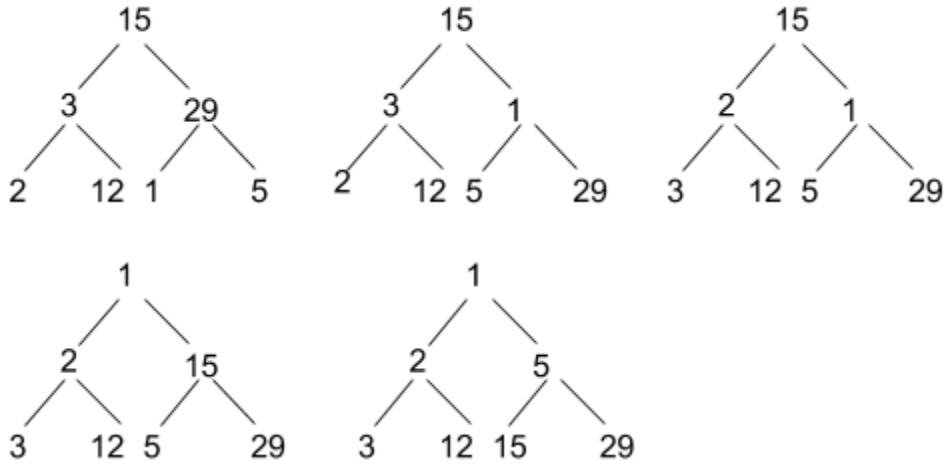
d. Which locations in the heap might contain the smallest value?
The leaf nodes. Position from 127 to 254.

e. Which locations in the heap might contain the second smallest value?
The leaf nodes. Position from 127 to 254.

4. (10 pts) Consider the array of numbers: [15, 3, 29, 2, 12, 1, 5]
   a. Draw a series of diagrams to show the min heap that would be constructed if we inserted each of the values from the array into the heap, one after the other, starting with an empty heap. Draw a new diagram every time a value walks up the tree.



   b. Draw a series of diagrams to show the min heap that would be constructed using the bottom-up heap building algorithm. Draw a new heap for every interior node.



5. (10 pts) Write out an algorithm in pseudocode that takes an array of numbers and checks to see if it is a max-heap. Be as efficient as you can. Explain the worst-case efficiency of the algorithm, giving its efficiency class in big-O notation.

```
Algorithm isMaxHeap(A[0...n-1])
        for i in 0 to  (n-2)/2
                If A[i] <= A[2i+1] || A[i]< = A[2i+2]
                        return false
        return true
```

The worst case happens when the tree follows the heap property except the last parent node with depth 2. In this case, we need to examine if every parent node is greater than its children. The time complexity is $\sum_{0}^{(n-2)/2} 1 = (n-2)/2 = \Theta(n)$.