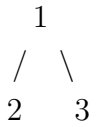
DAVP Week 3

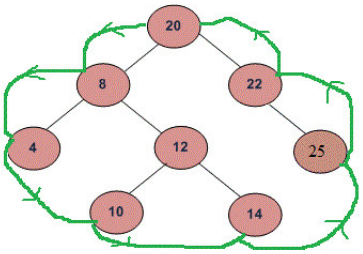
September 13, 2015

1. Given a binary tree and one of its nodes, find the path with the maximum sum that contains the given node. The path may start and end at any node in the tree. [Medium]
   Example:
   Input: Root of the tree below



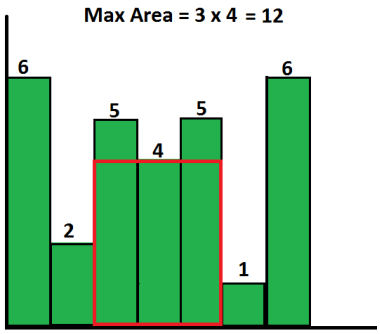
   Output: 6 (2+1+3)

2. **Boundary Traversal of binary tree** [Medium]
   Given a binary tree, print boundary nodes of the binary tree Anti-Clockwise starting from the root. For example, boundary traversal of the following tree is "20 8 4 10 14 25 22".



3. **Data Structure to represent sequence of pairs** [Medium-Hard]
   Operations supported are:
   - insert(pair): Insert element
   - find_pair(X): Find pair with the maximum value of second element given that first element is greater than X

4. **Largest Rectangular Area in a Histogram** [Hard]
   Find the largest rectangular area possible in a given histogram where the largest rectangle can be made of a number of contiguous bars. For simplicity, assume that all bars have same width and the width is 1 unit. For example, consider the following histogram with 7 bars of heights 6, 2, 5, 4, 5, 2, 6. The largest possible rectangle is 12 (see the below figure, the max area rectangle is highlighted in red).



   Try and work out an O(n) algorithm.

5. Given a sequence of $2n + 1$ numbers containing n distinct numbers twice, and a number exactly once, find the number occurring exactly once (also, $3n + 1$ and thrice).
   No more than constant extra space. [Easy]

6. K^{th} **largest element in a stream** [Medium]
   Given an infinite stream of integers, find the k^{th} largest element at any point of time. Extra space allowed is O(k).
   Example:
   Input:
   stream[] = {10, 20, 11, 70, 50, 40, 100, 5, ...}
   k = 3
   Output: {_, _, 10, 11, 20, 40, 50, 50, ...}

7. **Median in a stream of integers (running integers)** [Medium]
   Given that integers are read from a data stream. Find median of elements read so for in efficient way. For simplicity assume there are no duplicates. For example, let us consider the stream 5, 15, 1, 3 . . .
   After reading 1st element of stream - 5 -> median - 5
   After reading 2nd element of stream - 5, 15 -> median - 10
   After reading 3rd element of stream - 5, 15, 1 -> median - 5
   After reading 4th element of stream - 5, 15, 1, 3 -> median - 4, so on...
   Making it clear, when the input size is odd, we take the middle element of sorted data. If the input size is even, we pick average of middle two elements in sorted stream.
   Note that output is effective median of integers read from the stream so far. Such an algorithm is called online algorithm. Any algorithm that can guarantee output of i-elements after processing i^{th} element, is said to be online algorithm.

8. You are given a sequence of numbers a_1, a_2, ..., a_n and a number m.
   Check if it is possible to choose a non-empty subsequence a_{ij} such that the sum of numbers in this subsequence is divisible by m.

9. **Max Sum Subsequence** [Easy] xor-[Hard]
   Given a sequence of numbers find the subsequence with maximum sum.
   Now can you do the same to calculate maximum xor subsequence?
   Hint: The property of addition is that sum of after adding an element is elem + sum_so_far. But this does not hold in case of xor.

10. **Check if a given string is a rotation of a palindrome**[Easy]
    Given a string, check if it is a rotation of a palindrome. For example your function should return true for "aab" as it is a rotation of "aba".
    Examples:
    Input: str = "aaaad"
    Output: 1
    // "aaaad" is a rotation of a palindrome "aadaa"

    Input: str = "abcd"
    Output: 0
    // "abcd" is not a rotation of any palindrome.
    Try doing it in O(n)!