**Indian Institute of Technology, Guwahati**

**Department of Computer Science and Engineering**

**Data Structure Lab: (CS210)**

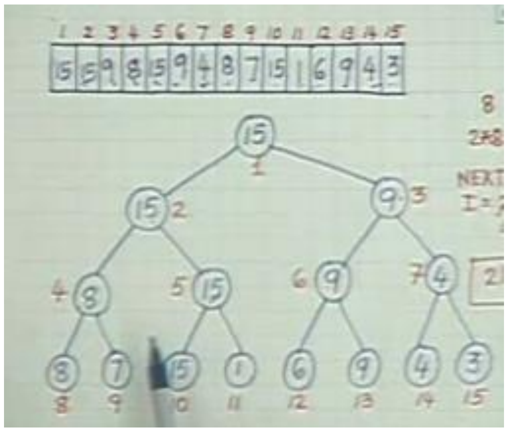
**Assignment: 4**

**Date: 1st September, 2016.        Total Marks: 20 (lab assignments) + 20 (offline assignments)**

**Deadline of Offline Assignment Submission: 7th September, 2016.**

**Lab Assignments:**

**1.** Implement **buildTornamentTree** function. This function will build a winner tree of n numbers using array where highest number is the winner. For n numbers, the array size would be 2n-1. The inputs will be stored in locations (n to 2n-1) of the array. Build winner tree in index space (1 to n-1). For child with index i and (i+1), parent index would be i/2. Following figure is for your reference. Your number of inputs may not be always $n = 2^k$. Assume that the all array elements are greater than 0. **(10)**



2. Implement **nextMax** function on the winner tree. This function will rebuild the winner tree by placing 0 at the winner position and rebuild along the path of the previous winner and find the next winner. You should use a log n function to identify the winner's original positions in the array. You should not use buildTornamentTree function from Problem 1. **(10)**

**Offline Assignments:**

3. (a) Implement sparse matrix using link list as described in class. Inputs are row and col number followed by number of non-zero elements (k) in the matrix. Consider random inputs for matrix elements. Your program will randomly choose k-distinct (i, j) positions in the matrix and will

place those number in those positions. All (i, j) values must be within range of row and col, respectively. (b) Write a function to find the highest number in the sparse matrix. **(5 + 5 =10)**
**Inputs:** example:
<row, col>
<no of non-zero elements>
For example:
6 7
22
**Output:**
<highest number>. For example
67

4. A Student was late to submit his assignment which was to be handed over in person to the Data Structure professor. Next day the professor was on leave and at home. The student needs to cross the jungle (may be good or bad) in order to reach professor's home. The jungle is good if it has exactly one point (Start Point) to enter into it and exactly one point (End Point) to exit out of it. In a good jungle, there must be exactly one way from start point to end point in order to reach professors' home. Entry point and Exit point must be on the border of the jungle/map. Initially the student is outside the jungle. Student can enter the jungle from any side. Given the rectangular (or square) map of jungle, will the student be able to submit his assignment? **(10)**

**Input Format:**

For every test case,
First line contains rows in map and columns in map separated by a space.
From 2nd line onwards, the map of jungle is given in which 'T' is a tree and student can't proceed from there and 'L' is land area on which student can move to.
Stop taking input when input rows and columns are both zero.

**Output Format:**

For every test case, print output on newline.
Print "Submitted!"(without quotes) if student was able to submit assignment else print "Bad Luck!"(without quotes).

**Examples:**
4 4
TTTT
TLLL
TLTT
TLTT
Submitted! (Entry point + Exit point exactly 2 and at least one path from one point to another)

1 1
L
Bad Luck! (Entry point + Exit point not exactly 2)

```
5 1
T
T
L
L
T
Submitted! (Entry point + Exit point exactly 2 and at least one path from one point to another)

2 2
TL
LT
Bad Luck! (No path from one point to another)

3 4
TLLT
TLTT
TLTT
Bad Luck! (Entry point + Exit point not exactly 2)
```