Homework Assignment #1

Due: January 16, 2020, by 5:30 pm

Jiahong Zhai 1005877561

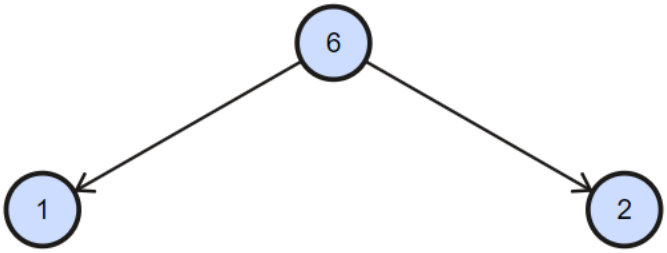
Question 2

Let I_n be the set of n integers {1, 2, 3, ……, n}, and n = 2.
Let S be the subset of I_n.
Let B be the vector(array) of n bits to represent S.

B[i] = 1 if i ∈ S
B[i] = 0 if i ∉ S

Use a complete binary tree T to store S. Each node in T is greater or equal to its children.

a. n = 8
I_n = {1, 2, 3, 4, 5, 6, 7, 8}
S = {1, 2, 6}
B = {1, 1, 0, 0, 0, 1, 0, 0}
T =



b.

INSERT(j): S doesn't allow duplicate, so we first need to see if j is in S. We can achieve this by run MEMBER(j) (we will take about MEMBER(j) later), which takes constant time O(1). If MEMBER(j) = True, then program terminated. If MEMBER(j) = False we do the following to insert j:

1. B[j] = 1 O(1)
2. Insert node j to the left-most available slot in the bottom level of the tree O(1)
3. If the new node is greater than its parent node, then swap them. Do this until the new node is not greater than its parent node.
   O(log n), because there are at most log n level in T

So worst-case time complexity of INSERT(j) is O(log n).

DELETE(j): We first need to see if j is in S. We can achieve this by run

MEMBER(j) (we will take about MEMBER(j) later), which takes constant time $O(1)$. If MEMBER(j) = False, then program terminated. If MEMBER(j) = True we do the following to delete j:

1. B[j] = 0 $O(1)$
2. Use the right-most node in the bottom level of T to replace the node of j.
3. If the new node is greater than its parent node, then swap them. If the new node is smaller than its child node, then swap them. Do this until the new node is not greater than its parent node or smaller than its child node.

$O(\log n)$, because there are at most log n level in T

So worst-case time complexity of DELETE(j) is $O(\log n)$

MAXIMUM(j): return the value of the root of T, which is $O(1)$

c. MEMBER(j): if B[j] = 1, then return True.
if B[j] = 0, then return False.

So worst-case time complexity of MEMBER (j) is $O(1)$