CS206A Data Structure

HW4

Student ID: 20160253

name: 박예기 (Park ye gi)

Problem1.

When the tree is full tree, the tree has the maximum number of nodes. On level i, the number of node is k^{i-1}. So, the maximum number of nodes in a k-ary tree of height h is

$\sum_{i=1}^{h} k^{i-1} = \frac{k^h-1}{k-1}$ (the answer is when the height of root is one.)

Problem2(a).

Pre-order : A B C D E F
In-order : F E D C B A
Post-order : F E D C B A

Problem2(b).

Pre-order : A B D H I E C F G
In-order : H D I B E A F C G
Post-order : H I D E B F G C A

Problem3.

I made BinaryTree class which contains three function: constructor function, preorder(), and postorder().

Constructor function is for making tree form. The form of tree is array of array like the below.

[[first-level element], [second-level element], [third-level element], …]

This function gets input for making the list. You should input the elements in order of level and if you want to remain some nodes empty, you should press the Enter.

Function preorder() prints out the result of pre-order traversal of the given binary tree. The function makes a list named stack. Firstly, in the stack, there is only a root. In the loop, it pops stack[-1] and append the right child and left child of popped element if popped element is not leaf node. And it sets stack[-1] as a new root. After the process is repeated in the loop, it prints out pre-order traversal.

Function postorder() prints out the result of post-order traversal of the given binary tree. The function makes two lists named stack1 and stack2. Firstly, in the stack1, there is only root. In the loop, it pops the element in stack1 and it appends the popped element in stack2. Next, it appends the left child and right child of popped element in stack1 if the popped element is not leaf node. Then it set the last element of stack1 as a new root. After the process is repeated in

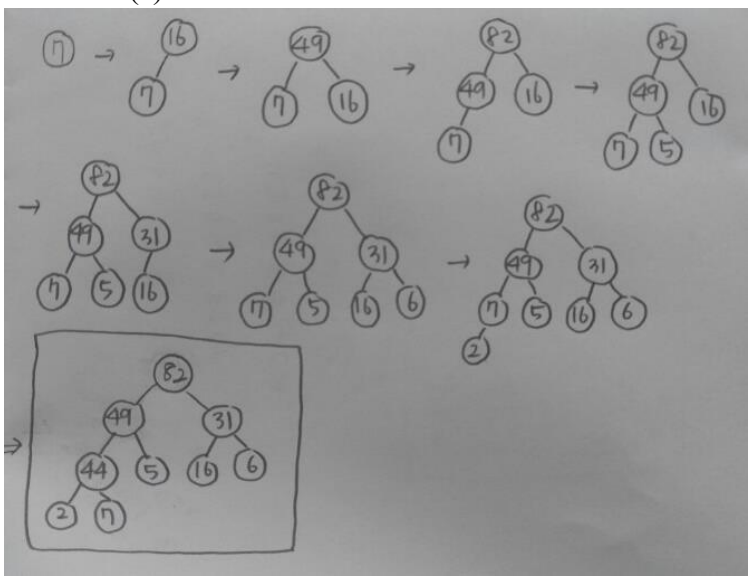
the loop, it prints out post-order traversal.

The below picture shows a binary tree and the result of preorder() and postorder() of that tree. (The example is the binary tree on problem2(b).)

```
[['A'], ['B', 'C'], ['D', 'E', 'F', 'G'], ['H', 'I', None, None, None, None, None, None]]
Pre-order traversal is
A B D H I E C F G

Post-order traversal is
H I D E B F G C A
Process finished with exit code 0
```

Problem4(a).



Problem4(b).