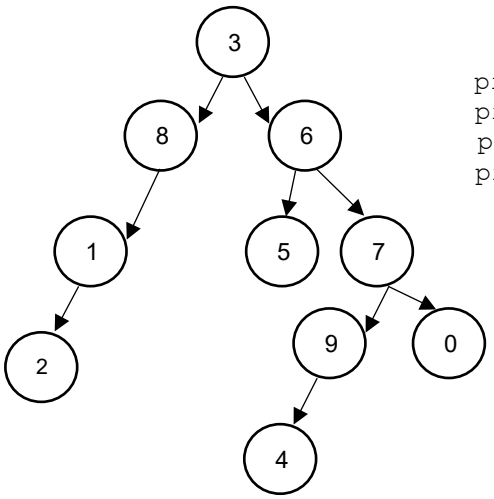
1. (Trees) Write a recursive method that prints the level of a given node in a binary tree. If the node is not found, it does not print anything. Iterative solutions will not be accepted.




```
printLevel(root, 8, 0); // It prints "2"
printLevel(root, 4, 0); // It prints "5"
printLevel(root, 9, 0); // It prints "4"
printLevel(root, 12, 0); // It prints nothing
```

```
struct node
{
  int value;
  struct node *left;
  struct node *right;
};


void printLevel(struct node* root, int number, int level){

if(root==NULL) return;

if(root->value == number)
      printf("%d", level);

printLevel(root->left, number, level+1);
printLevel(root->right, number, level+1);

}
```

2. (Recursion and Performance Analysis) According to the below recursive method, answer the following questions.

```
int recurse(int n)
{
    if (n <= 0)
        return 1;
    else
        return 1 + recurse(n/2);
}
```

a) What does the method return for n=8?

5

b) How many times will your method be called recursively for n=256?

8

c) What is the algorithm complexity of the recursive method in big Oh notation?

O(lgn)

d) What is the total variable space needed for the execution of the method for an input size of 256, if the integer size is 4 bytes?

(4 (for n) + 4 ( for return))*9 = 64 bytes