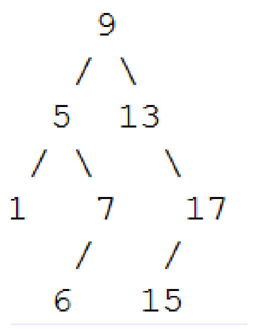
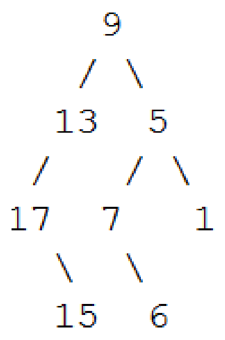
You are given the following tree:



Mirrored:



1. Write the function which traverses the whole tree and has the output: *1 6 7 5 15 17 13 9* – ***1p***
2. Write the function which calculates the size (number of elements) of this tree. – ***1p***
3. Write the function which searches for a node in the tree and outputs 1 if it found it and 0 if it did not. – ***2p***
4. Write the function which creates a mirrored tree from the first tree. – ***2p***
5. Given a binary tree, print out all of its root-to-leaf paths. – ***2p***
6. What is the complexity of the following operations on P.B.B.S.Ts: *get min/max*, *insert*. – ***1p***

Example of input/output:

Function1(root)	1 6 7 5 15 17 13 9
int size = size(root);	Size = 8

int found;
found = search(root, 5);
found ? printf("yes!") : printf("no!");
found = search(root, 10);
found ? printf("yes!") : printf("no!");
	yes!
	no!

mirror(root);
Function1(root);
	15 17 13 6 7 1 5 9

int * paths = (int*)malloc(sizeof(int)*100);
printPaths(root, paths, 0);
	9 13 17 15
	9 5 7 6
	9 5 1