Data Structures and Algorithms

Pierre Collet/ Manal ElZant-Karray

Practical work n°7 : Binary Tree

Exercise 1: Write a function to print all Root-to-Leaf paths in a binary tree.

Algorithm: Steps for print all paths from root to leaf are:

- `If node is null then return 0`
- `Put node->data in array and increment length by 1.`
- `If encountered leaf node (node->left is null and node->right is null) then print array.`
- `Recursively visit left subtree and right subtree`

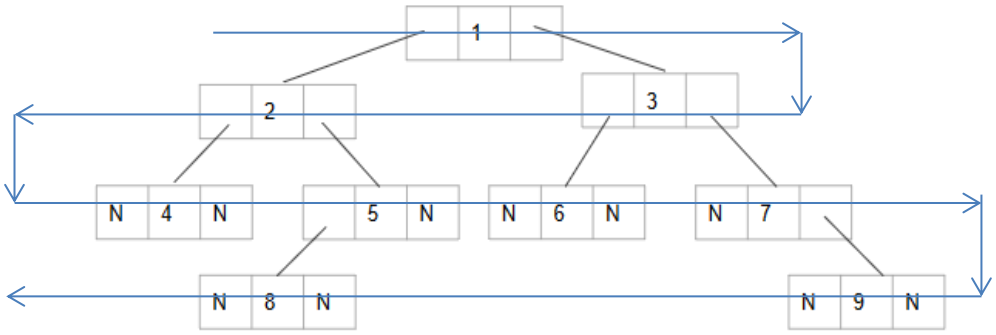
Exercise 2: Write a program to print all ancestors of a given node N in a binary tree. Use a recursive approach to print ancestors of a node.

`Let` **`root`** `be the pointer to the root node of given binary tree.`

- `if root is equal to NULL, return false (node not found).`
- `If root is equal to N, return true (node found).`
- `Recursively search N in left and right sub tree. If N is present in any of the sub tree, then root must be an ancestor of N.`
- `If neither left sub tree nor right sub tree contains N, then N is not ancestor of N.`

Exercise 3: Write a program to checks if a binary tree is max heap or not! Don't forget to check if the binary tree is complete.

Exercise 4: Complete the C program for recursive **level order traversal** in ***spiral form***.



1,3,2,4,5,6,7,9,8

Or

1,2,3,7,6,5,4,8,9

```
#include <stdbool.h>
#include <stdio.h>
#include <stdlib.h>
struct node {
      int data;
      struct node* left;
      struct node* right;
};
// Function to print spiral traversal of a tree
void printSpiral(struct node* root) {
….
}
// Print nodes at a given level
void printGivenLevel(struct node* root, int level, int lt) {
      …
}
int height(struct node* node) {
…
}
struct node* newNode(int data) {
      struct node* node = (struct node*)
      malloc(sizeof(struct node));
      node->data = data;
      node->left = NULL;
      node->right = NULL;
      return (node);
}
int main() {
      struct node* root = newNode(1);
      root->left = newNode(2);
      root->right = newNode(3);
      root->left->left = newNode(4);
      root->left->right = newNode(5);
      root->right->left = newNode(6);
      root->right->right = newNode(7);
      root->right->right->right = newNode(9);
      root->left->right->left = newNode(8);
      printf("The Spiral Order traversal of binary tree is \n");
      printSpiral(root);
      return 0;
}
```

Exercise 4: Write a function to mirror binary tree.