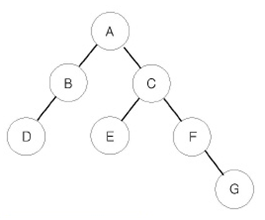
question 1

## **문제**

이진 트리를 입력받아 전위 순회(preorder traversal), 중위 순회(inorder traversal), 후위 순회(postorder traversal)한 결과를 출력하는 프로그램을 작성하시오.



예를 들어 위와 같은 이진 트리가 입력되면,

* 전위 순회한 결과 : ABDCEFG // (루트) (왼쪽 자식) (오른쪽 자식)
* 중위 순회한 결과 : DBAECFG // (왼쪽 자식) (루트) (오른쪽 자식)
* 후위 순회한 결과 : DBEGFCA // (왼쪽 자식) (오른쪽 자식) (루트)

가 된다.

## **입력**

첫째 줄에는 이진 트리의 노드의 개수 N(1≤N≤26)이 주어진다. 둘째 줄부터 N개의 줄에 걸쳐 각 노드와 그의 왼쪽 자식 노드, 오른쪽 자식 노드가 주어진다. 노드의 이름은 A부터 차례대로 영문자 대문자로 매겨지며, 항상 A가 루트 노드가 된다. 자식 노드가 없는 경우에는 .으로 표현된다.

## **출력**

첫째 줄에 전위 순회, 둘째 줄에 중위 순회, 셋째 줄에 후위 순회한 결과를 출력한다. 각 줄에 N개의 알파벳을 공백 없이 출력하면 된다.

answer 1 - good

import java.util.Scanner;

public class T\_1991 {

public static void main(String[] args) {

Scanner scan = new Scanner(System.in);

int number = scan.nextInt();

char[] node = new char[3];

Node root;

node[0] = scan.next().charAt(0);

node[1] = scan.next().charAt(0);

node[2] = scan.next().charAt(0);

root = new Node(node[0]);

root.left = new Node(node[1]);

root.right = new Node(node[2]);

for (int i = 0; i < number - 1; i++) {

node[0] = scan.next().charAt(0);

node[1] = scan.next().charAt(0);

node[2] = scan.next().charAt(0);

input(root, node);

}

potential(root);

System.out.println();

median(root);

System.out.println();

backword(root);

} // main end

private static void potential(Node root) {

System.out.print(root.node);

if(root.left==null){

return;

}

if (root.left.node != '.') {

potential(root.left);

}

if (root.right.node != '.') {

potential(root.right);

}

}

private static void backword(Node root) {

if(root.left==null){

return;

}

if (root.left.node != '.') {

backword(root.left);

}

if (root.right.node != '.') {

backword(root.right);

}

System.out.print(root.node);

}

private static void median(Node root) {

if(root.left==null){

return;

}

if (root.left.node != '.') {

median(root.left);

}

System.out.print(root.node);

if (root.right.node != '.') {

median(root.right);

}

}

private static void input(Node root, char[] node) {

if (root.node == node[0]) {

root.left = new Node(node[1]);

root.right = new Node(node[2]);

}else{

if(root.left==null){

return;

}

if (root.left.node != '.') {

input(root.left, node);

}

if (root.right.node != '.') {

input(root.right, node);

}

}

}

public static class Node {

private char node;

private Node left;

private Node right;

public Node(char node) {

this.node = node;

}

}

}