#include<iostream>

#include<vector>

using namespace std;

typedef struct Point

{

struct Point\* parent = NULL;

vector<Point\*> childs;

char data;

}point;

//typedef struct forestPoint

//{

// struct forestPoint\* parent = NULL;

// struct forestPoint\* child = NULL;

// struct forestPoint\* borther = NULL;

// char data;

//}forestPoint;

void createBinaryTree(Point\*& boot, Point\* parent)

{

char alpha;

cin >> alpha;

if (alpha == '#')

return;

boot = new Point;

Point\* lchild = NULL;

Point\* rchild = NULL;

boot->childs.push\_back(lchild);

boot->childs.push\_back(rchild);

boot->parent = parent;

boot->data = alpha;

createBinaryTree(boot->childs[0], boot);

createBinaryTree(boot->childs[1], boot);

}

void printBinaryTree(Point\* boot)

{

if (boot == NULL)

return;

cout << boot->data;

printBinaryTree(boot->childs[0]);

printBinaryTree(boot->childs[1]);

}

vector<Point\*> forest;

void printForest(Point\* boot)

{

if (boot == NULL)

return;

cout << boot->data;

int index = 0;

for (int x = 0; x < boot->childs.size(); x++)

if (boot->childs[x])

printForest(boot->childs[x]);

}

void binaryTreeToForest(Point\* boot)

{

if (boot->parent == NULL)

{

forest.push\_back(boot);

}

if (boot->childs[1])

{

boot->childs[1]->parent = boot->parent;

if (boot->parent)

{

boot->parent->childs.push\_back(boot->childs[1]);

}

Point\* temp = boot->childs[1];

boot->childs[1] = NULL;

binaryTreeToForest(temp);

}

if (boot->childs[0])

{

binaryTreeToForest(boot->childs[0]);

}

return;

}

int main()

{

Point\* boot = NULL;

createBinaryTree(boot, NULL);

binaryTreeToForest(boot);

for (auto x : forest)

{

printForest(x);

cout << endl;

}

return 0;

}

前序输入

