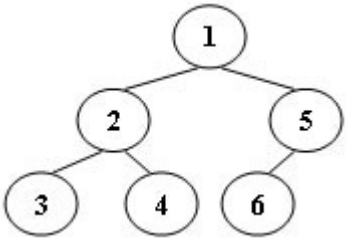
青少年软件编程（C 语言）等级考试试卷（七级）

1、模拟树遍历

二叉树的中序遍历可以借助一个堆栈来用非递归的方式实现。例如，对一棵有 6 个结点的二叉树（结点键值从 1 到 6）进行遍历，堆栈操作为：push(1); push(2); push(3); pop(); pop(); push(4); pop(); pop(); push(5); push(6); pop(); pop() —— 其中 push 为入栈，pop 为出栈。则这套操作对应了一棵唯一的二叉树，如下图所示。



你的任务是输出这棵树的后序遍历序列。

时间限制：1000
内存限制：262144

输入
输入第一行给出一个正整数 N（≤ 30），是二叉树中结点的个数（结点键值从 1 到 N）。随后 2N 行，每行给出一个堆栈操作：`Push X` 表示将键值为 `X` 的结点入栈，`Pop` 表示将一个结点出栈。

输出
在一行中输出该树后序遍历的序列。数字间以 1 个空格分隔，行首尾不得有多余空格。裁判保证输入数据一定对应了一棵树。

样例输入

```
6
Push 1
Push 2
Push 3
Pop
Pop
Push 4
Pop
Pop
```

```
Push 5
Push 6
Pop
Pop
```

样例输出

```
3 4 2 6 5 1
```

参考程序：

```
#include <iostream>
#include <stack>
#include <vector>

using namespace std;

void postorderTraversal(vector<int>& inorder, int start, int end) {
    if (start > end) return;
    int root = inorder[end];
    int index = end;
    for (int i = end - 1; i >= start; i--) {
        if (inorder[i] < root) {
            index = i;
            break;
        }
    }
    postorderTraversal(inorder, start, index);
    postorderTraversal(inorder, index + 1, end - 1);
    cout << root << " ";
}

int main() {
    int n;
    cin >> n;
    stack<int> s;
    vector<int> inorder;
    for (int i = 0; i < 2 * n; i++) {
        string operation;
        int value;
        cin >> operation;
```

```
        if (operation == "Push") {
            cin >> value;
            s.push(value);
        } else {
            value = s.top();
            s.pop();
            inorder.push_back(value);
        }
    }
    postorderTraversal(inorder, 0, n - 1);
    cout << endl;
    return 0;
}
```

2、寻宝图

给定一幅地图，其中有水域，有陆地。被水域完全环绕的陆地是岛屿。有些岛屿上埋藏有宝藏，这些有宝藏的点也被标记出来了。本题就请你统计一下，给定的地图上一共有多少岛屿，其中有多少是有宝藏的岛屿。

时间限制：1000
内存限制：262144

输入
输入第一行给出 2 个正整数 N 和 M（$1 < N \times M \le 10^5$），是地图的尺寸，表示地图由 N 行 M 列格子构成。随后 N 行，每行给出 M 位个位数，其中 `0` 表示水域，`1` 表示陆地，`2`-`9` 表示宝藏。 注意：两个格子共享一条边时，才是“相邻”的。默认地图外围全是水域。

输出
在一行中输出 2 个整数，分别是岛屿的总数量和有宝藏的岛屿的数量。

样例输入

```
10 11
01000000151
11000000111
00110000811
00110100010
00000000000
00000111000
```

```
00114111000
00110010000
00019000010
00120000001
```

样例输出

```
7 2
```

参考程序：

```
#include <iostream>
#include <vector>

using namespace std;

int n, m;
vector<vector<int>> grid;

void dfs(int i, int j) {
    if (i < 0 || i >= n || j < 0 || j >= m || grid[i][j] == 0)
        return;
    grid[i][j] = 0;
    dfs(i - 1, j);
    dfs(i + 1, j);
    dfs(i, j - 1);
    dfs(i, j + 1);
}

int countIslands() {
    int islands = 0;
    for (int i = 0; i < n; i++) {
        for (int j = 0; j < m; j++) {
            if (grid[i][j] > 0) {
                dfs(i, j);
                islands++;
            }
        }
    }
    return islands;
}
```

```
int countTreasureIslands() {
    int treasureIslands = 0;
    for (int i = 0; i < n; i++) {
        for (int j = 0; j < m; j++) {
            if (grid[i][j] > 1) {
                dfs(i, j);
                treasureIslands++;
            }
        }
    }
    return treasureIslands;
}

int main() {
    cin >> n >> m;
    grid.resize(n, vector<int>(m));
    for (int i = 0; i < n; i++) {
        for (int j = 0; j < m; j++) {
            char c;
            cin >> c;
            grid[i][j] = c - '0';
        }
    }
    int totalIslands = countIslands();
    int treasureIslands = countTreasureIslands();
    cout << totalIslands << " " << treasureIslands << endl;
    return 0;
}
```

3、小字辈

本题给定一个庞大家族的家谱，要请你给出最小一辈的名单。

时间限制：1000
内存限制：262144

输入
输入在第一行给出家族人口总数 N（不超过 100 000 的正整数） —— 简单起见，我们把家族成员从 1 到 N 编号。随后第二行给出 N 个编号，其中第 i 个编号对应第 i 位成员的父/母。家谱中辈分最高的老祖宗对应的父/母编号为 -1。一行中的数字间以空格分隔。

输出
首先输出最小的辈分（老祖宗的辈分为 1，以下逐级递增）。然后在第二行按递增顺序输出辈分最小的成员的编号。编号间以一个空格分隔，行首尾不得有多余空格。

样例输入

```
9
2 6 5 5 -1 5 6 4 7
```

样例输出

```
4
1 9
```

4、堆中的路径

将一系列给定数字插入一个初始为空的小顶堆`H[]`。随后对任意给定的下标`i`，打印从`H[i]`到根结点的路径。

时间限制：1000
内存限制：262144

输入
每组测试第 1 行包含 2 个正整数 N 和 M(≤ 1000)，分别是插入元素的个数、以及需要打印的路径条数。下一行给出区间[-10000, 10000]内的 N 个要被插入一个初始为空的小顶堆的整数。最后一行给出 M 个下标。

输出
对输入中给出的每个下标`i`，在一行中输出从`H[i]`到根结点的路径上的数据。数字间以 1 个空格分隔，行末不得有多余空格。

样例输入

```
5 3
46 23 26 24 10
5 4 3
```

样例输出

```
24 23 10
46 23 10
26 10
```