浙江大学2023-2024学年春夏学期 《数据结构基础》课程期末考试试卷 已结束

判断题 10 单选题 20 程序填空题 2 函数题 1

概览 题目列表 提交列表 排名

函数题 6-1-1 Height of Binary Search Tree 上一题 下一题

题目描述

6-1-1 Height of Binary Search Tree

分数 8

全屏浏览 作者 杨子祺 单位 浙江大学

You are supposed to write two functions to calculate the height of a binary search tree and find all the longest paths, respectively.

Format of struct:

The binary search tree is defined as follows:

```
typedef struct TreeNode *PtrTreeNode;
typedef struct TreeNode {
int key;
struct TreeNode *left;
struct TreeNode *right;
} TreeNode;
typedef PtrTreeNode BinarySearchTree;
```

Format of functions:

The definitions of the two functions you need to write are as follows:

```
int GetHeight(BinarySearchTree root);
void GetPath(BinarySearchTree root, int path[MAXN][MAXN], int height);
```

`root` is the root of a binary search tree. `height` is the height of the binary search tree. `path` is a two-dimensional array that stores all the longest paths, where the first dimension represents the paths and the second dimension represents the node keys in the paths.
For each test case, the function `GetHeight` returns the height of the binary search tree, and the function `GetPath` finds all the longest paths and stores them in the two-dimensional array `path`.
Note:

1. Each path is stored in order from the root node to the leaf node.
2. The storage order of different paths is not limited.
3. The key of each node is a positive integer.
4. The program will take the preorder sequence of the binary search tree as the input.
5. The height of the binary search tree is the number of edges in the longest path.

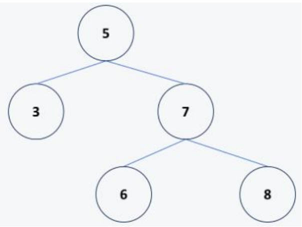
Sample program of judge:

```
#include <stdio.h>
#include <stdlib.h>
#define MAXN 150

typedef struct TreeNode *PtrTreeNode;
typedef struct TreeNode {
  int key;
  struct TreeNode *left;
  struct TreeNode *right;
} TreeNode;
typedef PtrTreeNode BinarySearchTree;

int GetHeight(BinarySearchTree root);
void GetPath(BinarySearchTree root, int path[MAXN][MAXN], int height);
void PrintPath(int path[MAXN][MAXN]);
BinarySearchTree BuildBST(int tree[], int size);

int main() {
  int treeArray[MAXN], N, i;
  scanf("%d", &N);
  for (i = 0; i < N; i++) scanf("%d", &treeArray[i]);
  int path[MAXN][MAXN] = {0};
  BinarySearchTree root = BuildBST(treeArray, N);
  int height = GetHeight(root);
  GetPath(root, path, height);
  printf("%d\n", height);
  PrintPath(path);
  return 0;
}
/* Your function will be put here */
```



Sample Input:

```
5
5 3 7 6 8
```

Sample Output:

```
2
5 7 6
5 7 8
```

The storage in the two-dimension array can be visualized as follows.

5	7	6	0	......	0
5	7	8	0	......	0
......					
0	0	0	0	......	0

代码长度限制	16 KB
时间限制	400 ms
内存限制	64 MB

C (gcc)

```
int GetHeight(BinarySearchTree root){
    if(root==NULL)
        return -1;
    int m=GetHeight(root->left);
    int n=GetHeight(root->right);
    return (m>n)? (m+1):(n+1);
}
void GetPath(BinarySearchTree root, int path[MAXN][MAXN], int height){
    printf("5 7 6\n");
    printf("5 7 8\n");
}
```

测试用例