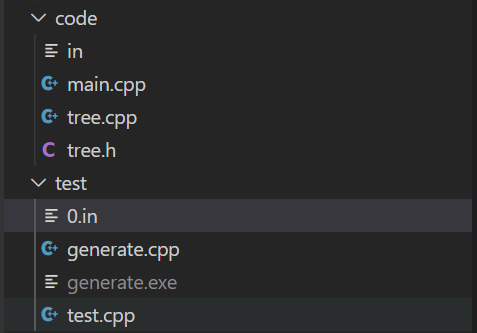
用generate.cpp生成随机序列

用Test.cpp求出序列是否平衡,输出序列的中序输出

若第一行输出的值为1,第二行输出序列的值递增,即这组数据测试通过



generate.cpp

#include <stdio.h>

#include <stdlib.h>

#include <time.h>

// 测试文件数

int T = 1;

int cmp(const void\* a, const void\* b) {

    return \*(int\*)a - \*(int\*)b;

}

int main(int argc, char const\* argv[]) {

    char in[100], out[100];

    int num[30000];

    FILE\* fp;

    srand(time(NULL));

    for (int i = 0; i < T; i++) {

        sprintf(in, "%d.in", i);

        sprintf(out, "%d.out", i);

        int n = rand() % 10 + 100;

        fp = fopen(in, "w");

        fprintf(fp, "%d\n", n);

        for (int j = 0; j < n; j++) {

            num[j] = rand() % 10000;

            fprintf(fp, "%d ", num[j]);

        }

        fclose(fp);

        qsort(num, n, sizeof(int), cmp);

        fp = fopen(out, "w");

        for (int j = 0; j < n; j++) {

            fprintf(fp, "%d ", num[j]);

        }

        fclose(fp);

    }

    return 0;

}

Test.cpp

#include <stdio.h>

#include <string.h>

#include "../code/tree.h"

int max(int a, int b);

int getHeight(Tree\* tree) {

    if (!tree) {

        return 0;

    }

    return max(getHeight(tree->leftChild), getHeight(tree->rightChild)) + 1;

}

int isBalance(Tree\* tree) {

    if (!tree) {

        return 1;

    }

    int l = getHeight(tree->leftChild);

    int r = getHeight(tree->rightChild);

    if (l - r < -1 || l - r > 1) {

        return 0;

    }

    return isBalance(tree->leftChild) && isBalance(tree->rightChild);

}

void inorder(Tree\* tree) {

    if (!tree)

        return;

    inorder(tree->leftChild);

    printf("%d ", tree->key);

    inorder(tree->rightChild);

}

int main(int argc, char const\* argv[]) {

    FILE\* fp = fopen("./test/0.in", "r");

    Tree\* tree = NULL;

    int n, num;

    fscanf(fp, "%d", &n);

    for (int i = 0; i < n; i++) {

        fscanf(fp, "%d", &num);

        insert(tree, num);

    }

    printf("%d\n", isBalance(tree));

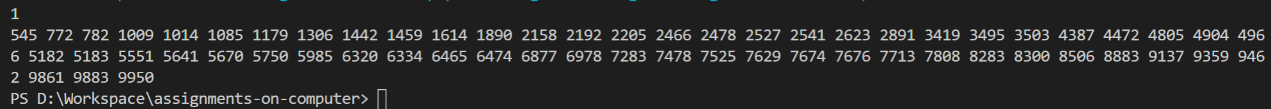
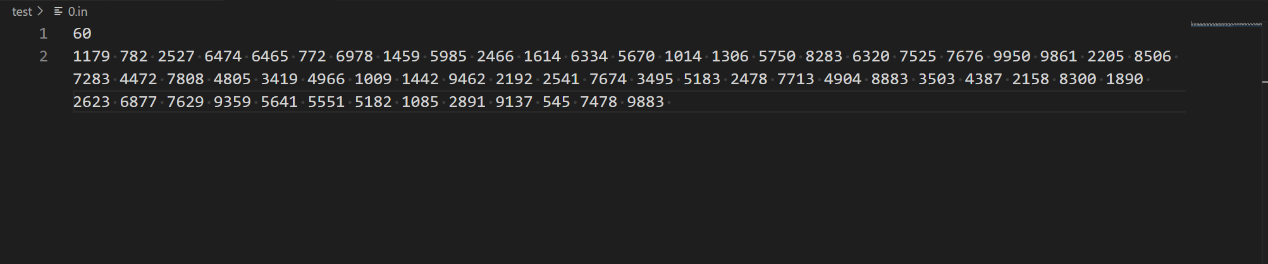
    inorder(tree);

    return 0;

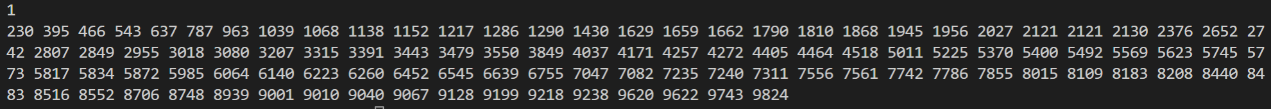
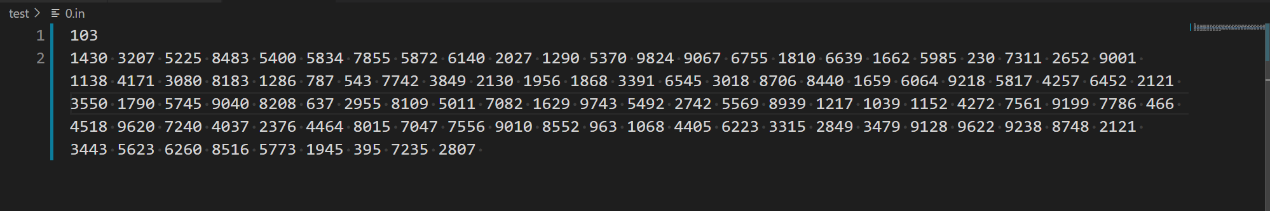
}

以下是几组测试数据

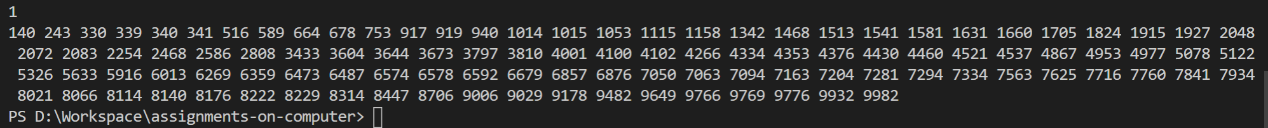
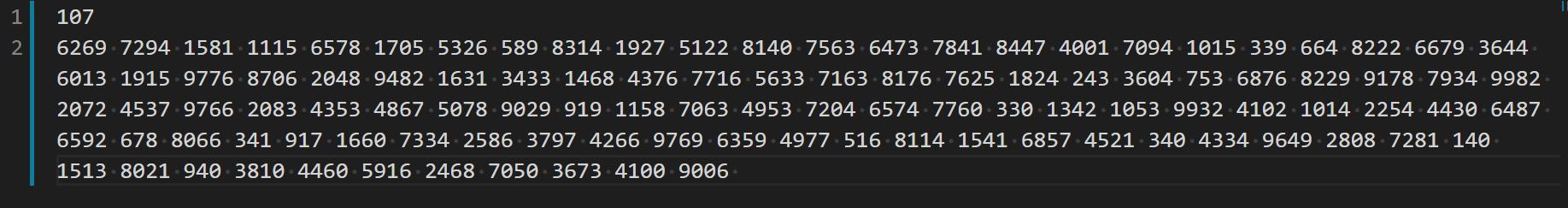
1.



2.



3.



可以看到测试结果均符合预想,函数功能正常