**要求代码和实验报告规范，在算法思想中：对实验涉及的数据结构进行有效设计和分析；对算法进行分析并给出时间、空间复杂度的结论；清晰表达实验思路、出现的问题及解决方法。**

一、调试成功程序及说明

1、

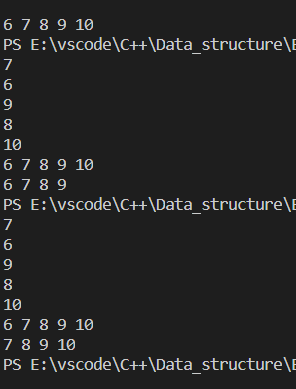
题目：实现二叉排序树的插入和删除。

算法思想：

（1）插入的思想比较简单，所有插入的结点都为叶子结点，也就是说插入一个数据，从跟结点开始向左右结点进行比较，如果小于就往左，大于就往右。

（2）删除的思想略微复杂，分为以下三种情况。①删除叶子结点，直接进行删除②有一棵子树的情况，直接另当前的树指针指向子树，释放这个结点。③具有两棵子树的情况，把右子树最左边的结点与当前的结点的值进行交换，然后删除右子树最左边的结点。

运行结果：



结果分析：运行结果正确，能够正确执行插入和删除操作。

附源程序。

#include <iostream>

#include <cstdlib>

#include <malloc.h>

using namespace std;

#define ElemType int

typedef struct node

{

ElemType data;

struct node \*lchild;

struct node \*rchild;

}\*BinarySortTree, BSNode;

BinarySortTree FindMin(BinarySortTree &BST)

{

if(BST == NULL)

{

return NULL;

}

BinarySortTree cur = BST;

while(cur != NULL)

{

if(cur->lchild ==NULL)

{

return cur;

}

else

{

cur = cur->lchild;

}

}

}

//寻找最小值，返回指针

BinarySortTree InsertBSTree(BinarySortTree &BST, ElemType data)

{

if(BST == NULL)

{

BST = new BSNode;

BST->data = data;

BST->lchild = NULL;

BST->rchild = NULL;

cout << BST->data<<endl;

}

else

{

if(data < BST->data)

{

BST->lchild = InsertBSTree(BST->lchild,data);

}

else if(data > BST->data)

{

BST->rchild = InsertBSTree(BST->rchild,data);

}

}

return BST;

}

//插入一个新的，值为data的结点

BinarySortTree DeleteNode(BinarySortTree &BST, ElemType data)

{

if(BST == NULL)

{

return BST;

}

else if(data < BST->data)

{

BST->lchild = DeleteNode(BST->lchild,data);

}

else if(data > BST->data)

{

BST->rchild = DeleteNode(BST->rchild,data);

}

else

{

if(BST->lchild != NULL && BST->rchild != NULL)

{

BinarySortTree tmp = FindMin(BST->rchild); //找到右子树的最小值，用来代替根节点

BST->data = tmp->data;

BST->rchild = DeleteNode(BST->rchild,tmp->data);

}

else

{

BinarySortTree tmp = BST;

if(tmp->rchild == NULL)

{

BST = BST->lchild;

}

else if(tmp->rchild == NULL)

{

BST = BST->rchild;

}

delete tmp;

}

}

return BST;

}

void InOrderTraverse(BinarySortTree &BST)

{

if(BST == NULL)

{

return ;

}

InOrderTraverse(BST->lchild);

cout << BST->data<<" ";

InOrderTraverse(BST->rchild);

return;

}

int main()

{

BinarySortTree BST = NULL,temp = NULL;

int a[30] = {-1,7,6,9,8,10,5,2,0,10,11};

for(int i = 1; i <= 5; i++)

{

InsertBSTree(BST,a[i]);

}

InOrderTraverse(BST);

int t = 6;

temp=FindMin(BST);

DeleteNode(BST,t);

cout << endl;

//cout << temp->data<<endl;

InOrderTraverse(BST);

return 0;

}

2、

题目：

算法思想：

运行结果：

结果分析：

附源程序。

......

3、

题目：

算法思想：

运行结果：

结果分析：

附源程序。

4、

题目：

算法思想：

运行结果：

结果分析：

附源程序。

5、

题目：

题目背景：

开学了，可是校园里堆积了不少垃圾杂物。

热心的同学们纷纷自发前来清理，为学校注入正能量～

题目描述：

通过无人机航拍我们已经知晓了n处尚待清理的垃圾位置，其中第i (1≤i≤n)处的坐标为(xi,yi)，保证所有的坐标均为整数。

我们希望在垃圾集中的地方建立些回收站。具体来说，对于一个位置(x,y)是否适合建立回收站，我们主要考虑以下几点：

1. (x,y)必须是整数坐标，且该处存在垃圾；
2. 上下左右四个邻居位置，即(x,y+1)、(x,y-1)、(x+1,y)和(x-1,y)处，必须全部存在垃圾；
3. 进一步地，我们会对满足上述两个条件的选址进行评分分数为不大于4的自然数，表示在(x±1,y±1)四个对角位置中有几处存在垃圾。

现在，请你统计一下每种得分的选址个数。

算法思想：按照要求，统计每个有效点周围的有效点个数，决定这个点是否可以选择为垃圾站，如果可以的话，就计算这个点的得分，最后统计答案。

运行结果：



结果分析：运行结果正确，通过CSP模拟测试。

附源程序。

#include <iostream>

#include <cstdlib>

using namespace std;

typedef struct node

{

int x;

int y;

int flag;

int value;

}Point;

int n;

Point spot[1005];

int ans[5];

bool check(Point &p)

{

for(int i = 1; i <= n ;i++)

{

if(spot[i].x == p.x - 1 && spot[i].y == p.y)

{

p.flag ++;

}

if(spot[i].x == p.x + 1 && spot[i].y == p.y)

{

p.flag ++;

}

if(spot[i].x == p.x && spot[i].y == p.y - 1)

{

p.flag ++;

}

if(spot[i].x == p.x && spot[i].y == p.y + 1)

{

p.flag ++;

}

if(spot[i].x == p.x - 1 && spot[i].y == p.y - 1)

{

p.value ++;

}if(spot[i].x == p.x + 1 && spot[i].y == p.y + 1)

{

p.value ++;

}if(spot[i].x == p.x - 1 && spot[i].y == p.y + 1)

{

p.value ++;

}if(spot[i].x == p.x + 1 && spot[i].y == p.y - 1)

{

p.value ++;

}

}

if(p.flag == 4)

{

return true;

}

else

{

return false;

}

}

int main()

{

//freopen("5.txt","r",stdin);

cin >> n;

for(int i = 1; i <= n; i++)

{

cin >> spot[i].x >> spot[i].y;

}

for(int i = 1; i <= n; i++)

{

if(check(spot[i]))

{

ans[spot[i].value] ++;

}

}

for(int i = 0; i < 5; i++)

{

cout << ans[i] <<endl;

}

return 0;

}

二、未调试成功程序及说明

1、

题目：

算法思想：

错误原因：

附源程序。

2、

题目：

算法思想：

错误原因：

附源程序。

......

三、代码行数及小结