哈尔滨工业大学计算机科学与技术学院

实验报告

课程名称： 数据结构与算法

课程类型： 必修

实验项目名称：BST 储存结构的建立（插入）、删除、查找算法的实现与应用

实验题目： 查找结构与排序算法

班级： 1303101

学号： 1130310128

姓名： 杨尚斌

设计成绩	报告成绩	指导老师
		张岩

一、 实验目的

1. 熟悉 BST 储存结构.

2. 掌握 BST 储存结构的建立（插入）、删除、查找算法的实现与应用.

3. 增强编程能力.

二、实验要求及实验环境

1.实验要求

(1) 设计 BST 的左右链存储结构；

(2) 实现 BST 左右链存储结构上的插入（建立）、删除、查找和排序算法。

(3) 利用 BST 结构和相应的操作算法，实现班级学习成绩管理（单科成绩管理，排名；加权绩点管理与排名等）

(4) 学生的基础成绩信息以文件形式保存；学生基础成绩信息和排名信息以文件形式存储；并能显示到屏幕。

2.测试环境

操作系统： windows x64 sp1

编译器： g++ 4.7.1

二、 设计思想

1. 逻辑设计

```
Struct Tree
{
    int data;
    Tree *lchild, *rchild;
};
Typedef Tree *BST;
```

//Tree 的数据结构，节点元素 data 和左右链接指针 lchild 与 rchild。

```
Void Insert(int R, BST *F)
```

//在树中插入新的节点，用来插入元素和 Create 树。

```
Void Create(BST *F)
```

//Create 树，需要用到 Insert 的函数

```
Void Display(BST F)
```

//展示当前树的内容，基本上在每个阶段都会调用

```
int Search(int k, BST F)
```

//寻找 K 是否在树中，如果是，返回地址，不是返回 0

```
BST Delete(int k, BST F)
```

//先查找 F 中是否有 k，有的话删掉

```
Void menu()
```

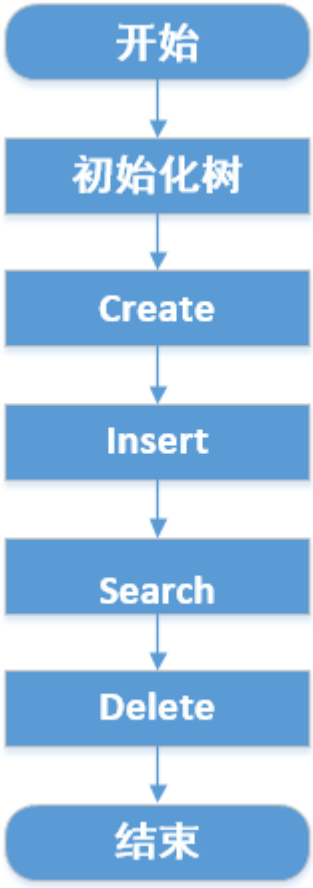
//主功能界面函数

Int main()

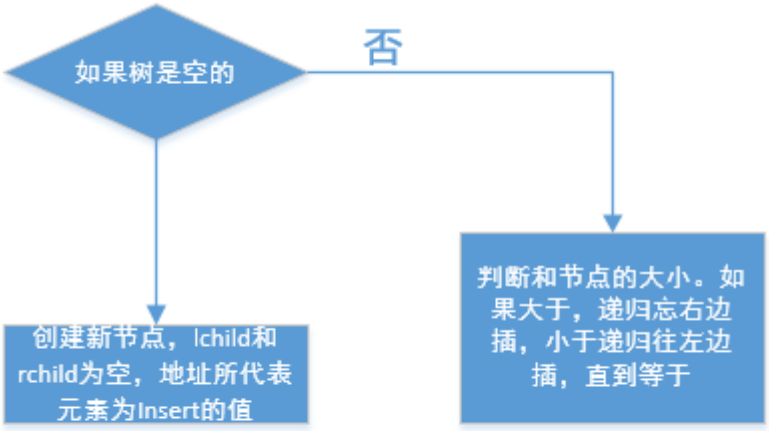
//主函数

2. 物理设计

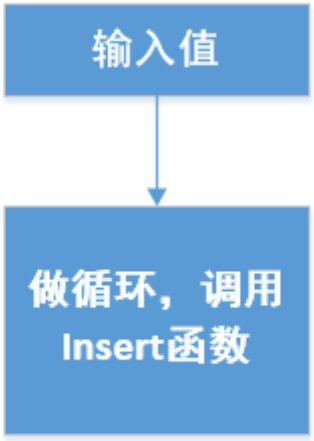
1） 整体设计



2） Insert

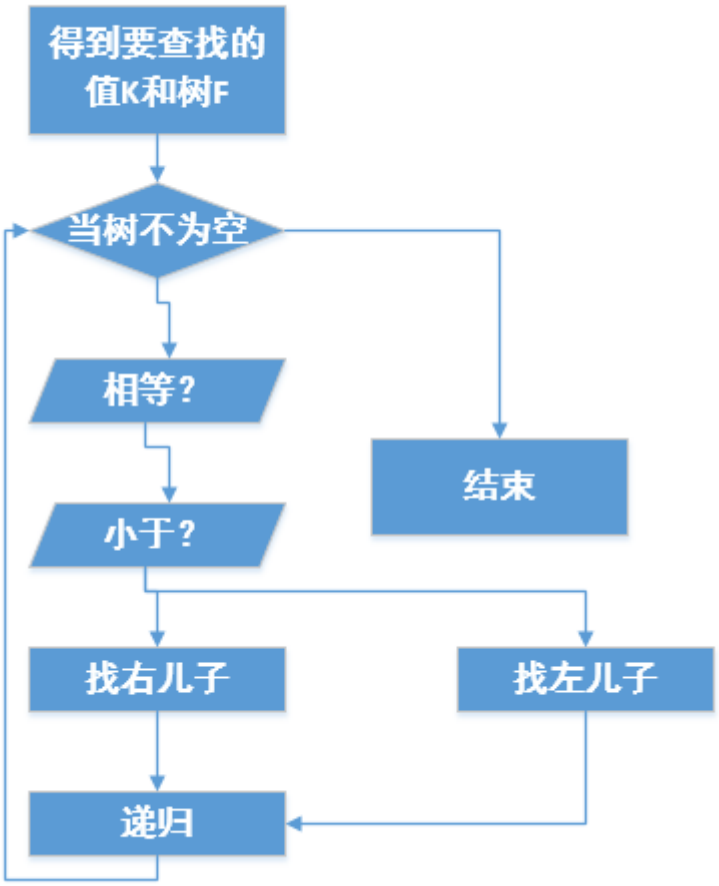


3） Create

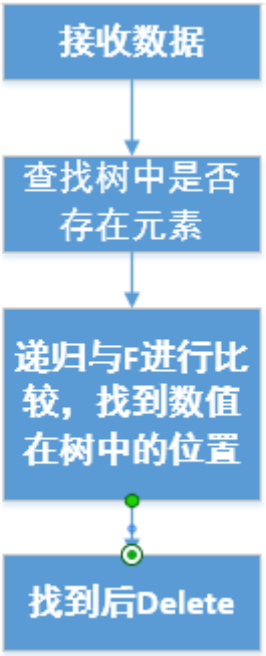


4） Display

5） **Search**

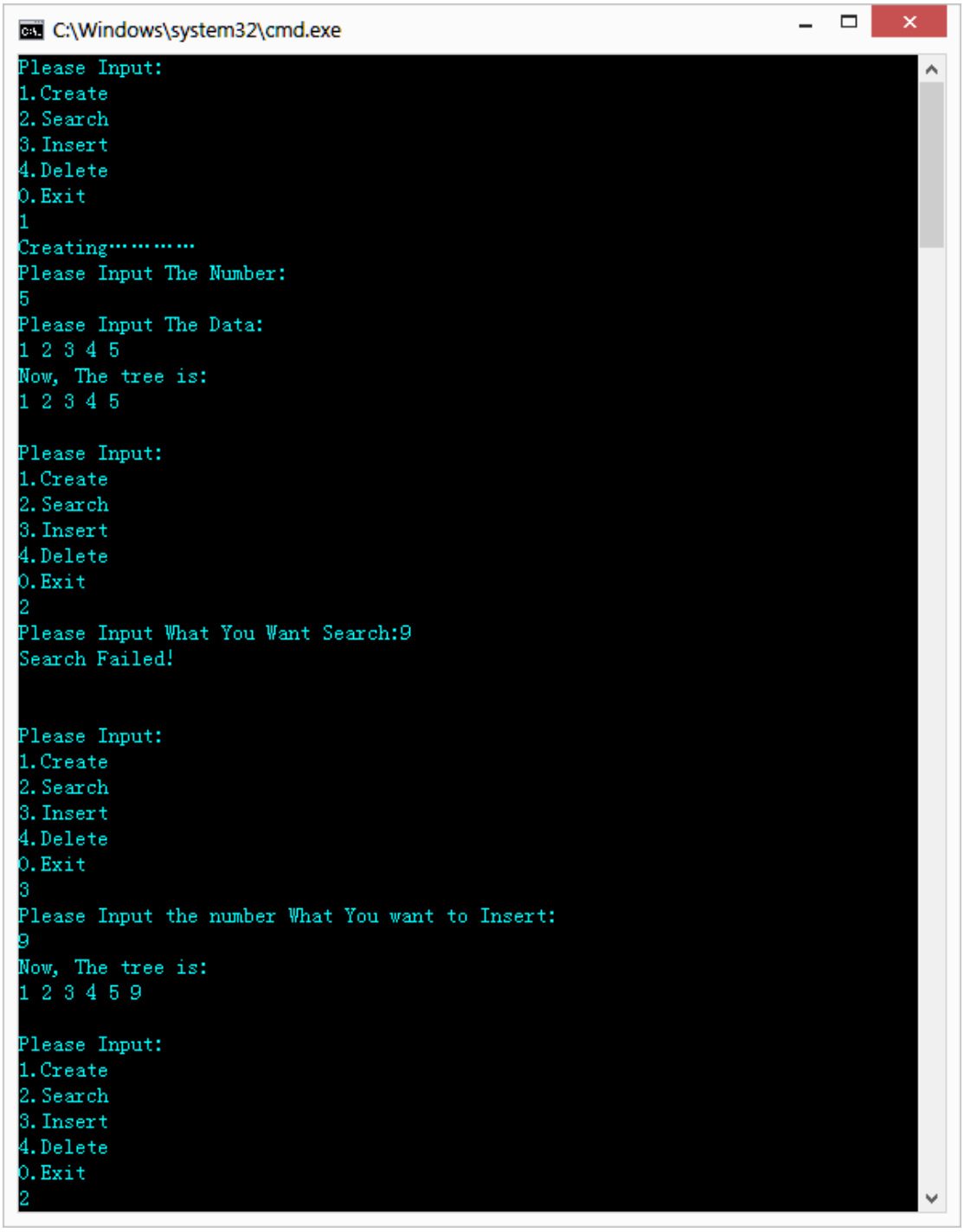


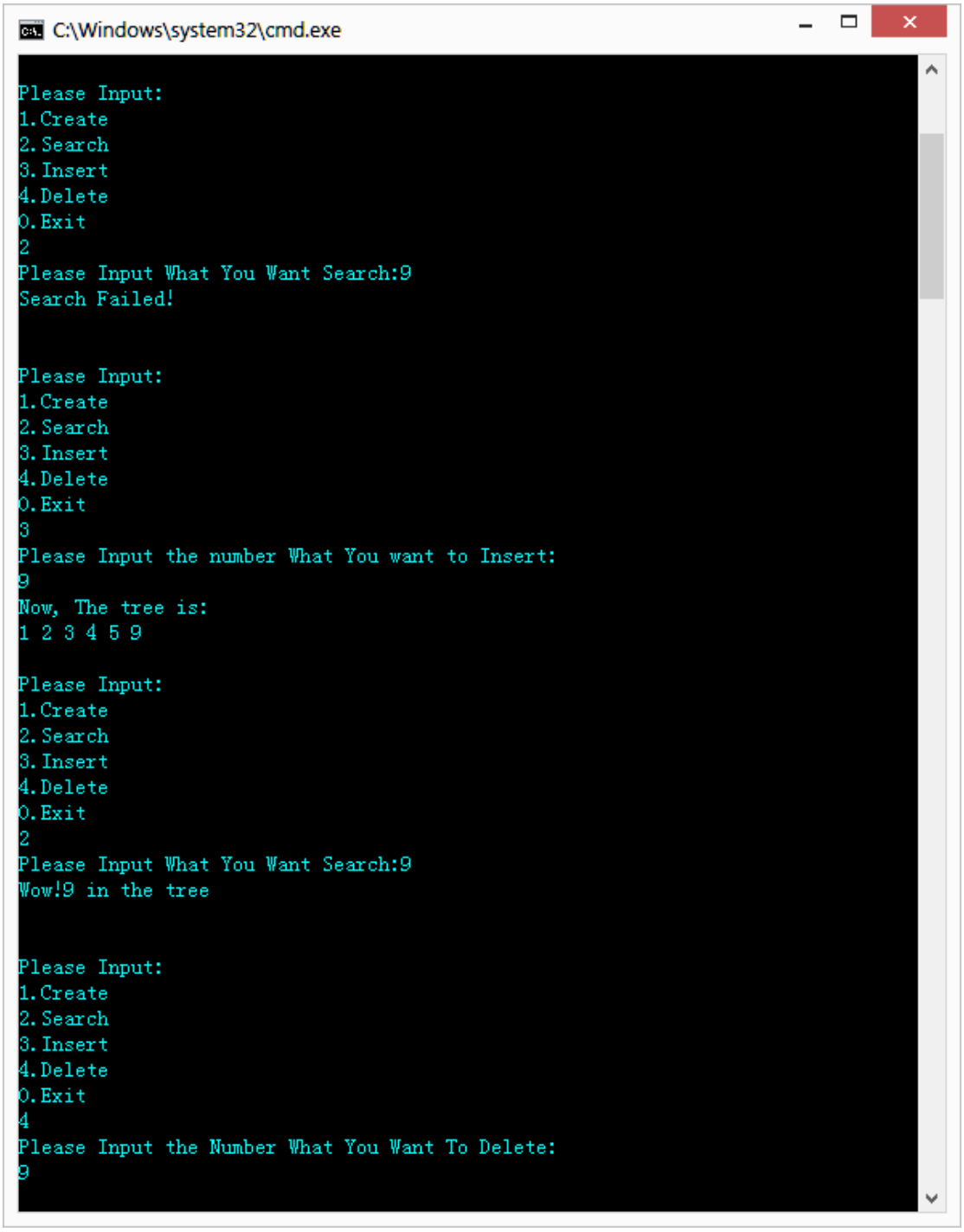
6） **Delete**



三、 测试结果

测试数据：1 2 3 4 5


C:\Windows\system32\cmd.exe
Please Input:
1.Create
2.Search
3.Insert
4.Delete
0.Exit
1
Creating……
Please Input The Number:
5
Please Input The Data:
1 2 3 4 5
Now, The tree is:
1 2 3 4 5
Please Input:
1.Create
2.Search
3.Insert
4.Delete
0.Exit
2
Please Input What You Want Search:9
Search Failed!
Please Input:
1.Create
2.Search
3.Insert
4.Delete
0.Exit
3
Please Input the number What You want to Insert:
9
Now, The tree is:
1 2 3 4 5 9
Please Input:
1.Create
2.Search
3.Insert
4.Delete
0.Exit
2



四、系统不足与经验体会

系统不足：

程序的健壮性存在一定的问题，例如当输入的二叉查找树中的值是相等的时候，将产生比较严重的错误，直接造成程序的退出。

程序的结构存在一定的问题，在整个代码中，重复使用了一些代码，使代码量比较大，尤其表现在 Delete 阶段，感觉做的不是很好，还可以进行进一步的优化。

经验体会：

二叉查找树是一种新的结构类型，以树的形式对数据进行了比较友好的排序。在以后的查找过程中极大了减少了比较繁忙的数据运算，在很大的程度上优化了数据查找的代码。

树贯彻着整个数据结构，在这个实验中，数据结构的学习也很快就告一段落了，但是他的思想始终伴随在我们左右，相信在后面会遇到更多的树的问题需要解决。

在数据结构的实验中实现书上的代码，可以在很高的程序上提高编程能力，加强对指针，c++，多样数据结构的的理解。

六、附录：源代码（带注释）

```
/*
admin:Shangbin Yang
Title:查找结构与排序算法
data:2014-12-04
*/
#include <iostream>
using namespace std;
/*
Define a Tree struction with data, lchild pointer and rchild pointer
*/
struct Tree
{
    int data;
    Tree *lchild, *rchild;
};
typedef Tree *BST;
/*
Insert the R to the Tree by size
```

```
*/
void Insert(int R, BST *F)
{
    BST S;
    //The tree is null, Create a new tree, and do some init
    if (*F == NULL)
    {
        S = new Tree;
        S->data = R;
        S->lchild = NULL;
        S->rchild = NULL;
        *F = S;
    }
    //By size, insert the R
    else if (R < (*F)->data)
    {
        Insert(R, &((*F)->lchild));
    }
    else if (R >(*F)->data)
    {
        Insert(R, &((*F)->rchild));
    }
}
//Create the tree, include [void Insert(int R, BST *F)]
void Create(BST *F)
{
    int number, R;
    *F = NULL;
    cout << "Please Input The Number:" << endl;
    cin >> number;
    cout << "Please Input The Data:" << endl;
    //by the void Insert(int R, BST *F)
    for (int i = 0; i < number; i++)
    {
        cin >> R;
        Insert(R, F);
    }
}
/*
Display the Tree, to show the tree any time
*/
void Display(BST F)
{
    if (F != NULL)
```

```
    {
        Display(F->lchild);
        cout << F->data << " ";
        Display(F->rchild);
    }
}
/*
Search the k from the Tree
*/
int Search(int k, BST F)
{
    BST p;
    p = F;
    while (p)
    {
        if (p->data == k)
        {
            return p->data;
        }
        if (p->data > k)
        {
            p = p->lchild;
        }
        else
            p = p->rchild;
    }
    return 0;
}
/*
Delete the k from the Tree
Should judge k in the Tree or not
*/
BST  Delete(int k, BST F)
{
    Tree *p, *f, *s, *q;
    p = F;
    f = NULL;
    if (!Search(k, F))
    {
        cout << "The " << k << " was not in the Tree !" << endl;
        return F;
    }
    while (p)
    {
```

```
    if (p->data == k)
        f = p;
    if (p->data > k)
        p = p->lchild;
    if (p->data < k)
        p = p->rchild;
}
if (p == NULL)
{
    cout << "Search Failed!" << endl;
    return F;
}
//Delete the k
if (p->lchild == NULL)
{
    if (f == NULL)
    {
        F = p->rchild;
    }
    else if (f->lchild == p)
    {
        f->lchild = p->lchild;
    }
    else
        f->rchild = p->rchild;
    delete p;
}
else
{
    q = p;
    s = p->lchild;
    while (s->rchild)
    {
        q = s;
        s = s->rchild;
    }
    if (q == p)
    {
        q->lchild = s->lchild;
    }
    else
    {
        q->rchild = s->rchild;
    }
```

```
        p->data = s->data;
        delete s;
    }
}
//Choose Menu
void menu()
{
    cout << "1.Create" << endl;
    cout << "2.Search" << endl;
    cout << "3.Insert" << endl;
    cout << "4.Delete" << endl;
    cout << "0.Exit" << endl;
}
int main()
{
    BST F = NULL;
    int n, t;
    int ch;
    cout << "Please Input:" << endl;
    menu();
    cin >> n;
    while (!(n >= 0 && n <= 4))
    {
        cout << "Error! Please Input Again: " << endl;
        menu();
        cin >> n;
    }
    while (n != 0)
    {
        switch (n)
        {
        case 1:
            cout << "Creating……" << endl;
            Create(&F);
            cout << "Now, The tree is:" << endl;
            Display(F);
            cout << endl;
            cout << endl;
            cout << "Please Input:" << endl;
            menu();
            cin >> n;
            break;
        case 2:
            cout << "Please Input What You Want Search:";
```

```
            cin >> ch;
            t = Search(ch, F);
            if (t != 0)
            {
                cout <<"Wow!" << t << " in the tree" << endl;
            }
            else
            {
                cout << "Search Failed!" << endl;
            }
            cout << endl;
            cout << endl;
            cout << "Please Input:" << endl;
            menu();
            cin >> n;
            break;
        case 3:
            cout << "Please Input the number What You want to Insert:"
<< endl;
            cin >> ch;
            Insert(ch, &F);
            cout << "Now, The tree is:" << endl;
            Display(F);
            cout << endl;
            cout << endl;
            cout << "Please Input:" << endl;
            menu();
            cin >> n;
            break;
        case 4:
            cout << "Please Input the Number What You Want To Delete:"
<< endl;
            cin >> ch;
            Delete(ch, F);
            cout << "Now, The tree is:" << endl;
            Display(F);
            cout << endl;
            cout << endl;
            cout << "Please Input:" << endl;
            menu();
            cin >> n;
            break;
        }
    }
```