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

试验项目名称：树型结构及应用

试验题目：二叉树的建立及遍历、哈夫曼编码

班级：1603006

学号：1160800724

姓名：池嘉洋

二叉树的建立及遍历

一、实验目的：

通过前序及中序表达式建立二叉树，加深对于二叉树结构的了解，同时深入理解数的前序和中序遍历与二叉树节点位置的关系，由此建立二叉树。深入理解各种不同遍历的特点，及其函数功能的实现。发现并了解函数递归与非递归的实现之间的共同点与差异。

二、实验要求及实验环境：

实验要求：

根据用户输入的二叉树的前序以及中序遍历序列，由这两个序列建立二叉树，并将建好的二叉树输出。建树分别有递归和非递归两种方式完成。

实验环境：

寝室+机房+Codeblocks

三、设计思想

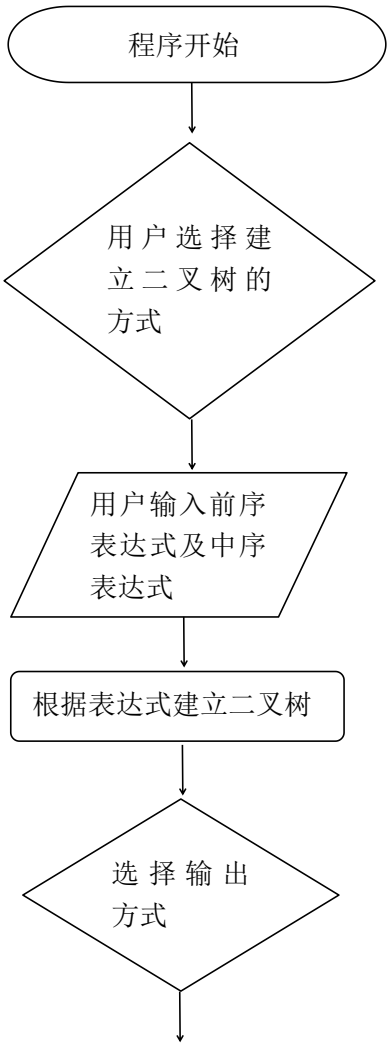
数据类型定义：

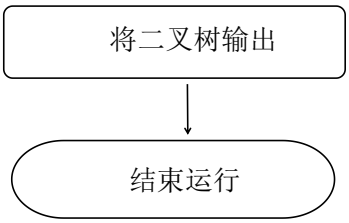
```
struct BiTNode
{
    BiTNode *lchild;
    BiTNode *rchild;
    char    data;
};
typedef BiTNode* BITree;
typedef struct sstack {
    BiTNode *db[N];
    int top;
} sstack;
typedef struct QueueNode {
    BiTNode* data;
    struct QueueNode* next;
}QueueNode;
typedef struct Queue {
    QueueNode* front;
    QueueNode* rear;
}LinkQueue;
```

程序用到的函数及其功能：

```
void createBiTree2(BITree &rT, string preStr, string inStr)
//由前序和中序表达式非递归建立二叉树
BiTNode* createBiTree(char *pre, char *in, int n)
//由前序和中序表达式递归建立二叉树
LinkQueue* InitQueue()
// 链队列的初始化
```

void DestoryQueue(LinkQueue* Que)
// 链队列的销毁
void EnQueue(LinkQueue* Que, BiTNode* node)
//数据入队
BiTNode* DeQueue(LinkQueue* Que)
//数据出队
void LayerOrderBiTree(struct BiTNode* root)
//层序遍历二叉树
void postTraverse(BiTNode *t)
//非递归后序遍历二叉树
void feipreorderTraverse(BiTNode*root)
//非递归先序遍历
void feiinorderTraverse(BiTNode *root)
//非递归中序遍历
void preOrder(BiTNode*root)
//递归先序遍历
void postorder(BiTNode*root)
//递归后序遍历
void inOrder(BiTNode*root)
//递归中序遍历
主程序流程图：





调用关系：
void LayerOrderBiTree(struct BiTNode* root)调用 LinkQueue* InitQueue()
void DestoryQueue(LinkQueue* Que)
void EnQueue(LinkQueue* Que, BiTNode* node)
BiTNode* DeQueue(LinkQueue* Que)
主函数调用 LinkQueue* InitQueue()
void DestoryQueue(LinkQueue* Que)
void EnQueue(LinkQueue* Que, BiTNode* node)
BiTNode* DeQueue(LinkQueue* Que)

四、结果测试：
**********菜单栏**********
1.根据先序和中序序列递归建立树
2.根据先序和中序序列非递归建立树
0.退出
****
请输入序号：1
请输入先序序列：
ABDHECFGIJ
请输入中序序列：
DHBEAFCIGJ

建树成功

**********菜单栏**********
3.先序遍历（递归）
4.中序遍历（递归）
5.后序遍历（递归）
6.先序遍历（非递归）
7.中序遍历（非递归）
8.后序遍历（非递归）
9.层序遍历

10.置空树
0.退出
****
请输入序号：3
A B D H E C F G I J

**********菜单栏**********
3.先序遍历（递归）
4.中序遍历（递归）
5.后序遍历（递归）
6.先序遍历（非递归）
7.中序遍历（非递归）
8.后序遍历（非递归）
9.层序遍历
10.置空树
0.退出
****
请输入序号：4
D H B E A F C I G J

**********菜单栏**********
3.先序遍历（递归）
4.中序遍历（递归）
5.后序遍历（递归）
6.先序遍历（非递归）
7.中序遍历（非递归）
8.后序遍历（非递归）
9.层序遍历
10.置空树
0.退出
****
请输入序号：5
H D E B F I J G C A

**********菜单栏**********
3.先序遍历（递归）
4.中序遍历（递归）
5.后序遍历（递归）
6.先序遍历（非递归）
7.中序遍历（非递归）
8.后序遍历（非递归）
9.层序遍历
10.置空树
0.退出

```
*****************************************
请输入序号：6
A B D H E C F G I J

*****************菜单栏*******************
     3.先序遍历（递归）
     4.中序遍历（递归）
     5.后序遍历（递归）
     6.先序遍历（非递归）
     7.中序遍历（非递归）
     8.后序遍历（非递归）
     9.层序遍历
     10.置空树
     0.退出
*****************************************
请输入序号：7
D H B E A F C I G J

*****************菜单栏*******************
     3.先序遍历（递归）
     4.中序遍历（递归）
     5.后序遍历（递归）
     6.先序遍历（非递归）
     7.中序遍历（非递归）
     8.后序遍历（非递归）
     9.层序遍历
     10.置空树
     0.退出
*****************************************
请输入序号：8
H D E B F I J G C A

*****************菜单栏*******************
     3.先序遍历（递归）
     4.中序遍历（递归）
     5.后序遍历（递归）
     6.先序遍历（非递归）
     7.中序遍历（非递归）
     8.后序遍历（非递归）
     9.层序遍历
     10.置空树
     0.退出
*****************************************
请输入序号：9
```

```
A
B C
D E F G
H I J
**********菜单栏**********
    1.根据先序和中序序列递归建立树
    2.根据先序和中序序列非递归建立树
    0.退出
****************************************
请输入序号：2
请输入先序序列：
ABDHECFGIJ
请输入中序序列：
DHBEAFCIGJ

建树成功

**********菜单栏**********
    3.先序遍历（递归）
    4.中序遍历（递归）
    5.后序遍历（递归）
    6.先序遍历（非递归）
    7.中序遍历（非递归）
    8.后序遍历（非递归）
    9.层序遍历
    10.置空树
    0.退出
****************************************
请输入序号：9
A
B C
D E F G
H I J
```

五、经验体会：

1、在开始写程序之前一定要大体了解函数算法的实现。

2、同一个函数功能，递归和非递归的实现存在很大的差异，并且递归形式较难理解，容易犯错。

3、写代码的过程中，始终都要保持高度的警惕，尽量防止一些错误的发生，出现错误后再改正要花费很长的时间。

4、如果函数过多，一定要对函数进行注释，尤其是代码书写的时间比较长。

六、附录：

源代码见文件

哈夫曼编码

一、实验目的：

了解掌握哈夫曼树的建立过程，深入了解哈夫曼编码的原理及其应用。并利用哈夫曼编码解决实际问题。

二、实验要求及实验环境

根据用户输入的或由文件中读取的英文文章，通过统计各个字符出现的频率确定不同字符的权重，并根据哈夫曼编码实现对英文文章占空间最小的编码，并对已有编码进行译码。

实验环境：

寝室+机房+Codeblocks

三、设计思想：

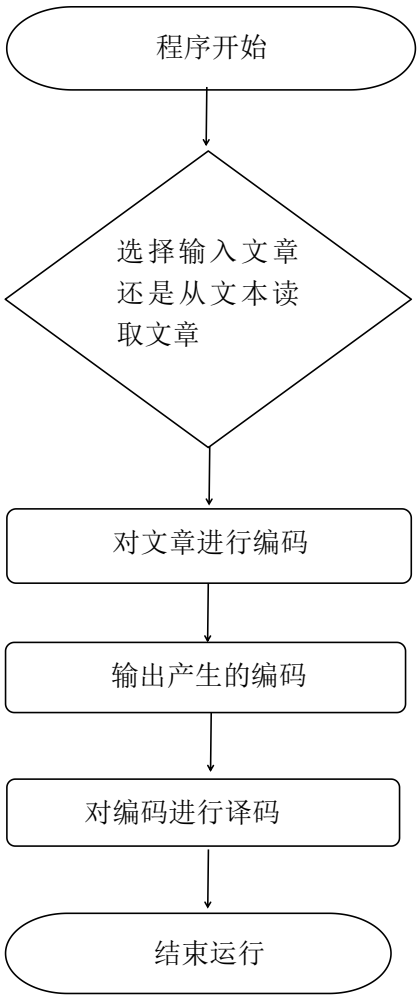
数据类型定义

```
struct HTnode
{
    int weight;
    int parent;
    int lchild;
    int rchild;
    char c;
};
struct HuffmanCode
{
    char ch;
    string bits;
};
```

程序用到的函数及其功能：

```
string compress(string passage)
//将文章中出现的字符进行排序便于对各字符权重的计算
int WeightingSort(HTnode HuffmanT[N],string PassageSort)
//计算各个字符的权重
void initial(HTnode HuffmanT[N])
//初始化哈夫曼树
void selectMin(HTnode HuffmanT[N],int n,int &p1,int &p2)
//挑选权重最小的两个节点
void creatHT(HTnode HuffmanT[N],int n)
//建立哈弗曼树
void Encode(HTnode HuffmanT[N],int n,HuffmanCode Hcode[N])
//对各个字符进行编码
string Decode(HTnode HuffmanT[N],string secret,int n)
//对编码进行译码
void menu()
//选择菜单
```

主程序流程图：



调用关系：

void creatHT(HTnode HuffmanT[N],int n) 调用 void selectMin(HTnode HuffmanT[N],int n,int &p1,int &p2)

主函数调用 string compress(string passage) int WeightingSort(HTnode HuffmanT[N],string PassageSort) void initial(HTnode HuffmanT[N]) void creatHT(HTnode HuffmanT[N],int n) void Encode(HTnode HuffmanT[N],int n,HuffmanCode Hcode[N]) string Decode(HTnode HuffmanT[N],string secret,int n) void menu()

四、测试结果：

1.输入文章
2.输出输入的文章
3.对文章进行编码
4.对编码进行译码
5.从文件读文章
0.退出
5
文件读取文章成功
1.输入文章
2.输出输入的文章

3.对文章进行编码
4.对编码进行译码
5.从文件读文章
0.退出
3

1100110

1110
,
00101
.
100101
?
10011
a
1010
b
01010
c
1100111
d
01011
e
1011
f
001000
g
0000
h
1111
i
1101
j
10000
l
110000
m
10001
n
0011
o
0100
s

0001
u
011
w
001001
x
100100
y
110001
z
110010
对文章编码如下
0001111101110000011100001101101100000100011001011111101000100001110001101000110001111101110010111101100010100011100100110110100011100001101010110111101110000110110111110001011001100100101000101001111001011111101010111010010011101111101011010011101100110000111000011111011010011100000111111011001110001101100101110010111110110000101111100001111101110001011111011001111111101001010011010110110100111011000010111110010101010111011000111010011000000001010110110010110011100111001111001 10
482
1. 输入文章
2. 输出输入的文章
3. 对文章进行编码
4. 对编码进行译码
5. 从文件读文章
0. 退出
4
对编码解码如下
shujujiegou,hafumanshu. youxianjiduilie ,?wobuzhidao haineng shuo shenme,zhege shumu chabuduo le ba yinggai.???
1. 输入文章
2. 输出输入的文章
3. 对文章进行编码
4. 对编码进行译码
5. 从文件读文章
0. 退出

1.输入文章
2.输出输入的文章
3.对文章进行编码
4.对编码进行译码
5.从文件读文章
0.退出
1
请输入英语文章,以#号结束
shujujiegou,hafumanshu#
1.输入文章
2.输出输入的文章
3.对文章进行编码
4.对编码进行译码
5.从文件读文章
0.退出
2

shujujiegou,hafumanshu
1.输入文章
2.输出输入的文章
3.对文章进行编码
4.对编码进行译码
5.从文件读文章
0.退出
3

11100
,
11101
a
1011
e
11110
f
11111
g
0000
h
100
i
0001
j
1100

m
0010
n
0011
o
1010
s
1101
u
01
对文章编码如下
11100110110001110001110000011111000001010011110110010111111101001010110011110110001
83
1.输入文章
2.输出输入的文章
3.对文章进行编码
4.对编码进行译码
5.从文件读文章
0.退出
4
对编码解码如下

shujujiegou,hafumanshu
1.输入文章
2.输出输入的文章
3.对文章进行编码
4.对编码进行译码
5.从文件读文章
0.退出

五、经验体会：
1、写程序的时候，必须首先想好程序都应包括什么功能，然后在书写函数以及寻找数据结构是有意的去帮助函数功能的实现。
2、写一个程序，首先尽量想好其大体的结构很重要，想好结构之后再去写相应的功能就会感觉简单很多。
3、写代码的过程中，始终都要保持高度的警惕，尽量防止一些错误的发生，出现错误后再改正要花费很长的时间。
六、附录：
源代码见文件