

**数据结构实验报告**

**实验报告题目：** 哈夫曼树的应用

**学生姓名**： 刘凌杰

**专 业： 软件工程**

**学 号**： 201720180411

**指导教师： 邹 国 华**

**2018年 11 月 18 日**

**一. 实验内容：**

* **哈夫曼树的建立**
* **哈夫曼编码**

**二. 实验算法实现：**

**结构体**

typedef struct Tree

{

DataType data;

struct Tree \*rchild;

struct Tree \*lchild;

}Tree;

**1.哈夫曼树创立**

Tree\* createHuftree(DataType \*huffman,int len)

{

sort(huffman,len);

Tree \*\*T=(Tree\*\*)malloc(len\*sizeof(Tree\*));

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

{

T[i]=(Tree\*)malloc(sizeof(Tree));

T[i]->data=huffman[i];

T[i]->lchild=NULL;

T[i]->rchild=NULL;

}

int j;

int dex=0;

Tree \*new\_ptr;

while(1)

{

new\_ptr=(Tree\*)malloc(1\*sizeof(Tree));

new\_ptr->data=T[dex]->data+T[dex+1]->data;

new\_ptr->lchild=T[dex];

new\_ptr->rchild=T[dex+1];

T[dex]=NULL;

T[dex+1]=NULL;

j=dex+2;

dex++;

if(dex==len-1)

{

break;

}

while(new\_ptr->data>T[j]->data)

{

T[j-1]=T[j];

j++;

if(j==len)

break;

}

T[j-1]=new\_ptr;

}

return new\_ptr;

}

**2.哈夫曼编码**

void interpret(Tree\* T,int len)

{

static int code[MAX];

if(T->lchild==NULL&&T->rchild==NULL)

{

cout<<T->data<<" ";

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

{

cout<<code[i];

}

cout<<endl;

}

else

{

code[len]=0;

interpret(T->lchild,len+1);

code[len]=1;

interpret(T->rchild,len+1);

}

}

**3.Main函数**

int main()

{

int len;

DataType \*huffman;

Tree \*T;

cout<<"请输入权值个数:"<<endl;

cin>>len;

huffman=new DataType [len];

cout<<"请输入如权值:"<<endl;

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

{

cin>>huffman[i];

}

T=createHuftree(huffman,len);

cout<<"前序遍历："<<endl;

qian\_list(T);

cout<<endl;

cout<<"解码："<<endl;

interpret(T,0);

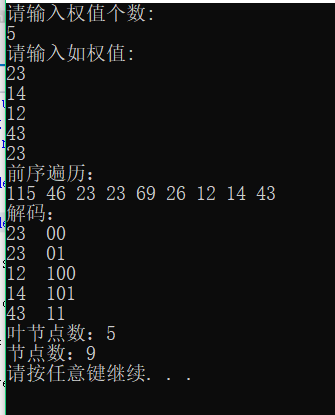
cout<<"叶节点数："<<yeNum(T)<<endl;

cout<<"节点数："<<jieNum(T)<<endl;

system("pause");

return 0;

}

****

**0 1**

**0 1 0 1**

**0 1**

**三．实验总结：**

经过一段时间的学习，哈夫曼树算法的实现基本掌握了，就是代码打得还不太熟练，还有待提高，有些功能不太熟要问同学或查资料，尤其在做创建时。以后在这方面我会多花点时间，让自己更加熟练。