

本科生课程论文(设计)

****

**题 目 数据结构**

学生姓名 沈宇豪

学 号 202083290386

学 院 计算机与软件学院

专 业 计算机类

授课教师 陈金辉

**二Ｏ二一 年 11 月 7 日**

目录

1. 需求分析-----------------------------------------------------------------------2
2. 概要设计-----------------------------------------------------------------------2
3. 详细设计-----------------------------------------------------------------------4
4. 调试分析-----------------------------------------------------------------------5
5. 附录代码-----------------------------------------------------------------------6

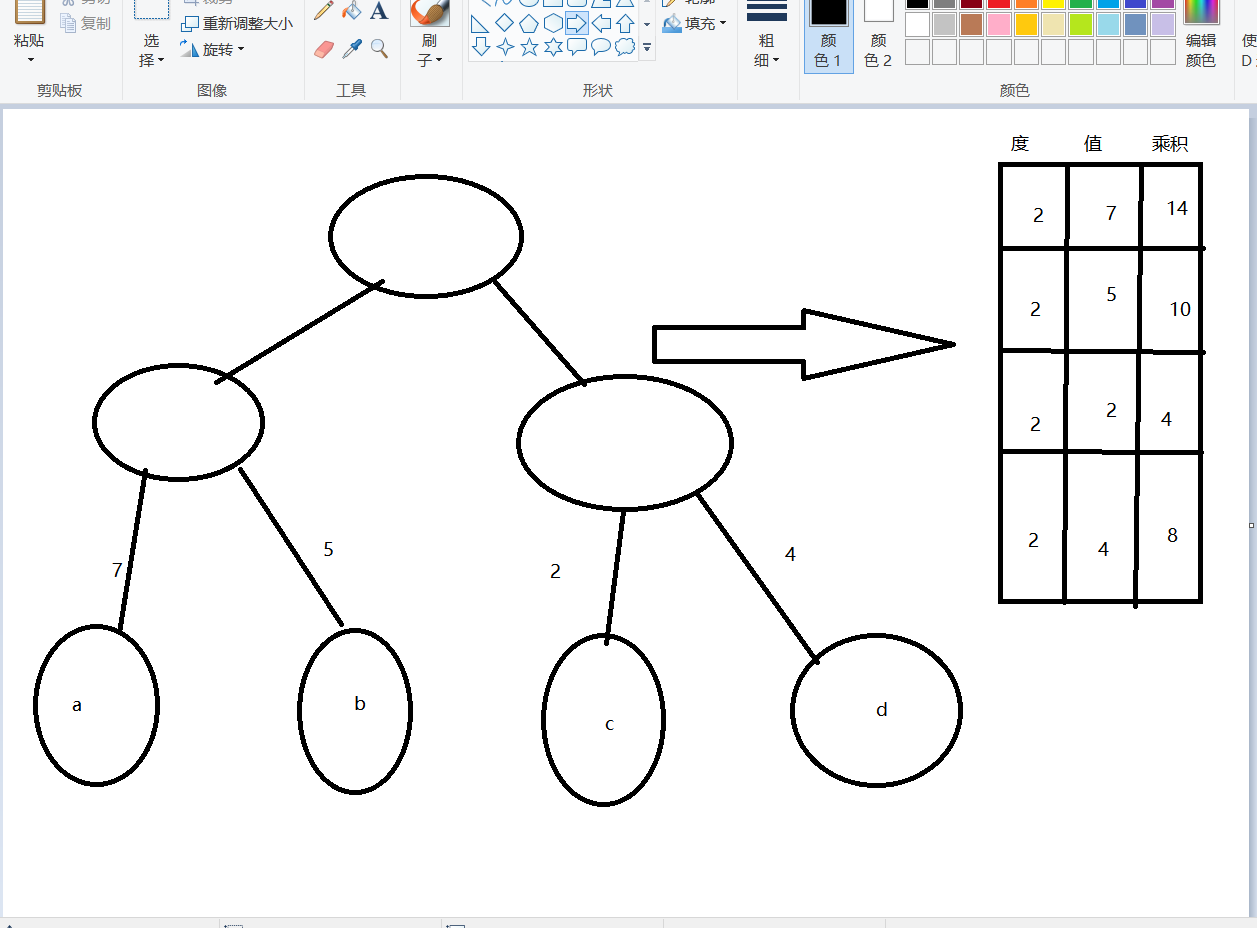
需求分析

1. 建立一棵二叉树
2. 输出其最优二叉树（哈夫曼树）

概要设计

1.首先需要建立一颗二叉树并且申请一个数组，由于建立的是二叉树树，存储结构选择链式存储结构

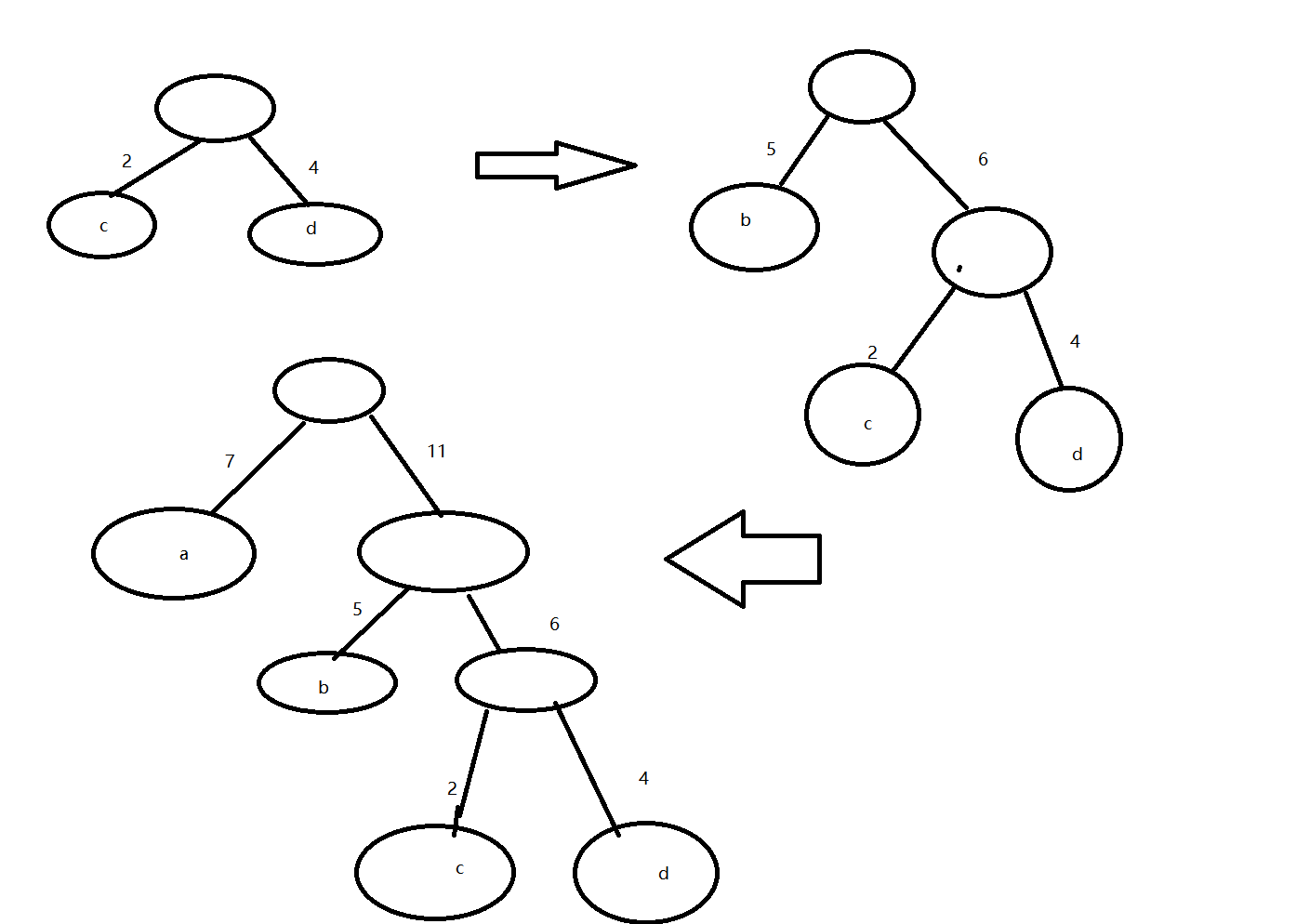
2.对二叉树进行遍历，将遍历的每个节点的值与权重的乘积存入数组



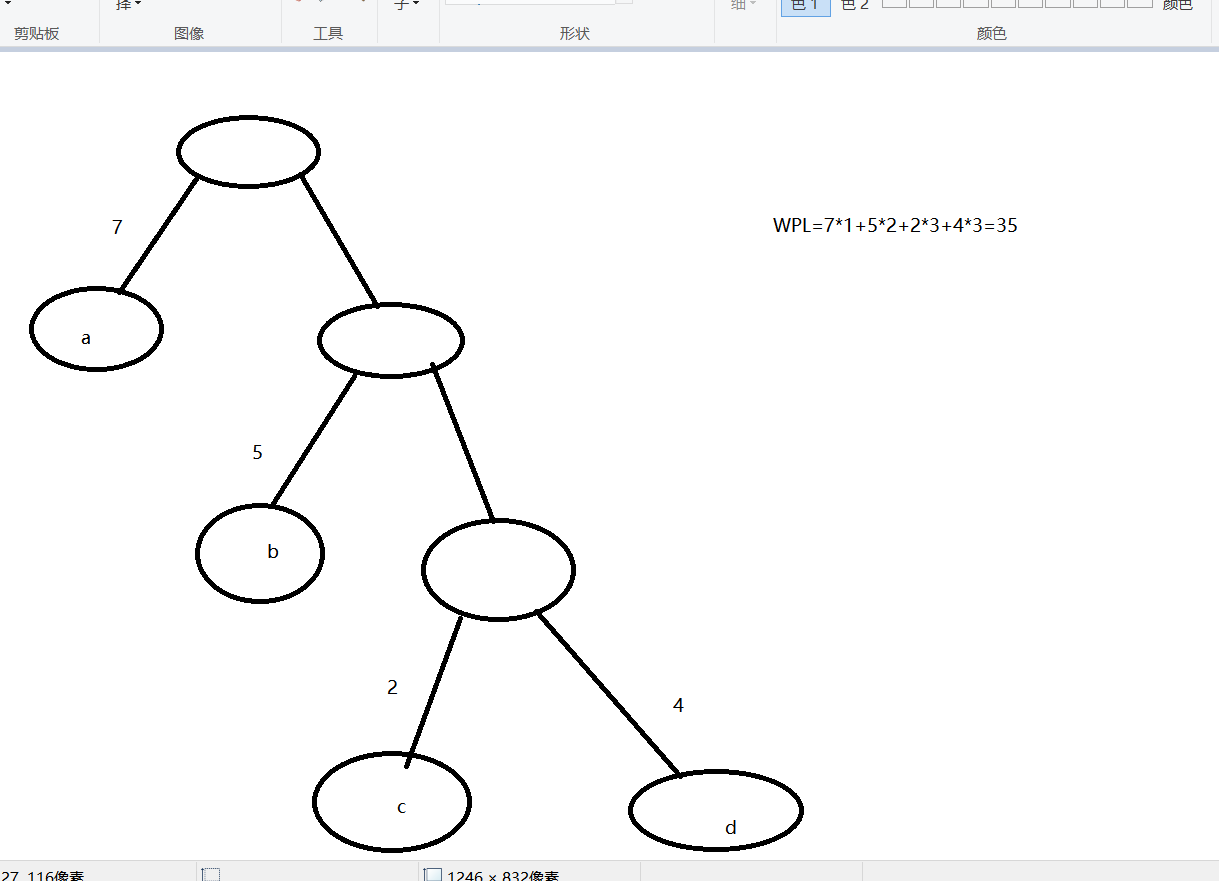
1. 对数组内的值进行比较，进行哈夫曼算法比较，组建出一颗新的树，即哈夫曼树

哈夫曼算法：

1. 在给定的n个权值{w1，w2，w3.....，wn}构成n棵二叉树的集合F={T1,T2,T3,.........Tn}，其中每棵二叉树的Ti中只有一个带权为Wi的根节点，其左右子树均空。
2. 在F中选去两棵根节点的权值最小的树作为左右子树构造一课新的二叉树，且置心得二叉树的根节点的权值为左右子树根节点上权值之和。
3. 在F中删除这两颗树，并将新得到的二叉树放入F。
4. 重复2和3，直到F只含一颗树为止。



这棵树就是哈夫曼树



详细设计

BTNode \*CreateHT(BTNode \*ht,int n) //创建哈夫曼树

{

int i;

BTqsort(ht,0,n);

for(int i=0; i<2\*n-1; i++) //n个权值的二叉树的哈夫曼树有2n-1个结点

{

ht[i].lchild=ht[i].rchild=ht[i].parent=NULL; //初始化

}

int min1=0; //排序后最小的两个权值的编号为0和1

int min2=1;

for( i=n; i<2\*n-1; i++)

{

p[min1]=p[min2]=1;

ht[i].lchild=&ht[min1];

ht[i].rchild=&ht[min2];

ht[i].data=ht[min1].data+ht[min2].data;

selectTwo(ht,i,min1,min2); //选择最小的两个权值

}

return &ht[i-1];

}

void selectTwo(BTNode \*ht,int end,int &min1,int &min2)

//选择两个最小的权值，序号由min1,min2表示

{

int minn=999999; //先将最小值设定足够大

for(int j=0;j<=end;j++) //选取最小值

{

if(ht[j].data<minn&&p[j]==0)

{

minn=ht[j].data;

min1=j;

}

}

minn=999999;

for(int j=0;j<=end;j++) //选取次小值

{

if(ht[j].data<minn&&j!=min1&&p[j]==0)

{

minn=ht[j].data;

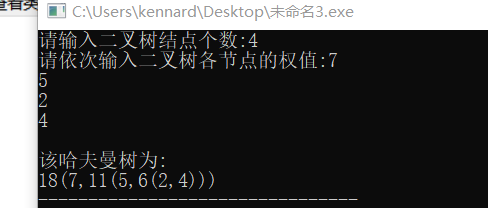
min2=j;

}

}

}

4.调试分析



附录

#include <iostream>

#include <cstdio>

#define N 20

using namespace std;

int p[100]={0}; //用来判断节点是否 被选取过

typedef struct node //建立哈夫曼树的结构体

{

int data;

struct node \*lchild; //定义各节点

struct node \*rchild;

struct node \*parent;

} BTNode;

void BTqsort(BTNode \*ht,int low,int high) //快排

{

if(low>=high-1)

return ;

int key=ht[low].data;

int j=high-1;

while(i<j)

{

while(i<j&&ht[j].data>key) j--; //前移

if(i<j)

ht[i].data=ht[j].data;

while(i<j&&ht[i].data<=key) i++; //后移

if(i<j)

ht[j].data=ht[i].data;

}

ht[i].data=key;

BTqsort(ht,low,i);

BTqsort(ht,i+1,high);

}

void selectTwo(BTNode \*ht,int end,int &min1,int &min2)

//选择两个最小的权值，序号由min1,min2表示

{

int minn=999999; //先将最小值设定足够大

for(int j=0;j<=end;j++) //选取最小值

{

if(ht[j].data<minn&&p[j]==0)

{

minn=ht[j].data;

min1=j;

}

}

minn=999999;

for(int j=0;j<=end;j++) //选取次小值

{

if(ht[j].data<minn&&j!=min1&&p[j]==0)

{

minn=ht[j].data;

min2=j;

}

}

}

BTNode \*CreateHT(BTNode \*ht,int n) //创建哈夫曼树

{

int i;

BTqsort(ht,0,n);

for(int i=0; i<2\*n-1; i++) //n个权值的二叉树的哈夫曼树有2n-1个结点

{

ht[i].lchild=ht[i].rchild=ht[i].parent=NULL; //初始化

}

int min1=0; //排序后最小的两个权值的编号为0和1

int min2=1;

for( i=n; i<2\*n-1; i++)

{

p[min1]=p[min2]=1;

ht[i].lchild=&ht[min1];

ht[i].rchild=&ht[min2];

ht[i].data=ht[min1].data+ht[min2].data;

selectTwo(ht,i,min1,min2); //选择最小的两个权值

}

return &ht[i-1];

}

void DispBTNode(BTNode \*b) //遍历

{

if(b!=NULL)

{

cout<<b->data;

if(b->lchild!=NULL||b->rchild!=NULL)

{

cout<<"("; //有左子树输出左括号

DispBTNode(b->lchild);

if(b->rchild!=NULL) //有右子树 输出右括号

cout<<",";

DispBTNode(b->rchild);

cout<<")";

}

}

}

int main()

{

int n,i,temp;

cout<<"请输入二叉树结点个数:";

cin>>n;

BTNode \*m;

BTNode w[2\*N-1];

if(n<N&&n>0)

{

cout<<"请依次输入二叉树各节点的权值:";

for(i = 0; i < n; i++)

{

cin>>temp;

w[i].data = temp;

}

m=CreateHT(w,n);

cout<<endl;

cout<<"该哈夫曼树为:"<<endl;

DispBTNode(m);

}

else cout<<"输入错误！"<<endl;

return 0;

}