实验报告

题目：写一个哈夫曼的编/译码系统

班级：电信1809 姓名：刘伟豪 学号：U201813494

1. **需求分析**  用结构体数组表示哈夫曼树，用二维数组来存储编码。需分别写出编码函数和译码函数，并在主函数里分别对其进行检验。
2. **概要设计**

首先初始化哈夫曼树，将所有节点的左右孩子和双亲都设为-1，再将所有所需信息存入结构体，再根据其权重构建哈夫曼树。再对所有字符进行编码存入二维数组中，然后利用print函数输出所有编码。在主函数里完成对所有函数的调用，并对编码译码函数进行检验。

1. **代码**

**#include<stdio.h>**

**#include<stdlib.h>**

**#define N\_LEAVE 26**

**#define N\_NODE (26\*2-1)**

**typedef struct \_Node**

**{**

**char character;**

**int weight;**

**int lchild;**

**int rchild;**

**int parent;**

**}Node;**

**typedef struct \_Code**

**{**

**int HufCode[N\_LEAVE];**

**int Start;**

**char Char;**

**}Code;**

**void Huffman(Node Ht[],int Wt[])**

**{**

**int i,j,x1,x2;**

**int min1,min2;**

**for(i=0;i<N\_NODE;i++)**

**{**

**Ht[i].parent = -1;**

**Ht[i].lchild = -1;**

**Ht[i].rchild = -1;**

**if(i<N\_LEAVE)**

**{**

**Ht[i].weight = Wt[i];**

**Ht[i].character = i+65;**

**}**

**else{**

**Ht[i].weight = 0;**

**Ht[i].character = '?';**

**}**

**}**

**for(i=1;i<=N\_LEAVE-1;i++){**

**min1 = min2 = 1000;**

**x1 = x2 = 0;**

**for(j=0;j<N\_LEAVE-1+i;j++){**

**if(Ht[j].parent == -1 && Ht[j].weight<min1 ){**

**min2 = min1;**

**x2 = x1;**

**min1 = Ht[j].weight;**

**x1 = j;**

**}**

**else{**

**if( Ht[j].parent == -1 && Ht[j].weight<min2 ){**

**min2 = Ht[j].weight;**

**x2 = j;**

**}**

**}**

**}**

**Ht[x1].parent = N\_LEAVE-1+i;**

**Ht[x2].parent = N\_LEAVE-1+i;**

**Ht[N\_LEAVE-1+i].weight = Ht[x1].weight + Ht[x2].weight;**

**Ht[N\_LEAVE-1+i].lchild = x1;**

**Ht[N\_LEAVE-1+i].rchild = x2;**

**}**

**}**

**void Code\_Ht(Node Ht[],Code Hc[])**

**{**

**int i,d,p,j;**

**Code x;**

**for(i=0;i<N\_LEAVE;i++)**

**{**

**x.Char = Ht[i].character;**

**x.Start = N\_LEAVE-1;**

**d = i;**

**p = Ht[i].parent;**

**while(1){**

**if(Ht[p].lchild == d)**

**x.HufCode[x.Start] = 0;**

**else if(Ht[p].rchild == d)**

**x.HufCode[x.Start] = 1;**

**else**

**printf("ERROR!");**

**d = p;**

**p = Ht[d].parent;**

**if(p == -1) break;**

**x.Start--;**

**}**

**for(j=x.Start;j<=N\_LEAVE-1;j++){**

**Hc[i].HufCode[j] = x.HufCode[j];**

**}**

**Hc[i].Start = x.Start;**

**Hc[i].Char = x.Char;**

**}**

**}**

**void translate(Node Ht[],char a[])**

**{**

**int i,j;**

**j=0;**

**Node root;**

**Node temp;**

**for(i=0;i<N\_NODE;i++){**

**if(Ht[i].parent==-1) root=Ht[i];**

**}**

**temp=root;**

**while(a[j]!='\0'){**

**if(a[j]=='0') {temp=Ht[temp.lchild];}**

**if(a[j]=='1') {temp=Ht[temp.rchild];}**

**if(temp.lchild==-1) {printf("%c",temp.character);temp=root;}**

**j++;**

**}**

**}**

**void PrintCode(Code Hc[])**

**{**

**int i,j;**

**for(i=0;i<N\_LEAVE;i++)**

**{**

**for(j=Hc[i].Start;j<N\_LEAVE;j++)**

**{**

**printf("%d",Hc[i].HufCode[j]);**

**}**

**printf("%5c\n",Hc[i].Char);**

**}**

**}**

**void FindCode(Code Hc[],char s[])**

**{**

**int i,j,k=0;**

**printf("\n");**

**printf("The code is :");**

**printf("\n");**

**while(s[k]!='\0'){**

**for(i=0;i<N\_LEAVE;i++)**

**{**

**if( s[k] == Hc[i].Char )**

**{**

**for(j=Hc[i].Start;j<N\_LEAVE;j++)**

**{**

**printf("%d",Hc[i].HufCode[j]);**

**}**

**}**

**}**

**k++;**

**}**

**printf("\n");**

**}**

**int main()**

**{**

**Node HufTree[N\_NODE];**

**Code HCode[N\_LEAVE];**

**int Wt[N\_LEAVE] = {64,13,22,32,103,21,15,47,57,1,5,32,20,57,63,15,1,48,51,80,23,8,18,1,16,1};**

**char s[]={"THISPROGRAMISMYFAVORITE"};**

**char a[]={"100"};**

**Huffman(HufTree,Wt);**

**Code\_Ht(HufTree,HCode);**

**PrintCode(HCode);**

**FindCode(HCode,s);**

**translate(HufTree,a);**

**}**

1. **调试分析**

测试结果：输入字符串THISPROGRAMISMYFAVORITE

输出结果：

0010101101001111100100110110111000101101110000101010011100010110011000111110000000110101101010001100