**新疆政法学院**

**数据结构与算法**

**实验报告**

班 级： 计算机科学与技术4班

学 号： 2124030160

姓 名： 孙久猛

指导老师： 张家琦

学 期： 2022-2023第一学期

|  |  |  |  |
| --- | --- | --- | --- |
| **实验题目** | 实验五：哈夫曼编码 | | |
| **实验时间** | **10:00** | **实验地点** | **东训0114** |
| **实验成绩** |  | **实验性质** | **□验证性 √设计性 □综合性** |
| 教师评价：  □算法/实验过程正确； □源程序/实验内容提交 □程序结构/实验步骤合理；  □实验结果正确； □语法、语义正确； □报告规范；  其他：  评价教师签名： | | | |
| 一、实验目的  1.理解哈夫曼树的概念及意义；  2.掌握二叉树应用（Huffman编码）的构建思想；  3.提高使用理论知识指导解决实际问题的能力。 | | | |
| **二、实验项目内容（实验题目）**  案例：已知在一段报文中有a、b、c、d、e、f六个字符，每个字符出现的频率依次为a：45，b：13，c：12，d：16，e：9，f：5。要求对每个字符进行编码要求所发出的报文总长度最短，并求该报文的平均码长。  （一）针对案例中的问题，寻找解决方法并进行理论解决。  1.利用哈夫曼编码对六个字符根据权重建立哈夫曼树。  3.对哈夫曼树进行编码。  4.根据哈夫曼树得出每个字符的哈夫曼编码。  （二）针对案例中的问题，将理论过程转化为代码程序，利用计算机解决该类问题。  1.在第一步的基础上，运用C语言，将第一步哈夫曼编码的过程转化为代码程序解决。  2.输入需要编码的字符个数及每个字符的权重，利用程序得出每个字符的哈夫曼编码。  **三、源程序（实验步骤/实验过程/算法）**  //Deom\_05  #include<stdio.h> /\* EOF(=^Z或F6),NULL \*/  #include<stdlib.h>  #include<string.h>  #include<limits.h> /\* INT\_MAX等 \*/  typedef struct  {  unsigned int weight; //defined the weight of node  unsigned int parent,lchild,rchild; //defined node 'parent,lchild,rchild'  }HTNode,\*HuffmanTree; /\* 动态分配数组存储赫夫曼树 \*/  typedef char \*\*HuffmanCode; /\* 动态分配数组存储赫夫曼编码表 \*/  int min1(HuffmanTree t,int i)  { /\* 函数void select()调用 \*/  int j,flag;  unsigned int k=UINT\_MAX; /\* 取k为不小于可能的值 \*/  for(j=1;j<=i;j++)  if(t[j].weight<k&&t[j].parent==0)  k=t[j].weight,flag=j;  t[flag].parent=1;  return flag;  }  void select(HuffmanTree t,int i,int \*s1,int \*s2)  { /\* s1为最小的两个值中序号小的那个 \*/  int j;  \*s1=min1(t,i);  \*s2=min1(t,i);  if(\*s1>\*s2)  {  j=\*s1;  \*s1=\*s2;  \*s2=j;  }  }  void HuffmanCoding(HuffmanTree \*HT,HuffmanCode \*HC,int \*w,int n)  { /\* w存放n个字符的权值(均>0),构造赫夫曼树HT,并求出n个字符的赫夫曼编码HC \*/  int m,i,s1,s2;  unsigned c,cdlen;  HuffmanTree p; //creat tree\_p of HuffmanTree  char \*cd;  if(n<=1)  return;  m=2\*n-1;  \*HT=(HuffmanTree)malloc((m+1)\*sizeof(HTNode)); /\* 0号单元未用 \*/  for(p=\*HT+1,i=1;i<=n;++i,++p,++w)  {  (\*p).weight=\*w;  (\*p).parent=0;  (\*p).lchild=0;  (\*p).rchild=0;  }  for(;i<=m;++i,++p)  (\*p).parent=0;  for(i=n+1;i<=m;++i) /\* 建赫夫曼树 \*/  { /\* 在HT[1~i-1]中选择parent为0且weight最小的两个结点,其序号分别为s1和s2 \*/  select(\*HT,i-1,&s1,&s2);  (\*HT)[s1].parent=(\*HT)[s2].parent=i;  (\*HT)[i].lchild=s1;  (\*HT)[i].rchild=s2;  (\*HT)[i].weight=(\*HT)[s1].weight+(\*HT)[s2].weight;  }  /\* 以下为无栈非递归遍历赫夫曼树,求赫夫曼编码\*/  \*HC=(HuffmanCode)malloc((n+1)\*sizeof(char\*));  /\* 分配n个字符编码的头指针向量([0]不用) \*/  cd=(char\*)malloc(n\*sizeof(char)); /\* 分配求编码的工作空间 \*/  c=m;  cdlen=0;  for(i=1;i<=m;++i)  (\*HT)[i].weight=0; /\* 遍历赫夫曼树时用作结点状态标志 \*/  while(c)  {  if((\*HT)[c].weight==0)  { /\* 向左 \*/  (\*HT)[c].weight=1;  if((\*HT)[c].lchild!=0)  {  c=(\*HT)[c].lchild;  cd[cdlen++]='0';  }  else if((\*HT)[c].rchild==0)  { /\* 登记叶子结点的字符的编码 \*/  (\*HC)[c]=(char \*)malloc((cdlen+1)\*sizeof(char));  cd[cdlen]='\0';  strcpy((\*HC)[c],cd); /\* 复制编码(串) \*/  }  }  else if((\*HT)[c].weight==1)  { /\* 向右 \*/  (\*HT)[c].weight=2;  if((\*HT)[c].rchild!=0)  {  c=(\*HT)[c].rchild;  cd[cdlen++]='1';  }  }  else  { /\* HT[c].weight==2,退回 \*/  (\*HT)[c].weight=0;  c=(\*HT)[c].parent;  --cdlen; /\* 退到父结点,编码长度减1 \*/  }  }  free(cd);  }  int main()  {  HuffmanTree HT; //define the HuffmanTree HT  HuffmanCode HC; //declare 'HC' the table of HuffmanCode  int \*w,n,i;  printf("请输入权值的个数(>1)：");  scanf("%d",&n);  w=(int \*)malloc(n\*sizeof(int));  printf("请依次输入%d个权值(整型)：\n",n);  for(i=0;i<=n-1;i++)  scanf("%d",w+i);  HuffmanCoding(&HT,&HC,w,n);  for(i=1;i<=n;i++)  puts(HC[i]);  return 0;  }  **四、运行结果**  //Picture\_01    //Picture\_02 | | | |

日 期： 2022年10月26日