**广东理工学院信息工程系**

**课程实验报告**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 姓 名 | 黄梓铭 | 学 号 | 55 | 实验学时 | 2 |
| 实验时间 | 2017年11月16日 | 实验地点 | 42410 | 指导教师 | 江涛 |
| 课程名称 | 数据结构与算法 | 专 业班 级 | 16计算机科学与技术10班 | 成绩 |  |
| 实验项目 | 平衡树、哈夫曼树 | | | | |
| 实  验  目  的 | 目的：  1．掌握平衡树的基本性质；掌握哈夫曼树的基本性质。  2．掌握平衡树的构造；掌握哈夫曼树的构造以及哈夫曼编码。 | | | | |
| 实  验  内  容  和  要  求 | 1、平衡树是左子树与右子树的高度之差的绝对值小于等于1的二叉排序树。现有输入序列：35 80 50 90 45 102 20 40 49 10 39  请根据输入序列，画出其对应的平衡树。  2、已知某报文使用到的字符有8个：A,T,S,H,G,B,R,K，各种字符分别出现3次、1次、4次、1次、7次、2次、5次和3次，请以这8个字符的出现次数作为字符的权值，构造出哈夫曼树(注意：哈夫曼树形态不唯一物品栏)，写出这8个字符对应的哈夫曼编码（确定了哈夫曼树形态后，哈夫曼编码唯一）。   |  |  |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | --- | --- | | 字符 | A | T | S | H | G | B | R | K | | 哈夫曼编码 | 100 | 00000 | 00 | 00001 | 01 | 000 | 11 | 101 |   3、对第二小题编程，输入这8个字符以及它们的权值，输出这8个字符的哈夫曼编码。调试参考程序并给出运行结果，运行结果参考下图。 | | | | |
| 实  验  过  程 | 1、平衡树是左子树与右子树的高度之差的绝对值小于等于1的二叉排序树。现有输入序列：35 80 50 90 45 102 20 40 49 10 39  请根据输入序列，画出其对应的平衡树。   |  | | --- | |  |   2、已知某报文使用到的字符有8个：A,T,S,H,G,B,R,K，各种字符分别出现3次、1次、4次、1次、7次、2次、5次和3次，请以这8个字符的出现次数作为字符的权值，构造出哈夫曼树，写出这8个字符对应的哈夫曼编码。   |  |  |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | --- | --- | | 字符 | A | T | S | H | G | B | R | K | | 哈夫曼编码 |  |  |  |  |  |  |  |  |   3、对第二小题编程，输入这8个字符以及它们的权值，输出这8个字符的哈夫曼编码。  给出源代码以及运行截图：  #include <iostream>  #include<stdlib.h>  using namespace std;  #define MAX 50  #define MAXNUM 60  typedef struct //哈夫曼树结点结构  {  char data;  int weight;  int parent;  int lchild;  int rchild;  }HuffNode;  typedef struct  {  char cd[MAX]; //存放字符的编码  int start; //编码的起始位置  }HuffCode;  void Select (HuffNode huffTree[ ], int k, int &i1,int &i2 ) {  /\*选择根结点权值最小的两个结点。\*/  int i,j;  for(i=0;i<k;i++)  if(huffTree[i].parent ==-1) {i1=i;break;}  for(j=i+1;j<k;j++)  if(huffTree[j].parent ==-1) {i2=j;break;}  for(i=0;i<k;i++)  if((huffTree[i].parent==-1)&&(huffTree[i].weight<huffTree[i1].weight)&&(i2!=i))  {  i1=i;  }  for(j=0;j<k;j++)  if((huffTree[j].parent==-1)&&(huffTree[j].weight<huffTree[i2].weight)&&(i1!=j))  {  i2=j;  }  }  void HuffmanTree(HuffNode huffTree[ ], int n ) {  /\*构造哈夫曼树huffTree [n]，w[ ]存放n个权值。\*/  int i,k;  for (i = 0; i <2\*n-1; i++)  { huffTree [i].parent = -1;  huffTree [i].lchild = -1;  huffTree [i].rchild = -1;  }  for (k = n; k < 2\*n-1; k++) { int i1,i2;  Select(huffTree, k,i1, i2);  /\* 在huffTree [0] ~ huffTree [k-1] 的范围内选择两个parent为-1且  weight最小的结点，其序号分别赋值给i1、i2返回\*/  huffTree[k].weight = huffTree[i1].weight+huffTree[i2].weight;  huffTree[i1].parent = k; huffTree[i2].parent = k;  huffTree[k].lchild = i1; huffTree[k].rchild = i2;  }  }  void Encoding(HuffNode ht[ ],HuffCode hcd[],int n)  {  HuffCode d;  int i,k,f,c;  for(i=0;i<n;i++)  {  d.start=n+1;  c=i;  f=ht[i].parent;  while(f!=-1)  {  if(ht[f].lchild==c)  { d.start--;d.cd[d.start]='0';}  else  {  d.start--;d.cd[d.start]='1';  }  c=f;  f=ht[f].parent;  }  hcd[i]=d;  }  cout<<"输出哈夫曼编码"<<endl;  for(i=0;i<n;i++)  {  cout<<ht[i].data<<":";  for(k=hcd[i].start;k<=n;k++)  {  cout<<hcd[i].cd[k];  }  cout<<endl;  }  }  int main()  {  int n,i;  HuffNode huffTree[MAXNUM];  HuffCode hcd[MAXNUM];  cout<<"请输入字符个数"<<endl;  cin>>n;  for(i=0;i<n;i++)  {  cout<<"输入第"<<i+1<<"个字符:";  cin>>huffTree[i].data;  cout<<"输入第"<<i+1<<"个字符的权值:";  cin>>huffTree[i].weight;  }  HuffmanTree(huffTree,n);  Encoding(huffTree,hcd,n);  system("pause");  return 0;  } | | | | |
| 实  验  结  果 | **（记录程序执行的结果，分析结果）** | | | | |
| 实  验  总  结 | **（记录程序在调试过程中出现的问题以及解决方法、总结收获和心得）**  这节课掌握了很多  掌握平衡树的基本性质；掌握哈夫曼树的基本性质。  掌握平衡树的构造；掌握哈夫曼树的构造以及哈夫曼编码。 | | | | |