数据结构实践课报告

题目：哈夫曼编/译码器

班级：17052317 姓名：於文卓 学号17061833 完成日期：2018年11月30日

1. **需求分析**
2. 问题描述

利用哈夫曼编码进行通信可以大大提高信道利用率，缩短信息传输时间，降低传输成本。但是，这要求在发送端通过一个编码系统对待传数据预先编码，在接受端将传来的数据进行译码。试为这样的信息发收站写一个哈夫曼的编/译码系统

1. 基本要求

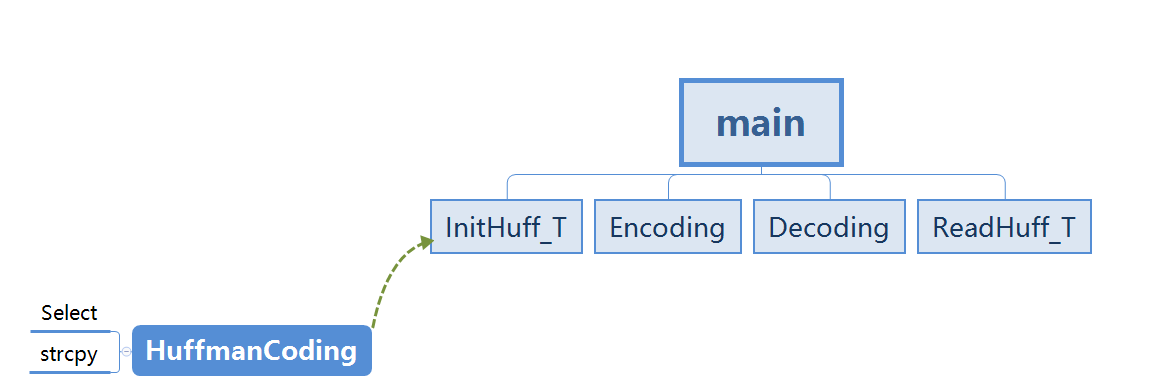
一个完整的系统应该具有以下功能

1. 初始化
2. 编码
3. 译码
4. 打印代码文件
5. 打印哈夫曼树
6. 测试数据：

　见课本p149

THIS PROGRAM IS MY FAVORITE

1. **概要设计**



1. 函数简单介绍

|  |
| --- |
| void strcpy(char \*S1, char \*S2) //将字符串S2复制到S1  void Select(HuffmanTree HT, int t, int &s1, int &s2) //在HT[1]到HT[t-1]中找出权值最小的两个S1和S2  int HuffmanCoding( HuffmanTree &HT, HuffmanCode &HC, int \*w, int n) //根据各个字符的权值构造赫夫曼树HT，将对应的赫夫曼编码存储在HC中  void InitHuff\_T( HuffmanTree &HT, HuffmanCode &HC, char ch[], int &n ) //初始化赫夫曼数，要求用户输入字符和相应权值  void Encoding(HuffmanTree &HT, HuffmanCode &HC, char ch[]) //根据赫夫曼编码将用户指定的文件中的字符编成相应的编码,并将所得编码存储到用户指定文件  void Decoding(HuffmanTree HT, char ch[], int n) //对指定的存储由赫夫曼编码表示的信息的文件进行译码，翻译成相应的字符表示，并存储到指定文件  void ReadHuff\_T( HuffmanTree &HT, HuffmanCode &HC, char ch[], int &n) //从文件读取赫夫曼树 |

1. 部分函数代码和注释
2. HuffmanCoding

|  |
| --- |
| int HuffmanCoding( HuffmanTree &HT, HuffmanCode &HC, int \*w, int n) //根据各个字符的权值构造赫夫曼树HT，将对应的赫夫曼编码存储在HC中  {  int s1, s2, m, i, start;  unsigned int c, f;  HTNode \*p;  char \*cd;  if( n <= 1 ) return 0;  m = 2 \* n - 1; //赫夫曼树的总结点树为m  HT = (HuffmanTree)malloc((m + 1) \* sizeof(HTNode)); //申请存储赫夫曼树的空间  for(p = HT + 1, i = 1; i <= n; ++i, ++p, ++w) //将各个叶子结点的weight赋以相应的权值，parent,lchild,rchild均赋为0  {  p->weight = \*(w + 1);  p->parent = p->lchild = p->rchild = 0;  }  for( ; i <= m; ++i, ++p ) //将各个非叶子结点的weight,parent,lchild,rchild均赋为0  {  p->weight = p->parent = p->lchild = p->rchild = 0;  }  for( i = n + 1; i <= m; ++i ) //构造赫夫曼树，给各个非叶子结点赋值  {  Select(HT, i - 1, s1, s2);  HT[s1].parent = i; HT[s2].parent = i;  HT[i].lchild = s1; HT[i].rchild = s2;  HT[i].weight = HT[s1].weight + HT[s2].weight;  }  HC = (HuffmanCode)malloc((n + 1) \* sizeof(char \*)); //申请空间，用于存储指向存储各个字符相应赫夫曼编码的字符数组的指针  cd = (char \*)malloc(n \* sizeof(char)); //申请用于求赫夫曼编码  cd[n - 1] = '\0'; //编码结束符  for( i = 1; i <= n; ++i) //逐个字符求赫夫曼编码  {  start = n - 1; //编码在数组cd[]中的最前位置  for(c = i, f = HT[i].parent; f != 0; c = f, f = HT[f].parent) //从叶子到根逆向求编码  if(HT[f].lchild == c)  cd[ --start ] = '0';  else  cd[ --start ] = '1';  HC[i] = (char \*)malloc((n - start) \* sizeof(char)); //为第i个字符编码分配空间  strcpy(HC[i], &cd[start]); //将cd[]数组的start位置到n-1位置复制给HC[i]  }  free(cd); //释放空间  return 1;  } |

1. Encoding

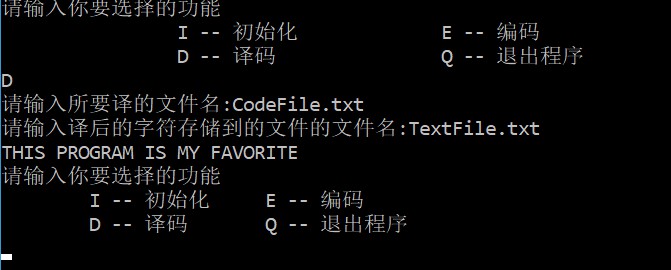
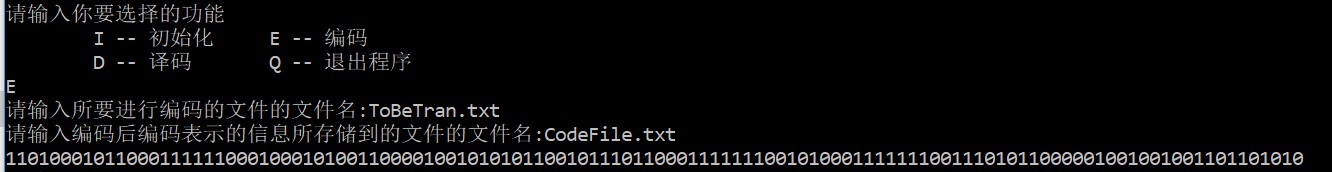
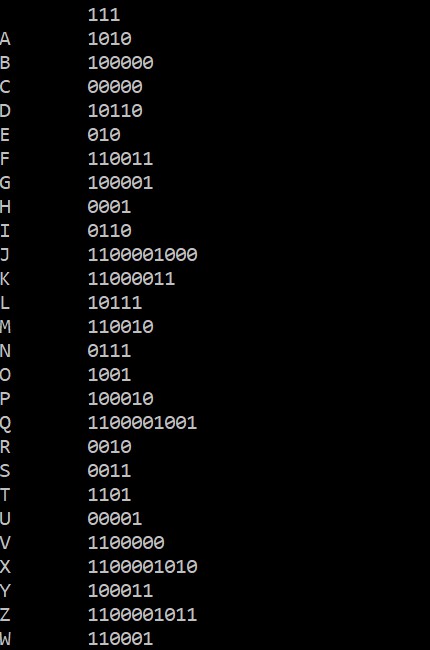
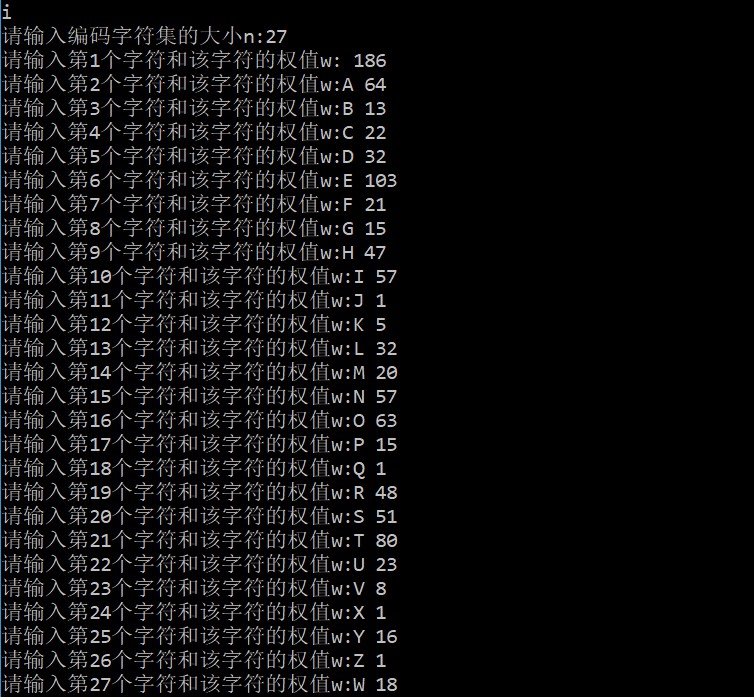
|  |
| --- |
| void Encoding(HuffmanTree &HT, HuffmanCode &HC, char ch[]) //根据赫夫曼编码将用户指定的文件中的字符编成相应的编码,并将所得编码存储到用户指定文件  {  FILE \*ToBeTran, \*CodeFile;  char ToBeTran\_Name[100], CodeFile\_Name[100]; //存储用户指定文件的文件名  int i;  char c;  printf("请输入所要进行编码的文件的文件名:");  scanf("%s", ToBeTran\_Name); //获得所要进行编码的文件的文件名  if(( ToBeTran = fopen(ToBeTran\_Name, "r")) == NULL ) //打开文件  {  printf("Open file fail......\n");  exit(0);  }  printf("请输入编码后编码表示的信息所存储到的文件的文件名:");  scanf("%s", CodeFile\_Name); //获得编码后编码表示的信息所存储到的文件的文件名  if(( CodeFile = fopen(CodeFile\_Name, "w")) == NULL ) //打开文件  {  printf("Open file fail......\n");  exit(0);  }  c = fgetc(ToBeTran); //从文件读取一个字符  while( c != EOF ) //对文件中的各个字符进行编码，直至文件结尾  {  i = 1;  while( c != ch[i] && ch[i] != '\0' ) //在ch[]数组中查找从文件读取的字符  i++;  if(ch[i] == '\0') //未找到，c不在ch[]数组中，c无法被识别，程序出错，退出  {  printf("字符%c无法识别，程序将退出。\n", c);  exit(0);  }  fputs(HC[i], CodeFile); //若找到，则将c相应的赫夫曼编码写入到文件中  printf("%s", HC[i]); //将c相应的赫夫曼编码输出到屏幕  c = fgetc(ToBeTran); //读入文件中的下一个字符  }  printf("\n");  fclose(ToBeTran);  fclose(CodeFile);  } |

1. Decoding

|  |
| --- |
| void Decoding(HuffmanTree HT, char ch[], int n) //对指定的存储由赫夫曼编码表示的信息的文件进行译码，翻译成相应的字符表示，并存储到指定文件  {  int p, i = 1;  char code[1000], c;  char CodeFile\_Name[100], TextFile\_Name[100]; //存储用户指定文件的文件名  p = 2 \* n - 1;  FILE \*CodeFile, \*TextFile;  printf("请输入所要译的文件名:");  scanf("%s", CodeFile\_Name); //获得所要译的文件的文件名  if(( CodeFile = fopen(CodeFile\_Name, "r")) == NULL ) //打开文件  {  printf("Open file fail......\n");  exit(0);  }  printf("请输入译后的字符存储到的文件的文件名:");  scanf("%s", TextFile\_Name); //获得译后的字符存储到的文件的文件名  if(( TextFile = fopen(TextFile\_Name, "w")) == NULL ) //打开文件  {  printf("Open file fail......\n");  exit(0);  }  c = fgetc(CodeFile);  while( c != EOF )  {  code[i] = c;  i++;  c = fgetc(CodeFile);  }//从文件中读取已经编码好的代码  code[i] = '\0'; //从文件读取字符，存储在code[]数组中  i = 1;  while ( code[i] != '\0' && p != 0 ) //对数组code[]中的赫夫曼编码进行译码  {  if ( code[i] == '0' )  p = HT[p].lchild; //进入左分支  else  p = HT[p].rchild; //进入右分支  if (!HT[p].lchild && !HT[p].rchild) //进入叶子结点  {  fputc(ch[p], TextFile); //将相应的字符写入到文件中  printf("%c", ch[p]); //将相应的字符输出到屏幕  p = 2 \* n - 1; //重新从树根出发进行译码  }  i++;  }  printf("\n");  } |

1. **用户手册**
2. **本程序运行环境为win10操作系统，执行文件为main.exe**
3. **用户界面和测试数据如下**



****

1. **附件**

main.c

Huffman\_Tree.h

htfTree

ToBeTran.txt

TextFile.txt