**实验八 Huffman Tree的构造及huffman编码的实现**

假设通信所用的字符集合为: a, b, c, d, e, f, g, h;

各自使用的频率为: 17, 2, 7, 5, 25, 11, 19,14

编写程序实现字符集对应的哈夫曼树，输出各字符的编码。

**提示：哈夫曼树的结点和BinNode一样。**

**构造过程如下：**

1. **构建各字符频率值对应的叶子结点；**

**同时建立表V(或者两个表)**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **17** | **2** | **7** | **5** | **25** | **11** | **19** | **14** |
| **a** | **b** | **c** | **d** | **e** | **f** | **g** | **h** |

1. **将各叶子结点放入优先队列；**
2. **从优先队列中取出结点1，结点2；构建新子树（结点1为左子树，结点2为右子树，子树根的值为左右子值之和），将新子树根放入优先队列；**
3. **重复3，直到队列为空。构建的子树根即为huffman树的树根。**

**参考的类如下：**

**class BinNode{**

**public:**

**BinNode(int d,BinNode\* L=NULL,BinNode \*R=NULL):data(d),left(L),right(R)**

**{ }**

**private:**

**int data;**

**BinNode \*left,\*right;**

**};**

**可以自定义一个HuffmanTree类，以建立的V作为初始化树依据构建哈夫曼树；**

**或者**

**可以写一个独立函数，以建立的V作为初始化依据构建哈夫曼树；**

**然后写一个成员函数 或者 一个独立函数**

**读出各叶子对应的编码；**

**\*有兴趣的同学还可以继续做 以传输字符串 为依据的传输信息生成以及解码的工作。分别可以写成类的两个成员函数或者两个独立的函数Code(string), DeCode(string)。**