**实验报告**

一、Huffman Tree

1.思路说明：

构造哈夫曼树：

利用了STL中的优先队列，通过最小堆实现哈夫曼树，即每次取权值最小的两个结点生成新结点，再将其加入队列中，优先队列会自动为其排序。直至最后队列中只剩一个元素，则哈夫曼树构造完成。

求哈夫曼编码：

为每个节点设置string变量存储哈夫曼编码，从根节点开始，左子树等于根节点的编码加0，右子树等于根节点编码加1，如此循环直至到达叶子节点，即求得哈夫曼编码。我再次利用优先队列，但这次是以ID大小作为排序依据，从而实现字典序输出。

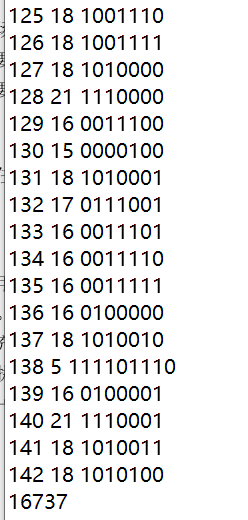
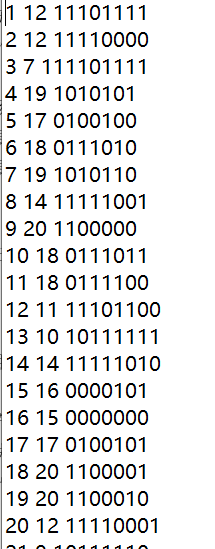
2.文件说明：

“data.txt”是所用数据

“Huffman.cpp”是程序代码

“Huffmanout.txt”是输出结果

3.运行结果展示：展示一部分，其余详见“Huffmanout.txt”



二、KMP

1.思路说明：

就是利用KMP算法进行模式匹配

2.文件说明：

“data2.txt”是所用数据

“randomwalk.cpp”是随机游走的程序代码，用来生成“input.txt”

“input.txt”是KMP实验的输入文件

（其中我选择的模式串T是50 25 62 36 49）

“KMP.cpp”利用KMP算法的程序代码

“KMPout.txt”存放KMP.cpp的运行输出结果

（PS：由于随机生成的input真的很随机……，所以模式串T的出现频率并不高，不得不说经常会只有一大段-1）

3.结果展示：

“randomwalk.cpp”的运行结果放在“input.txt”中（因为是随机生成，所以每次运行的结果都不同）

“KMP.cpp”的运行结果详见“KMPout.txt”，这里展示一部分

