文件压缩实验报告

1. 问题描述

利用Huffman树和Huffman编码，完成文件的压缩和解压缩

1. 设计思路

Huffman树压缩：

首先统计文件中每种字符的词频，并将其作为树节点放入优先队列。读取完成后，从优先队列中取出两个频率域最小的节点，将其作为两个叶子结点，根节点的频率为这两个节点频率之和。如此循环，直到优先队列中只剩下一个节点，此时该节点为Huffman树的根节点。 从该根节点开始遍历Huffman树，向左编码标0，向右编码标1，到根节点时的编码即为该根节点对应字符的编码。把文件字节的哈夫曼编码以二进制形式写入目标文件中。给压缩文件头部写入元数据，解压缩时需使用这些数据。

把每个字符的哈夫曼编码一个一个过，如果是字符’0’，就转换为二进制的0；如果是字符’1’，就转换为二进制的1。转换结果从第1位依次赋值给unsigned char，若已经八位（一个字节）了，就写入文件中。如果最后一次不满一个字节，依然需要写到文件中。由于最后一个字节可能不满八位，因此在写入最后一个字节后，再写入一个字节表示最后一个字节中的有效位数。

Huffman树解压缩：

获取压缩文件头部元数据，得到Huffman编码，生成Huffman树。

生成解压缩文件，一位一位地从压缩文件中读取信息，若是0则向左子树走，若是1则向右子树走。走到叶子结点时输出字符到文件中。

功能设计：

自定义Huffman树类。

定义void printMeta(Node\* root, string str); 输出元数据到文件

void printCode(int size, string originalFile, string compressedFile); 输出文件的Huffman编码

void huffmanCode(vector<char> data, vector<int> freq, int size); 构建Huffman树

void encodeFile(string originalFile); 遍历文件，统计词频

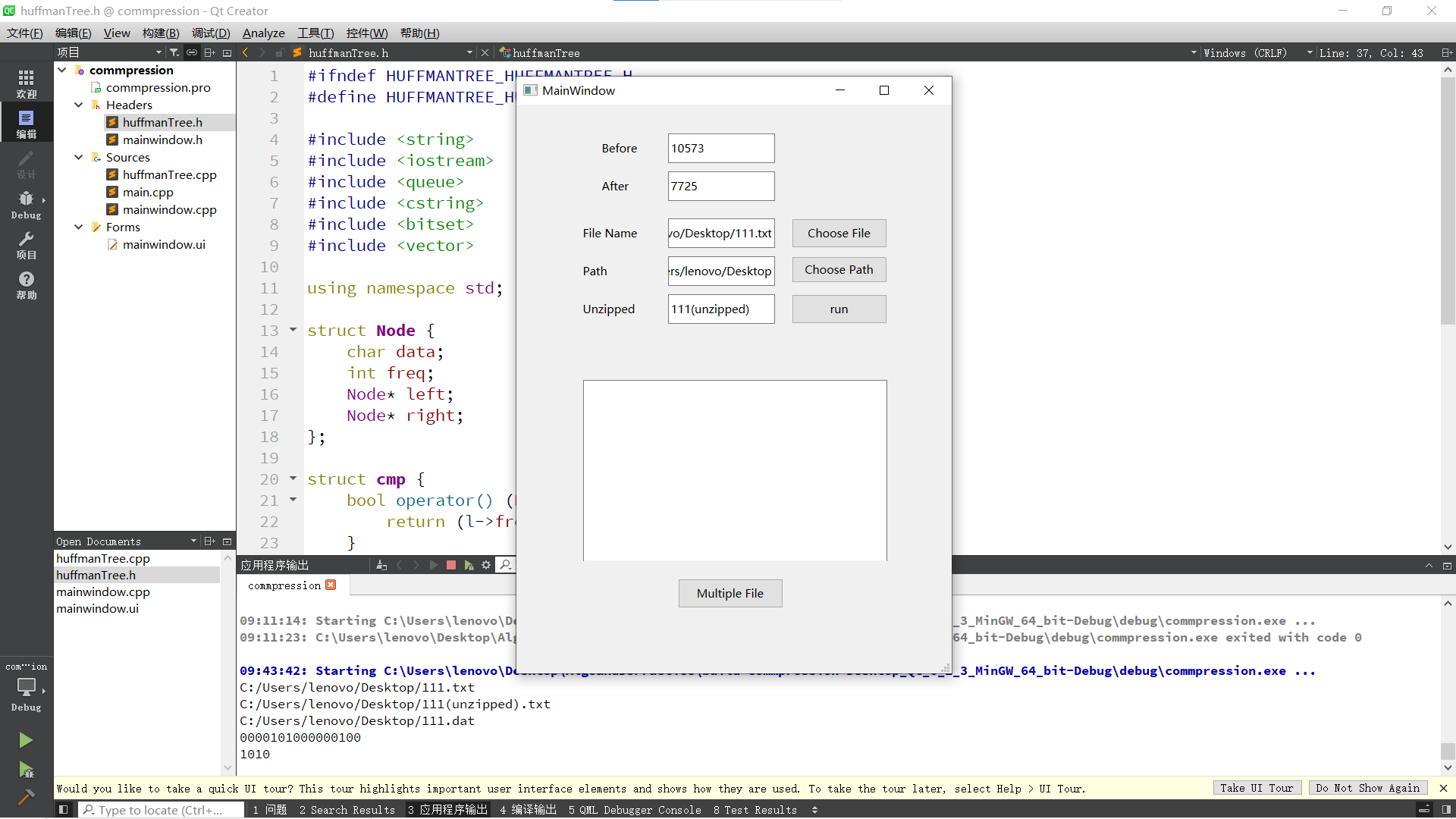
void decodeFile(string compressedFile, string newFile); 解码压缩文件

void out(string data, string newFile); 输出字符到解压文件

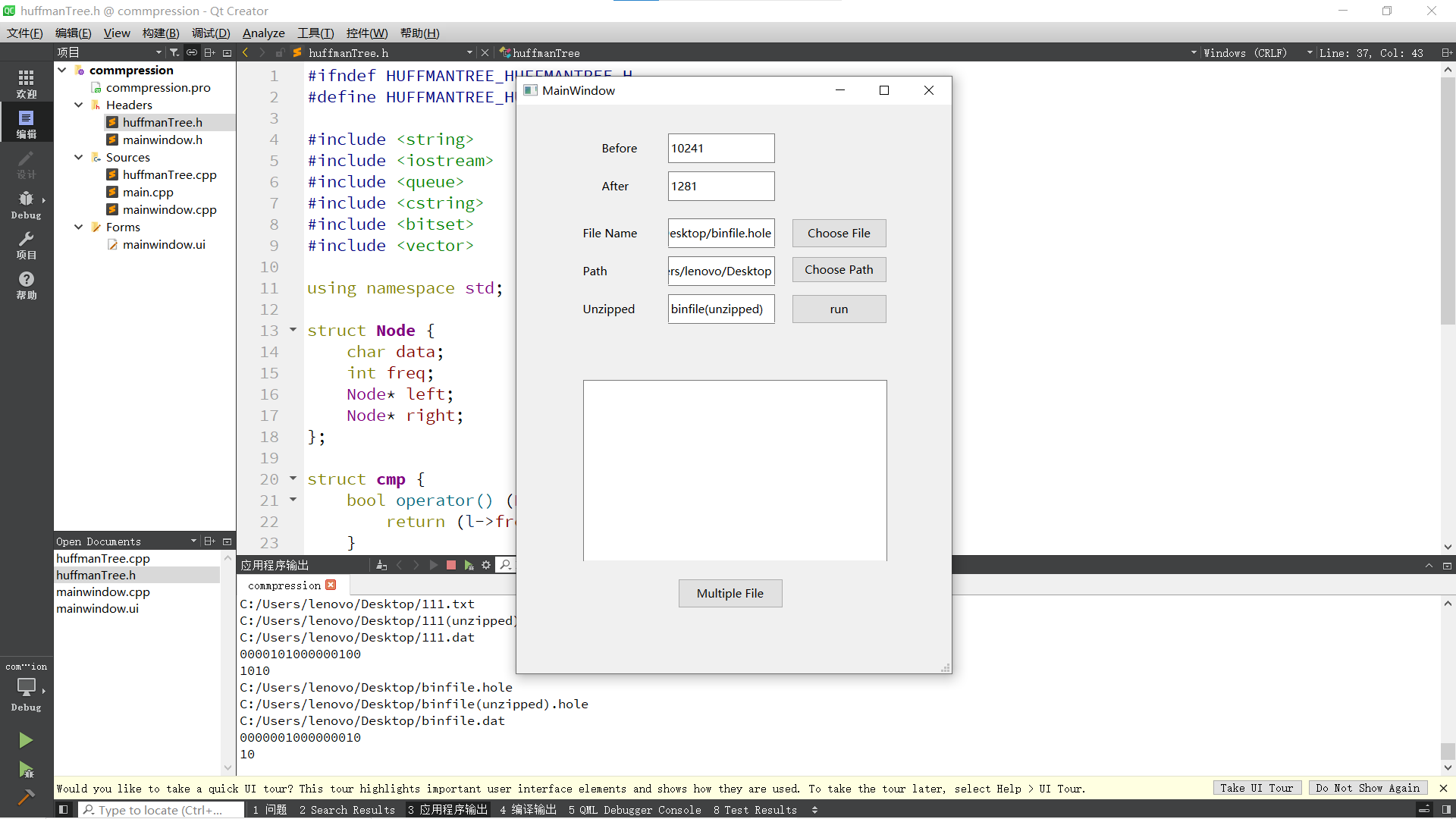
实现多文件压缩解压：使用qt自带函数选择多个文件，使用QStringList存储文件名，使用for循环分别压缩解压各个文件

1. 实验结果截图

压缩文本文件



压缩二进制文件



多文件压缩解压

