# 

# 程序设计与算法综合训练

# 实验报告

# 实验名称： 哈夫曼编/译码系统

# 专业班级：人工智能二班

# 学号：WA2214014

# 姓名：杨跃浙

目录

[一、实验内容及要求 3](#_Toc15120)

[1.1 实验目的 3](#_Toc22670)

[1.2 实验内容 3](#_Toc29297)

[1.3 实验要求 3](#_Toc3750)

[1.4 实验任务 4](#_Toc31134)

[任务二:尝试统计包含大小写英文的字符串文件中的字符频率 4](#_Toc16990)

[二、 任务1 4](#_Toc4691)

[2.1 问题描述 4](#_Toc113)

[2.2 提示 4](#_Toc29338)

[2.3编码结果 5](#_Toc25667)

[2.4字符串哈夫曼编码 5](#_Toc32202)

[三、 任务2 6](#_Toc20706)

[3.1 问题描述 6](#_Toc30467)

[3.2 代码修改 6](#_Toc27411)

[3.3 代码调试 6](#_Toc19832)

[3.4 分析结果 7](#_Toc21692)

[四、任务3 7](#_Toc13578)

[4.1问题描述 7](#_Toc28032)

[4.2步骤1 8](#_Toc17650)

[4.3步骤2 11](#_Toc21838)

[五、实验总结 15](#_Toc21650)

# 一、实验内容及要求

## 1.1 实验目的

加深对树和二叉树数据结构的理解，强化学生的逻辑思维能力和动手能力，巩固良好的编程习惯，掌握工程软件设计的基本方法，为后续课程的学习打下坚实基础。

## 1.2 实验内容

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，在接收端将传来的数据进行译码（解码）。对于双工信道（即可以双向传输信息的信道），每端都需要一个完整的编/译码系统。试为这样的信息收发站设计一个哈夫曼编译码系统。

## 1.3 实验要求

基本要求：

（1）编码（EnCoding）。用已建好的哈夫曼树，对文件ToBeTran.data中的文本进行编码形成报文，将报文写在文件Code.txt中；

（2）译码（Decoding）。利用已建好的哈夫曼树，对文件CodeFile.data中的代码进行解码形成原文，结果存入文件Textfile.txt中；

（3）输出（Output）。输出DataFile.data中出现的字符以及各字符出现的频度（或概率）；输出ToBeTran.data及其报文Code.txt；输出CodeFile.data及其原文Textfile.txt。

## 1.4 实验任务

任务一:利用给出的字符和字符频率对字符串进行哈夫曼编码。

任务二:尝试统计包含大小写英文的字符串文件中的字符频率

任务三：

从文件DataFile.data中读取字符和对应权值，建立哈夫曼树

对文件ToBeTran.data中的文本进行编码形成报文，写入文件Code.txt

对文件CodeFile.data中的代码进行解码形成原文，存入文件Textfile.txt

输出DataFile.data中出现的字符和字符出现的频度；

输出ToBeTran.data及其报文Code.txt；

输出CodeFile.data及其原文Textfile.txt。

# 任务1

## 2.1 问题描述

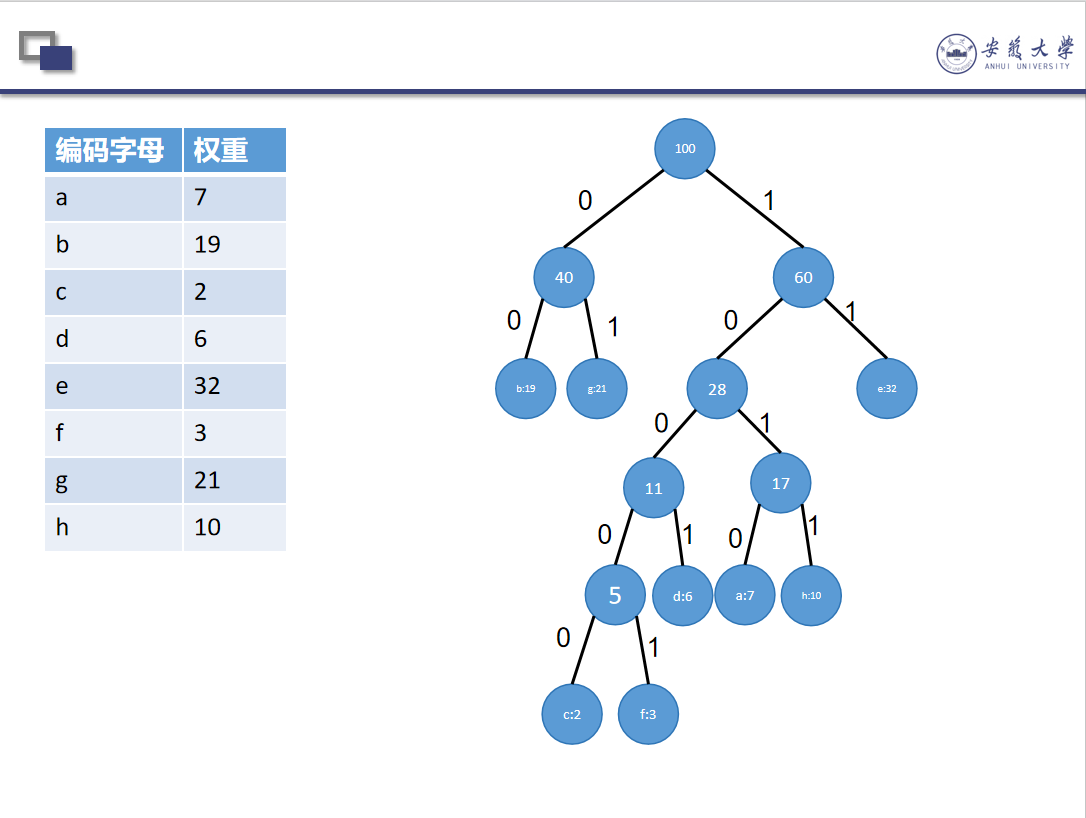
先给出8个字母 {a, b, c, d, e, f, g, h} ，它们出现的概率分别为{ 0.07, 0.19, 0.02, 0.06, 0.32, 0.03, 0.21, 0.10}，组成字符串为:abcdfhgedecf,请设计哈夫曼编码并输出编码后二进制码。

## 2.2 提示

先将概率放大100倍，以方便构造哈夫曼树。  
权值集合 w={7, 19, 2, 6, 32, 3, 21, 10}，

按哈夫曼树构造规则（合并、删除、替换），可得到哈夫曼树。

## 2.3编码结果



* a: 1010
* b: 00
* c: 10000
* d: 1001
* e: 11
* f: 10001
* g: 01
* h: 1011

## 2.4字符串哈夫曼编码

abcdfhgedecf：

10100010000100110001101101111001111000010001

# 任务2

## 3.1 问题描述

试着统计一下:

包含大小写英文的字符串文件中的字符频率

要求：要将每个字符的大小写频率都要统计到一起

## 3.2 代码修改

在统计时要自动判断大小写并统一

int letterCount[26] = { 0 };

char c;

while (inFile.get(c))

{

if (isalpha(c))

{

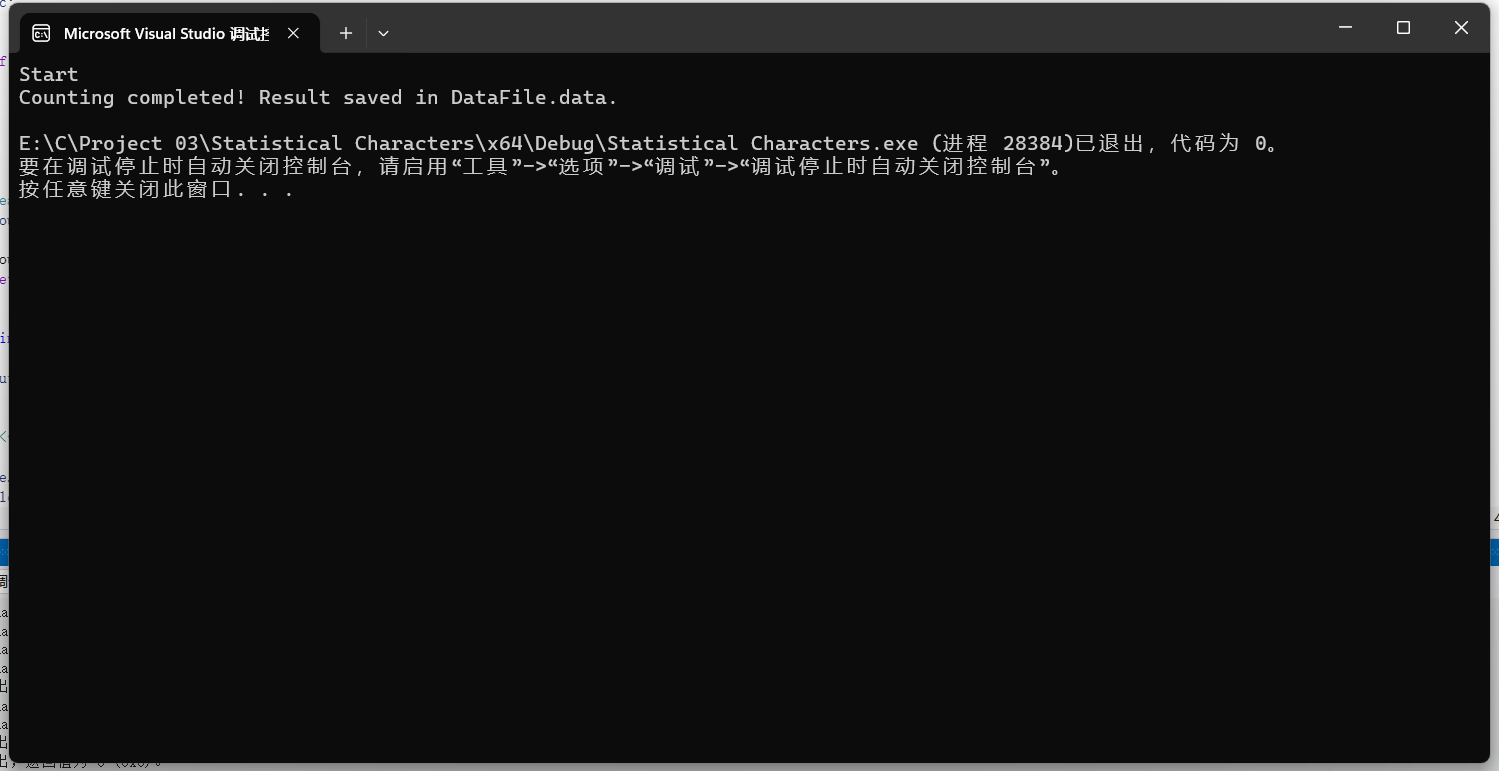
c = tolower(c);

letterCount[c - 'a']++;

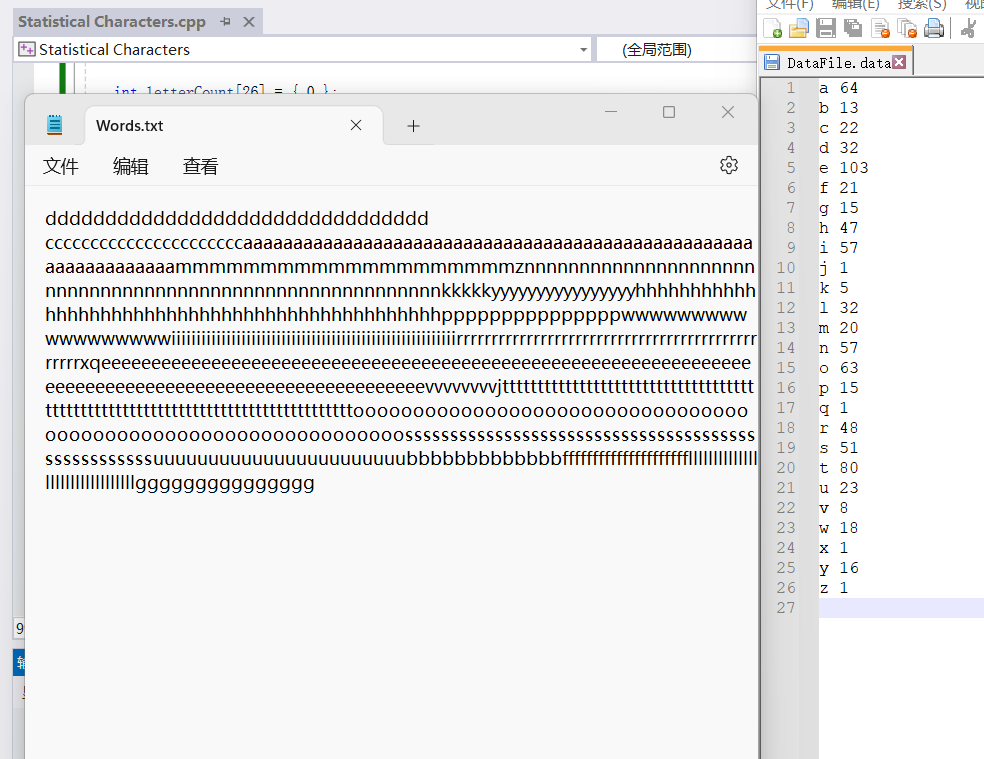
}

}

## 3.3 代码调试



## 3.4 分析结果



结果是正确的

# 四、任务3

## 4.1问题描述

从文件DataFile.data中读取字符和对应权值，建立哈夫曼树

对文件ToBeTran.data中的文本进行编码形成报文，写入文件Code.txt

对文件CodeFile.data中的代码进行解码形成原文，存入文件Textfile.txt

输出DataFile.data中出现的字符和字符出现的频度；

输出ToBeTran.data及其报文Code.txt；

输出CodeFile.data及其原文Textfile.txt。

## 4.2步骤1

从文件DataFile.data中读取字符和对应权值，建立哈夫曼树

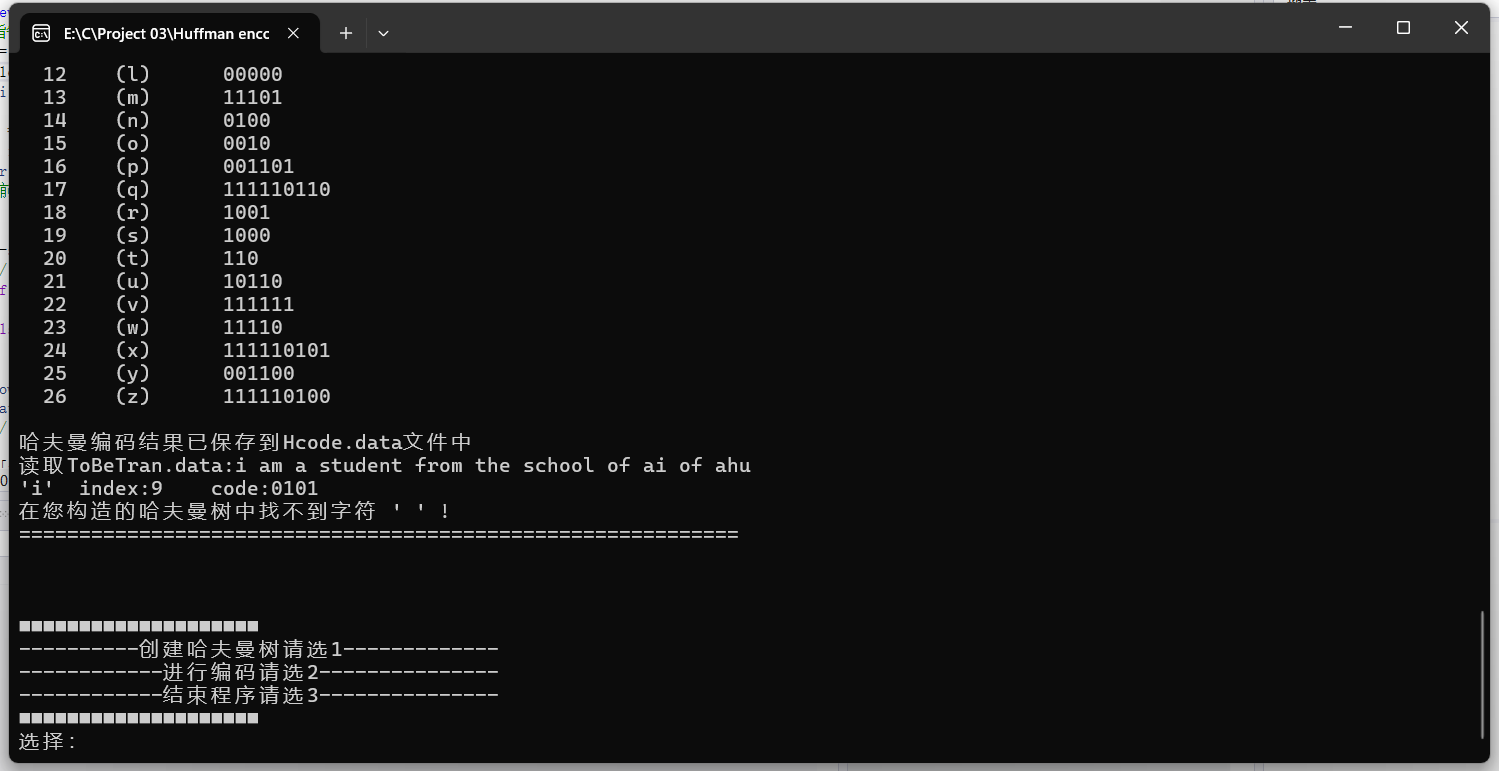
对文件ToBeTran.data中的文本进行编码形成报文，写入文件Code.txt

输出DataFile.data中出现的字符和字符出现的频度；

输出ToBeTran.data及其报文Code.txt；

修改代码文件路径以及错误代码

需要注意文件中的空格，可能会导致代码错误：



为此 应该修改第一步的代码，在除了统计字母之外，加上空格的统计：

int letterCount[27] = { 0 };

char c;

while (inFile.get(c))

{

if (isalpha(c))

{

c = tolower(c);

letterCount[c - 'a']++;

}

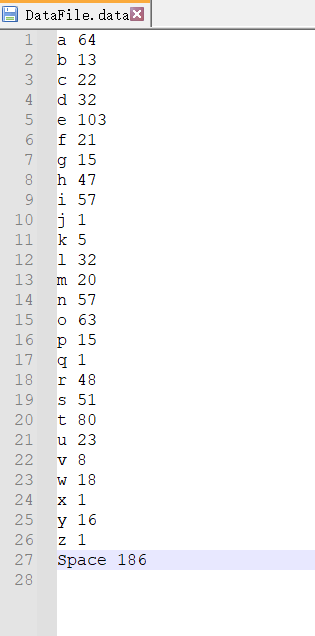
else if (c == ' ')

{

letterCount[26]++;

}

}



然后同时修改初始化的代码：

while (getline(infile, line)) {

stringstream ss(line);

string character;

int weight;

ss >> character;

if (character == "Space") {

ss >> weight;

character = ' ';

}

else {

ss >> weight;

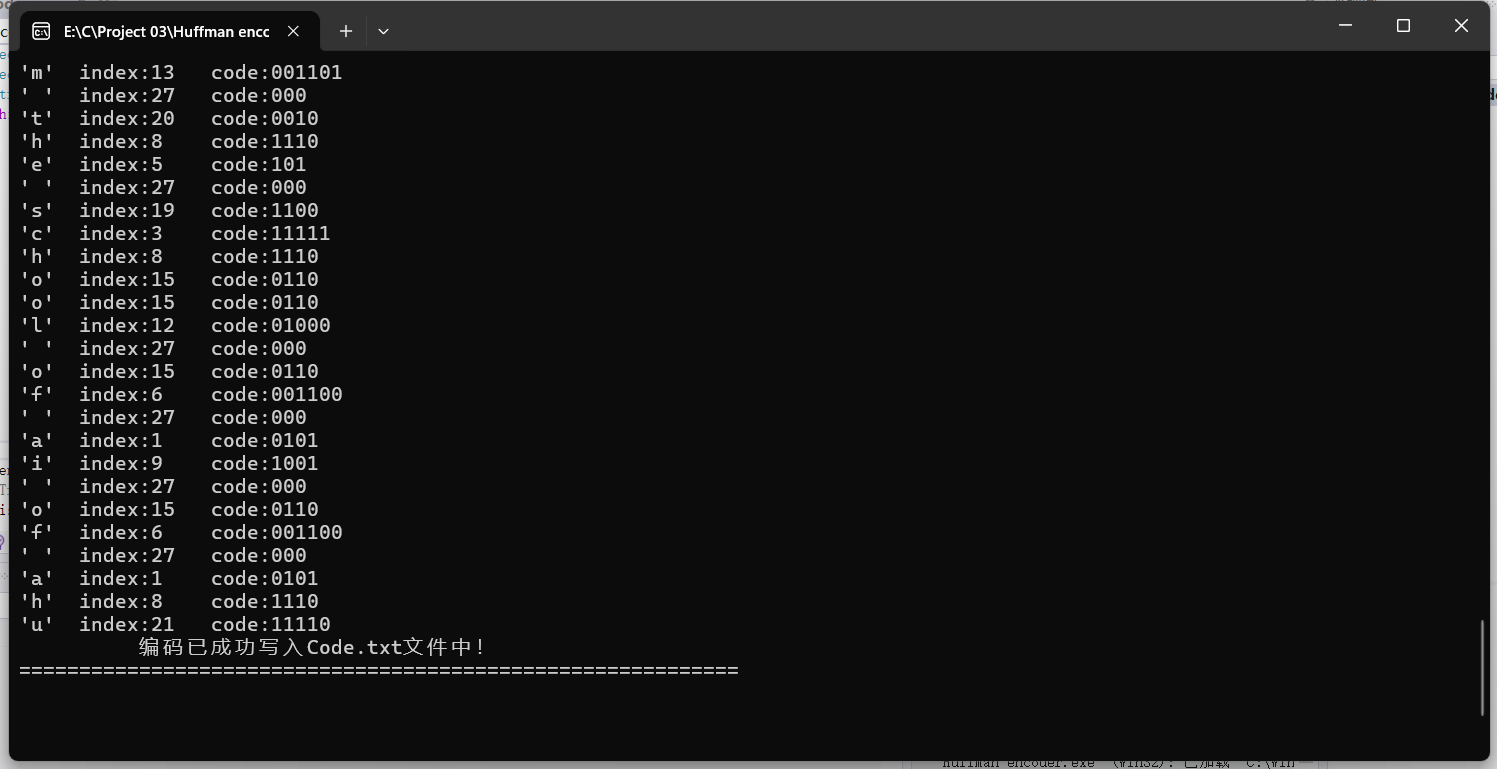
}

cout << "'" << character << "': " << weight << endl;

characters.push\_back(character[0]);

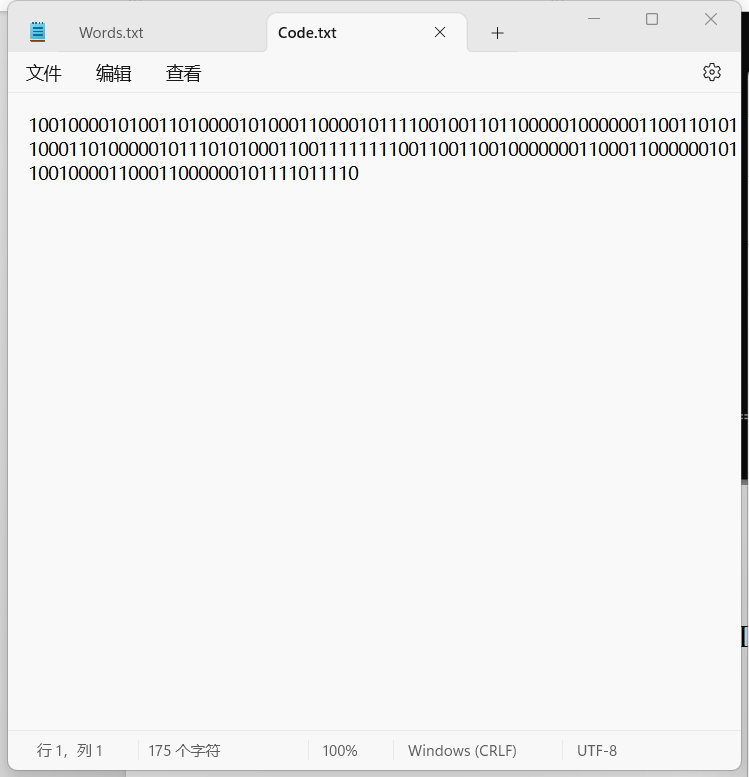
weights.push\_back(weight);

}



此时的运行结果是正确的

编码结果如下图：



## 4.3步骤2

对文件CodeFile.data中的代码进行解码形成原文，存入文件Textfile.txt

输出CodeFile.data及其原文Textfile.txt。

代码中存在的问题：

解码逻辑：现在代码会正确地遍历哈夫曼树，从根节点开始，根据每个编码位移动到左子树或右子树，并在到达叶节点时输出对应的字符。

HuffManCode结构：code字段从int改为string，这是必要的，因为哈夫曼编码是由0和1的序列组成，应该用字符串处理。

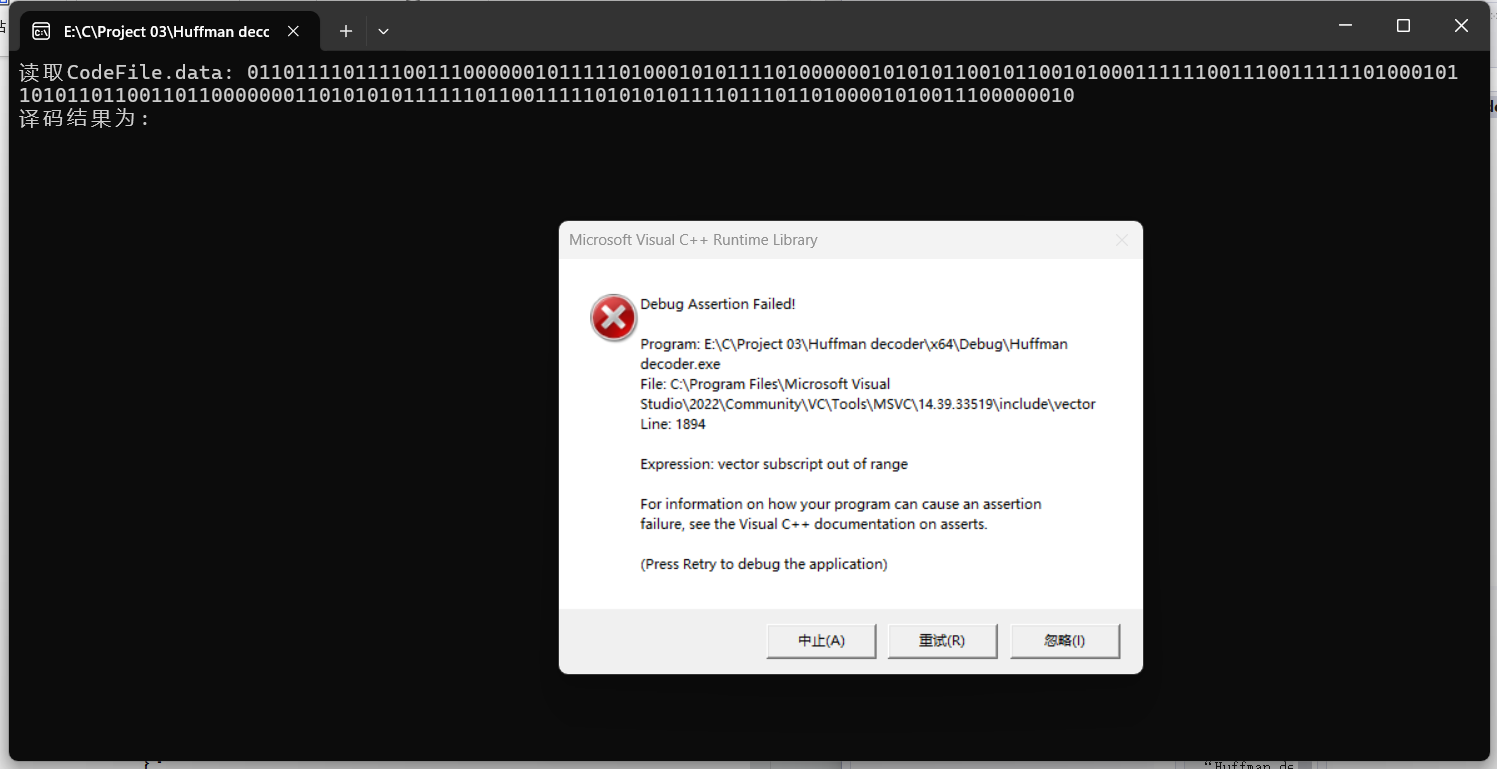
输出：在找到一个有效的哈夫曼编码后，立即将其写入文件并重置到根节点继续解析。

struct HuffManCode {

string ch;

string code;

};



错误信息表明在运行程序时，尝试访问vector的索引超出了其有效范围。这种类型的错误通常发生在使用数组或容器元素时，没有正确检查索引是否有效。

如果节点的lchild或rchild值是0（代表没有子节点），但代码尝试用这些值作为索引访问数组，将引发越界错误。在哈夫曼树中，0通常不是有效的数组索引，因为C++中的数组索引从0开始，但在哈夫曼树表示中，索引从1开始。

因此修改代码：

vector<HuffmanTreeNode> readHuffmanData() {

vector<HuffmanTreeNode> nodes;

ifstream infile("E:/C/Project 03/HuffmanTree.data");

nodes.push\_back({ 0, 0, 0, 0 }); // 虚拟零节点，索引从1开始

int weight, parent, lchild, rchild;

while (infile >> weight >> parent >> lchild >> rchild) {

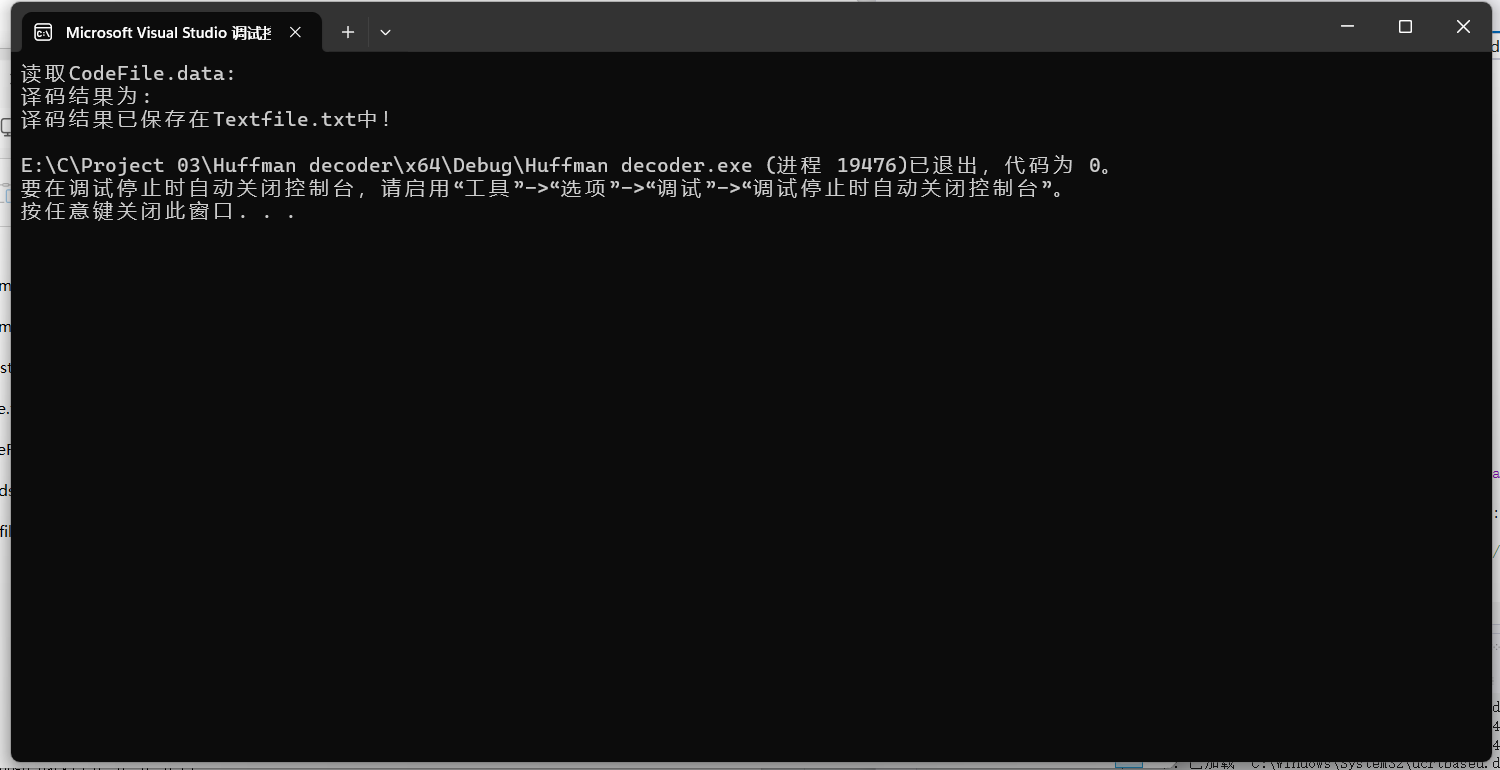
nodes.push\_back({ weight, parent, lchild, rchild });

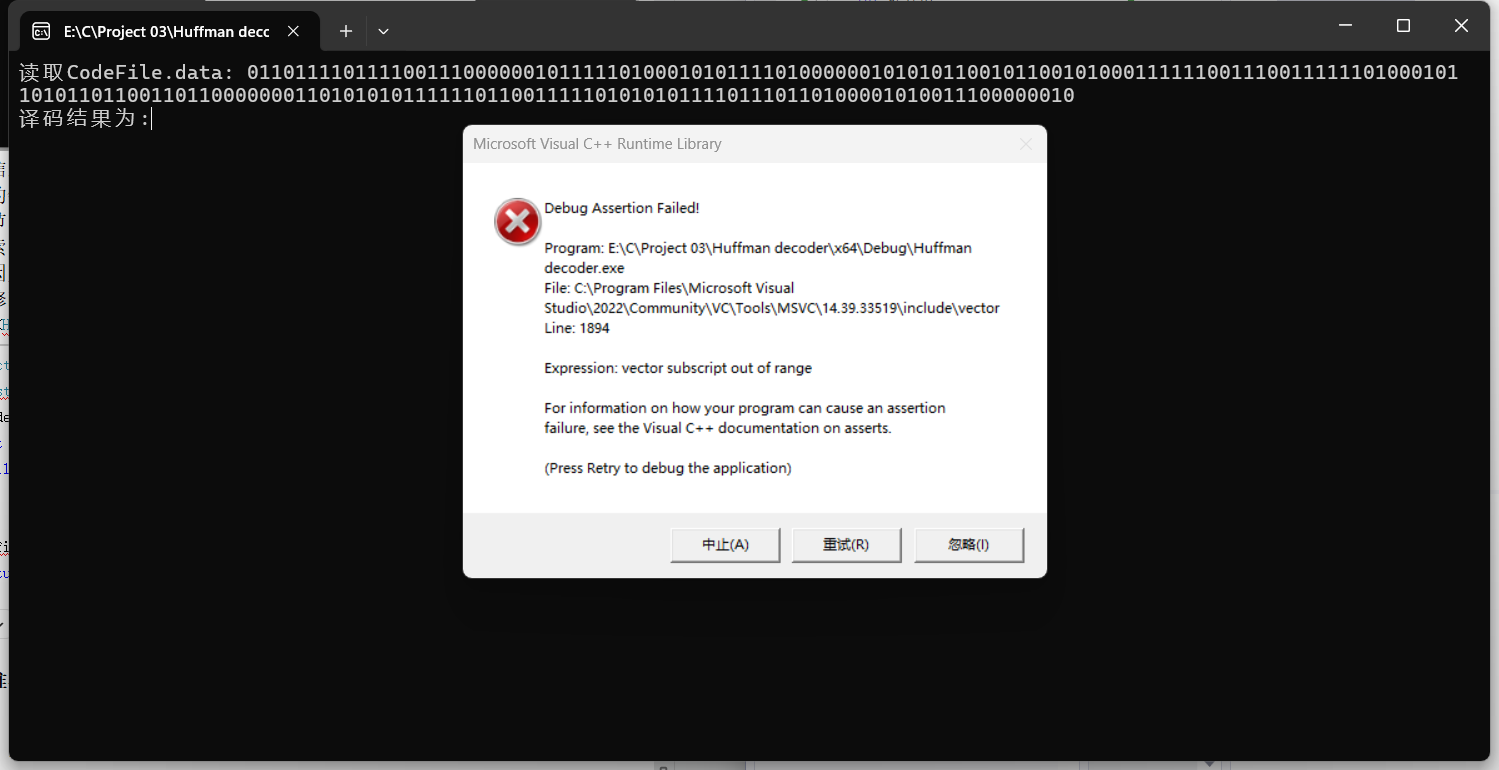
}

infile.close();

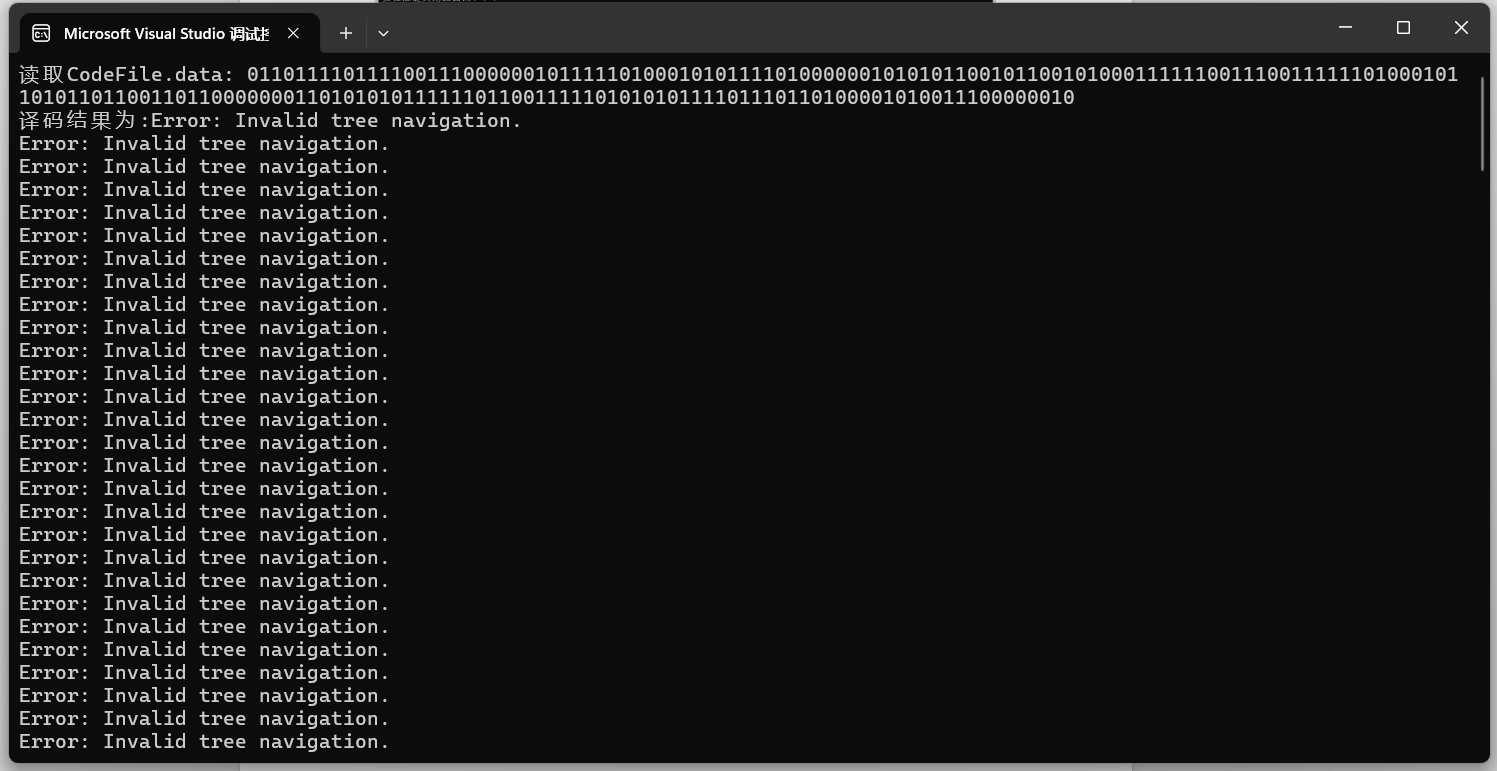
return nodes;

}





仍然报错，增加检查



无法正确解码

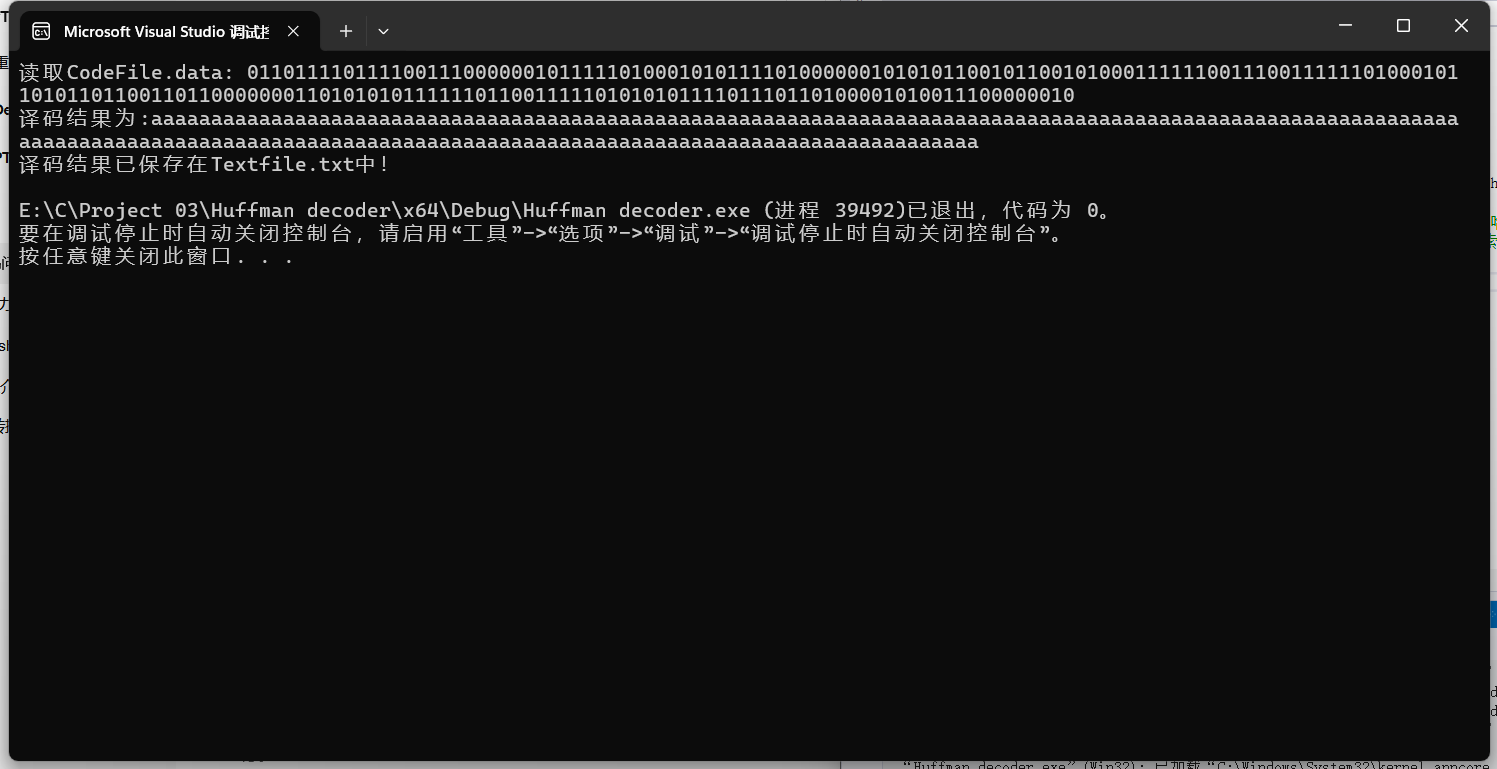
可以检查的地方：

检查哈夫曼树数据文件：确保 "HuffmanTree.data" 文件中的数据按照正确的格式排列，每一行代表一个节点，包括权重、父节点索引、左子节点索引和右子节点索引。特别要注意根节点的信息，它应该没有父节点。

检查哈夫曼编码数据文件：确认 "HCode.data" 文件中的数据按照正确的格式排列，每一行包含一个字符和它对应的哈夫曼编码。

检查程序的文件路径：确保程序能够正确地找到和读取这两个文件。你可以尝试输出文件路径来检查程序是否正确地指向了这两个文件。

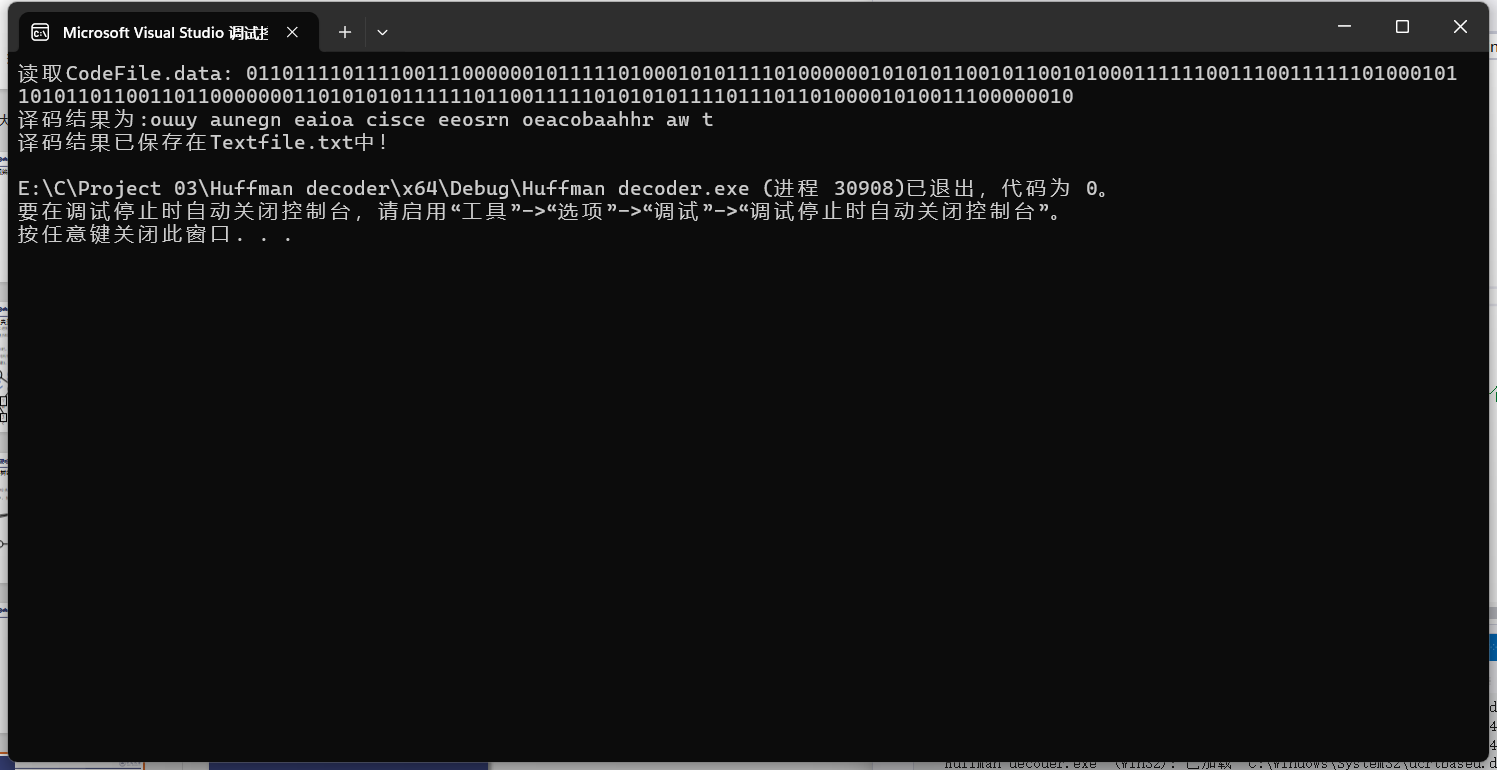
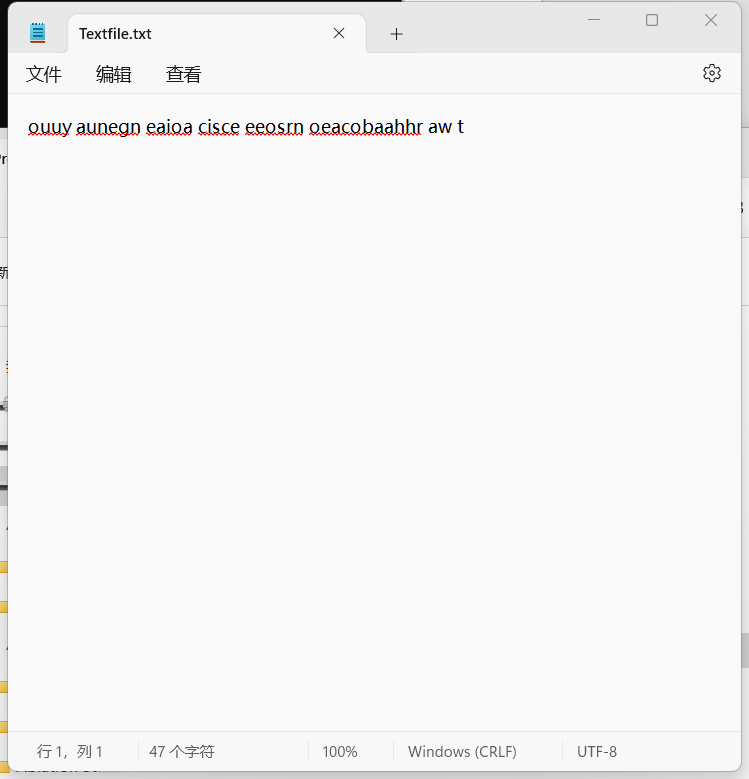
检查数据的正确性：在程序中添加一些输出语句，输出从文件中读取的哈夫曼树数据和哈夫曼编码数据，然后手动比较一下数据是否和期望的一致。



修改解码逻辑之后发现，输出不符合预期

实际上，解码程序可以只依赖于HCode.data

在正确修改程序之后 程序能够正确解码

****

能够正确解码

# 五、实验总结

这次实验耗时较长，主要是针对哈夫曼编码的设计与实现，其中包括编码、译码以及建立哈夫曼树等任务。在实验过程中，我遇到了一些挑战，特别是在任务三中，需要从文件中读取字符和对应的权值，然后建立哈夫曼树。在处理文件路径和空格统计方面，我遇到了一些问题，但通过仔细检查代码和调试，最终解决了这些困难。

在任务一中，我成功地利用给出的字符和字符频率对字符串进行了哈夫曼编码，并输出了编码后的二进制码。这涉及到对字符频率进行排序、建立哈夫曼树和编码的过程，需要考虑各种特殊情况和边界条件。

在任务二中，我尝试统计了包含大小写英文字母的字符串文件中的字符频率，并进行了相应的修改和调试。通过自动判断大小写并统一字符频率统计，我成功地完成了任务，得到了正确的结果。

在任务三中，我从文件中读取字符和对应的权值，建立了哈夫曼树，并对文件中的文本进行了编码和解码操作。在处理文件路径和空格统计时，我遇到了一些问题，但通过仔细分析和修改代码，最终成功解决了这些困难。

通过这次实验，我不仅加深了对哈夫曼编码原理的理解，还提高了自己的编程能力和问题解决能力。在未来的学习和工作中，我将继续努力，不断提升自己的技能水平，为实现更多的目标而努力。