数据结构课程设计

项目说明文档

**关键字检索系统**



同济大学

Tongji University

姓名： 林觉凯

学号： 2253744

指导老师： 张颖

学院专业： 软件学院 软件工程

**目录**

**1.项目分析-------------------------------------------------------------------------3**

**1.1 项目背景分析-------------------------------------------------------------------------3**

**1.2项目功能分析--------------------------------------------------------------------------3**

**1.2.1项目功能要求--------------------------------------------------------------------------------3**

**1.2.2项目输入要求--------------------------------------------------------------------------------3**

**1.2.3项目输出要求--------------------------------------------------------------------------------3**

**1.2.4项目示例--------------------------------------------------------------------------------------3**

**2.项目设计-------------------------------------------------------------------------4**

**2.1 数据结构设计-------------------------------------------------------------------------4**

**2.2类设计-----------------------------------------------------------------------------------4**

**2.2.1字典树结点类(Trienode)-------------------------------------------------------------------4**

**2.2.2字典树类(Trietree)--------------------------------------------------------------------------4**

**2.3成员与操作设计-----------------------------------------------------------------------5**

**2.4系统流程设计--------------------------------------------------------------------------5**

**3.项目实现-------------------------------------------------------------------------6**

**3.1 三个基础函数的实现----------------------------------------------------------------6**

**3.2 字典树基本功能的实现-------------------------------------------------------------6**

**3.3 文件处理的实现----------------------------------------------------------------------8**

**3.4 main函数中搜索的实现 ------------------------------------------------------------9**

**4.项目测试------------------------------------------------------------------------10**

**5.项目的心得与体会------------------------------------------------------------11**

**1.项目分析**

**1.1项目需求分析**

建立一个文本文件，文件名由用户用键盘输入，输入一个不含空格的关键字，统计输出关键字在文本中的出现次数。

**1.2 项目功能分析**

**1.2.1项目功能要求**

本项目的设计要求可以分成两个部分实现：首先建立一个文本文件，文件名由用户用键盘输入；然后输入一个不含空格的关键字，统计输出该单词在文本中的出现次数。在统计关键词出现频次的过程中，要做到大小写不敏感、前缀后缀敏感（大小写相同视为同一个词，带有不同前缀后缀（比如-ing,-ed,-er,-or，-ture，等等）的视为不同的词）。

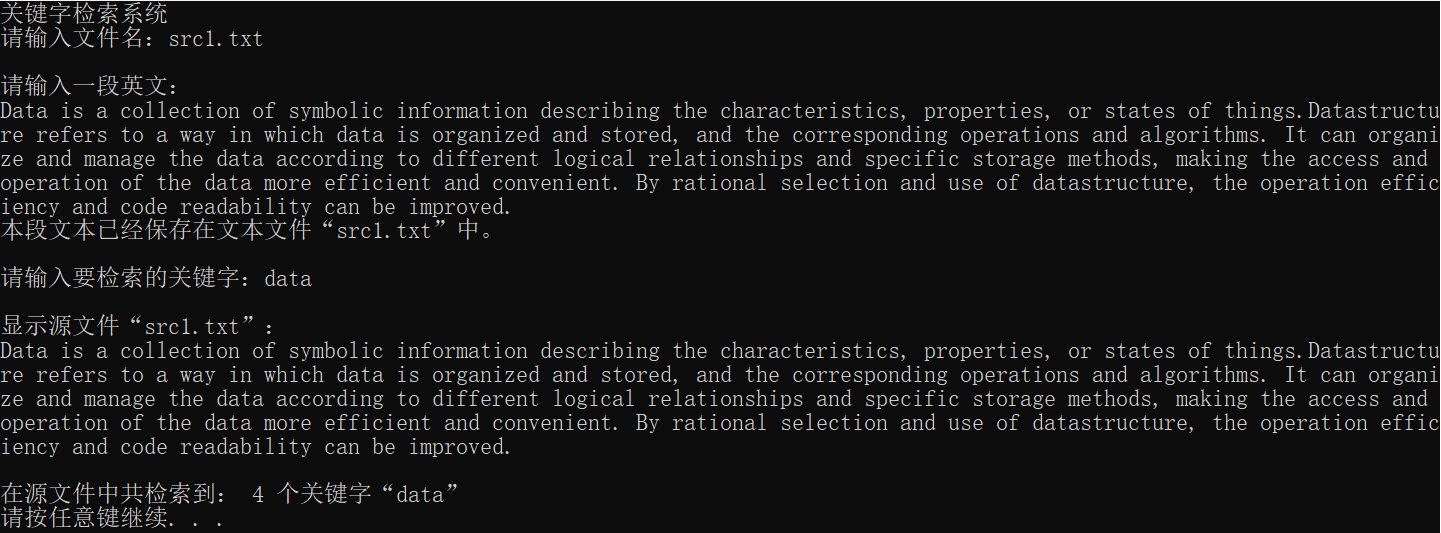
**1.2.2项目输入要求**

首先建立一个文本文件，文件名由用户用键盘输入；然后输入一个不含空格的关键字。输入为一个文本文件的名称、具体的文本内容和要检索的关键字。

**1.2.3项目输出要求**

按照用户的要求，输出该关键词出现的频次。

**1.2.4项目示例**



解释：依据大小写不敏感、前缀后缀敏感的原则，data和Data视为同一个词，data和datastructure视为不同的词。

**2.项目设计**

**2.1 数据结构设计**

此次项目需要统计在一个文本中某一个关键字出现的频次，由于要避免这个关键字可能是某一个长字符中的某一段所引起的不必要的计算，使用本题不能采取简单的字符串匹配算法或者高级优化过的KMP算法。

字典树（Trie树）是一种多叉树数据结构，用于高效地存储和检索字符串集合。字典树的每个节点表示一个字符，从根节点到叶子节点的路径构成一个字符串。根节点不包含字符，每个非根节点都包含一个字符。字典树的一个重要特点是共享公共前缀，这使得它在存储和查找字符串时非常高效。本程序使用字典树这一数据结构来实现检索关键字出现频次的目的。

**2.2类设计**

为了实现此次字典树，我设计了两个类：字典树结点类和字典树类，而两个类之间的耦合关系可以采用嵌套、继承等多种关系。本程序也构造了这两个基本的字典树结点和字典树类来进行相应的操作。

**2.2.1字典树结点类**(Trienode)

\*每个字典树结点包含着指向下一个节点的26个指针和该结点的词频。

class Trienode

{

public:

int Word\_times;

Trienode\* Next[NUM\_OF\_WORDS];

Trienode()

{

Word\_times = 0;

for (int i = 0; i < NUM\_OF\_WORDS; i++)

Next[i] = NULL;

}

};

**2.2.2字典树类**(Trietree)

class Trietree

{

public:

Trietree();

void Insert(string word);

int Search(string word);

private:

Trienode\* root;

};

**2.3成员与操作设计**

字典树结点类的公有成员为该词出现的频数和指向下一个字母的26个指针；家谱树类的私有成员为根节点，公有操作为插入词语和查找词频。

**2.4系统流程设计**

系统操作的整体流程大致如下：

**程序开始运行**

**输入文件名称**

**建立文件并打开**

**输入文章内容**

**写入文件并关闭**

**打开文件**

**输入关键词**

**关键词查找**

**显示查找结果**

**3.项目实现**

**3.1 三个基础函数的实现**

//判断某一个字符是否为字母

bool Is\_alphabet(char ch)

{

if ((ch >= 'a' && ch <= 'z') || (ch >= 'A' && ch <= 'Z'))

return true;

else

return false;

}

//将某个字符大写转小写的函数

char Transform\_lowcase(char ch)

{

if (ch >= 'A' && ch <= 'Z')

ch += ('a' - 'A');

return ch;

}

//将一个字符串转化为小写

string Transform\_str(string str)

{

string retstr = "";

for (unsigned int i = 0; i < str.length(); i++)

retstr += Transform\_lowcase(str[i]);

return retstr;

}

由于要做到在搜索关键词出现的频次的时候做到大小写不敏感，这里先将文本和关键词全部转化为小写字母的形式。同时还需要有一个判断该字符是否为英文字母的函数来忽略文本中的标点符号和空格。

**3.2 字典树基本功能的实现**

//将单词word插入字典树

void Trietree::Insert(string word)

{

Trienode\* current = root;

for (unsigned int i = 0; i < word.length(); i++)

{

char element = word[i];

if (current->Next[element - 'a'] == NULL)

{

current->Next[element - 'a'] = new Trienode;

if (current->Next[element - 'a'] == NULL)

{

cout << "内存分配失败！" << endl;

system("pause");

return;

}

}

current = current->Next[element - 'a'];

}

current->Word\_times++;

}

该函数的功能是将某一个单词插入到字典树中去，并且相应的词频得到相应的增加。首先设定根节点为初始的current结点，利用循环将word中的每一个字符插入到字典树内。如果某个结点指向下一个结点中的26个指针中相应的指针不存在，就相应地给它分配一个新的结点，从而将该词中的每一个字母都插入进去，再是current指针后移；如果存在，就指向下一个结点，从而达到遍历这个单词中的每一个字母的效果。最后将这个词的词频加一。

//搜索单词word出现的次数

int Trietree::Search(string word)

{

Trienode\* current = root;

for (unsigned int i = 0; i < word.length(); i++)

{

char element = word[i];

if (current->Next[element - 'a'] == NULL)

return 0;

current = current->Next[element - 'a'];

}

return current->Word\_times;

}

该函数的功能是搜索某一个单词再字典树中出现的频率。由于再先前的操作中已经将每一个单词的信息和频率储存在字典树的相应的位置中去，这个函数将其中储存的词频树搜索并打印出来即可。同样先使current指针指向root结点。利用循环查找该词中的每一个字母是否再字典树中都存在，如果某个字母不存在的话就直接return 0，表示该词不存在；如果全都遍历完，说明该词存在，并显示此时current结点的词频。

**3.3 文件处理的实现**

部分源代码：

FILE\* fp;

char File\_name[Max\_filename];

cout << "请输入文件名：";

cin.getline(File\_name, Max\_filename);

fp = fopen(File\_name, "w"); //建立并打开空文件File\_name

if (fp == NULL)

{

cout << "文件打开失败" << endl;

system("pause");

return 0;

}

cout << endl << "请输入一段英文：" << endl;

char File\_content[1024];

cin.getline(File\_content, 1024);

fputs(File\_content, fp); //将输入的内容保存在文本文件中

cout << "本段文本已经保存在文本文件“" << File\_name << "”中。" << endl << endl;

fclose(fp);

string KeyWord;

cout << "请输入要检索的关键字："; //获得关键字

cin >> KeyWord;

cout << endl << "显示源文件“" << File\_name << "”：" << endl; //显示文本文件中的内容

fp = fopen(File\_name, "r");

fgets(File\_content, 1024, fp);

cout << File\_content << endl;

本题需要建立文件、打开文件、文件写入内容等等有关文件的操作，我们分别利用fopen()函数和fclose()函数来打开和关闭文件，利用fputs()和fgets()函数来实现字符串文本的写入和读取操作。

**3.4 main函数中搜索的实现**

Trietree trietree;

int i = 0;

while (File\_content[i] != '\0')

{

if (!Is\_alphabet(File\_content[i])) //如果文章中的某个位置不是单词，则跳过

{

i++;

continue;

}

string current = "";

while (Is\_alphabet(File\_content[i]))

{

current += Transform\_lowcase(File\_content[i]); //将单词current转化为小写后

i++;

}

if (current != " ") //单词末尾遇到空格，说明这个单词结束

trietree.Insert(current); //将单词插入字典树

}

string Key\_word = Transform\_str(KeyWord); //将要搜索的关键词转化为小写

cout << endl << "在源文件中共检索到： " << trietree.Search(Key\_word) << " 个关键字“" << KeyWord << "”" << endl;

main函数中在输入关键词后进行检索前需要将读取的文本内容写入到实例化的字典树trietree中，我们利用循环，从文本起点开始，每遇到一个单词，就将这个单词的每一个字母变为小写之后写入到相应的位置current字符串中去；每遇到“ ”，就说明该单词已经结束，将这个单词用Insert函数插入到字典树中去。之后在利用Search函数检索该KeyWord出现的频率。

**4.项目测试**

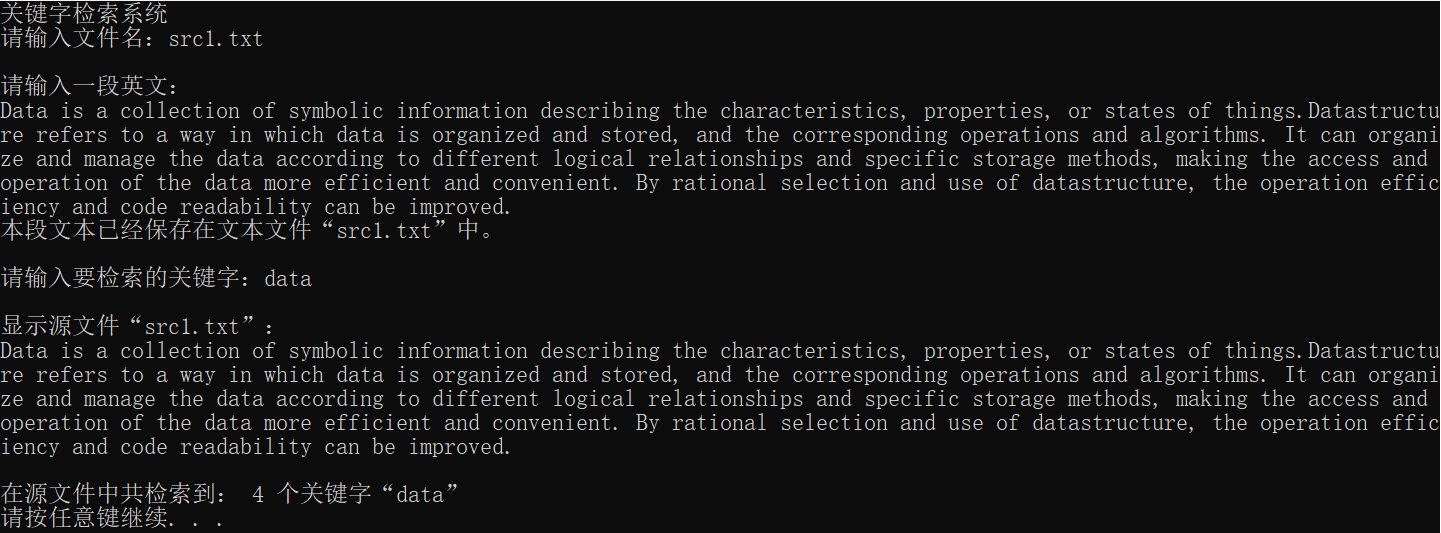
测试用例1：

预期效果：考虑到data与Data是同一个检索词，和datastructure不是同一个检索词

关键词：data

共检索到4个关键词“data”

Data is a collection of symbolic information describing the characteristics, properties, or states of things.Datastructure refers to a way in which data is organized and stored, and the corresponding operations and algorithms. It can organize and manage the data according to different logical relationships and specific storage methods, making the access and operation of the data more efficient and convenient. By rational selection and use of datastructure, the operation efficiency and code readability can be improved.

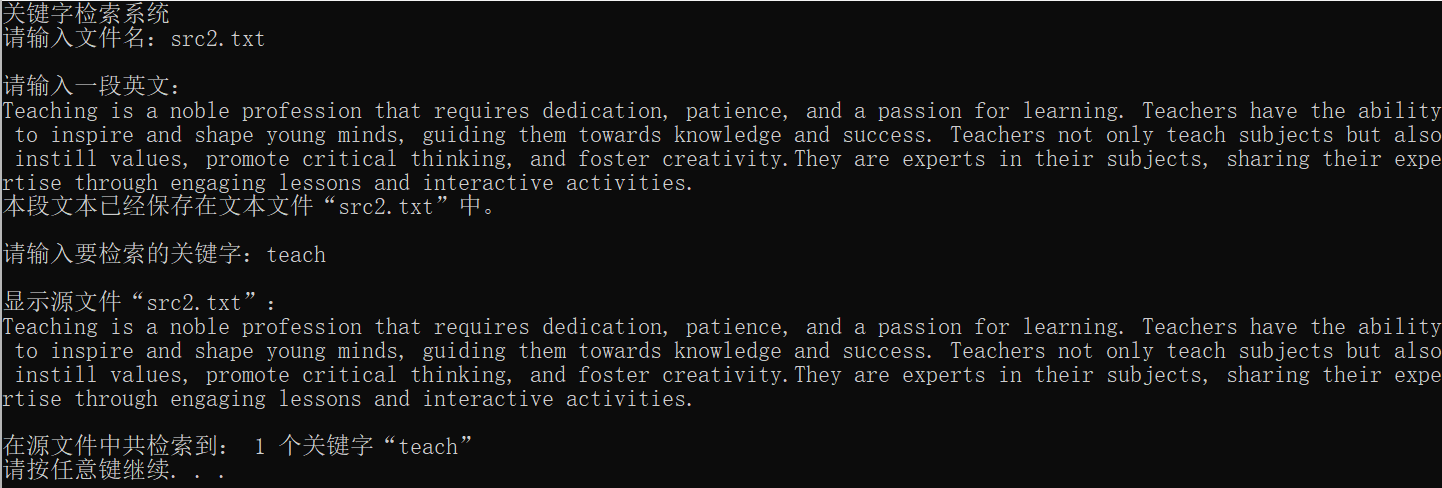


测试用例2：

Teaching is a noble profession that requires dedication, patience, and a passion for learning. Teachers have the ability to inspire and shape young minds, guiding them towards knowledge and success. Teachers not only teach subjects but also instill values, promote critical thinking, and foster creativity.They are experts in their subjects, sharing their expertise through engaging lessons and interactive activities.

预期效果：考虑到teach与teaching、teacher不是同一个检索词

关键词：teach

共检索到1个关键词“teach”

**5.项目心得与体会**

本次项目我学习到了比较多的东西。首先，我通过查找资料学到了有关于文件操作的一些函数，比如fopen()、fclose()、fputs()和fgets(),这些函数帮助我实现该项目部分操作的完成；本次项目最主要的是我学习到了一种新的树的类型——字典树，字典树是一种多叉树，主要有插入字符串和查找字符串词频的函数操作，可以帮助我们更好地检索关键词。