**实验四 文本编辑**

**计算机166班 李彪 6130116194**

## 4.1 问题描述

要求建立一个文本文件，每个单词子串不包含空格且不跨行，单词子串由字符序列构成，且区分大小写；统计给定单词子串在文本文件中出现的总次数；检索输出某个单词子串出现在文本中的行号、在该行中出现的次数以及位置。

## 4.2 输入与输出

输入：从键盘循环输入字符串，创建到文本文件中

从键盘输入要计数的单词子串以及要定位的单词子串

输出：将统计结果以及定位的结果输出到屏幕

## 4.3 需求分析

1.定义字符串的存储结构；

2.编写主函数、菜单函数，确定函数之间的接口；

3.编写创建文件函数、单词匹配函数、单词计数函数、单词定位函数。使用多套数据，进行系统调试；

## 4.4 开发工具与环境

硬件设备：微型计算机系统

操作系统debian linux

编译器 g++

## 4.5 概要设计

1. 数据结构定义

#define maxsize 10000

typedef unsigned char sstring[maxsize];

1. 各函数模块

void wordcount();

操作结果：统计单词出现的总次数

void creattextfile();

操作结果：创建文本文件

int choose( );

操作结果：输入选项

int index\_BF(sstring s,sstring t,int pos);

操作结果：串匹配，返回匹配的位置

void strfind();

操作结果：定位单词出现在某行，某列以及次数

## 4.6 详细设计

#include <fstream>

#include <iostream>

#include <cstring>

// #include <stren>

using namespace std;

const int maxsize = 10000;

string content[maxsize]; // 字符串数组，存储文件内容

int lineIndex = 0;

typedef char sstring[maxsize];

// using namespace std;

void createTextFile()

{

// C++ 通过以下几个类支持文件的输入输出：

// ofstream: 写操作（输出）的文件类

// ifstream: 读操作（输入）的文件类

// fstream: 可同时读写操作的文件类

sstring data;

int i = 0;

char temp;

ofstream file;

file.open("exp.txt");

cout << "Write file, input '#' to stop: " << endl;

while (true)

{

temp = getchar();

if (temp != '#')

{

data[i++] = temp;

}

else

{

break;

}

}

getchar();

file << data << endl;

file.close();

}

int choose()

{

int c;

cout << "\t文本编辑系统菜单🈳️" << endl

<< endl;

cout << "\t1. 创建文本文件" << endl;

cout << "\t2. 文件单词计数" << endl;

cout << "\t3. 文件单词定位" << endl;

cout << "\t0. exit" << endl;

cout << "请输入你的选择: ";

cin >> c;

cin.ignore();

return c;

}

void getFileContent()

{

sstring data;

int index = 0;

ifstream infile("exp.txt");

while (infile.getline(data, maxsize))

{

content[index++] = data;

}

lineIndex = index;

}

int getLength(sstring s)

{

return strlen(s);

}

int getIndex(string s, sstring t, int pos)

{

// 返回子串t在主串s中第pos个字符之后的位置。

// 1 <= pos <= sLen

// 匹配失败返回-1

int i = pos - 1, j = 0, sLen = s.length(), tLen = getLength(t);

while (i < sLen && j < tLen)

{

if (s[i] == t[j])

{

i++;

j++;

}

else

{

i = i - j + 1;

j = 0;

}

}

if (j == tLen)

{

return i - tLen + 1;

}

return 0;

}

void wordCount()

{

sstring queryStr;

int i = 0;

int count = 0; // 计数

cout << "输入你要计数的字符串: ";

cin >> queryStr;

while (i < lineIndex)

{

int index = 1;

while (true)

{

index = getIndex(content[i], queryStr, index) + 1;

if (index - 1 == 0)

{

break;

}

count++;

}

i++;

}

cout << count << endl;

}

void strFind()

{

sstring s;

int row, col, index = 1;

cout << "请输入需要定位的字符串: ";

cin >> s;

for (int i = 0; i < lineIndex; i++)

{

int count = 0;

while (true)

{

index = getIndex(content[i], s, index) + 1;

if (index - 1 != 0)

{

count++;

printf("字符串 %s 在第 %d 行，第 %d 列.\n", s, i + 1, index - 1);

continue;

}

break;

}

printf("在第%d行共出现了%d次\n", i + 1, count);

}

}

int main()

{

getFileContent();

bool t = true;

while (t)

{

switch (choose())

{

case 1:

{

createTextFile();

getFileContent();

cout << endl;

break;

}

case 2:

{

wordCount();

cout << endl;

break;

}

case 3:

{

strFind();

cout << endl;

break;

}

case 0:

{

t = false;

cout << "exit success" << endl;

break;

}

default:

{

cout << "输入有误，请重新输入" << endl;

}

}

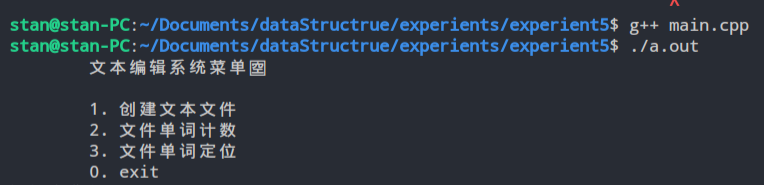
}

return 0;

}

**实验运行结果**

编译，执行可执行文件 a.out 选择功能

****