《数据结构课程设计》

实验报告

姓名 艾斯兰

班级: 2018291301

学号: 2018290011

2020年 月 日

目录

[题目：基于顺序存储的线性表、链式存储的线性表和散列表的低频词过滤系统 1](#_Toc44080246)

[(一) 课程设计内容： 1](#_Toc44080247)

[(二) 系统功能要求： 1](#_Toc44080248)

[(三) 实现方案设计及说明 1](#_Toc44080249)

[(四) 数据结构说明 2](#_Toc44080250)

[(五) 主要模块设计说明 2](#_Toc44080251)

[(六) 运行截图 2](#_Toc44080252)

[(七) 自己的收获和程序改进的想法 2](#_Toc44080253)

[(八) 提交材料清单 2](#_Toc44080254)

# 题目：基于顺序存储的线性表、链式存储的线性表和散列表的低频词过滤系统

## 课程设计内容：

对于一篇给定的英文文章，分别利用顺序存储的线性表、链式存储的线性表和散列表（选做）三种方法来实现单词频率的统计，实现低频词的过滤，并比较三种方法的效率。

## 系统功能要求：

1. 读取英文文章文件（aa.txt）作为输入文件，从输入文件中读入数据，并识别其中的单词，过滤掉标点符号。
2. 分别利用顺序存储的线性表、链式存储的线性表和散列表（选做）构建单词的存储结构。
3. 当识别出一个单词后，若顺序存储的线性表、链式存储的线性表或者散列表（选做）中没有该单词，则在适当的位置上添加该单词；若该单词已经被识别，则增加其出现的频率。
4. 统计结束后
   1. 将文章中所有单词及其出现次数按照从高到低的次序输出到文件中（sample.txt）
   2. 将出现次数小于5次的单词在屏幕上输出，需同时输出单词和次数
5. 程序执行时，需提示用户输入要读取的文章名称，输入文章名称后开始执行并在屏幕输出和在文件中输出。

## 实现方案设计及说明

描述一下计划用什么样的程序结构来完成：

main函数做什么，main函数里面分成几个部分，分别做什么；

第一个代码 ：

用了数组结构为了存储词和单词的次数

函数:GetWord 为了取单词，swap函数排序时为了单词的交换

main函数里包含：

——读取输入输出文件

——存放由GetWord函数得到的每个单词

——两次循环为了消除同样单词和统计

——两次循环为了排序

——然后就在console输出每个单词和次数的cout操作

——关闭两个文件

第二个代码 顺序存储的线性表方式

函数:

bool Inlist(list &)

bool DestroyList(list &)

void modify(FILE \*,list &)

void sort (list&)

void swap(dataElem &,dataElem &)

void print\_(list &,FILE \*)

bool GetWord(FILE \*,char \*)

main 函数：

——Inlist函数 是初始化

——定义了两个文件指针 fpInput,fpOutput

第一个是读取文件，第二个是输出文件

——调用了modify 函数，modify 函数的功能就是读取单词存放到线性表，多个出现的单词消除，还有做统计

——sort函数 是为了排序 里面包含函数swap

——print函数 是为了输出还有线性表的内容放到fpOutput 文件，次数小于5的单词不输出也不存放。

——DestroyList 函数 it destroys the list

fclose(fpInput)

fclose(fpOutput)

第三个代码 用链式方式做的

函数:

bool SingleLinkListInit(LinkList &)

bool ListDestroy(LinkList &)

bubble\_sort(LinkList )

swap(struct LNode \* ,struct LNode \* )

count\_remove(LinkList )  
 void printlist(FILE \*,Linklist)

bool GetWord(FILE \* ,char \* )

main 函数:

——fpInput,fpOutput 两个文件

——一次循环为存放单词到连表去

——count\_remove 函数 为了统计和消除同样的单词

——printlist 输出，写到输出文件

——ListDestroy

fclose(fpInput)

fclose(fpOutput)

## 数据结构说明

初始化使用了c++ 的new 新建空间

使用了c++ 的delete

fgetc 从文件读取单词

我试用了 cstring 里面的一些结构:

strcpy-把所指向的字符串复制到 线性表或连表

strcmp – 比较两个单词

用了c++的 std::cout,和c 的printf

fopen 为了打开文件

fclose 关闭文件

## 主要模块设计说明

我第一次为了更好理解题目的功能用了最简单的自己熟练的方法写了一个代码。

我的结构如下：

typedef struct

{

char word[128];

into num =1;

}sqlist[1000];

我用了数组结构的方式存储读取的词和次数。然后为了存储我写了这段代码：

sqlist L;

int cl=0; //长度

while(!fof(fpInput) //文件不结束时

{

char sWord[128];

if(GetWord(fpInput,W)

{

strcpy(L[cl].word,sWord);

cl++; //读取后增加长度

}

}

存储后我只显示了我的结果

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

{

std::cout<<L[i].word<<L[i].num<<std::endl;

显示成功后我写了一段代码为了比较，增加和删除相同的词。

代码如下：

int j;

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

{  
 j=i+1; // j是第一个比较词的下一个位置

while(j<cl)

{

if(strcmp(L[i].word,L[j].word)==0) //如果相同

{

strcpy(L[j].word,L[cl-1].word); //把第二个比较的词放到最后

L[i].num++; //增加第一个比较词的次数

cl--; //长度减1，为了删除刚放的词

j-- ； //长度减少了，j 的 直也要减少

}

j+; //j增加

}

}

这样我就算了每个词的次数，还有删除了相同的词。

成功这段代码后，我写了一个bubble sort ，和交换swap

程序如下

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

{

for(j=0;j<cl-i-1;j++)

{

if(L[j].num<L[j+1].num)

{

swap(L[j+1],L[j]);

}

}

}

但是做swap 时我遇到问题，因为我的结构 直接定义了一个数组类型 sqlist[ ], 所以写函数的时候就不知道怎么写函数参数的类型。所以问老师的时候它推荐给我用 这样的方式

typedef struct

{

char word[128];

into num =1;

}sqlist[1000],dataElem; /////加上了数据元素的类型 dataElem

所以写swap 的时候就没问题了

void swap(dataElem &pa,dataElem &pb)

{

dataElem temp=pa;

pa=pb;

pb=temp;

}

这样我对题目和我要做的事儿搞清楚了。

第二版

做完第一个后我用顺序存储方式做了我的实现。一开始用了如下的定义

typedef struct

{

char word[32];

int num ;

}El,dataElem;

typedef struct list

{

El \*elem;

unsigned int length;

unsigned int max;

}\*ptr;

这样我就不浪费我的空间，就是在初始化时 用new

bool Inlist(list &l)

{  
 l.elem=new El[MAX];

if(!l.elem)

{  
 return false;

}

l.length = 0;

l.max = MAX;

return true;

}

还有最后用DestroyList 函数

DestroyList(list &L)

{

if（L.elem）

{

free(L.elem);

return true;

}

else return false;

}

如下一段代码的其本思想和功能描述：

while(!feof(fpInput))

{

char sWord[128];

if (GetWord(fpInput,sWord ))

{

int i=0;

for(i=0;i<L.length;i++)

{

if(strcmp(L.elem[i].word,sWord)==0)

{

L.elem[i].num++;

break;

}

if(i==L.length)

{

if(L.length<L.max)

{

strcpy(L.elem[L.length].word,sWord);

L.elem[L.length].num=1;

L.length++;

}

}

} }

在这儿 if(strcmp(L.elem[i].wprd,sWord)==0)

线性表的第i个词与读取的词比较，若他们相等那它的次数加加，后break;

若他们不相同那就读取的词放在线性表尾巴上，尾巴上词的次数加1

长度也加1

这样我们就不用删除相同的词了，就直接放，直接加次数，

做这个后只要写排序就行，排序跟第一个代码一样，所以不用解释了

第三版

第三个代码我用了连表方法实现

线定义了node结构

typedef struct {

char word[32];

int count;

}el;

typedef struct LNode

{

el data;

struct LNode \*next;

}LNode;

typedef LNode \*LinkList;

代码的基本思想：我先把每一个词放在连表去

如下代码：

LinkList temp,p;

if(SingleLinkListInit(L))

{

temp=L;

while(!feof(fpInput))

{

char sWord[128];

if (GetWord(fpInput,sWord ))

{

p=(LinkList)new(LNode);

strcpy( p->data.word,sWord);

p->data.count=1;

temp->next=p;

temp=p;

}

}

temp->next=NULL;

}

先定义一个LinkList temp,LinkList p; //node指针

初始化成功后新建空间 p=(LinkList)new(LNode);

以后就把读取的词放在 p->data.word,p->data.count=1,次数增加

temp->next=p,temp=p 是给L->next p的node,

循环结束后我们就存放了所有的词，temp->next=NULL 是L->next=NULL;

下一步是为了删除相同的词，还有增加次数

count\_remove 函数

代码如下

void count\_remove(LinkList L)

{

LinkList ptr,p2,temp;

ptr=L->next;

while( ptr->next!=NULL&&ptr!=NULL)

{

p2=ptr;

while(p2->next!=NULL)

{//cout<<p->data.word<<endl;

//cout<<p2->data.word<<endl;

if(strcmp(ptr->data.word,p2->next->data.word)==0)

{

// cout<<p2->data.word;

temp=p2->next;

p2->next=p2->next->next;

delete(temp);

ptr->data.count++;

}

else

p2=p2->next;

}

ptr=ptr->next;

}

}

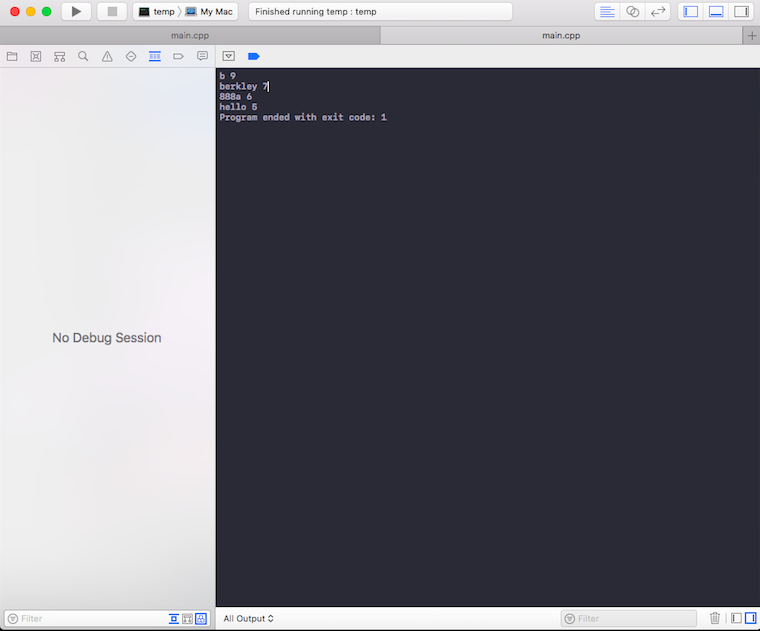
我不去解释了因为这个思想和第一个代码的思想一样

然后排序，就可以了

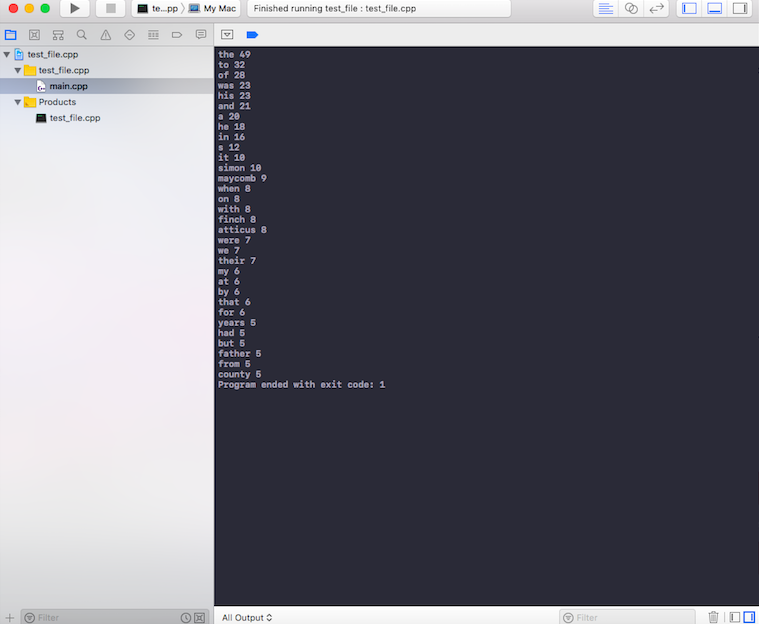
当然这个代码可以简化，像第二个方法一样做，但是没去改代码。

## 运行截图

输入文件aa.txt 文件



输入文件sample\_input.txt



## 自己的收获和程序改进的想法

写完这些代码后我对线性表和连表有了清晰的视野，还有对这些内容有了很大提高。对文件也比以前熟悉得更好。

整个代码由几个小块组成的，以后如果需要写一个大的代码，我也就从小块去写，就是一步一步地去做。

也非常感谢老师的助教。

## 提交材料清单

|  |  |  |
| --- | --- | --- |
| 提交材料 | 是否提交 | 文件名称 |
| 1、实验报告 | □ |  |
| 2、可执行程序 | □ |  |
| 3、源程序，如果是多个文件要压缩到一个文件 | □ |  |