**实验二 文章相似度检验**

1. 实验题目：给出两篇文章求其相似程度。
2. 解题思路：

1）首先将文章按段，句，字三维动态存入三维数组。

2）1. 文章相似度为最大段落相似度之和/二者最大段落数。

2. 段落相似度为最大句子相似度之和/二者最大句子数。

3. 句子相似度为最长公共字串之和/二者最小字符数。

3) 最长公共字串之和：首先生成串1和串2的矩阵，在矩阵中求最长对角线，删除公共子串，重复操作，直至串1 串2再无公共子串，求得所有公共子串之和。

1. 运行结果

#include **<iostream>**#include**<stdlib.h>**#include**<malloc.h>**#include **<process.h>**#include **<stdio.h>**#include **<string.h>  
typedef struct** {  
 **int** \*maxlen;  
 **int** \*xpos;  
 **int** \*ypos;  
 **int** size;  
 **int** MaxSize;  
}LIST;  
**void** Initlist(LIST &S);  
**void** AgainMalloc(LIST &S);  
**void** ListInsert(LIST &S, **int** a, **int** b, **int** c);  
**bool** GetMartix(**char** \*\*&C, **char** \*str1, **char** \*str2);  
**void** GetDiag(**char** \*\*C,**char**\* str1,**char**\* str2, LIST &S);  
**void** printlist(LIST &S);  
**void** GetMaxLen(LIST &S, **int** &xmax, **int** &ymax, **int** &lenmax);  
**void** StrDelete(**char** \*&str, **int** pos, **int** len);  
**double** matchS(**char** \*str1, **char**\*str2);*//求句子相似度***void** readin(**char** \*\*\*&C,**const** char\* p,**int** &pmax,**int** &smax,**int** &cmax);*//将文章读入三维数组， pmax，smax，cmax分别为其段落，句子，字符的最长标号*  
**double** AriMatch(**int** Apmax,**int** Bpmax,**int** Asmax,**int** Bsmax, **int** Acmax, **int** Bcmax, **char** \*\*\*A,**char** \*\*\*B);*//求文章相似度***double** ParaMatch(**char** \*\*A, **char**\*\*B,**int** Asmax,**int** Bsmax, **int** Acmax, **int** Bcmax);*//求段落相似度***int** main() {  
 **char** \*\*\*A;  
 **int** Apmax=0;  
 **int** Asmax=0;  
 **int** Acmax=0;  
 readin(A,”test1.txt”,Apmax,Asmax,Acmax);**char** \*\*\*B;  
 **int** Bpmax=0;  
 **int** Bsmax=0;  
 **int** Bcmax=0;  
 readin(B,”test2.txt”Bpmax,Bsmax,Bcmax);**float** m=**float**(100\*AriMatch(Apmax,Bpmax,Asmax,Bsmax,Acmax,Bcmax,A,B));**int** i,j;  
 **for**(i = 0; i <= Apmax; ++i)  
 **for**(j = 0; j <= Asmax; ++j)  
 free(A[i][j]);  
 **for**(i = 0; i <= Apmax; ++i)  
 free(A[i]);  
 free(A);  
 **for**(i = 0; i <= Bpmax; ++i)  
 **for**(j = 0; j <= Bsmax; ++j)  
 free(B[i][j]);  
 **for**(i = 0; i <= Bpmax; ++i)  
 free(B[i]);  
 free(B);  
 printf(**"%.2f%%"**,m);  
}  
**bool** GetMartix(**char** \*\*&C, **char** \*str1, **char** \*str2)  
{  
 **int** i,j;  
 **if**(C!=**NULL**)  
 {C=**NULL**;  
 }  
 **int** len1=strlen(str1);  
 **int** len2=strlen(str2);  
 **if**(strlen(str1)==0||strlen(str2)==0)  
 **return false**;  
 C=(**char** \*\*)malloc(strlen(str1)\***sizeof**(**char** \*));  
 **for**(i=0;i<strlen(str1);i++ )  
 {  
 \*(C+i)=(**char** \*)malloc(strlen(str2)\***sizeof**(**char**));  
 **for** (j=0;j<strlen(str2);j++)  
 {  
 **if** (\*(str1+i)==\*(str2+j))  
 \*(\*(C+i)+j) = **'1'**;  
 **else** \*(\*(C+i)+j) = **'0'**;  
 }  
 }  
 **return true**;  
}  
  
**void** Initlist(LIST &S)  
{  
 **if**(S.maxlen!=**NULL**)  
 {  
 S.maxlen=**NULL**;  
 }  
 **if**(S.xpos!=**NULL**)  
 {  
 S.xpos=**NULL**;  
 }  
 **if**(S.ypos!=**NULL**)  
 {  
 S.ypos=**NULL**;  
 }  
 S.maxlen=(**int** \*)malloc(**sizeof**(**int**));  
 S.xpos=(**int** \*)malloc(**sizeof**(**int**));  
 S.ypos=(**int** \*)malloc(**sizeof**(**int**));  
 S.MaxSize=1;  
 S.size=0;  
}  
  
**void** GetDiag(**char** \*\*C,**char**\* str1,**char**\* str2, LIST &S)  
{  
 **int** i,j,r,c=0,count=0;  
 **int** len1=strlen(str1);  
 **int** len2=strlen(str2);  
 **for**(i=0;i<len1;i++)  
 {  
 **for**(j=0;j<len2;j++)  
 {  
 **if**(C[i][j]==**'1'**)  
 {  
 count=0;  
 c=i;r=j;  
 **while**(C[c][r]==**'1'**)  
 {  
 count++;  
 C[c][r] = **'2'**;  
 c++;r++;  
 **if**(c>=len1||r>=len2)  
 **break**;  
 }  
 ListInsert(S,i,j,c-i);  
 }  
  
 }  
 }  
}  
  
**void** AgainMalloc(LIST &S)  
{  
 S.maxlen=(**int** \*)realloc(S.maxlen,(S.MaxSize+1)\* **sizeof**(**int**));  
 S.xpos=(**int** \*)realloc(S.xpos,(S.MaxSize+1)\* **sizeof**(**int**));  
 S.ypos=(**int** \*)realloc(S.ypos,(S.MaxSize+1)\* **sizeof**(**int**));  
 S.MaxSize++;  
}  
  
**void** ListInsert(LIST &S, **int** a, **int** b, **int** c)  
{  
 **if** (S.size == S.MaxSize)  
 AgainMalloc(S);  
 S.xpos[S.size] = a;  
 S.ypos[S.size] = b;  
 S.maxlen[S.size] = c;  
 S.size++;  
}  
  
**void** printlist(LIST &S)  
{  
 **int** i;  
 **for**(i=0;i<S.size;i++)  
 printf(**"%d "**,S.maxlen[i]);  
}  
  
**void** GetMaxLen(LIST &S, **int** &xmax, **int** &ymax, **int** &lenmax)  
{  
 **int** i;  
 lenmax=0;xmax=0;ymax=0;  
 **for**(i=0; i<S.size; i++)  
 {  
 **if**(S.maxlen[i]>lenmax)  
 {  
 lenmax=S.maxlen[i];  
 xmax=S.xpos[i];  
 ymax=S.ypos[i];  
 }  
 }  
}  
  
**void** StrDelete(**char** \*&str, **int** pos, **int** len)  
{  
 **int** i;  
 **for**(i=pos+len;i<strlen(str);i++)  
 {  
 str[i-len]=str[i];  
 }  
 str[strlen(str)-len]=**'\0'**;  
}  
  
**double** matchS(**char** \*str1, **char** \*str2)  
{  
 **char** \*\*C;  
 *//char \*sub;* LIST S;  
 Initlist(S);  
 **int** xmax=0,ymax=0,lenmax=0,count=0;  
 **int** len;  
 **if**(strlen(str1)>strlen(str2))  
 len=strlen(str2);  
 **else** len=strlen(str1);  
 **if**(GetMartix(C,str1,str2)==**false**)  
 **return** 0;  
 GetDiag(C,str1,str2,S);  
 GetMaxLen(S, xmax, ymax, lenmax);StrDelete(str1,xmax,lenmax);  
 StrDelete(str2,ymax,lenmax);  
 count=count+lenmax;  
 **while**(lenmax!=0)  
 {  
 Initlist(S);  
 **if**(GetMartix(C,str1,str2)==**false**)  
 **break**;  
 GetDiag(C,str1,str2,S);  
 GetMaxLen(S, xmax, ymax, lenmax);StrDelete(str1,xmax,lenmax);  
 StrDelete(str2,ymax,lenmax);  
 count=count+lenmax;  
 }  
 **int** i;  
 **if**(C!=**NULL**){  
 **for**(i=0;i<strlen(str1);i++)  
 free(C[i]);  
 free(C);  
 }  
 **delete** S.xpos;  
 **delete** S.ypos;  
 **delete** S.maxlen;  
 **return float**(count)/**float**(len);  
}  
**void** readin(**char** \*\*\*&A,**const** char\* p,**int** &pmax,**int** &smax,**int** &cmax)  
{  
 FILE \*fp;  
 **char** ch;  
 **int** flag,start=0,ctemp=0,stemp=0,i,j,k;  
 A=(**char** \*\*\*)malloc(**sizeof**(**char** \*\*));  
 A[0]=(**char** \*\*)malloc(**sizeof**(**char** \*));  
 A[0][0]=(**char** \*)malloc(**sizeof**(**char**));  
 **if**((fp=fopen(**p**,**"r+"**))==**NULL**){  
 printf(**"cannot open the file\n"**);  
 exit(0);  
 }  
 **while**((ch=fgetc(fp))!=**EOF**)  
 {  
 **if**(ch==**' '**&&start==0)  
 flag=0;  
 **else if**(ch==**'.'**)  
 { flag=2;  
 start=0;  
 }  
 **else if**(ch==**'\n'**)  
 {  
 flag=3;  
 start=0;  
 }  
 **else** {  
 flag=1;  
 **if**(ch==**','**) start=0;  
 **else** start=1;  
 }  
 **switch**(flag){  
 **case** 0: **break**;  
 **case** 1: {  
 A[pmax][stemp][ctemp]=ch;  
 ctemp++;  
 **if**(ctemp>cmax)  
 {  
 cmax++;  
 A=(**char** \*\*\*)realloc(A,(pmax+1)\***sizeof**(**char** \*\*));  
 A=(**char** \*\*\*)realloc(A,(pmax+1)\***sizeof**(**char** \*\*));  
 **for**(i=0;i<=pmax;i++)  
 { A[i]=(**char** \*\*)realloc(A[i],(smax+1)\***sizeof**(**char** \*));  
 A[i]=(**char** \*\*)realloc(A[i],(smax+1)\***sizeof**(**char** \*));  
 }  
 **for**(i=0;i<=pmax;i++)  
 **for**(j=0;j<=smax;j++) {  
 A[i][j]=(**char** \*) realloc(A[i][j], (cmax + 1) \* **sizeof**(**char**));  
 A[i][j]=(**char** \*) realloc(A[i][j], (cmax + 1) \* **sizeof**(**char**));  
 }  
 }  
 }  
 **break**;  
 **case** 2: {  
 A[pmax][stemp][ctemp]=**'\0'**;  
 ctemp=0;  
 stemp++;  
 **if**(stemp>smax) {  
 smax++;  
 A = (**char** \*\*\*) realloc(A, (pmax + 1) \* **sizeof**(**char** \*\*));  
 A = (**char** \*\*\*) realloc(A, (pmax + 1) \* **sizeof**(**char** \*\*));  
 **for** (i = 0; i <= pmax; i++)  
 {  
 A[i] = (**char** \*\*) realloc(A[i], (smax+ 1) \* **sizeof**(**char** \*));  
 A[i] = (**char** \*\*) realloc(A[i], (smax+ 1) \* **sizeof**(**char** \*));  
 }  
 **for** (i = 0; i <= pmax; i++)  
 **for** (j = 0; j <= smax; j++){  
 A[i][j]= (**char** \*) realloc(A[i][j], (cmax + 1) \* **sizeof**(**char**));  
 A[i][j]= (**char** \*) realloc(A[i][j], (cmax + 1) \* **sizeof**(**char**));  
 }  
 }  
 }  
 **break**;  
 **default**: {  
 A[pmax][stemp][0]=**'\0'**;  
 stemp=0;  
 pmax++;  
 A=(**char** \*\*\*)realloc(A,(pmax+1)\***sizeof**(**char** \*\*));  
 A=(**char** \*\*\*)realloc(A,(pmax+1)\***sizeof**(**char** \*\*));  
 **for**(i=0;i<=pmax;i++)  
 { A[i]=(**char** \*\*)realloc(A[i],(smax+1)\***sizeof**(**char** \*));  
 A[i]=(**char** \*\*)realloc(A[i],(smax+1)\***sizeof**(**char** \*));}  
 **for**(i=0;i<=pmax;i++)  
 **for**(j=0;j<=smax;j++)  
 {A[i][j]=(**char** \*)realloc(A[i][j],(cmax+1)\***sizeof**(**char** ));  
 A[i][j]=(**char** \*)realloc(A[i][j],(cmax+1)\***sizeof**(**char** ));  
 }  
 }  
 }  
 }  
 A[pmax][stemp][0]=**'\0'**;  
 fclose(fp);  
}  
**double** AriMatch(**int** Apmax,**int** Bpmax,**int** Asmax,**int** Bsmax, **int** Acmax, **int** Bcmax, **char** \*\*\*A,**char** \*\*\*B)  
{  
 **int** i=0,j=0;  
 **double** count=0,temp,para;  
 **for**(i=0;i<=Apmax;i++){  
 temp=0;  
 **for**(j=0;j<=Bpmax;j++)  
 {  
 **if**((para=ParaMatch(A[i],B[j],Asmax,Bsmax,Acmax,Bcmax))>temp)  
 temp=para;  
 }  
 count=count+temp;  
 }  
 **if**(i>j)  
 **return double**(count/i);  
 **else  
 return double**(count/j);  
}  
**double** ParaMatch(**char** \*\*A, **char**\*\*B,**int** Asmax,**int** Bsmax, **int** Acmax, **int** Bcmax)  
{  
 **int** i=0,j=0;  
 **double** count=0,temp=0,s;  
 **char** \*str1=**NULL**;  
 **char** \*str2=**NULL**;  
 **for**(i=0;i<=Asmax;i++){  
 **if**(A[i][0]==**'\0'**)  
 **break**;  
 temp=0;  
 **for**(j=0;j<=Bsmax;j++){  
 **if**(B[j][0]==**'\0'**)  
 **break**;  
 str1=(**char** \*)malloc(strlen(A[i])\***sizeof**(**char**));  
 strcpy(str1,A[i]);  
 str2=(**char** \*)malloc(strlen(B[j])\* **sizeof**(**char**));  
 strcpy(str2,B[j]);  
 **if**((s=matchS(str1,str2))>temp)  
 {  
 temp=s;  
 }  
 }  
 count+=temp;  
 }  
 str1=**NULL**;  
 str2=**NULL**;  
 **if**(i>j)  
 **return double**(count/i);  
 **else  
 return double**(count/j);  
}

1. 运行结果

输入:

test1.txt:

I like to travel to different places, I can see beautiful scenery and understand different cultures. I always travel with friends, we book a lot of plans in advance. But the plan is always interrupted by a variety of unexpected events, such as bad weather, holidays being cancelled, etc., so we have to spend time waiting for the next trip. I can ignore these problems when I travel alone. I can travel at any time, just pick up my backpack and buy a ticket. How free is that, I don't have to wait for other people to go where I want to go. It’s great to travel alone.

Summertime is the besttime of the year.There is no school for monthsI get to do what I want.With no tests no homework,I'm as free as a bird.I do many things during the summer vacation.I relax by reading books watching TV.I also hang out with my friends travel with my family.However,I don't play in summer.I take advantage of the free time to learn more.For example,last summer I learned to swim.This summer I might study computers or English.Summer vacation flies by fast,so it's important to do as as you can.

Everyone has their own dreams, I am the same. But my dream is not a lawyer, not a doctor, not actors, not even an industry. Perhaps my dream big people will find it ridiculous, but this has been my pursuit! My dream is to want to have a folk life! I want it to become a beautiful painting, it is not only sharp colors, but also the colors are bleak, I do not rule out the painting is part of the black, but I will treasure these bleak colors! Not yet, how about, a colorful painting, if not bleak, add color, how can it more prominent American? Life is like painting, painting the bright red color represents life beautiful happy moments. Painting a bleak color represents life difficult, unpleasant time. You may find a flat with a beautiful road is not very good yet, but I do not think it will. If a person lives flat then what is the point? Life is only a short few decades, I want it to go Finally, Each memory is a solid.

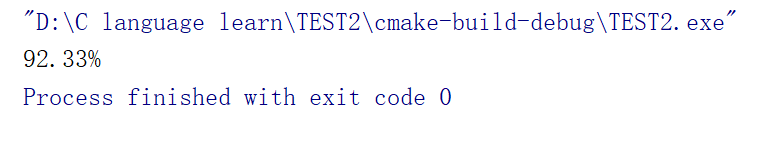
test1.txt:

I like to travel to different places, because I can see beautiful scenery and understand different cultures. I always travel with friends, and we make many plans ahead of time. But plans are always interrupted by unexpected events, such as bad weather, cancelled holidays and so on, so we have to spend time waiting for the next trip. Traveling alone can ignore these problems, and I can travel anytime I want, just with a backpack and a ticket. It's so free. I don't have to wait for anybody else to go where I want to go. It's nice to travel alone.

Summer is the best time of the year. No school, I want to do what I want to do, no homework, I like a free bird, I do a lot of things, summer vacation, through reading and watching TV to relax themselves, I also travel with my friends and my family. However, I do not play summer. I use my spare time to learn more. For example, I learned to swim last summer. I may learn computer or English this summer. Summer vacation is fast, so the important thing is to do it, you can.

Everyone has their own dreams, so do I. But my dream is not a lawyer, not a doctor, not an actor, or even an industry. My dream may be ridiculous to adults, but this is what I have been pursuing. My dream is to have a colorful life. I want it to be a beautiful painting, it not only has bright colors, but also have dim colors, I do not rule out that this painting has a part of the black, but I will cherish these dim colors! Isn't it? How can a brightly colored painting stand out more beautifully if it doesn't have a bit of darkness? Life is like painting. The bright colors in the picture represent the beautiful and happy life. The dim colors in the painting represent difficulties and unpleasantness in life. Maybe you would think that it's not good to have a flat and beautiful road, but I don't think so. What does it mean for a person to have a flat life? Life is only a short span of decades. I want him to come to the last moment. Every memory is full.

输出:



1. 心得体会
2. realloc函数：
3. 只能对已申请的空间调整大小。
4. realloc申请到的空间必须赋值给指针，以防止其指针漂移。
5. 指针数组要逐级申请，逐级释放。

2）指针操作

1. 申请的内存必须释放否则内存泄漏。

2. 指针最好赋初值NULL。

3. 不要对NULL指针操作。

3）fclose函数包含了对指针内存的释放，无需在free文件指针。