历

年

真

题

目录

[2005 统计篇文章中各英文字母的个数 4](#_Toc510015889)

[读字符 4](#_Toc510015890)

[统计相同的字符 5](#_Toc510015891)

[2006 6](#_Toc510015892)

[找出100到1000内的不含9的素数，存到result文件中。 6](#_Toc510015893)

[写数字 6](#_Toc510015894)

[2007 素数 7](#_Toc510015895)

[2008 8](#_Toc510015896)

[除去特定单词 9](#_Toc510015897)

[首字母大写 10](#_Toc510015898)

[除去重复的单词 10](#_Toc510015899)

[读字符串（） 10](#_Toc510015900)

[写字符串 11](#_Toc510015901)

[2009 12](#_Toc510015902)

[读特定格式里面的内容---如用逗号分隔按照字符读出） 12](#_Toc510015903)

[除去因按照字符读出而读出的空格或其他内容 13](#_Toc510015904)

[八进制转化成十进制 13](#_Toc510015905)

[将字符串格式的内容转化成数字 13](#_Toc510015906)

[冒泡排序 14](#_Toc510015907)

[写入数字 14](#_Toc510015908)

[2010 15](#_Toc510015909)

[二进制写数字 15](#_Toc510015910)

[二进制读数字 16](#_Toc510015911)

[判断素数 16](#_Toc510015912)

[得到数组中所有素数 17](#_Toc510015913)

[冒泡排序 17](#_Toc510015914)

[2011 18](#_Toc510015915)

[2012 22](#_Toc510015916)

[二进制写数字 22](#_Toc510015917)

[二进制读数字 23](#_Toc510015918)

[冒泡排序 25](#_Toc510015919)

[2013 26](#_Toc510015920)

[读特殊格式的内容---如括号里面的单词（思路用字符读取） 27](#_Toc510015921)

[递归判断路线计划是否可行 30](#_Toc510015922)

[文本写内容 30](#_Toc510015923)

[2014 31](#_Toc510015924)

[二进制写数字 31](#_Toc510015925)

[二进制读数字 31](#_Toc510015926)

[所有点到某一点的距离 32](#_Toc510015927)

[冒泡排序 32](#_Toc510015928)

[写数字 33](#_Toc510015929)

[2016 34](#_Toc510015930)

[读单词 34](#_Toc510015931)

[统计数组中每个单词出现的次数 35](#_Toc510015932)

[冒泡排序 35](#_Toc510015933)

[写内容 36](#_Toc510015934)

[2015 36](#_Toc510015935)

[二进制写数字 37](#_Toc510015936)

[二进制读数字 38](#_Toc510015937)

[得到有效点 38](#_Toc510015938)

[写入格式 39](#_Toc510015939)

# 2005 统计篇文章中各英文字母的个数

//统计篇文章中各英文字母的个数

#include<stdio.h>

#include<stdlib.h>

typedef struct word

{

char w;

int time;

}word;

## 读字符

void read(char filename[],char all\_w[],int \*n)

{

FILE \*read;

char a;

int i;

\*n=0;

if((read=fopen(filename,"r+"))==NULL)

{

printf("Couldn't open the file！ê?\n");

system("pause");

exit(0);

}

while(!feof(read))

{

a=getc(read);

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

all\_w[\*n]=a,++\*n;

}

fclose(read);

}

int check(word w[],char a,int count)

{

int i;

for(i=0;i<count;++i)

{

if(w[i].w==a)

return i+1;

}

return 0;

}

## 统计相同的字符

void del(word w[],char all\_c[],int count\_all,int \*count)

{

int i,j,k;

\*count=0;

i=0,j=0,k=0;

w[0].w=all\_c[i];

w[0].time=1;

++j;

++\*count;

for(i=1;i<count\_all;++i)

{

if(check(w,all\_c[i],\*count))

{

++(w[check(w,all\_c[i],\*count)-1].time);

}

else

{

w[j].w=all\_c[i];

w[j].time=1;

++j;

++\*count;

}

}

\*count=j;

printf("一°?共2有®D字Á?母? %d 个?\n",count\_all);

printf("除y去¨£¤重?复¡ä字Á?母?有®D %d 个? \n",\*count);

for(i=0;i<\*count;++i)

{

printf(" %c : %d \n",w[i].w,w[i].time);

}

}

int main()

{

char filename[20],all\_c[1000];

word w[200];

int count\_all,count;

printf("Please enter the name of file: ");

scanf("%s",filename);

read(filename,all\_c,&count\_all);

del(w,all\_c,count\_all,&count);

system("pause");

return 0;

}

# 2006

## 找出100到1000内的不含9的素数，存到result文件中。

#include<stdio.h>

#include<stdlib.h>

int check(int n)

{

int i;

if(n==0||n==1) return 0;

for(i=2;i<=n/2;++i)

{

if(n%i==0)

return 0;

}

if(i>n/2)

return 1;

}

int check\_9(int n)

{

while(n)

{

if(n%10==9)

return 0;

n/=10;

}

return 1;

}

void get\_su(int a[],int \*n)

{

int i,j;

j=0;

for(i=100;i<1000;i++)

{

if(check(i)&&check\_9(i))

a[j]=i,++j;

}

\*n=j;

}

## **写数字**

void write(int a[],int n)

{

FILE \*write;

int i;

if((write=fopen("G:\\file\_2006.txt","w+"))==NULL)

{

printf("The file couldn't be openned!\n");

system("pause");

exit(0);

}

for(i=0;i<n;++i)

{

fprintf(write,"%d",a[i]);

fprintf(write,"%s "," ");

if((i+1)%20==0)

fprintf(write,"%s ","\n");

}

}

int main()

{

int data[1000],count;

get\_su(data,&count);

write(data,count);

system("pause");

return 0;

}

# 2007 素数

//把10到1000之间满足以下两个条件的数，存到result.txt文件中

//1、是素数。

//2、它的反数也是素数，如：123的反数是321。

#include<stdio.h>

#include<stdlib.h>

int check(int n)

{

int i;

if(n==0||n==1) return 0;

for(i=2;i<=n/2;++i)

{

if(n%i==0) return 0;

}

if(i>n/2) return 1;

}

int dao(int n)

{

int k=0;

while(n)

{

k=k\*10+n%10;

n/=10;

}

return k;

}

void write()

{

FILE \*write;

int i,k;

if((write=fopen("G:\\file\_2007.txt","w+"))==NULL)

{

printf("The file could'n be openned!\n");

system("pause");

exit(0);

}

for(i=10,k=1;i<1000;++i)

{

if(check(i)&&check(dao(i)))

{

fprintf(write," %d ",i),++k;

if(k%20==0)

{ fprintf(write,"%c",'\n');}

}

}

fclose(write);

}

int main()

{

write();

system("pause");

return 0;

}

# 2008

（1）用IE从FTP上下载org.dat，并保存在D盘的根目录中。

（2）此文件中按文本方式存放了一段其他文章，其中有若干长度小于15的英文单词，单词之间用空格分开，无其他符号。

（3）顺序读取这段文章的不同的单词（大小写敏感），同时在读取的过程中排除所有的单词THE以及变形，即这些单词不能出现在读取的结果中。

（4）将读取的所有单词的首字母转大写后，输出D根目录下new.txt，每个单词一行。

------------------------

那段文字可以点右键打开方式中用记事本打开，内容是：

The constructor is used to initialize the object The destructor is used to delete the Object the calling seqence of constructor is opposite to the calling sequence of destructor

个人感觉今年出题挺有水平的，苏大一个重要的方向就是信息处理，这个应该就是分词技术的基础把，我做的一般，不过听说好象有将近满分的，还当然也有很多兄弟没做出来。

-----------------

正确结果应该是

Constructor

Is

Used

To

Initialize

Object

Destructor

Delete

Object

Calling

Seqence

Of

Opposite

注意有两个Object

因为第一次读取是大小写敏感的，后来又进行了首字母大写转化

#include<stdio.h>

#include<stdlib.h>

#include<string.h>

## 除去特定单词

void del\_the(char all\_word[][15],int \*count)

{

int i,j;

for(i=0;i<\*count;)

{

if(strcmp(all\_word[i],"The")==0||strcmp(all\_word[i],"the")==0)

{

for(j=i;j<\*count-1;++j)

strcpy(all\_word[j],all\_word[j+1]);

--\*count;

}

else

++i;

}

}

## 首字母大写

void a\_t\_A(char all\_word[][15],int \*count)

{

int i;

for(i=0;i<\*count;++i)

{

if(all\_word[i][0]>='a'&&all\_word[i][0]<='z')

all\_word[i][0]-=32;

}

for(i=0;i<\*count;++i)

{

printf("%s ",all\_word[i]);

}

printf("%s","\n");

}

## 除去重复的单词

void del\_same(char all\_word[][15],int \*count)

{

int i,j,k;

for(i=0;i<\*count;++i)

{

for(j=i+1;j<\*count;)

if(strcmp(all\_word[i],all\_word[j])==0)

{

for(k=j;k<\*count-1;++k)

strcpy(all\_word[k],all\_word[k+1]);

--\*count;

}

else

++j;

}

}

## 读字符串（）

void read(char all\_word[][15],int \*count)

{

FILE \*read;

char word[15];

char ch;

int i;

\*count=0;

if((read=fopen("G:\\file\_2008.txt","r+"))==NULL)

{

printf("The file couldn't be openned!\n");

system("pause");

exit(0);

}

while(!feof(read))

{

fscanf(read,"%s",word);

strcpy(all\_word[\*count],word);

printf("%s ",word);

++\*count;

}

printf("\n");

printf("%d\n",\*count);

fseek(read,-1,2);

ch=fgetc(read);

if(ch==' ')

--\*count;

for(i=0;i<\*count;++i)

printf("%s ",all\_word[i]);

printf("%d\n",\*count);

fclose(read);

}

## 写字符串

void write(char all\_word[][15],int \*count)

{

FILE \*write;

int i;

if((write=fopen("G:\\file\_2008\_w.txt","w+"))==NULL)

{

printf("The file couldn't be openned!\n");

system("pause");

exit(0);

}

for(i=0;i<\*count;++i)

{

fprintf(write,"%s",all\_word[i]);

if(!(i==\*count-1))

fprintf(write,"%c",'\n');

}

}

int main()

{

char all\_word[100][15];

int count\_all;

read(all\_word,&count\_all);

del\_the(all\_word,&count\_all);

del\_same(all\_word,&count\_all);

a\_t\_A(all\_word,&count\_all);

write(all\_word,&count\_all);

system("pause");

return 0;

}

# 2009

（1）用IE浏览器从FTP上下载org.dat，并保存在D盘的根目录下。

（2）此文件中按文本方式存放了一段其他文章，其中有若干长度小于15的十进制或八进制数字，数字之间用“,”分开，数字内部存在且仅存在空格。

（3）八进制数以起始位“0”作为标示与十进制数区分。

（4）顺序读取这些数字将他们转变为十进制数后按从大到小的顺序排序后，输出到D盘根目录下new.txt，每个数字一行。 eg：\_235\_,34\_\_2,\_043\_1\_,1\_3

分别是：十进制235，十进制342.，八进制431，十进制13。\_代表空格。

#include<stdio.h>

#include<stdlib.h>

#include<string.h>

## 读特定格式里面的内容---如用逗号分隔按照字符读出）

void read(char num[][10],int \*count)

{

FILE \*read;

char ch;

int i;

if((read=fopen("G:\\file\_2009\_r.txt","r+"))==NULL)

{

printf("The file could be openned ！ê?\n");

system("pause");

exit(0);

}

\*count=0;

ch=fgetc(read);

while(!feof(read))

{

i=0;

while(ch!=','&&!feof(read))

{

num[\*count][i]=ch;

++i;

ch=fgetc(read);

}

num[\*count][i]='\0';

++\*count;

ch=fgetc(read);

while(ch==','&&!feof(read))

ch=fgetc(read);

}

}

## 除去因按照字符读出而读出的空格或其他内容

void del\_b(char num[][10],int \*count,char num1[][10])

{

int i,j,k;

for(i=0;i<\*count;++i)

{

for(j=0,k=0;num[i][j]!='\0';++j)

if(num[i][j]!=' ')

num1[i][k]=num[i][j],++k;

num1[i][k]='\0';

}

for(i=0;i<\*count;++i)

{

printf("%s\n",num1[i]);

}

}

## 八进制转化成十进制

int e\_t\_o(char num[])

{

int data=0;

int i=0;

while(num[i]!='\0')

{

data=data\*8+num[i]-'0';

++i;

}

return data;

}

## 将字符串格式的内容转化成数字

void turn(char num1[][10],int \*count,int data[])

{

int i;

for(i=0;i<\*count;++i)

{

if(num1[i][0]!='0')

data[i]=atoi(num1[i]);

else

data[i]=e\_t\_o(num1[i]);

}

for(i=0;i<\*count;++i)

{

printf("%d\n",data[i]);

}

}

## 冒泡排序

void sort(int data[],int n)

{

int i,j,t;

for(i=0;i<n-1;++i)

{

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

if(data[j]<data[j+1])

t=data[j],data[j]=data[j+1],data[j+1]=t;

}

for(i=0;i<n;++i)

{

printf("%d\n",data[i]);

}

}

## 写入数字

void write(int data[],int n)

{

int i;

FILE \*write;

if((write=fopen("G:\\file\_2009\_w.txt","w+"))==NULL)

{

printf("The file couldn't be openned!\n");

system("pause");

}

for(i=0;i<n;++i)

{

fprintf(write,"%d\n",data[i]);

}

}

int main()

{

char num[10][10],num1[10][10];

int data[10];

int count;

read(num,&count);

del\_b(num,&count,num1);

turn(num1,&count,data);

sort(data,count);

write(data,count);

system("pause");

return 0;

}

# 2010

(1) FTP上下载make.exe和org.dat，运行make.exe输入准考证后三位生成data.txt，文件为二进制编码。

(2) data.txt内存有2048个整数，其中前n哥为非0数，后2048-n个数为0，将其读入数组，计算非零数的个数n。

(3) 选出n个数中的最大数&最小数。

(4) 选出n个数中最大素数。

(5) 将n哥数从大到小排序，并平均分成三段(若n非3的整数倍，则不考虑最后的1-2个数)，选出中间段的最大数和最小数。

#include<stdio.h>

#include<stdlib.h>

## 二进制写数字

void write()

{

FILE \*write;

int data[]={1,2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0};

int n=sizeof(data)/4;

int i;

if((write=fopen("G:\\file\_2010.txt","wb+"))==NULL)

{

printf("The file couldn't be openned!\n");

system("pause");

exit(0);

}

for(i=0;i<n;++i)

{

//printf("%d ",data[i]);

fwrite(&data[i],4,1,write);

}

fclose(write);

}

## 二进制读数字

void read(int all[],int \*count\_all)

{

FILE \*read;

int i;

int data;

if((read=fopen("G:\\file\_2010.txt","rb+"))==NULL)

{

printf("The file couldn't be openned!\n");

system("pause");

exit(0);

}

\*count\_all=0;

while(!feof(read))

{

//fscanf(read,"%d",&data);

fread(&data,4,1,read);

//printf("%d ",data);

if(data!=0)

{

all[\*count\_all]=data;

++\*count\_all;

}

}

//for(i=0;i<\*count\_all;++i)

//{

// printf("%d\n",all[i]);

//}

}

## 判断素数

int check\_su(int n)

{

int i;

if(n<2) return 0;

for(i=2;i<=n/2;++i)

{

if(n%i==0) return 0;

}

if(i>n/2) return 1;

}

## 得到数组中所有素数

void get\_su(int all[],int su[],int count\_all,int \*count\_su)

{

int i,j;

for(i=0,j=0;i<count\_all;++i)

if(check\_su(all[i]))

su[j]=all[i],++j;

\*count\_su=j;

}

## 冒泡排序

void sort(int data[],int n)

{

int i,j,t;

for(i=0;i<n-1;++i)

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

{

if(data[j]>data[j+1])

t=data[j],data[j]=data[j+1],data[j+1]=t;

}

//for(i=0;i<n;++i)

//{

// printf("%d\n",data[i]);

//}

}

int main()

{

int all[2048],su[2048];

int count\_all,count\_su;

int max,min;

int mid\_max,mid\_min;

int max\_su;

write();

read(all,&count\_all);

get\_su(all,su,count\_all,&count\_su);

sort(all,count\_all);

max=all[count\_all-1];

min=all[0];

sort(su,count\_su);

max\_su=su[count\_su-1];

mid\_max=all[count\_all/3\*2-1];

mid\_min=all[count\_all/3];

printf("Max number is : %d \n",max);

printf("Min number is : %d \n",min);

printf("Max\_su number is : %d \n",max\_su);

printf("Mid\_max number is : %d \n",mid\_max);

printf("Mid\_min number is : %d \n",mid\_min);

system("pause");

return 0;

}

# 2011

第一道：

输出1000-9999中满足以下条件的所有数：

(1) 该数是素数。

(2) 十位数和个位数组成的数是素数，百位数和个位数组成的数是素数。

(3) 个位数和百位数组成的数是素数，个位数和十位数组成的数是素数。

比如1991，个位和十位组成的数就是19.

#include<stdio.h>

#include<stdlib.h>

int check\_su(int n)

{

int i;

if(n<2) return 0;

for(i=2;i<=n/2;++i)

if(n%i==0) return 0;

if(i>n/2) return 1;

}

void operation()

{

int i,count=0;

for(i=1000;i<9999;++i)

if(check\_su(i)&&check\_su(i%100)&&check\_su(i/100%10\*10+i%10)&&check\_su(i/100%10+i%10\*10)&&check\_su(i%10\*10+i/10%10))

{

printf("%d ",i),++count;

if(count%10==0)

printf("%s","\n");

}

printf("%s","\n");

}

int main()

{

operation();

system("pause");

return 0;

}

第二道：

从ftp上下载一个文本info.txt，其中是一段关于计算机历史的英文文档(不过是加过密的，打开全是乱码)，要求密匙cip1和cip2以及解密后的第一个句子。条件：

(1) cip1和cip2都是八位无符号整数；

(2) 加密过程：每次从文本中读出八位字符，然后将该字符和密钥交替异或便是该位置的密文。

如第0个字节与cip1异或，第一个字节和cip2异或，第三个字节和cip1异或，。。。，如此循环至结束。

最后输出cip1和cip2以及第一段完整的英文句子。

#include<stdio.h>

#include<stdlib.h>

void write()

{

FILE \*write;

char ch;

char article[]="Success and failure is and walkers. When you experience the enjoyment of victory, success and failure in the far away from you hide the corner, or wait in the waiting outside, you consider it. Therefore, when one victory, enjoying the mood of the holiday, the more times, when the attack is over failure. The former, the latter more lax degree will succeed. The former, the latter will not slack away a shot harvest it \"success\". Who is to come, success and failure in the \"twin\", manipulation, and stand, vigorous... Step after of the road of life.";

int n=sizeof(article);

int i;

if((write=fopen("G:\\file\_2011.txt","wb+"))==NULL)

{

printf("The file couldn't be openned!\n");

system("pause");

exit(0);

}

for(i=0;i<n-1;++i)

{

ch=article[i];

if(i%2) ch^=9;

else ch^=8;

fwrite(&ch,1,1,write);

//printf("%c",article[i]);

}

fclose(write);

}

void read(char article[20000],int \*count)

{

FILE \*read;

char ch;

int i;

if((read=fopen("G:\\file\_2011.txt","rb+"))==NULL)

{

printf("The file couldn't be openned!\n");

system("pause");

exit(0);

}

i=0;

while(!feof(read))

{

fread(&ch,1,1,read);

if(!feof(read))

{

article[i]=ch;

++i;

}

}

\*count=i;

//for(i=0;i<\*count;++i)

//{

// if(i%2)

// printf("%c",article[i]^8);

// else

// printf("%c",article[i]^9);

//}

}

int check(char a,int n)

{

if(((a^n)>='A'&&(a^n)<='Z')||((a^n)>='a'&&(a^n)<='z')||((a^n)=='\"')||((a^n)=='.')||((a^n)==',')||((a^n)=='!')||((a^n)=='?')||((a^n)==' '))

return 1;

else

return 0;

}

void get\_cip(char article[],int count,int \*cip1,int \*cip2)

{

int i,j,k;

for(i=0;i<256;++i)

{

for(k=0;k<count;++k)

if(k%2)

{

if(check(article[k],i));

else

break;

}

if(k>=count)

break;

}

for(j=0;j<256;++j)

{

for(k=0;k<count;++k)

if(!(k%2))

{

if(check(article[k],j));

else

break;

}

if(k>=count)

break;

}

if(j>=256)

printf("Fail to get cip1\n");

else

{

\*cip1=j;

printf("cip1=%d\n",\*cip1);

}

if(i>=256)

printf("Fail to get cip2\n");

else

{

\*cip2=i;

printf("cip2=%d\n",\*cip2);

}

}

void print(char article[20000],int count,int cip1,int cip2)

{

int i;

printf("The first juzi: ");

for(i=0;i<count;++i)

{

if(i%2)

{

printf("%c",article[i]^cip2);

if((article[i]^cip2)=='.') break;

}

else

{

printf("%c",article[i]^cip1);

if((article[i]^cip1)=='.') break;

}

}

printf("\n");

}

int main()

{

char article[20000];

int count;

int cip1,cip2;

write();

read(article,&count);

get\_cip(article,count,&cip1,&cip2);

print(article,count,cip1,cip2);

system("pause");

return 0;

}

# 2012

从服务器上下载数据文件org.dat文件以二进制方式存放一系列整数，每个整数占4个字节。从第一个整数开始，第一个整数和第二个整数构成一个坐标点，以此类推，数据文件中保存了许多坐标点数据。

一组一组的读，保存在二维数组里，后面要用

文本文件 单个读或者一行一行读 循环读 条件 文件指针没有到文件尾

问题1：规定处于第一象限的坐标点为有效点，请问数据文件中所有点的个数n为多少？有效点的个数k为多少？

问题2：每个有效点与坐标原点构成一个的矩形，请问k个有效点与坐标原点构成的k个矩形的最小公共区域面积为多少？

问题3：寻找有效点钟符合下列条件的点：以该点为坐标原点，其它有效点仍然是有效点即处于第一象限（不包括坐标轴上的点）。输出这些点。

问题4：对所有有效点进行分组，每个有效点有且只有属于一个分组，分组内的点符合下列规则：若对组内所有点的x坐标进行排序，点p1(x1,y1)在点p2(x2,y2)后面，即x1>x2那么y1>y2，请输出所有的分组。

#include<stdio.h>

#include<stdlib.h>

typedef struct point

{

int x;

int y;

int flag;

}point;

## 二进制写数字

void write()

{

FILE \*write;

int a[] = { 20, 9, 1, 1, -9, 56, 8, 9, -3, -7, 90, 2, -2, 8, 6, 8, 80, 3, -9, -44, -23, 10, 20, 3, -23, 89, 5, -5, 4, 3, 22, 77, 90, 10, -4, -9, -43, 9, -2, 6, -5, 9, -7, 90, 1, -2, 8, 6, 8, 90, 77, 99, 10, -4, -9, 99, -9, 56, 8, 9, -3, -7};

int n=sizeof(a)/4;

int i;

if((write=fopen("G:\\file\_2012.txt","wb+"))==NULL)

{

printf("The file couldn't be openned!\n");

system("pause");

exit(0);

}

for(i=0;i<n;++i)

{

putw(a[i],write);

}

fclose(write);

//printf("%d\n",n);

}

## 二进制读数字

void read(point all\_p[],int \*count)

{

FILE \*read;

int a;

\*count=0;

if((read=fopen("G:\\file\_2012.txt","rb+"))==NULL)

{

printf("The file couldn't be openned!\n");

system("pause");

exit(0);

}

while(!feof(read))

{

fread(&a,4,1,read);

if(!feof(read))

{ //printf("%d ",a);

all\_p[\*count].x=a;

}

fread(&a,4,1,read);

if(!feof(read))

{ //printf("%d ",a);

all\_p[\*count].y=a;

++\*count;

}

}

printf("The total number of points is : %d\n",\*count);

fclose(read);

}

void check\_rea(point all\_p[],int count\_all\_p,point true\_p[],int \*count\_true\_p)

{

int i;

\*count\_true\_p=0;

for(i=0;i<count\_all\_p;++i)

{

if(all\_p[i].x>0&&all\_p[i].y>0)

true\_p[\*count\_true\_p]=all\_p[i],++\*count\_true\_p,true\_p[\*count\_true\_p].flag=0;

}

//for(i=0;i<\*count\_true\_p;++i)

//{

// printf("( %d , %d )\n",true\_p[i].x,true\_p[i].y);

//}

}

void get\_min\_s(point true\_p[],int count\_true\_p,int \*s)

{

int min\_x,min\_y,i;

min\_x=true\_p[0].x;

min\_y=true\_p[0].y;

for(i=1;i<count\_true\_p;++i)

{

if(min\_x>true\_p[i].x) min\_x=true\_p[i].x;

if(min\_y>true\_p[i].y) min\_y=true\_p[i].y;

}

\*s=min\_x\*min\_y;

}

void operate3(point true\_p[],int count\_true\_p,point \*p)

{

int min\_x,min\_y,i;

min\_x=0;

min\_y=0;

for(i=1;i<count\_true\_p;++i)

{

if(true\_p[min\_x].x>true\_p[i].x) min\_x=i;

if(true\_p[min\_y].y>true\_p[i].y) min\_y=i;

}

if(min\_x==min\_y)

{

(\*p).x=true\_p[min\_x].x;

(\*p).y=true\_p[min\_x].y;

}

else

{

(\*p).x=0;

(\*p).y=0;

}

}

## 冒泡排序

void sort(point true\_p[],int count\_true\_p)

{

int i,j;

point p;

for(i=0;i<count\_true\_p-1;++i)

for(j=0;j<count\_true\_p-1-i;++j)

{

if(true\_p[j].x>true\_p[j+1].x)

p=true\_p[j],true\_p[j]=true\_p[j+1],true\_p[j+1]=p;

}

}

void operate4(point true\_p[],int count\_true\_p)

{

int i,j,k;

int count=1,count2;;

for(i=0;i<count\_true\_p;++i)

{

true\_p[i].flag=0;

}

for(i=0;i<count\_true\_p;++i)

{

if(!true\_p[i].flag)

{

count2=count;

for(j=i+1;j<count\_true\_p;++j)

{

if(true\_p[i].flag==0&&true\_p[j].flag==0&&true\_p[i].x<true\_p[j].x&&true\_p[i].y<true\_p[j].y)

{

printf("The %d group are : ( %d , %d ) , ( %d , %d ) ",count,true\_p[i].x,true\_p[i].y,true\_p[j].x,true\_p[j].y);

true\_p[i].flag=true\_p[j].flag=1;

++count;

k=j;

}

else if(true\_p[k].flag==1&&true\_p[j].flag==0&&true\_p[k].x<true\_p[j].x&&true\_p[k].y<true\_p[j].y)

{

printf("( %d , %d ) ",true\_p[j].x,true\_p[j].y);

true\_p[j].flag=1;

k=j;

}

}

if(count2!=count)

printf("%c",'\n');

}

}

printf("程¨¬序¨°运?行D结¨¢束º?！ê?\n");

}

int main()

{

point all\_p[100],true\_p[100];

point p;

int count\_all\_p,count\_true\_p;

int min\_s;

write();

read(all\_p,&count\_all\_p);

check\_rea(all\_p,count\_all\_p,true\_p,&count\_true\_p);

printf("The real points' number is : %d\n",count\_true\_p);

get\_min\_s(true\_p,count\_true\_p,&min\_s);

printf("The min\_s is : %d \n",min\_s);

operate3(true\_p,count\_true\_p,&p);

if(p.x==0)

printf("Don't has point like this which meet the operation3!\n");

else

printf("The point which meet the operation3 is ( %d , %d )\n",p.x,p.y);

sort(true\_p,count\_true\_p);

operate4(true\_p,count\_true\_p);

system("pause");

return 0;

}

# 2013

Your Task

Your program should determine if a particular destination airport can be reached from a particular originating airport within a particular number of hops.

A hop (leg of a flight) is a flight from one airport to another on the path between an originating and destination airports.

For example, the flight plan from PVG to PEK might be PVG → CAN → PEK. So PVG → CAN would be a hop and CAN → PEK would be a hop.

Input Data Files

Path Input File(PathInput.txt)

This input file will consist of a number of single origination/destination airport pairs (direct flights). The first line of the file will contain an integer representing the total number of pairs in the rest of the file.

6

[PVG, CAN]

[CAN, PEK]

[PVG, CTU]

[CTU, DLC]

[DLC, HAK]

[HAK, LXA]

Path Request File(PathRequest.txt)

This input file will contain a sequence of pairs of origination/destination airports and a max number of hops. The first line of the file will contain an integer representing the number of pairs in the file.

2

[PVG, DLC, 2]

[PVG, LXA, 2]

Output File(Output.txt)

For each pair in the Path Request File, your program should output the pair followed by “YES” or “NO” indicating that it is possible to get from the origination to destination airports within the max number of hops or it is not possible, respectively.

[PVG, DLC, YES]

[PVG, LXA, NO]

#include<stdio.h>

#include<stdlib.h>

#include<string.h>

typedef struct line

{

char ori[5];

char des[5];

}line;

typedef struct plan

{

char ori[5];

char des[5];

int max;

}plan;

## 读特殊格式的内容---如括号里面的单词（思路用字符读取）

void read(line li[],int \*count\_li,plan pl[],int \*count\_pl)

{

FILE \*read\_li,\*read\_pl;

char ch;

int i=0;

int j;

if((read\_li=fopen("G:\\file\_2013\_line.txt","r+"))==NULL)

{

printf("The file couldn't be openned!\n");

system("pause");

exit(0);

}

if((read\_pl=fopen("G:\\file\_2013\_plan.txt","r+"))==NULL)

{

printf("The file couldn't be openned!\n");

system("pause");

exit(0);

}

ch=fgetc(read\_li);

while(!feof(read\_li))

{

if(ch=='[')

{

j=0;

ch=fgetc(read\_li);

while(ch!=',')

{

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

{

li[i].ori[j]=ch;

++j;

}

ch=fgetc(read\_li);

}

li[i].ori[j]='\0';

}

if(ch==',')

{

j=0;

ch=fgetc(read\_li);

while(ch!=']')

{

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

{

li[i].des[j]=ch;

++j;

}

ch=fgetc(read\_li);

}

li[i].des[j]='\0';

++i;

}

ch=fgetc(read\_li);

}

\*count\_li=i;

i=0;

ch=fgetc(read\_pl);

while(!feof(read\_pl))

{

if(ch=='[')

{

j=0;

ch=fgetc(read\_pl);

while(ch!=',')

{

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

{

pl[i].ori[j]=ch;

++j;

}

ch=fgetc(read\_pl);

}

pl[i].ori[j]='\0';

}

if(ch==',')

{

j=0;

ch=fgetc(read\_pl);

while(ch!=',')

{

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

{

pl[i].des[j]=ch;

++j;

}

ch=fgetc(read\_pl);

}

pl[i].des[j]='\0';

}

if(ch==',')

{

j=0;

ch=fgetc(read\_pl);

while(ch!=']')

{

if(ch>='0'&&ch<='9')

pl[i].max=ch-'0';

ch=fgetc(read\_pl);

}

++i;

}

ch=fgetc(read\_pl);

}

\*count\_pl=i;

fclose(read\_li);

fclose(read\_pl);

for(i=0;i<\*count\_li;++i)

printf("[ %s , %s ]\n",li[i].ori,li[i].des);

for(i=0;i<\*count\_pl;++i)

printf("[ %s , %s , %d ]\n",pl[i].ori,pl[i].des,pl[i].max);

}

## 递归判断路线计划是否可行

int check(line li[],int count\_li,char ori[10],char des[10],int k)

{

int i,flag=0;

for(i=0;i<count\_li;++i)

if(k>=0&&strcmp(li[i].ori,ori)==0&&strcmp(li[i].des,des)==0)

{

flag=1;

break;

}

else if(k>=1&&strcmp(li[i].ori,ori)==0&&strcmp(li[i].des,des)!=0)

flag=check(li,count\_li,li[i].des,des,--k);

return flag;

}

## 文本写内容

void out(line li[],int count\_li,plan pl[],int count\_pl)

{

FILE \*write;

int i;

if((write=fopen("G:\\file\_2013\_output.txt","w+"))==NULL)

{

printf("The file couldn't be openned!\n");

system("pause");

exit(0);

}

for(i=0;i<count\_pl;++i)

if(check(li,count\_li,pl[i].ori,pl[i].des,pl[i].max))

fprintf(write,"[ %s , %s , YES ]\n",pl[i].ori,pl[i].des);

else

fprintf(write,"[ %s , %s , NO ]\n",pl[i].ori,pl[i].des);

}

int main()

{

line li[10];

plan pl[10];

int count\_li,count\_pl;

read(li,&count\_li,pl,&count\_pl);

out(li,count\_li,pl,count\_pl);

system("pause");

return 0;

}

# 2014

从网页上下载input.dat文件，里面是用二进制编写的，里面放了一堆int 型的数，每个数占4个字节，每次读取两个，这两个数构成一个坐标。

（1）规定处于第一象限的数是有效点，（即x>0,y>0的坐标），问这么多点中有效点有多少个？

（2）现在用户从键盘输入一个坐标和一个数字K，设计算法输出K个离该坐标距离最近的点的坐标和每个坐标到该点的距离，写入到output.txt文件中

#include<stdio.h>

#include<stdlib.h>

#include<math.h>

typedef struct point

{

int x;

int y;

double dis;

}point;

## 二进制写数字

void write()

{

FILE \*write;

int a[] = { 20, 9, 1, 1, -9, 56, 8, 9, -3, -7, 90, 2, -2, 8, 6, 8, 80, 3, -9, -44, -23, 10, 20, 3, -23, 89, 5, -5, 4, 3, 22, 77, 90, 10, -4, -9, -43, 9, -2, 6, -5, 9, -7, 90, 1, -2, 8, 6, 8, 90, 77, 99, 10, -4, -9, 99, -9, 56, 8, 9, -3, -7};

int n=sizeof(a)/4;

int i;

if((write=fopen("G:\\file\_2014.txt","wb+"))==NULL)

{

printf("The file couldn't be openned!\n");

system("pause");

exit(0);

}

for(i=0;i<n;++i)

{

putw(a[i],write);

}

fclose(write);

//printf("%d\n",n);

}

## 二进制读数字

void read(point all\_p[],int \*count\_all)

{

FILE \*read;

int i=0;

if((read=fopen("G:\\file\_2014.txt","rb+"))==NULL)

{

printf("The file could be openned!\n");

system("pause");

exit(0);

}

while(!feof(read))

{

if(!feof(read))

fread(&all\_p[i].x,4,1,read);

if(!feof(read))

fread(&all\_p[i].y,4,1,read),++i;

}

\*count\_all=i;

for(i=0;i<\*count\_all;++i)

printf("( %d ,%d )\n",all\_p[i].x,all\_p[i].y);

}

void get\_tr(point all\_p[],int count\_all,point all\_tr[],int \*count\_tr)

{

int i,j;

for(i=0,j=0;i<count\_all;++i)

if(all\_p[i].x>0&&all\_p[i].y>0)

all\_tr[j]=all\_p[i],++j;

\*count\_tr=j;

}

## 所有点到某一点的距离

void get\_dis(point all\_tr[],int count\_tr,point p)

{

int i;

for(i=0;i<count\_tr;++i)

{

all\_tr[i].dis=sqrt((all\_tr[i].x-p.x)/1.0\*(all\_tr[i].x-p.x)/1.0+(all\_tr[i].y-p.y)/1.0\*(all\_tr[i].y-p.y)/1.0);

}

}

## 冒泡排序

void sort(point all\_tr[],int count\_tr)

{

int i,j;

point p;

for(i=0;i<count\_tr-1;++i)

for(j=0;j<count\_tr-1-i;++j)

{

if(all\_tr[j].dis>all\_tr[j+1].dis)

{

p=all\_tr[j];

all\_tr[j]=all\_tr[j+1];

all\_tr[j+1]=p;

}

}

}

## 写数字

void output(point all\_tr[],int count\_tr)

{

FILE \*write;

int i;

if((write=fopen("G:\\file\_2014\_output.txt","w+"))==NULL)

{

printf("The file couldn't be openned!\n");

system("pause");

exit(0);

}

for(i=0;i<count\_tr;++i)

fprintf(write,"( %d , %d ) %lf\n",all\_tr[i].x,all\_tr[i].y,all\_tr[i].dis);

fclose(write);

printf("The code is over!\n");

}

int main()

{

point all\_p[100],all\_tr[100],p;

int count\_all,count\_tr;

int i,k;

write();

read(all\_p,&count\_all);

printf("The number of all points is : %d \n",count\_all);

get\_tr(all\_p,count\_all,all\_tr,&count\_tr);

printf("The number of true points is : %d \n",count\_tr);

printf("Please enter a point (format:x,y):");

scanf("%d,%d",&p.x,&p.y);

get\_dis(all\_tr,count\_tr,p);

printf("Please enter a number :");

scanf("%d",&k);

sort(all\_tr,count\_tr);

output(all\_tr,k);

system("pause");

return 0;

}

# 2016

文本文件input.txt由若干英文单词和分隔符（空格，回车，换行）构成。根据如下说明编写程序统计不同单词出现的次数（频度）。将统计结果按出现频度从高到低排序，并将出现频度大于5的单词及其频度输出到文件output.txt中。文件格式如图1所示。

说明：

(1) 多个连续的分隔符被视为一个分隔符。

(2) 单词大小写敏感。

(3) 每个单词的长度不超过20个字符。

(4) 单词的数量未知。如使用定义静态大数组的方式来统计，将被扣除5分。

#include<stdio.h>

#include<stdlib.h>

#include<string.h>

typedef struct word

{

char w[20];

int count;

}word;

## 读单词

void read(char all\_w[100][20],int \*count\_all)

{

FILE \*read;

char word[20];

char ch;

int i;

if((read=fopen("G:\\file\_2016\_r.txt","r+"))==NULL)

{

printf("The file couldn't be openned!\n");

system("pause");

exit(0);

}

\*count\_all=0;

while(!feof(read))

{

fscanf(read,"%s",word);

strcpy(all\_w[\*count\_all],word);

++\*count\_all;

}

fseek(read,-1L,2);

ch=fgetc(read);

if(ch==' ') --\*count\_all;

for(i=0;i<\*count\_all;++i)

printf("%s ",all\_w[i]);

printf("\n%d\n",\*count\_all);

fclose(read);

}

int check\_ha(word w[],int count,char data[])

{

int i;

for(i=0;i<count;++i)

{

if(strcmp(w[i].w,data)==0)

return i+1;

}

return 0;

}

## 统计数组中每个单词出现的次数

void del\_re(word w[],int \*count,char all\_w[][20],int count\_all)

{

int i,j;

strcpy(w[0].w,all\_w[0]);

w[0].count=1;

\*count=1;

for(i=1,j=1;i<count\_all;++i)

if(check\_ha(w,\*count,all\_w[i]))

++w[check\_ha(w,\*count,all\_w[i])-1].count;

else

{

strcpy(w[j].w,all\_w[i]);

printf("%s ",all\_w[i]);

w[j].count=1;

++j;

++\*count;

}

printf("%d\n",\*count);

printf("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n");

for(i=0;i<\*count;++i)

printf("%s %d\n",w[i].w,w[i].count);

}

## 冒泡排序

void sort(word w[],int count)

{

int i,j;

word t;

for(i=0;i<count-1;++i)

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

if(w[j].count<w[j+1].count)

{

t=w[j];

w[j]=w[j+1];

w[j+1]=t;

}

}

## 写内容

void write(word w[],int count)

{

FILE \*write;

int i;

if((write=fopen("G:\\file\_2016\_w.txt","w+"))==NULL)

{

printf("The file couldn't be openned！ê?\n");

system("pause");

exit(0);

}

for(i=0;i<count;++i)

{

if(w[i].count>=5)

fprintf(write,"%s : %d\n",w[i].w,w[i].count);

}

}

int main()

{

char all\_w[100][20];

word w[100];

int count\_all,count;

read(all\_w,&count\_all);

del\_re(w,&count,all\_w,count\_all);

sort(w,count);

write(w,count);

system("pause");

return 0;

}

# 2015

和14年 12年 题目差不多。苏大特别喜欢出这种题目，这种题目不难，但按规律每年有1/3的人做不出来。

题目是简单，加上当时的紧张，还有键盘不熟悉（我一哥们用苹果电脑，ctrl+c、ctrl+v按法和标准键盘不样，导致...），所以我建议，用一般电脑进行练习。

15年题目:

从网页上下载input.dat文件，里面是用二进制编写的，里面放了一堆int型的数，每个数占4个字节，每次读取两个，这两个数构成一个坐标。

(1)规定处于第一象限的数是有效点，（即x>0,y>0的坐标），问这么多点中有效点有多少个？

当时答案是 352个。(说到这我不得不提醒大家，苏大不要求程序具有很强的通用性（这句话不是我说的，是要求上这么写的），所以不要想着让程序多么多么的完善，在那么短的时间内加上紧张，难免会……，所以就按给定的数据能运行成功就算成功。我当时是用#define MAX 200定义的，结果就出现了自己都意响不到的错误，程序跑飞了。害了我调了1个小时才把这个错误找到，你们用 数组时 一定要定义大一点啊！有的同学是用链表了，他考虑到的是程序的通用性。 不过链表处理起来可能比数组要考虑的事情更多了。但一般情况下有效点的个数不会超过1000个的，只是一般…… 有些同学担心怎么办，因为不考虑程序的通用性，所以你往程序里写入10000个二进制数看看大小是几KB,到考试时把文件下载下来，看看多大，估计下大概有多少个数据，然后再定义大点) 这是我的方法，当然你也可以采用链表，第一问大概 90%-95%的同学都做起来了。

(2)从键盘上输入k和n,从第一问中的有效点中找出 距离小于n，距离小于n的点的个数要大于k, 将它们以文本格式输出到文件中(导师就直接运行，运行完看你的文本文件中的答案,最后再看你的注释等等，总之先看结果，结果不对才分部给分；结果对至少65分+ )

第二问大约有 %50的人没做起来…… 所以你初试考了很低的分，你第二问做起来了，就相当于比没做起来的多加 30－40分，多加这么多分，你再看看你的排名呢。

这题我当时是 用到了 二重循环

举例 (1,3) (3,4) (2,8) (3,3)

输入 k=1 n=4

文件中的答案为：（意思为:每个点距离小于4的点的个数，个数大于1个则输出 ） （2，8）距其余三个点的距离都大于4，所以个数为0，所以不输出

X Y 个数

1 3 2

3 4 2

3 3 2

#include<stdio.h>

#include<stdlib.h>

#include<math.h>

typedef struct point

{

int x;

int y;

int count;

}point;

## 二进制写数字

void write()

{

FILE \*write;

int a[] = { 20, 9, 1, 1, -9, 56, 8, 9, -3, -7, 90, 2, -2, 8, 6, 8, 80, 3, -9, -44, -23, 10, 20, 3, -23, 89, 5, -5, 4, 3, 22, 77, 90, 10, -4, -9, -43, 9, -2, 6, -5, 9, -7, 90, 1, -2, 8, 6, 8, 90, 77, 99, 10, -4, -9, 99, -9, 56, 8, 9, -3, -7};

int n=sizeof(a)/4;

int i;

if((write=fopen("G:\\file\_2015.txt","wb+"))==NULL)

{

printf("The file couldn't be openned!\n");

system("pause");

exit(0);

}

for(i=0;i<n;++i)

{

putw(a[i],write);

}

fclose(write);

//printf("%d\n",n);

}

## 二进制读数字

void read(point p\_all[],int \*count\_all)

{

FILE \*read;

int i;

if((read=fopen("G:\\file\_2015.txt","rb+"))==NULL)

{

printf("The file couldn't be openned!\n");

system("pause");

exit(0);

}

i=0;

while(!feof(read))

{

if(!feof(read))

fread(&p\_all[i].x,4,1,read);

if(!feof(read))

fread(&p\_all[i].y,4,1,read),++i;

}

\*count\_all=i;

for(i=0;i<\*count\_all;++i)

{

printf("( %d , %d ) ",p\_all[i].x,p\_all[i].y);

if(i%6==0&&i!=0)

printf("\n");

}

printf("\n");

fclose(read);

}

## 得到有效点

void get\_tr(point p\_all[],int count\_all,point p\_tr[],int \*count\_tr)

{

int i,j;

for(i=0,j=0;i<count\_all;++i)

if(p\_all[i].x>0&&p\_all[i].y>0)

p\_tr[j]=p\_all[i],++j;

\*count\_tr=j;

for(i=0;i<\*count\_tr;++i)

{

printf("( %d , %d ) ",p\_tr[i].x,p\_tr[i].y);

if(i%6==0&&i!=0)

printf("\n");

}

printf("\n");

}

double get\_dis(point p1,point p2)

{

return sqrt((p1.x-p2.x)\*(p1.x-p2.x)+(p1.y-p2.y)\*(p1.y-p2.y));

}

void get\_count(point p\_tr[],int count\_tr,int n)

{

int i,j;

for(i=0;i<count\_tr;++i)

p\_tr[i].count=0;

for(i=0;i<count\_tr;++i)

for(j=0;j<count\_tr;++j)

if(get\_dis(p\_tr[i],p\_tr[j])>n)

++p\_tr[i].count;

}

## 写入格式

void output(point p\_tr[],int count\_tr,int k)

{

FILE \*write;

int i;

if((write=fopen("G:\\file\_2015\_output.txt","w+"))==NULL)

{

printf("The file couldn't be openned!\n");

system("pause");

exit(0);

}

fprintf(write,"%-5c %-5c %-5s\n",'X','Y',"个数");

for(i=0;i<count\_tr;++i)

{

if(p\_tr[i].count>k)

fprintf(write,"%-5d %-5d %-5d\n",p\_tr[i].x,p\_tr[i].y,p\_tr[i].count);

}

}

int main()

{

point p\_all[1000],p\_tr[1000];

int count\_all,count\_tr;

int n,k;

write();

read(p\_all,&count\_all);

get\_tr(p\_all,count\_all,p\_tr,&count\_tr);

printf("The number of true point is : %d \n",count\_tr);

printf("Please enter the n and k (format:n,k):");

scanf("%d,%d",&n,&k);

get\_count(p\_tr,count\_tr,n);

output(p\_tr,count\_tr,k);

system("pause");

return 0;

}