# 实验8 文件实验

## 8.1 实验目的

（1）熟悉文本文件和二进制文件在磁盘中的存储方式；

（2）熟练掌握流式文件的读写方法。

## 8.2 实验内容

8.2.1 文件类型的程序验证题

设有程序：

#include <stdio.h>

int main(void)

{

short a=0x253f,b=0x7b7d;

char ch;

FILE \*fp1,\*fp2;

fp1=fopen("d:\\abc1.bin","wb+");

fp2=fopen("d:\\abc2.txt","w+");

fwrite(&a,sizeof(short),1,fp1);

fwrite(&b,sizeof(short),1,fp1);

fprintf(fp2,"%hx %hx",a,b);

rewind(fp1); rewind(fp2);

while((ch = fgetc(fp1)) != EOF)

putchar(ch);

putchar('\n');

while((ch = fgetc(fp2)) != EOF)

putchar(ch);

putchar('\n');

fclose(fp1);

fclose(fp2);

return 0;

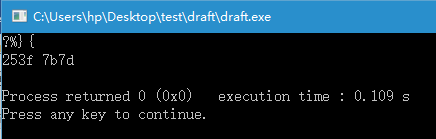
}

1. 请思考程序的输出结果，然后通过上机运行来加以验证。
2. 将两处sizeof(short)均改为sizeof(char)结果有什么不同，为什么？
3. 将fprintf(fp2,"%hx %hx",a,b) 改为 fprintf(fp2,"%d %d",a,b)结果有什么不同。

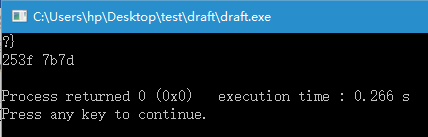
解答:

1. 思考程序的输出结果为：?%}{（依次为ASCII码中0x3f，0x25，0x7d，0x7b对应的字符），换行后输出253f 7b7d。

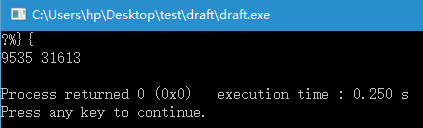
程序运行结果如图：



1. 将两处sizeof(short)均改为sizeof(char)后，只将0x3f和0x7d写入文件abc1.bin，故输出结果如图：



1. 将fprintf(fp2,"%hx %hx",a,b) 改为 fprintf(fp2,"%d %d",a,b)，即将0x253f与0x7b7d以整数形式写入txt文件，故输出结果如图：



8.2.2 源程序修改替换题

将指定的文本文件内容在屏幕上显示出来，命令行的格式为：

type filename

1. 源程序中存在什么样的逻辑错误（先观察执行结果）？对程序进行修改、调试，使之能够正确完成指定任务。
2. #include<stdio.h>
3. #include<stdlib.h>
4. int main(int argc, char\* argv[])
5. {
6. char ch;
7. FILE \*fp;
8. if(argc!=2){
9. printf("Arguments error!\n");
10. exit(-1);
11. }
12. if((fp=fopen(argv[1],"r"))==NULL){ /\* fp 指向 filename \*/
13. printf("Can't open %s file!\n",argv[1]);
14. exit(-1);
15. }
16. while(ch=fgetc(fp)!=EOF) /\* 从filename中读字符 \*/
17. putchar(ch); /\* 向显示器中写字符 \*/
18. fclose(fp); /\* 关闭filename \*/
19. return 0;
20. }

（2）用输入输出重定向freopen改写上述源程序中的main函数。

解答：

（1）第16行的语句逻辑存在问题，!=的优先级高于赋值‘=’，不添加括号会导致无法判断返回值，应修改为while((ch=fgetc(fp))!=EOF)；

替换后的程序如下所示：

#include<stdio.h>

#include<stdlib.h>

int main(int argc, char\* argv[])

{

char ch;

FILE \*fp;

if(argc!=2){

printf("Arguments error!\n");

exit(-1);

}

if((fp=fopen(argv[1],"r"))==NULL){ /

printf("Can't open %s file!\n",argv[1]);

exit(-1);

}

while((ch=fgetc(fp))!=EOF)

putchar(ch);

fclose(fp);

return 0;

}

创建一个文本文件x.txt，在cmd内通过type命令展示文本内容，后再通过程序输出文本内容；

运行截图：



（2）利用freopen改写（1）中的main函数，即不需使用文件指针\*fp，直接通过输入输出流向文本中存储数据。

修改后源程序：

#include<stdio.h>

#include<stdlib.h>

int main(int argc, char\* argv[])

{

char ch;

if(argc!=2){

printf("Arguments error!\n");

exit(-1);

}

if(freopen(argv[1],"r",stdin)==NULL){

printf("Can't open %s file!\n",argv[1]);

exit(-1);

}

freopen(argv[2],"w",stdout);

while((ch=getchar())!=EOF)

putchar(ch);

return 0;

}

重复1中操作以通过程序在cmd中输出文本内容；

程序运行结果：



8.2.3 程序设计

（1）从键盘输入一行英文句子，将每个单词的首字母换成大写字母，然后输出到一个磁盘文件“test”中保存。

解答：

1. 解题思路：
2. 定义文件指针\*fout；
3. 通过函数fopen新建文件，即：fout=fopen("d:\\test.txt","w+")；
4. 通过函数gets输入语句。若输入的不是文件尾则对每个单词首字母进行处理，随后通过fprintf将其输入文件test中保存，具体处理流程转3.1；否则转4。
   1. 先单独判断首字母是否为小写字母，是则将其替换为对应的大写字母，即：if(a[0]>='a'&&a[0]<='z')a[0]=a[0]+'A'-'a'；
   2. 对每个字符进行判断，若字符为小写字母且其之前的一个字符为空格，则将其替换为对应的大写字母，即：

for(i=1;a[i]!='\0';i++)

if(a[i-1]==' '&&a[i]>='a'&&a[i]<='z')

a[i]=a[i]+'A'-'a';

* 1. 通过fprintf将一行处理后的字符串输出至文件test.txt中；

1. 识别到文件尾，停止输入并关闭文件；
2. 结束。
3. 程序清单：

#include<stdio.h>

int main(void)

{

char a[100];

int i=0;

FILE \*fout;

fout=fopen("d:\\test.txt","w+");

while(gets(a)!=NULL){

if(a[0]>='a'&&a[0]<='z')a[0]=a[0]+'A'-'a';

for(i=1;a[i]!='\0';i++)

if(a[i-1]==' '&&a[i]>='a'&&a[i]<='z')

a[i]=a[i]+'A'-'a';

fprintf(fout,"%s\n",a);

}

fclose(fout);

}

1. 测试
2. 测试数据：

为方便分析测试结果，选择被处理的字符串分别为空格开头，小写字母开头，多行字符。如表8-1所示。

表8-1 编程题8.2.3.(1)的测试数据

|  |  |  |
| --- | --- | --- |
| 测试  用例 | 程 序 输 入（数据组数N=3） | 理 论 结 果 |
| 字符串 |
| 用例1 | aaa bbb ccc | Aaa Bbb Ccc |
| 用例2 | aaa bbb ccc | Aaa Bbb Ccc |
| 用例3 | aaa bbb ccc  ddd  hello world ! | Aaa Bbb Ccc  Ddd  Hello World ! |

1. 对应测试用例一的运行结果如图8-1所示。

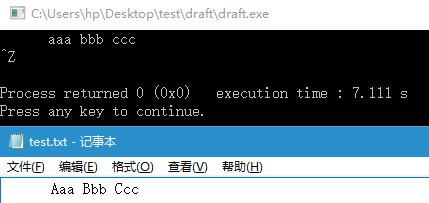


图8-1 编程题8.2.3.(1)的测试用例一的运行结果

1. 对应测试用例二的运行结果如图8-1所示。

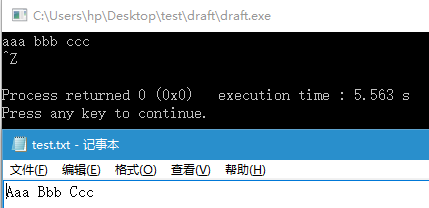


图8-2 编程题8.2.3.(1)的测试用例二的运行结果

1. 对应测试用例三的运行结果如图8-1所示。

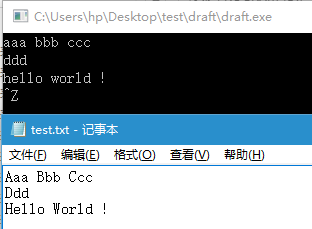


图8-3 编程题8.2.3.(1)的测试用例三的运行结果

说明上述的运行结果与理论分析吻合，验证了程序的正确性。

## 8.3 自设题

**（1）**自设实验题目：设有两个含一行字符串的文件A.txt与B.txt，对两行字符串中的字母按照从小到大的顺序进行排列，并将排列结果存入新文件C.txt中。

**（2）**实验目的：通过设计实验程序，熟练对文件的读与写的操作，从而掌握对文件进行处理的方式。

**（3）**实验程序：

#include <stdio.h>

#include <stdlib.h>

int main (void)

{

FILE \*fp;

int i,j,m,n;

char c[100],t,ch;

if ((fp=fopen("D:\A.txt","r"))==NULL){

printf("\ncan not open file A\n");

exit(0);

}

printf("file A :\n");

for (i=0;(ch=fgetc(fp))!=EOF;i++){

c[i]=ch;

putchar(c[i]);

}

fclose(fp);

m=i;

if ((fp=fopen("D:\B.txt","r"))==NULL){

printf("\ncan not open file B\n");

exit(0);

}

printf("\nfile B:\n");

for (;(ch=fgetc(fp))!=EOF;i++){

c[i]=ch;

putchar(c[i]);

}

fclose(fp);

n=i;

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

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

if (c[i]>c[j]){

t=c[i];

c[i]=c[j];

c[j]=t;

}

printf("\nfile C :\n");

fp=fopen("D:\C.txt","w+");

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

putc(c[i],fp);

putchar(c[i]);

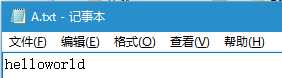
}

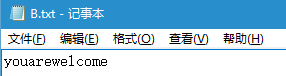
printf("\n");

fclose(fp);

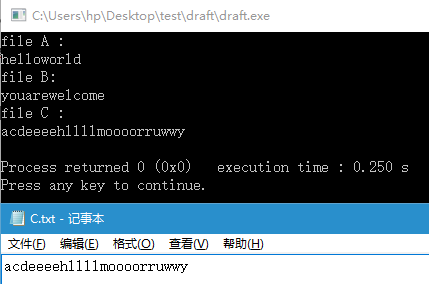
}

**（4）**实验用例：如图





程序运行结果：



**（5）**实验结论：验证了程序的正确性。

## 8.4 实验小结

在实验中主要遇到的问题有：

1. 在8.2.1文件类型程序验证的问题（1）中，开始因为对文件的写入与读取机制不够了解误判为输出%？{}，后在运行程序验证时发现问题并重新思考后解决。
2. 在8.2.2源程序修改替换时，开始没有理解题目的意思，后通过cmd与主函数参数解决了问题，另外改问题也可以通过codeblock中创建项目，设定arguments来解决。
3. 在解决问题8.2.3程序设计（1）时，通过fopen与w+创建文件时开始因为文件地址的问题无法新建，后换为更简单的地址成功新建了文件。

体会：

1. 在对文件进行相关操作时可以利用不同的函数完成同样的要求，如8.2.2源程序修改替换中的问题（2），可以通过函数freopen直接对输入输出流进行操作从而完成同样的要求，而不新建文件指针，在编程时多多注意这方面解决问题的多样性能够有效的提升自己的编程能力；
2. 解决问题时应多加注意普适性，如8.2.3程序设计题（1）中涉及将输入的字符串处理后输出到文件当中进行存储，应尽量设计能够处理多行字符串的程序而不是仅仅设计能够处理一行字符的程序；
3. 因为刚开始接触C语言，尚未利用C语言进行大型程序或者功能的设计，目前做的练习涉及文件的还较少。但在以后通过代码实现指定功能时便必定需要程序员对文件的相关操作足够熟练，这还需要我们对自身多加锻炼。