# 8文件实验

## 8.1实验目的

熟悉文本文件和二进制文件在磁盘中的存储方式；掌握流式文件的读写方法。

## 8.2 必做题

### 8.2.1 文件程序验证

**【题目】**

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。

**【答案】**

（1）预测程序运行输出结果-分别为？%} { 和253f 7b7d。

实际输出结果如下图8-1、图8-2和图8-3所示。与预测结果一致。

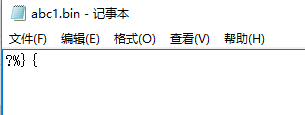


图8-1文件输出结果

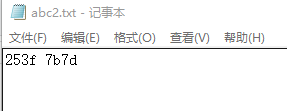


图8-1文件输出结果

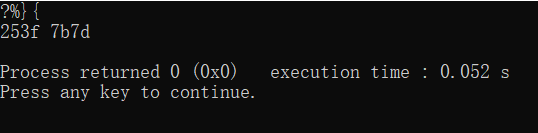


图8-3 控制台输出结果

（2）第一个更改为sizeof(char)后，读入时只读入一个大小为char的数据，从而导致输出的长度减半，即为?}；而第二个更改为sizeof(char)后，因为后面会被fprintf覆盖，所以不会造成影响。输出结果如下图8-4、图8-5所示。

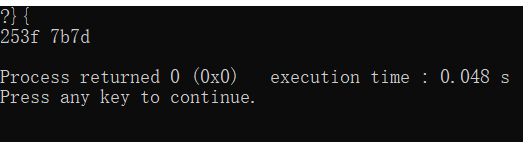


图8-4 实验八必做题8.2.1（2）修改第一个后输出结果

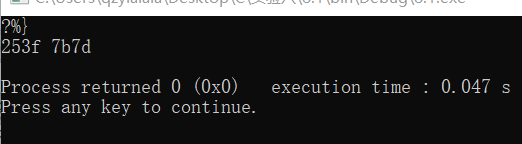


图8-5 实验八必做题8.2.1（2）修改第二个后输出结果

（3）改为%d后，将以十进制形式输出a，b，预测结果为9535 31613。实际结果如下图8-4.

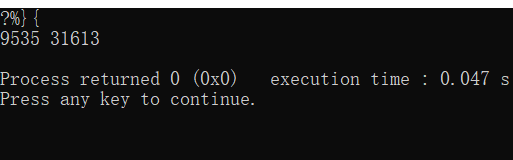


图8-6 实验八必做题8.2.1（3）输出结果

### 8.2.2源程序修改替换

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。

1. #include<stdio.h>
2. #include<stdlib.h>
3. int main(int argc, char\* argv[])
4. {
5. char ch;
6. FILE \*fp;
7. if(argc!=2){
8. printf("Arguments error!\n");
9. exit(-1);
10. }
11. if((fp=fopen(argv[1],"r"))==NULL){ /\* fp 指向 filename \*/
12. printf("Can't open %s file!\n",argv[1]);
13. exit(-1);
14. }
15. while(ch=fgetc(fp)!=EOF) /\* 从filename中读字符 \*/
16. putchar(ch); /\* 向显示器中写字符 \*/
17. fclose(fp); /\* 关闭filename \*/
18. return 0;
19. }

（2）用输入输出重定向freopen改写上述源程序中的main函数。

**【错误分析及修改】**

1.命令行参数中的argv[0]被文件名占用，所以输入的type应该为argv[1]，并且总参数应该是3个。应该将if(argc!=2)改为if (argc != 3)，同时将判断filename均改为argv[2]。

2.读入字符时没有考虑到=与!=的优先级。应在ch=fgetc(fp)外加（）来保证运算优先级。

3、源程序存在一个缺陷。由于仅仅判断是否有两个参数，故输入t filename的时候也会正常执行。修改这个缺陷只要加入判断参数argv[1]是否为type的代码。即为：

if (strcmp(argv[1], "type") != 0)

{

printf("%s is not a right command", argv[1]);

exit(-1);

}

**【修改后代码】**

#include<stdio.h>

#include<stdlib.h>

#include<string.h>

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

{

char ch;

FILE \*fp;

if (argc != 3) {

printf("Arguments error!\n");

exit(-1);

}

if (strcmp(argv[1], "type") != 0)

{

printf("%s is not a right command", argv[1]);

exit(-1);

}

if ((fp = fopen(argv[2], "r")) == NULL) { /\* fp 指向 filename \*/

printf("Can't open %s file!\n", argv[2]);

exit(-1);

}

while ((ch = fgetc(fp)) != EOF) /\* 从filename中读字符 \*/

putchar(ch); /\* 向显示器中写字符 \*/

fclose(fp); /\* 关闭filename \*/

return 0;

}

### 8.2.3 大小写转化

**【题目】**

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。

**【程序清单】**

#include<stdio.h>

enum \_state { SPACE, COPY };

void change(char str[]);

int main()

{

FILE \*fp1;

char str[100];

fp1 = fopen("D:\\m.txt", "w+");

fgets(str, 100, stdin);

change(str);

fprintf(fp1, "%s", str);

fclose(fp1);

return 0;

}

void change(char str[])

{

int i = 0;

enum \_state state;

if(str[0]==' ') state = SPACE;

else {

str[0] ^= 32;

state = COPY;

}

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

{

switch (state)

{

case SPACE:

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

str[i] ^= 32;

if (str[i] == ' ')

state = SPACE;

else state = COPY;

break;

case COPY:

if (str[i] == ' ')

state = SPACE;

else state = COPY;

break;

}

}

}

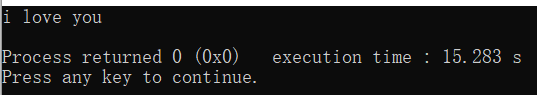
**【测试样例、测试理论结果及测试结果】**

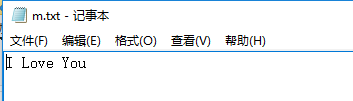
1、在控制台输入“i love you.”，预计在文件中输出“I Love You.”；

2、在控制台输入“ you love me, aren’t you?”，预计在文件中输出“You Love Me, Aren’t You?”；

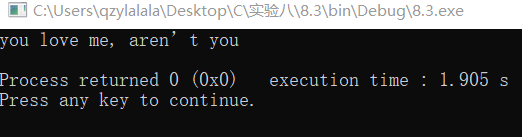
3、在控制台输入“ i love you and the world. Haha!”，空格保留，大写保留，预计在文件中输出“ I Iove You And The World. Haha!”；

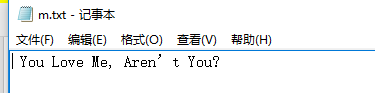
实际输出分别如图8-，图8.5-2，图8.5-3所示，完全符合预测**。**



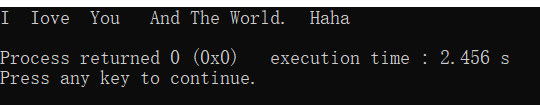


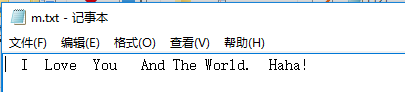
实验八必做题8.2.3样例1控制台输入及文件输出





实验八必做题8.2.3样例2控制台输入及文件输出





实验八必做题8.2.3样例3控制台输入及文件输出

## 8.3小结

通过完成这几个文件实验，我对文件的一些基础函数有了较好的掌握，同时真正理解了文件的读与写。