

## 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 实验题目及要求

#### 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');

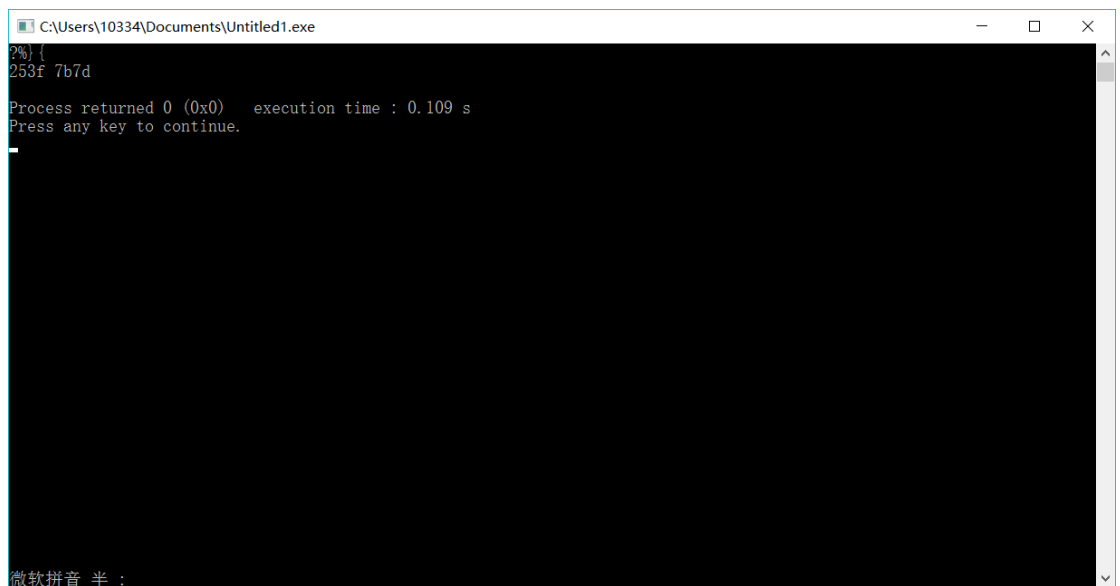
    while((ch = fgetc(fp2)) != EOF)
        putchar(ch);
```

```
    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。

- a) ?%}{
- b) 253f 7b7d



- c) 微软拼音 半 :

(2) 将两处 sizeof(short)均改为 sizeof(char)结果有什么不同，为什么？

- a) ?}
- b) 253f 7b7d

```
C:\Users\10334\Documents\Untitled1.exe
?}
253f 7b7d
Process returned 0 (0x0)   execution time : 1.244 s
Press any key to continue.
```

c) 微软拼音 半 :

- d) 原因: char 类型为一个字节, short 类型为两个字节, 那么将 short 改为 char 后, 运行第一个 fwrite() 时, 只将 a (0x253f) 的第一个字节的信息, 即 3f 存储到了 d:\abc1.bin 文件中, 在运行第二个 fwrite() 时, 只将 b = 0x7b7d 的第一个字节的信息 (即 7d) 存储在了 abc1.bin 中, 因此在执行 while((ch = fgetc(fp1)) != EOF) putchar(ch); 时只输出了 3f 对应的 acii 码 ‘?’ 和 7d 对应的 acii 码 ‘}’。

(3) 将 fprintf(fp2,"%hx %hx",a,b) 改为 fprintf(fp2,"%d %d",a,b) 结果有什么不同。

a) ?%}{

b) 9535 31613

```
C:\Users\10334\Documents\Untitled1.exe
?%){
9535 31613
Process returned 0 (0x0)   execution time : 0.135 s
Press any key to continue.
```

c) 微软拼音 半 :

- d) 第一次输出 253f 7b7d 是以 16 进制数存储的, 将 %hx 改为 %d 后是以 10

进制数存储的。

### 8.2.2. 源程序修改替换题

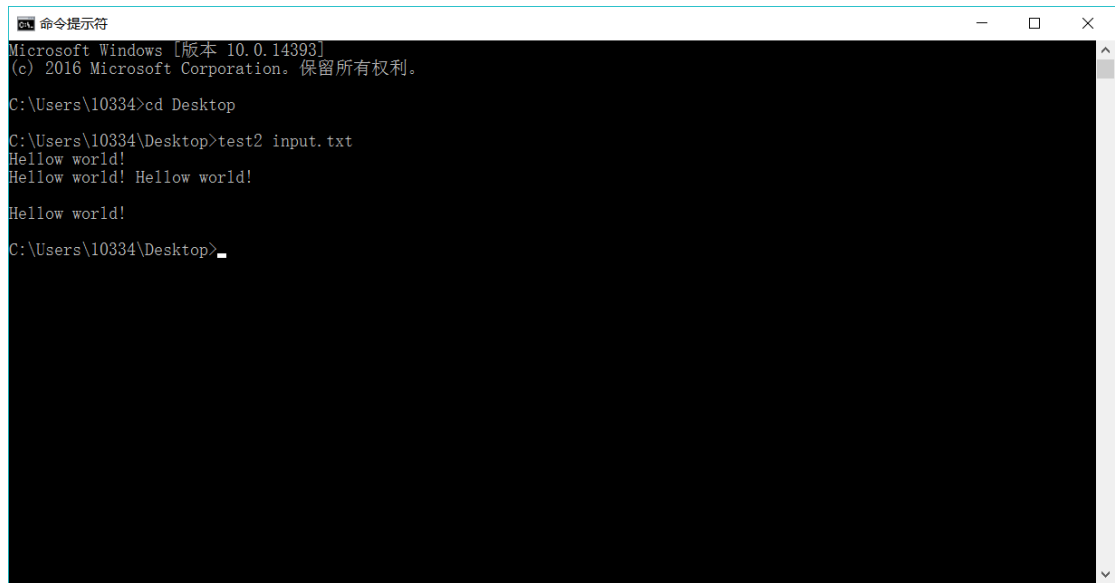
将指定的文本文件内容在屏幕上显示出来，命令行的格式为：

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。

```
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                                                     */
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 }
```





```
命令提示符
Microsoft Windows [版本 10.0.14393]
(c) 2016 Microsoft Corporation. 保留所有权利。

C:\Users\10334>cd Desktop
C:\Users\10334\Desktop>test2 input.txt
Hellow world!
Hellow world! Hellow world!

Hellow world!
C:\Users\10334\Desktop>
```

### 3. 编程设计题

(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。

程序清单：

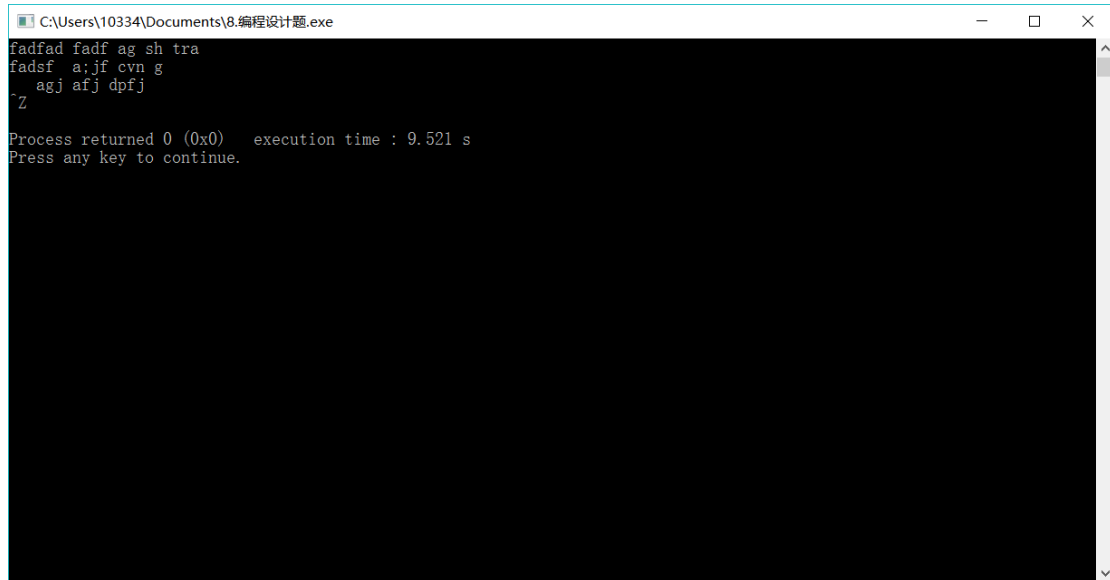
```
#include<stdio.h>

#include<ctype.h>

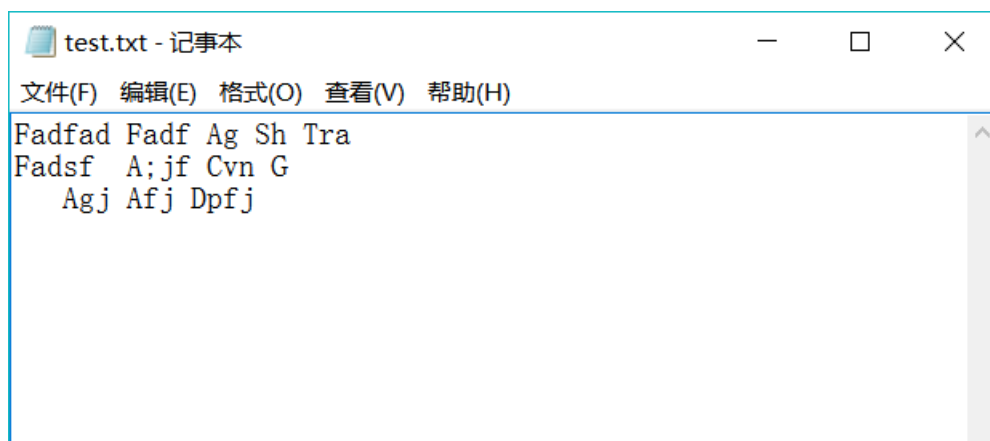
int main()
{
    FILE * fp;
    fp=fopen("test.txt", "w", stdout);
    char c, last;
    last=' ';
    while((c=getchar())!=EOF)
    {
        if((last==' ' || last=='\n') && c!=' ')
            putchar(toupper(c));
        else putchar(c);
        last=c;
    }
    fclose(fp);
}
```

```
    return 0;  
}
```

运行结果:



```
C:\Users\10334\Documents\8.编程设计题.exe  
fadfad fadf ag sh tra  
fadsf a;jf cvn g  
agj afj dpfj  
~Z  
Process returned 0 (0x0) execution time : 9.521 s  
Press any key to continue.
```



```
test.txt - 记事本  
文件(F) 编辑(E) 格式(O) 查看(V) 帮助(H)  
Fadfad Fadf Ag Sh Tra  
Fadsf A;jf Cvn G  
Agj Afj Dpfj
```

#### 8.4 实验感想

文件的读写操作让 c 程序的数据得以保存，让数据的存取变得简单易行，让我感觉自己写的程序进入了更高的台阶，程序的生命周期得以延长，要加强对文件读写操作的练习，争取熟练掌握。