



院 系 数据科学与计算机学院 班 级 19 学 号 19335174 姓 名 施天予

【实验题目】文件打包实验

【实验目的】掌握文件打包的方法。

【实验说明】

- ♦ 把源程序和可执行文件放在相应的上交源码目录中。
- ♦ 截屏用按键 (Ctrl+Alt+PrintScreen) 截取当前窗口
- ♦ 把每段具有独立功能的代码单独写入一个函数有助于编码和调试。

【参考资料】

- ♦ C 语言字符串函数: [http://msdn.microsoft.com/zh-cn/library/f0151s4x\(v=vs.110\).aspx](http://msdn.microsoft.com/zh-cn/library/f0151s4x(v=vs.110).aspx)
- ♦ C 语言程序设计: <http://www.runoob.com/cprogramming/>
- ♦ C 语言函数分类: [http://msdn.microsoft.com/zh-cn/library/2aza74he\(v=vs.110\).aspx](http://msdn.microsoft.com/zh-cn/library/2aza74he(v=vs.110).aspx)

【实验环境】

- ♦ Windows + VS 2017
- ♦ Linux + gcc

【实验内容】

1、(StructSave.cpp)把输入的结构数据保存到文件中。

▪ 实验要求:

循环输入员工(Person)的信息, 每输入一个员工的信息, 立即写入文件(Persons.stru), 直到输入的姓名为 **exit** 时跳出循环。

Person 的信息表示:

```
struct Person {  
    char username[USER_NAME_LEN];    // 员工名  
    int level;                        // 工资级别  
    char email[EMAIL_LEN];           // email 地址  
    DWORD sendtime;                  // 发送时间  
    time_t regtime;                  // 注册时间  
};
```

▪ 字符串函数和自定义函数(仅作参考):

```
printf(), scanf_s(), strcpy_s()  
int inputOnePerson(Person *personSent){...}
```

▪ 参考运行结果:



```
C:\实验测试\StructSave\Debug>StructSave  
name: zhang  
level: 2  
email: zhang@sina.com  
name: li  
level: 10  
email: li@163.com  
name: wang  
level: 9  
email: wang@mail.sysu.edu.cn  
name: exit  
press any key to continue...
```

▪ 完成后截屏运行结果:



Microsoft Visual Studio 调试控制台

```
name: zhang
level: 8
email: zhang@sina.com

name: li
level: 10
email: li@163.com

name: wang
level: 9
email: wang@mail.sysu.edu.cn

name: exit
struct copy finished!
press any key to continue...
```

源代码:

```
#include <stdio.h>
#include <string.h>
#include <stdlib.h>
#include <time.h>

#define FNAME_LEN 300
#define USER_NAME_LEN 20
#define EMAIL_LEN 80
#define TIME_BUF_LEN 30
typedef unsigned long DWORD;

typedef struct {
    char username[USER_NAME_LEN];    // 员工名
    int level;                       // 工资级别
    char email[EMAIL_LEN];           // email地址
    DWORD sendtime;                  // 发送时间
    time_t regtime;                  // 注册时间
} Person;

int ReadPerson(Person* personSent) {
    int ReadPerson(Person* personSent) {
        char pts[TIME_BUF_LEN];      /* pointer to time string */
        time_t now;                  /* current time */
        (void)time(&now);             /* 取得系统时间 */
        ctime_s(pts, TIME_BUF_LEN, &now); /* 把时间转换为字符串 */
        char inputBuf[100];
        int inputNumber;
        /* 输入员工记录 */
        printf("name: ");
        scanf_s("%s", inputBuf, USER_NAME_LEN); // 输入用户名
        if (!strcmp(inputBuf, "exit")) return 0;
        strcpy_s(personSent->username, inputBuf);
        printf("level: ");
        scanf_s("%d", &inputNumber); // 输入用户级别
        personSent->level = inputNumber;
        printf("email: ");
        scanf_s("%s", inputBuf, EMAIL_LEN);
        strcpy_s(personSent->email, inputBuf); // 输入电子邮件
        personSent->sendtime = (DWORD)now;    // 设置发送时间
        personSent->regtime = now;            // 设置注册时间
        printf("\n");
        return 1;
    }
}
```



```
int main() {
    FILE* pFile;
    Person per;

    // 打开要写的二进制文件(w-write b-binary), 没有则创建, 有则覆盖
    if ((fopen_s(&pFile, "e:\\temp\\aaa.stru", "wb")) != NULL) {
        printf("cant open the file!\n");
        getchar();
        exit(0);
    }

    while (true) {
        if (!ReadPerson(&per)) break;
        if (fwrite(&per, sizeof(Person), 1, pFile) != 1) {
            printf("file write error!\n");
        }
    }

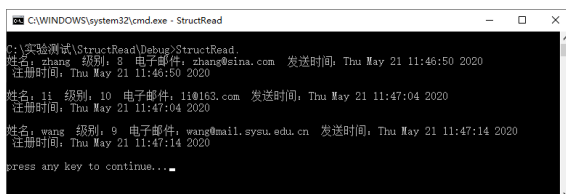
    fclose(pFile);
    printf("struct copy finished!\n");
    printf("press any key to continue...");
    getchar();
    return 0;
}
```

2、(StructRead. cpp)从文件读出结构数据并显示出来。

实验要求:

读出上个步骤在文件(Persons.stru)中保存的结构数据并显示出来。

参考运行结果:



完成后截屏运行结果:

E:\VSource\ConsoleApplication1\Release\ConsoleApplication1.exe

```
用户名: zhang
级别: 8
Email地址: zhang@sina.com
发送时间: Sat Mar 6 19:03:17 2021
注册时间: Sat Mar 6 19:03:17 2021
用户名: li
级别: 10
Email地址: li@163.com
发送时间: Sat Mar 6 19:03:39 2021
注册时间: Sat Mar 6 19:03:39 2021
用户名: wang
级别: 9
Email地址: wang@mail.sysu.edu.cn
发送时间: Sat Mar 6 19:03:48 2021
注册时间: Sat Mar 6 19:03:48 2021
Press any key to continue...
```



■ 源代码:

```
1  #include <stdio.h>
2  #include <string.h>
3  #include <stdlib.h>
4  #include <time.h>
5
6  #define FNAME_LEN 300
7  #define USER_NAME_LEN 20
8  #define EMAIL_LEN 80
9  #define TIME_BUF_LEN 30
10 typedef unsigned long DWORD;
11
12 typedef struct {
13     char username[USER_NAME_LEN]; // 员工名
14     int level; // 工资级别
15     char email[EMAIL_LEN]; // email地址
16     DWORD sendtime; // 发送时间
17     time_t regtime; // 注册时间
18 } Person;
19
20 void PrintPerson(Person* per) {
21     /* 读出并显示一个员工记录 */
22     char regtime[TIME_BUF_LEN];
23     char sendtime[TIME_BUF_LEN];
24     // 显示员工记录
25     printf("用户名: %s\r\n", per->username); // 显示用户名
26     printf("级别: %d\r\n", per->level); // 显示级别
27     printf("Email地址: %s\r\n", per->email); // 显示email
28
29     time_t t1 = (time_t)per->sendtime;
30     ctime_s(sendtime, TIME_BUF_LEN, &t1);
31     printf("发送时间: %s", sendtime); // 显示发送时间
32     ctime_s(regtime, TIME_BUF_LEN, &per->regtime);
33     printf("注册时间: %s", regtime); // 显示注册时间
34 }
35
36 int main() {
37     FILE* pFile;
38     Person per;
39     // 打开要读取的二进制文件(r-read b-binary)
40     if (fopen_s(&pFile, "e:\\temp\\Persons.stru", "rb") != NULL) {
41         printf("读入文件未找到! \n");
42         printf("按任意键继续...");
43         getchar();
44         getchar();
45         exit(1);
46     }
47     while (fread(&per, sizeof(Person), 1, pFile) == 1) {
48         PrintPerson(&per);
49     }
50     fclose(pFile);
51     printf("\r\nPress any key to continue...");
52     getchar();
53     getchar();
54     return 0;
55 }
```



3、打包文件 (FilePack.cpp)

实验要求:

- (1) 先输入打包文件名 (含路径);
- (2) 循环输入要打包的文件名 (含路径), 每输入一个, 就把该文件的文件名 (最多 300 字节)、文件大小(long) 和文件内容写入文件 FileSet.pak 中, 当输入文件名为 `exit` 时跳出循环。

老师用到的字符串函数和自定义函数(仅作参考):

`strcpy_s()`, `scanf_s()`, `printf()`, `strcmp()`, `sprintf_s()`—用于多个字符串和整数合并成一个字符串
`strrchr()`—反向查找字符

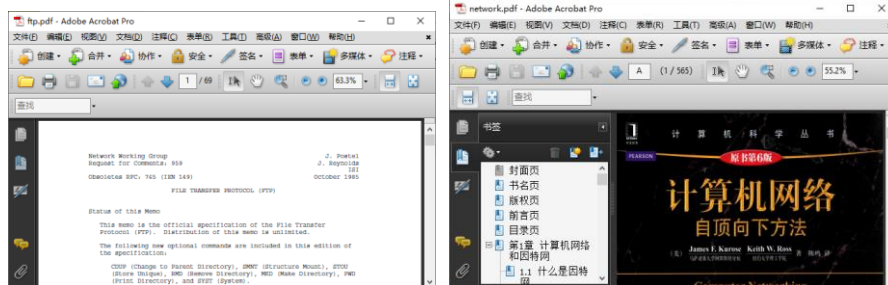
```
struct FileStruct {  
    char fileName[300];  
    __int64 fileSize;  
};  
  
__int64 getFileSize(char * fileName) {...}  
char * getFileName(char *pathName) {...}  
int packFile(char *srcFileName, FILE * destFile) {...}
```

参考运行结果:

c:\Test 下有两个文件 ftp.pdf 和 network.pdf, c:\Test\s 为空文件夹。

此电脑 > OS (C:) > Test				
名称	修改日期	类型	大小	
s	2020/5/21 14:26	文件夹		
ftp.pdf	2002/3/28 4:53	PDF 文件	84 KB	
network.pdf	2017/10/30 16:47	PDF 文件	121,686 KB	

打开 ftp.pdf 和 network.pdf



运行程序:

```
C:\WINDOWS\system32\cmd.exe - FilePack  
C:\实验测试>cmd  
Microsoft Windows [版本 10.0.18362.836]  
(c) 2019 Microsoft Corporation. 保留所有权利。  
  
C:\实验测试>FilePack  
输入目标文件名(含路径): c:\Test\FilePack.pak  
输入要打包的#1文件(含路径): c:\Test\ftp.pdf  
输入要打包的#2文件(含路径): c:\Test\network.pdf  
输入要打包的#3文件(含路径): c:\Test\ftp.pdf  
输入要打包的#4文件(含路径): c:\Test\network.pdf  
输入要打包的#5文件(含路径): c:\Test\ftp.pdf  
输入要打包的#6文件(含路径): exit  
打包结束!  
按任意键继续...
```

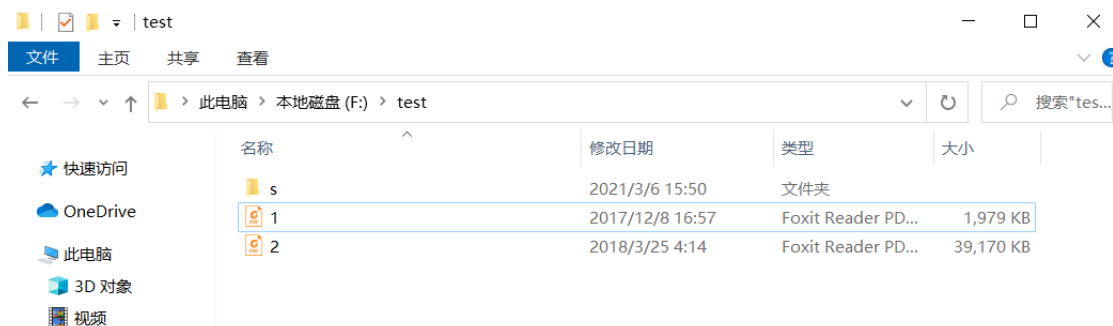
c:\Test 的内容变为:



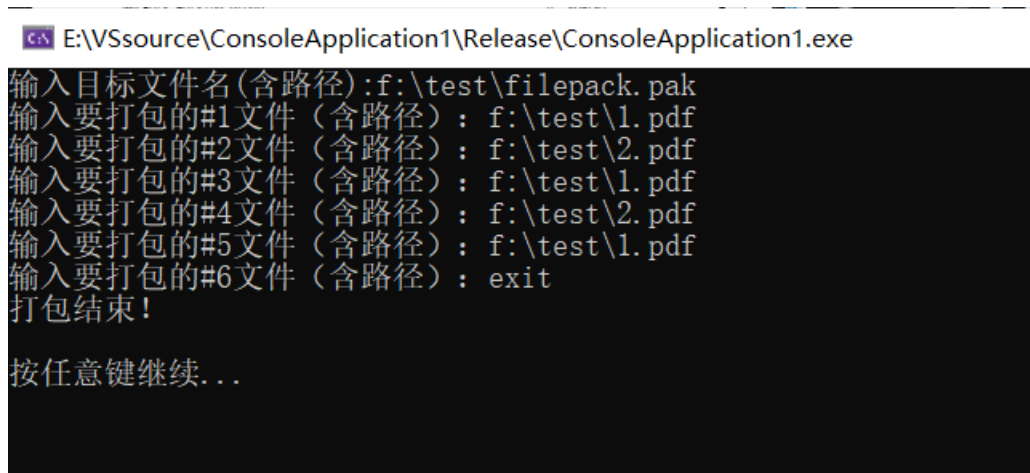
名称	修改日期	类型	大小
s	2020/5/21 14:26	文件夹	
FilePack.pak	2020/5/21 14:45	PAK 文件	243,622 KB
ftp.pdf	2002/3/28 4:53	PDF 文件	84 KB
network.pdf	2017/10/30 16:47	PDF 文件	121,686 KB

完成后测试截屏：

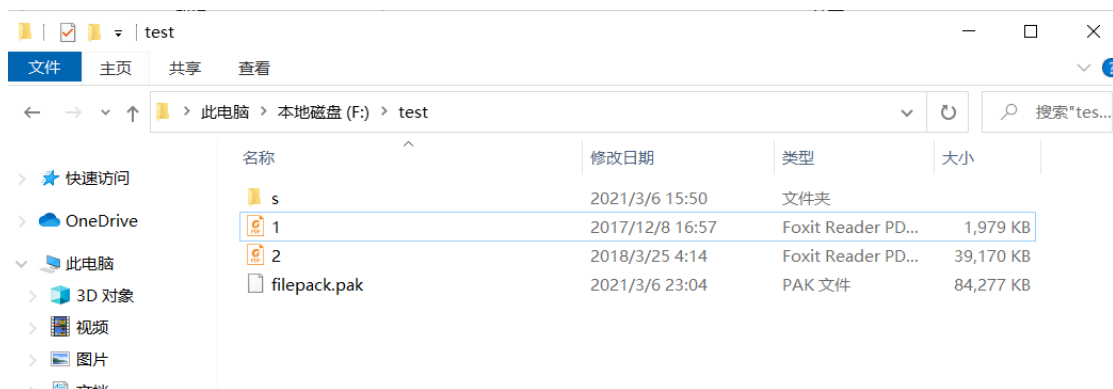
f:\test 下有两个文件 1. pdf 和 2. pdf, f:\test\s 为空文件夹。



运行程序：



f:\test 的内容变为：



源代码：



```
1  #include <stdio.h>
2  #include <string.h>
3  #include <stdlib.h>
4
5  #define FILE_NAME_LEN 300
6  #define BUF_LEN 20
7
8  typedef struct {
9      char fileName[FILE_NAME_LEN];
10     __int64 fileSize;
11 } FileStruct;
12
13 char name[FILE_NAME_LEN];
14 char buf[BUF_LEN];
15
16 __int64 getFileSize(char* fileName) {
17     FILE* fp;
18     if (fopen_s(&fp, fileName, "rb") != NULL) {
19         printf("文件打开失败!\n");
20         return -1;
21     }
22     fseek(fp, 0L, SEEK_END);
23     int s = ftell(fp);
24     fclose(fp);
25     return s;
26 }
27
28 void getFileName(char* name) {
29     printf("输入目标文件名(含路径):");
30     scanf_s("%s", name, FILE_NAME_LEN);
31 }
32
33 int packFile(char* srcFileName, FILE* destFile) {
34     FILE* sfile = NULL;
35     if (fopen_s(&sfile, name, "wb") != NULL) {
36         printf("文件打开失败!\n");
37         return 0;
38     }
39     int len = 0;
40     while ((len = fread(buf, 1, BUF_LEN, sfile)) >= BUF_LEN)
41         fwrite(buf, 1, BUF_LEN, destFile);
42     fwrite(buf, 1, len, destFile);
43     fclose(sfile);
44     return 1;
45 }
46
47 int main() {
48     FILE* dfile = NULL;
49     FileStruct f;
50     int cnt = 0;
51     char srcFileName[FILE_NAME_LEN] = "f:\\test\\filepack.pak";
```



```
52     getFileName(srcFileName);
53     if ((fopen_s(&dfile, srcFileName, "wb")) != NULL) {
54         printf("文件打开失败!\n");
55         return 0;
56     }
57     while (true) {
58         printf("输入要打包的%d文件 (含路径): ", ++cnt);
59         scanf_s("%s", name, FILE_NAME_LEN);
60         if (strcmp(name, "exit") == 0) break;
61         f.fileSize = getFileSize(name);
62         if (f.fileSize == -1) return 0;
63         strcpy_s(f.fileName, FILE_NAME_LEN, strrchr(name, '\\') + 1);
64         if (fwrite(&f, sizeof(FileStruct), 1, dfile) != 1)
65             printf("file write error!\n");
66         packFile(srcFileName, dfile);
67     }
68     printf("打包结束! \n");
69     printf("\n");
70     printf("按任意键继续...\n");
71     fclose(dfile);
72     getchar();
73     getchar();
74     return 0;
75 }
```

4、解包文件 (FileUnpack.cpp)

■ 实验要求:

- (1) 输入解包路径
- (2) 输入要解包的文件名, 然后解包其中每一个文件, 有重名文件时文件名加上序号 (从 2 开始)。

■ 老师用到的字符串函数和自定义函数(仅作参考):

strcpy_s(), scanf_s(), printf(), sprintf_s()

```
struct FileStruct {
    char fileName[300];
    __int64 fileSize;
};

// 拷贝filePathName中前面长度为len的字符串到fileFullName
int mystrcpy(char * fileFullName, int len, char * filePathName) {...}
void getUniqueName(char *newFileName, char *filePathName) {...}
int unpackFile(FILE *srcFile, char *Path) {...}
```

■ 参考运行结果:

运行程序:

```
C:\WINDOWS\system32\cmd.exe - FileUnpack
C:\实验测试>cd
Microsoft Windows [版本 10.0.18362.836]
(c) 2019 Microsoft Corporation. 保留所有权利。

C:\实验测试>FileUnpack
输入目标文件夹: c:\Test\s
输入要解包的文件: c:\Test\FilePack.pak
解包结束! 按任意键继续...
```



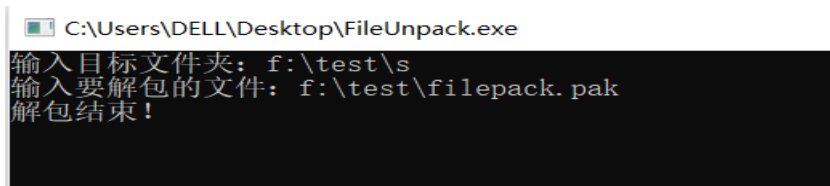

c:\Test\s 的内容变为:

名称	修改日期	类型	大小
ftp(2).pdf	2020/5/21 14:58	PDF 文件	84 KB
ftp(3).pdf	2020/5/21 14:58	PDF 文件	84 KB
ftp.pdf	2020/5/21 14:58	PDF 文件	84 KB
network(2).pdf	2020/5/21 14:58	PDF 文件	121,686 KB
network.pdf	2020/5/21 14:58	PDF 文件	121,686 KB

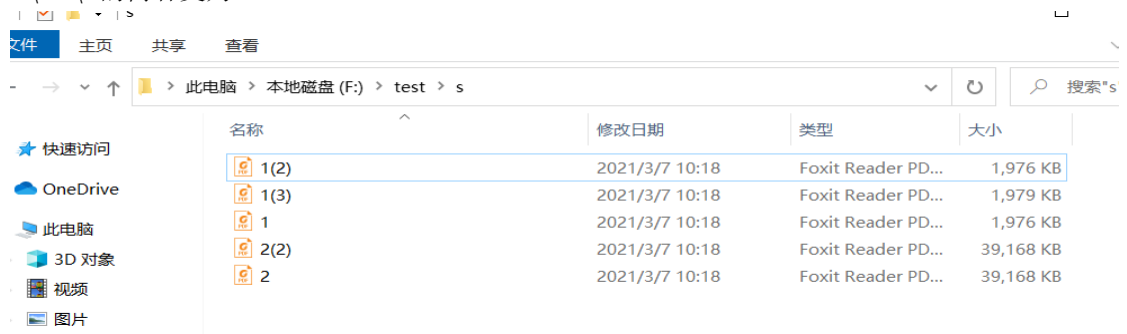
c:\Test\s 下的五个文件均可打开, 显示内容正常, 字节数与源文件相同。

■ 完成后截屏运行结果:

运行程序:



f:\test\s 的内容变为:



f:\test\s 下的五个文件均可打开, 显示内容正常, 字节数与源文件相同。

■ 源代码:

```
1  #include <stdio.h>
2  #include <string.h>
3  #include <stdlib.h>
4
5  #define FILE_NAME_LEN 300
6  #define BUF_LEN 20
7  #define HASH_SIZE 19260817
8
9  typedef struct {
10     char fileName[FILE_NAME_LEN];
11     __int64 fileSize;
12 } FileStruct;
13
14 int HashFunc(char* str) {
15     int s = 131, hash = 0;
16     while (*str)
17         hash = hash * s + (*str++);
18     return (hash & 0x7FFFFFFF) % HASH_SIZE;
19 }
20
21 char ans[BUF_LEN];
22 char* div(char* filePathName) {
23     memset(ans, '\0', sizeof(ans));
24     char* p = strrchr(filePathName, '.');
25     for (int i = 0; i < strlen(filePathName) - strlen(p); ++i)
26         ans[i] = filePathName[i];
27     return ans;
28 }
29
```



```
30 int cnt[HASH_SIZE] = { 0 };
31 int unpackFile(FILE* sfile, char* Path) {
32     FILE* dfile = NULL;
33     FileStruct f;
34     while (fread(&f, sizeof(FileStruct), 1, sfile) == 1) {
35         char temp[BUF_LEN];
36         char part[BUF_LEN] = "(0)";
37         strcpy_s(temp, FILE_NAME_LEN, Path);
38         strcat_s(temp, FILE_NAME_LEN, "\\");
39         strcat_s(temp, FILE_NAME_LEN, div(f.fileName));
40         int res = HashFunc(f.fileName);
41         if (cnt[res]++) {
42             part[1] += cnt[res];
43             strcat_s(temp, FILE_NAME_LEN, part);
44         }
45         strcat_s(temp, FILE_NAME_LEN, strrchr(f.fileName, '.'));
46         if (fopen_s(&dfile, temp, "wb") != NULL) {
47             printf("文件解包失败! \n");
48             getchar();
49             getchar();
50             return 0;
51         }
52         int left = f.fileSize;
53         char buf[BUF_LEN];
54         while (left) {
55             int Size = left > BUF_LEN ? BUF_LEN : left;
56             fread(buf, Size, 1, sfile);
57             fwrite(buf, Size, 1, dfile);
58             left -= Size;
59         }
```

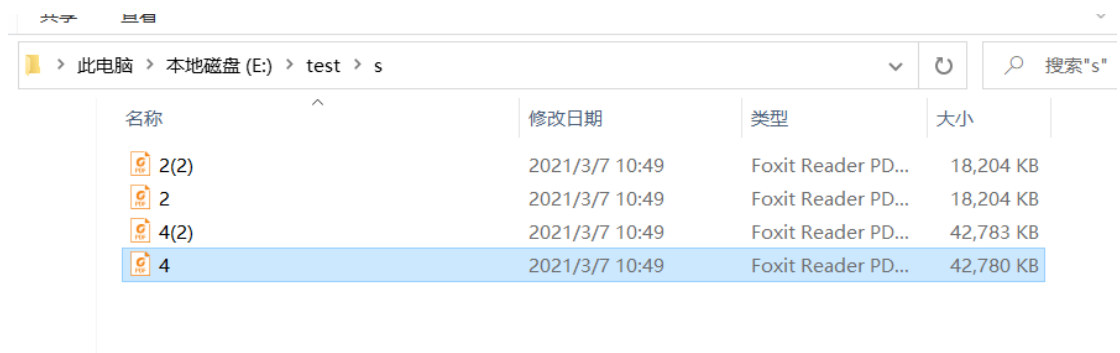
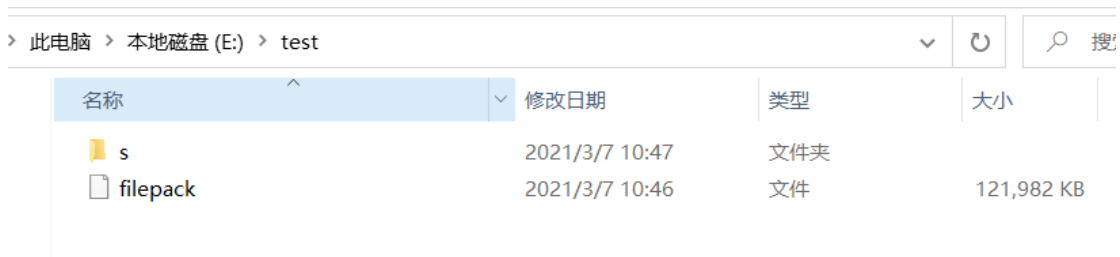
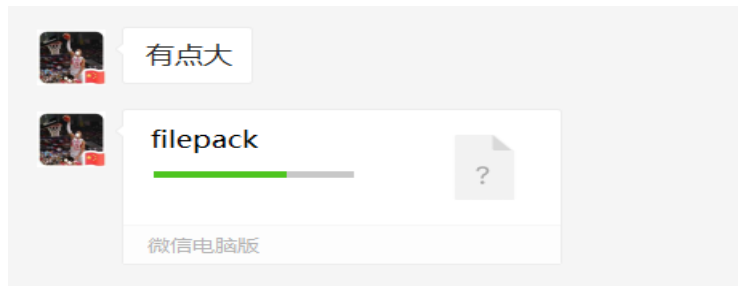
```
60     }
61     fclose(dfile);
62 }
63
64 int main() {
65     FILE* sfile = NULL;
66     char dic[FILE_NAME_LEN];
67     char name[FILE_NAME_LEN];
68     printf("输入目标文件夹: ");
69     scanf_s("%s", dic, FILE_NAME_LEN);
70     printf("输入要解包的文件: ");
71     scanf_s("%s", name, FILE_NAME_LEN);
72     if (fopen_s(&sfile, name, "rb") != NULL) {
73         printf("文件打开失败! \n");
74         getchar();
75         getchar();
76         return 0;
77     }
78     unpackFile(sfile, dic);
79     printf("解包结束! \n");
80     fclose(sfile);
81     getchar();
82     getchar();
83     return 0;
84 }
```



5、与同学互测并截屏运行结果：

把打包的文件发给同学，看他是否可以取出其中文件，同样测试是否可以读出并取出同学发来的打包文件。

截屏同学发来的文件和解包结果：





【完成情况】

是否完成以下步骤? (√完成 ×未做)

(1) [√] (2) [√] (3) [√] (4) [√] (5) [√]

互测同学的学号姓名: 19335201 王志文

【实验体会】

写出实验过程中遇到的问题, 解决方法和自己的思考; 并简述实验体会 (如果有的话)。

这次实验主要是利用 C 语言进行文件的一些操作。文件的操作我本来就不是很熟练, 再加上 C 语言太久没用了, 字符串函数也忘得一干二净, 所以刚开始做的时候有些吃力。前面两个实验还算容易, 借助老师的代码一下子就做了出来。后面两个有些困难, 但我在搜集一些资料学习后还是完成了。实验过程中, 我不小些把一个 `fopen_s` 函数的最后一个参数从 “rb” 写成了 “wb”, 导致把要读取的文件直接变成了 0kb, 浪费了我许多时间。总体来说, 这次实验让我收获颇丰, 希望以后也能再接再厉!

【交实验报告】

每位同学单独完成本实验内容并填写实验报告。

交作业地点: <http://172.18.187.251/netdisk/default.aspx?vm=19net>

\文件上交\编程实验\文件打包

截止日期: 2021 年 3 月 9 日 23:00 (周二)。

上传文件: 学号_姓名_文件打包.doc

学号_姓名_文件打包.rar (源程序和可执行程序)