程序设计分组训练 实验 3 实验报告

专　　业：计算机科学与技术

班　　级：_____

学生姓名：_____

学　　号：_____

北京交通大学计算机与信息技术学院

2021 年 10 月 26 日

目录

1.相关知识准备

1.1 判断文件夹存在与否——access 函数

确定文件或文件夹的访问权限。即，检查某个文件的存取方式，比如说是只读方式、只写方式等。如果指定的存取方式有效，则函数返回 0，否则函数返回 -1，即当参数 1 满足参数 2 条件时候返回 0，不满足返回-1；当判断是否是存在的文件时，使 mode 为 0；

```
int access(const char *pathname, int mode);
```

1.2 创建文件夹——mkdir 函数

函数名: mkdir

功 能: 建立一个目录

用 法: int mkdir(const char *dirname);

头文件库：direct.h

返回值：创建一个目录，若成功则返回 0，否则返回-1

1.3 C 语言文件路径的“\”“/”

“\”一般是表示本地目录的，比如你电脑里面的 C:\windows\..

“/”主要表示远程电脑或者网络上的路径。

windows 下文件路径是用\分隔的，比如 C:\Windows\System32\就是一个标准的 windows 路径。在 C 语言中，文件路径会被写作字符串形式，但是由于 C 语言的字符串规则中，\字符是转义字符的引导，所以直接写

"C:\Windows\System32\"会出错。而字符'\\'才是表示\。于是在写路径的时候，就必须是"C:\\Windows\\System32\\"

2.实验操作

2.1 思路

以 switch 语句分为命令行参数分别为 1、2、3 的情况进行分析解决。最关键的就是严格参照实验指导中的程序框图。

2.2 区分相对路径与绝对路径

如“知识准备”中的例子，利用‘/’‘\\’进行条件判断。

代码如下：

```
01:for(int i=len-1; i>=0; --i){
02:        if(pave[i]=='/'){
03:            for(int j=0; j<=i; ++j)
04:                filepave[j] = pave[j];
05:            flag =0;
06:            break;
07:        }
08:        else if(pave[i]=='\\'){
09:            for(int j=0; j<=0; ++j)
10:                filepave[j]= pave[j];
11:            flag =0;
12:            break;
13:        }
14:    }
15:
```

设置 FLIE*pf 以‘w’模式打开一个 txt 文件，遍历结构体写入文件中，然后关闭。随机数以 rand 函数生成。代码如下。

```
   01:void Fileopreate(int number, char txtaddress[]){
02:// 生成文件指针 以w模式打开
03:    FILE*pf = fopen(txtaddress, "w");
04:    int(*array)[3] = (int(*)[3])malloc(sizeof(int)*number*3);
05:
06:
07:    for(int i=0; i<number; i++ ){
08:                for(int j=0; j<3; j++){
09:                    array[i][j] =rand()%100;
10:                }
```

```
11:    }
12://空指针的操作
13:    if(pf==NULL){
14:        printf("open error\n");
15:        exit(0);
16:    }
17:    fprintf(pf, "%d\n", number);
18:    for(int i=0; i<number; i++){
19:        for(int j=0; j<3; j++){
20:            fprintf(pf,"%d, ", array[i][j]);
21:        }
22:        fprintf(pf, "\n");
23:    }
24:    if(fclose(pf)!=0){
25:        printf("close error\n");
26:    }
27:    //标记性语句，便于调试
28:    printf("opreate success!\n");
29:    free(array);
30:}
```

3.实验中问题回答

3.1 如何判断输入的数据条数是否合法

题目：a) 图 3-1 的流程中多处地方涉及到了判断输入参数是否合法的程序逻辑（红色字体部分），在本实验中由命令行输入的参数主要包括数据记录条数和记录文件名两个参数。当需要由用户输入数据记录条数时，实验 3 程序需提示用户可输入数字或者字母 r，当用户输入字母 r 时，记录条数由程序随机指定，但需要满足 CONF 结构体中的相关参数约束，当用户输入数字时，记录条数由用户输入的数值确定。请同学们思考程序是如何判断用户输入的数据记录条数是否合法的，请在实验报告中论述你的判断逻辑并将代码片段的截图粘贴在实验报告中；

以字符串的形式读入用户输入，遍历字符串。

先判断是否是‘r’，再进行遍历。检验字符在‘0’和‘9’与否，只要有一个不是判定为非法，最后得到合法。

代码如下：

```
01:int num_judge(char *num) {
02:    int i;
03:    //1 auto random number
04:    //2 normal input
05:    //3 error input
06:    if(strcmp(num, "r")==0 ) return 1;
07:    for(i=0; i<strlen(num); i++){
08:        if(num[i]>'9' || num[i] < '0')
09:            break;
10:    }
11:    if(i == strlen(num)) return 2;
12:    return 3;
13:}
```

3.2 生成数据记录文件函数的输入参数和返回值说明

题目：a) 图 3-1 中“生成数据记录文件函数”的输入参数和返回值应该是什么？请在实验报告中给出该函数的函数声明，并对函数的输入参数及返回值加以解释说明；

输入参数 CONF 结构体 （其中， 所需的参数都在结构体中）

无返回值

函数原型 void file_write(CONF conf)

3.3 生成指定范围随机整数的函数的封装

题目：c) 如何封装“获取一个指定范围内的随机整数的函数”（图 3-2 红色部分）， 它的输入参数和返回值应该是什么，请在实验报告中给出

你实现的该函数的声明截图，对函数输入参数和返回值加以说明；

以 random 函数为母体，把上下限封装在其中。上下限是保存在 CONF 中的默认参数。

```
int random_int(int min, int max) {
    return rand()%(max-min)+min;
}
```

3.4 不同代码功能的统计以及对鲁棒性的理解

题目：d) 请同学们统计你所编写代码的代码行数，尝试区分哪些代码是用于保障程序的健壮性，哪些代码是用来实现程序的 功能，比较两部分代码数量，看二者的比例是多少，在实验报告中结合以上数据分析说明你对程序健壮性的理解；

Run 函数代码总行数 163，其中对健壮性的保证在一半左右。子函数中，对于输入命令行参数的保证在一半以上，judge 类函数就有两个。
由此推断，保证程序健壮性的代码是程序的主要部分之一，这是对程序实现的保障，和程序功能的实现是同样重要的。两者并重的程序才是好程序。

3.5 结构体变量的作用

题目：e) 在程序中，配置信息结构体变量起到了什么作用，请同学们在实验报告中就你对这一问题的理解加以说明。

CONF 结构体起到相关数据的整合作用，保证了代码的可读性和变量名的良好管理。在 file_write 函数中跟保证了参数传递的高效性和便捷性。

3.6 可以被封装的函数

题目：f) 在流程图 3-1 中，有哪些流程可以封装为子函数，请同学们加以分析，并尝试将一些流程封装为函数，给出函数的声明并解释说明函数功能及输入输出参数。

代码展示：

```
01://判断输入参数的合法性
02:int num_judge(char *num) {
03:    int i;
04:    //1 auto random number
05:    //2 normal input
06:    //3 error input
07:    if(strcmp(num, "r")==0 ) return 1;
08:    for(i=0; i<strlen(num); i++){
09:        if(num[i]>'9' || num[i] < '0')
10:            break;
11:    }
12:    if(i == strlen(num)) return 2;
13:    return 3;
14:}
15:
16:int random_int(int min, int max) {
17:    return rand()%(max-min)+min;
18:}
19://对文件路径的合法性判断
20:int file_judge(char *name) {
21:    //0 normal
22:    //-1 error
23:    int len = strlen(name);
24:    char tem[5];
25:    char *tem1 =":,?;*&^%$#@!()?><";
26:
27:    if((name[len-4]=='.')&&(name[len-3]=='t')&&(name[len-2]=='x')&&(name[len-
   1]=='t')){
28:        for(int i=0; i<len; i++){
```

```
29:           for(int j=0; j< strlen(tem1); j++){
30:               if(name[i]-tem1[j] == 0)
31:                   return -1;
32:           }
33:       }
34:       return 0;
35:   }
36:   if((name[len-4]=='.')&&(name[len-3]=='d')&&(name[len-
37:2]=='a')&&(name[len-1]=='t')){
38:       for(int i=0; i<len; i++){
39:           for(int j=0; j< strlen(tem1); j++){
40:               if(name[i]-tem1[j] == 0)
41:                   return -1;
42:           }
43:       }
44:       return 0;
45:   }
46:   return -1;
47:}
48://文件写入
49:void file_write(CONF conf) {
50:   FILE *pf;
51:   strcat(conf.filesavepath, conf.filename);
52:   if (access(conf.filesavepath,0)!=0) //等于0，文件存在 不等于0，文件不存在
53:   {
54:       mkdir(conf.filesavepath);
55:   }//判断文件存在与否
56:   pf = fopen(conf.filesavepath, "w");
57:   fprintf(pf, "%d\n", conf.number );
58:   for(int i =0; i<conf.number; i++){
59:       for(int j=0; j<3; j++){
60:           fprintf(pf, "%d, ", random_int(conf.minvalue1, conf.maxvalue1));
61:       }
62:       fprintf(pf, "\n");
63:   }
64:   fclose(pf);
65:}
66:// 将文件路径分出
67:char *get_filepave(char * pave){
68:   int len = strlen(pave);
69:   char filepave[100];
70:   int flag=1;
71:   for(int i=len-1; i>=0; --i){
```

```
 72:         if(pave[i]=='/'){
 73:             for(int j=0; j<=i; ++j)
 74:                 filepave[j] = pave[j];
 75:             flag =0;
 76:             break;
 77:         }
 78:         else if(pave[i]=='\\'){
 79:             for(int j=0; j<=0; ++j)
 80:                 filepave[j]= pave[j];
 81:             flag =0;
 82:             break;
 83:         }
 84:     }
 85:     if(flag==1)
 86:         strcpy(filepave,"\\Lab3data");
 87:     return filepave;
 88:}
 89:// 分出文件名
 90:char *get_filename(char *name) {
 91:     int len =strlen(name);
 92:     char filename[100];
 93:     int flag=1;
 94:     for(int i = len-1; i>=0; --i){
 95:         if(name[i]=='/'){
 96:             for(int j=i+1; j<=len-1; ++j){
 97:                 filename[j-i-1] = name[j];
 98:             }
 99:             flag=0;
100:             break;
101:         }
102:         else if(name[i]=='\\'){
103:             for(int j= i+1; j<len -1; ++j){
104:                 filename[j-i-1] = name[j];
105:             }
106:             flag=0;
107:             break;
108:         }
109:     }
110:     if(flag==1)
111:         strcpy(filename, name);
112:     return filename;
113:
114:}
```

3.7 Outputdata 文件夹的路径设置

题目：在本实验中，结构体 CONF 的 filesavepath 分量用于存储生成的数据记录文件的存放位置，请学习绝对路径、相对 路径的相关知识，将默认情况下生成的数据记录文件存放在“与实验 3 应用程序同级的 OutputData 目录下”，请在实验报 告中加以论述 filesavepath 的初始值应该如何设置。请同学们思考以下问题：

要求默认文件生成于 lab_3.exe 同级的文件夹中，则使用相对路径设置为“Outputdata\\”

3.8 用户输入的路径中有未创建的文件夹时的运行情况

题目：a) 当用户输入文件名参数时输入的相对路径或绝对路径中包含有未创建的文件夹，程序是否能够正常运行？请在实验报告中通过对程序的实际测试加以说明， 并阐述如果想解决这一问题，你的程序改进思路是什么。

当用户的输入路径中的文件夹有未创建文件夹时，分两种情况讨论。未创建文件夹为最后级时，可以用 mkdir 函数创建原本不存在的文件夹。而未创建文件夹不是最后一级时，程序无法运行。推荐循环判断文件夹是否存在，不存在则调用 mkdir 创建相应文件夹。

3.9 输入参数要求变化时合法性检验的变化

题目：

b) 如果程序强制要求文件名参数只能允许用户输入文件名参数，不能

输入任何含有路径的文件名，那么在对文件名参数合法性校验的程序逻辑部分，需要做怎样的改动？请在实验报告中加以说明。

此时可以把路径中的‘\\’‘/’也加入到合法性判断中，只要含有上述两个字符则不合法。

4.心得体会

4.1 自身收获

本次实验中，最磨人的莫过于文件路径和文件名的分离，一个从头开始截取，一个则从尾部开始截取。File_write 函数中，在相对路径中创建 txt 文件的也废了一会功夫。一开始，总是把 txt 文件新建为文件夹，最后把 filepath 和 filename 合一的顺序改在 mkdir 函数之后问题就决解了。

这次的实验，本身实现部分并不难，但保证输入合法性的部分实在太磨人了，这样让我对企业级程序有了更深的认识。

4.2 小组互评

这次临近期中，小组成员其他杂事都比较多，导致实验进度缓慢。由此，我们应借助小组合作机制建立相互督促机制。

张开元同学的报告最后还是完成的比较好的，他的代码清晰，构成逻辑严密。因为是相互交流比较多，所以不借助注释也可以很快看明白。这恰恰说明小组合作的重要性。

王麒越同学的报告完成的很好，只是在变量命名上还不能做到见名知义，还需要一起多努力。