C语言之文件操作集合

2019年03月20日 17:43:24 小龙魂 阅读数：51

文件打开

```
FILE * open(const char *filename, const char *mode)
```

第一个参数：文件名

第二个参数：

文件读写模式	含义
“r"或者"rt”	只读
“rb”	只读二进制文件
“w"或者"wt”	只写打开或者建立一个文件
“wb”	只写打开或者建立二进制文件
"a"或者“at”	追加打开文本文件，并在文件末尾添加数据
“ab”	追加打开二进制文件，并在文件末尾添加数据
"r+“或"rt+”	读写打开文本文件，允许读写
“rb+”	读写打开二进制文件，允许读写
"w+“或者"wt+”	读写打开或建立文本文件，允许读写
“wb+”	读写打开或建立二进制文件，允许读写
"a+“或者"at+”	读写打开或建立文本文件，允许读写，或在文件末尾添加数据
“ab+”	读写打开或建立二进制文件，允许读写，或在文件末尾添加数据

实例：

```
FILE *fp
fp = open("1.txt", "r");
```

文件关闭

```
int fclose(FILE * filename); // 成功返回0， 错误返回EOF(-1)
```

文件检测

```
feof(fp);  // 如果到文件末尾，返回非0值； 否则返回0
```

实例

```
while(!feof(fp)) {
    // 文件读写
}
```

文件文件出错函数

```
ferror(fp); // 检测文件输入/输出调用是否错误，正常返回0， 错误返回非0值
```

文件读写操作

- 字符读写函数：fgetc() / fputc()
- 字符串读写函数： fgets() / fputs()
- 格式化读写函数： fscanf() / fprintf()
- 数据块读写函数： fread() / fwrite()

字符读写函数

```
#include<stdio.h>
int main () {
    char tmp;
    FILE *d, *w;
    d = fopen("1.txt", "r");
    w = fopen("2.txt", "w+");
    if(d == NULL && w == NULL) {
        printf("Open file failed,can't go on\n");
        return;
    }
    tmp=fgetc(d);
    while(EOF != tmp) {
        printf("%c", tmp);
        fputc(tmp, w);
        tmp = fgetc(d);
    }
    fclose(d);
    fclose(w);
}
```

字符串读写函数

```
char * fgets(char * str, int n, FILE * fpIn);
int fputs(const char * str, FILE *fpOut);
```

```
#include<stdio.h>
int main() {
    char *tmp = NULL;
    char buf[100];
    FILE *w = NULL, *d = NULL;
    d = fopen("2.txt", "r");
    w = fopen("3.txt", "w+");
    if (d == NULL && w == NULL) {
        printf("打开文件失败! ");
        return;
    }
    while(!feof(d)) {
        tmp = fgets(buf, 20, d);
        printf("%s\n", tmp);
        fputs(tmp, w);
    }
    fclose(w);
    fclose(d);
    return 0;
}
```

格式化读写函数

```
fscanf（可以从一个文件流中格式化读出数据，遇到空格或回车就停止）
原型： int fscanf(FILE *stream, const char *format, ...); //fscanf(文件流指针,格式字符串,输出表列);
参数：    FILE *stream ：文件流指针
        const char *format, ... ：字符串的格式
例子 :
fscanf(fp,"%s %s %d",new1->number,new1->staddress,&new1->price);
（这样写的话数据输入到文件中时每个数据中间就会有一个空格）
或者：
fscanf(fp,"%s,%s,%d",new1->number,new1->staddress,&new1->price);
（这样写的话数据输入到文件中时每个数据中间就会有一个‘,’）
```

```
----------------------------------------------------------------
fprintf（可以向一个文件中格式化写入数据）
原型： int fprintf(FILE *stream, const char *format, ...); //fprintf(文件流指针,格式字符串,输出表列);
参数：   FILE *stream ：文件流指针
        const char *format, ... ：字符串的格式
例子 ：
fprintf(fp,"%s %s %d\n",new->number,new->staddress,new->price);
或者：
fprintf(fp,"%s,%s,%d\n",new->number,new->staddress,new->price);
```

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

#define LENGTH 100 //数组的大小

typedef struct
{
    char number[LENGTH];
    char staddress[LENGTH];
    int price;
}*node, Node;

int main(int argc,char *argv[])
{
    FILE *fp;
    fp = fopen("test.txt","a+"); //以读写的权限打开文件（如果文件不存在则创建）
    if(fp == NULL)
    {
        perror("fopen");
        exit(1);
    }
    node new,new1; //new用来存放写入到文件中的数据，new1用来存放从文件中读出的数据

    //为两个结构体指针分配空间
    new = (node)malloc(sizeof(Node));
    new1 = (node)malloc(sizeof(Node));

    //清空
    memset(new,0,sizeof(node));
    memset(new1,0,sizeof(node));

    strcpy(new->number,"20170816");
    strcpy(new->staddress,"南宁");
    new->price = 100;
    fprintf(fp,"%s %s %d\n",new->number,new->staddress,new->price); //格式化写入数据到文件中
    fseek(fp, 0,  SEEK_SET); //文件指针重置，因为上面把数据写入文件的时候已经把文件流指针定位到文件尾了，所以要重新定位到文件头

    fscanf(fp,"%s %s %d",new1->number,new1->staddress,&new1->price);//格式化从文件中读出数据
    printf("%s %s %d\n",new1->number,new1->staddress,new1->price);
    //释放两个结构体指针
    free(new);
    free(new1);
    fclose(fp); //关闭文件
    return 0;
}
```

数据块读写函数

```
size_t fread ( void * ptr, size_t size, size_t count, FILE * stream );
其中，ptr：指向保存结果的指针；size：每个数据类型的大小；count：数据的个数；stream：文件指针
函数返回读取数据的个数。

size_t fwrite ( const void * ptr, size_t size, size_t count, FILE * stream );
其中，ptr：指向保存数据的指针；size：每个数据类型的大小；count：数据的个数；stream：文件指针
函数返回写入数据的个数。
```

写数据

```c
#include <stdio.h>
#include <string.h>
#include <stdlib.h>
typedef struct{
    int age;
    char name[30];
}people;

int main ()
{
    FILE * pFile;
    int i;
    people per[3];
    per[0].age=20;strcpy(per[0].name,"li");
    per[1].age=18;strcpy(per[1].name,"wang");
    per[2].age=21;strcpy(per[2].name,"zhang");

    if((pFile = fopen ("5.txt", "wb"))==NULL)
    {
        printf("cant open the file");
        exit(0);
    }

    for(i=0;i<3;i++)
    {
        if(fwrite(&per[i],sizeof(people),1,pFile)!=1)
            printf("file write error\n");
    }
    fclose(pFile);
    return 0;
}
```

读数据

```c
#include <stdio.h>
#include <string.h>
#include <stdlib.h>
typedef struct{
    int age;
    char name[30];
}people;

int main () {
    FILE * fp;
    people per;
    if((fp=fopen("5.txt","rb"))==NULL)
    {
      printf("cant open the file");
      exit(0);
    }

    while(fread(&per,sizeof(people),1,fp)==1)    //如果读到数据，就显示；否则退出
    {
        printf("%d %s\n",per.age,per.name);
    }
    return 0;
}
```

文件随机读写

fseek() 用来移动文件流的读写位置.

```
int fseek(FILE * stream, long offset, int whence);
第一个参数file指针
第二个参数移动的偏移量
第三个参数移动到哪里
fseek(fp,100L,SEEK_SET);把fp指针移动到离文件开头100字节处；
fseek(fp,100L,SEEK_CUR);把fp指针移动到离文件当前位置100字节处；
fseek(fp,100L,SEEK_END);把fp指针退回到离文件结尾100字节处。
```

起始点	表示符号	数字
文字首	SEEK-SET	0
当前位置	SEEK-CUR	1
文件末尾	SEEK-END	2

ftell() 函数用来获取文件读写指针的当前位置

```
long ftell(FILE * stream);
【参数】stream 为已打开的文件指针。
【返回值】成功则返回当前的读写位置，失败返回 -1。
```

```
#include <stdio.h>
#include <stdlib.h>
#define SLEN 81
int main(void)
{
    char file[SLEN];
    char ch;
    FILE *fp;
    long count, last;
    if ((fp = fopen("3.txt","rb")) == NULL)
    {
        printf("reverse can't open %s\n", file);
        return ;
    }

    fseek(fp, 0L, SEEK_END);
    last = ftell(fp);
    for (count = 1L; count <= last; count++)
    {
        fseek(fp, -count, SEEK_END);
        ch = getc(fp);
        printf("%c", ch);
    }
    fclose(fp);
    return 0;
}
```

rewind()

rewind() 用来将位置指针移动到文件开头

```
void rewind ( FILE *fp );
```

```
// 从键盘输入一行字符，追加写入到一个文件中，再把该文件内容读出显示在屏幕上。
#include<stdio.h>
int main()
{
    FILE  *fp;
    char  ch;
    if((fp=fopen("3.txt","ab+"))==NULL)
    {
        printf("\nCannot open file\nstrike any key exit\n");
        getchar();
        return 1;
    }
    printf("input a string:\n");
    ch=getchar();
    while(ch!='\n')
    {
        fputc(ch,fp);
        ch=getchar();
    }
    rewind(fp);
    ch=fgetc(fp);
```

```
    while(ch!=EOF)
    {
        putchar(ch);
        ch=fgetc(fp);
    }
    fclose(fp);
    return 0;
}
```

想对作者说点什么