

C语言程序设计

文件



程序运行的思考



存在问题：

- 1 运行结果一闪而过，怎样长久保存？
- 2 已有数据能否直接读取，不用再重新输入？

使用文件



文件的概念

➤ **文件** 存储在外部介质上数据的集合,是操作系统数据管理的单位。

➤ **文件无处不在**

C语言源程序文件（.c或.cpp）、

执行文件（.exe）、图片文件（.jpg）

➤ **文件的作用**

数据文件的改动不引起程序的改动——**程序与数据分离**

不同程序可以访问同一数据文件中的数据——**数据共享**

能**长期保存**程序运行的中间数据或结果



文件的分类

➤ 按文件的逻辑结构

记录文件：由具有一定结构的记录组成（定长和不定长）

流式文件：由一个个字符（字节）数据顺序组成

➤ 按数据的组织形式

文本文件：ASCII文件，每个字节存放一个字符的ASCII码

二进制文件：数据按其在内存中的存储形式原样存放



文件的操作流程

- 打开文件
- 读取或写入

读取：将某个文件中的数据输入到程序中

写入：把程序的运行内容输出到某个文件中

- 关闭文件



01

文件的打开与关闭



文件指针

每个被使用的文件都在内存中开辟一个区，用来存放文件的有关信息（文件名、文件状态、文件的当前位置等），可以用一个文件指针指着。

```
typedef struct
{
    short    level;        /* 缓冲区“满”或“空”的程度 */
    unsigned flags;        /* 文件状态标志 */
    char     fd;           /* 文件描述符 */
    unsigned char hold;    /* 如无缓冲区不读取字符 */
    short    bsize;        /* 缓冲区大小 */
    unsigned char *buffer; /* 缓冲区的位置 */
    unsigned char *curp;   /* 指针，当前的指向 */
    unsigned  istemp;      /* 临时文件，指示器 */
    short     token;       /* 用于有效性检查 */
} FILE;
```

FILE *fp

FILE *fr

FILE *fw



文件的打开与关闭

➤ 文件打开函数 fopen ()

语法: FILE *fp;

fp=fopen("文件名", "操作方式");

“文件名”是指要打开（或创建）的文件名。

如果使用字符数组（或字符指针），
则不使用双引号,直接写数组名或字符指针名。

➤ 注意

使用文件函数必须 #include "stdio.h"

对文件进行读写之前，必须先打开该文件；



文件的打开方式

文件使用方式	含义
“r/rb” (只读)	为 输入 打开一个文本/二进制文件
“w/wb” (只写)	为 输出 打开或建立一个文本/二进制文件
“a/ab” (追加)	向文本/二进制文件尾 追加 数据
“r+/rb+” (读写)	为读/写打开一个文本/二进制文件
“w+/wb+” (读写)	为读/写建立一个文本/二进制文件
“a+/ab+” (读写)	为读/写打开或建立一个文本/二进制文件

➤ 注意：r+ 与 w+ 的区别

r+：若文件不存在，r+方式无法打开文件

w+：若文件不存在，w+方式会创建一个文件
若文件已存在，w+方式会清空原有文件内容



打开文件

➤ 例：若要打开磁盘中的out.txt文件，并写入数据

根据语法： `FILE *fp;`

`fp=fopen("文件名", "操作方式");`

程序代码为： `FILE *fp;`

`fp=fopen("out.txt", "w+");`



文件的关闭

➤ 文件关闭函数 `fclose()`

语法: `fclose(fp);`

代码实现

```
FILE *fp;
```

```
fp=fopen("out.txt","w+");
```

```
.....
```

```
fclose(fp); /*关闭fp所指向的文件*/
```

注意:

文件结束使用后, 应立即关闭, 以免数据丢失。



02

文件的读写



文件的读写

读/写1个字符（或字节）数据时：

- 选用fgetc()和fputc()函数

读/写1个字符串时：

- 选用fgets()和fputs()函数

读/写1个（或多个）不含格式的数据时：

- 选用fread()和fwrite()函数

读/写1个（或多个）含格式的数据时：

- 选用fscanf()和fprintf()函数



fgetc()与fputc()函数

fgetc()函数：从文件中读一个字符

用法： fgetc(文件指针);

使用 ch=fgetc(fp);

从fp所指向的文件中，读取一个字符，给内存变量ch

注意：对二进制文件执行读入操作时，必须使用库函数feof()来判断是否遇到文件尾。

如果遇到文件尾，feof返回1；否则，则返回0

fputc()函数：将一个字符写到文件中

用法： fputc(字符数据，文件指针);

使用 fputc(ch,fp);

将字符变量ch的值 输出到fp所指向的文件中去



feof()判断文件结束函数

对二进制文件执行读入操作时，必须使用库函数feof()来判断是否遇到文件尾。如果遇到文件尾，feof()函数返回值 **1**；否则，则返回值 **0**。

用法

```
while(!feof(fp))  
    {c=fgetc(fp);  
    ...    }
```

对文本文件

```
while((ch=fgetc(fp))!=EOF)    注意： EOF是文件结束标志  
    putchar(ch);
```



fgets()和fputs()函数

fgets()——从文件中读一个字符串

用法： fgets(指针, 串长度+1, 文件指针);

功能： 从指定文件中读入一个字符串，存入“字符数组”中，并在尾端自动加一个结束标志'\0'；同时，将读写位置指针向前移动（字符串长度）+1个字节。

如果在读入规定长度之前遇到文件尾EOF或换行符，读入即结束。

fputs()——向指定文件输出一个字符串

用法： fputs(字符串, 文件指针);

“字符串” 可以是一个字符串常量，或字符数组名，或字符指针变量名

功能： 向指定文件输出一个字符串，同时将读写位置指针向前移动（字符串长度）个字节。

如果输出成功，则函数返回值为0；否则，为非0值



fread() 和fwrite()函数

用法： `fread(buffer, size, count, fp);`
 `fwrite(buffer, size, count, fp);`

功能：

`fread()`——从fp所指向文件的当前位置开始，一次读入size个字节，重复count次，并将读入的数据存放到从buffer开始的内存中；同时，将读写位置指针向前移动size* count个字节。buffer是存放读入数据的起始地址（即存放何处）

`fwrite()`——从buffer开始，一次输出size个字节，重复count次，并将输出的数据存放到fp所指向的文件中；同时，将读写位置指针向前移动size* count个字节。其中，buffer是要输出数据在内存中的起始地址（即从何处开始输出）。



fscanf()和fprintf()函数

用法:

fscanf(文件指针, "格式符", 输入变量首地址表);

fprintf(文件指针, "格式符", 输出参量表);

例:

```
int i=100; float f=8.80;  
.....  
fprintf(fp,"%3d,%6.2f", i, f);
```

表示将变量i按%3d格式、变量f按%6.2f格式，
以逗号作分隔符，输出到fp所指向的文件中：
100,□□8.80（□表示1个空格）



文件读写的应用-冒泡排序

```
#include <stdio.h>
#define NUM 10
void main ( )
{   int a[NUM], i, j, t;
    printf ("input %d numbers: \n", NUM);
    for (i = 0; i < NUM; i++) //输入NUM个整数
        scanf ("%d", &a[i]);
    for (i = 1; i < NUM; i++) //趟数，共NUM-1趟
        for (j = 0; j < NUM - i; j++) //实现一次冒泡操作
            if (a[j] > a[j+1]) //交换a[j]和a[j+1]
            {   t = a[j];
                a[j] = a[j+1];
                a[j+1] = t;    }
    printf ("the sorted numbers:\n");
    for (i = 0; i < NUM; i++) //输出排好序的数据
        printf ("%d ", a[i]); }
```

上机调试

遇到的问题

- 调试一次，输入10个数据，很烦
- 结果一闪而过，没有保存



读写部分修改

➤ 从文件中读入数据

```
fp=fopen("shuju.txt","r");  
for(i=0;i<NUM;i++)  
fscanf(fp,"%d",&a[i]);  
fclose(fp);
```

➤ 将结果写入文件

```
fp=fopen("out.txt","w+");  
for(i=0;i<NUM;i++)  
fprintf(fp,"%4d",a[i]);  
fclose(fp);
```

用VC6.0调试



03

其他文件函数



文件的定位

文件中有一个读写位置指针，指向当前的读写位置。

每次读写1个（或1组）数据后，系统自动将位置指针移动到下一个读写位置上。

如果想改变系统这种读写规律，可使用有关文件定位的函数。

- 对于流式文件，既可以顺序读写，也可随机读写，关键在于控制文件的位置指针。
- 顺序读写是指，读写完当前数据后，系统自动将文件的位置指针移动到下一个读写位置上。
- 随机读写是指，读写完当前数据后，可通过调用 `fseek()` 函数，将位置指针移动到文件中任何一个地方。



rewind()

位置指针复位函数rewind()

- 用法: rewind(文件指针);
 - 功能: 使文件的位置指针返回到文件头
- 使用: rewind(fp);

例 对一个磁盘文件进行显示和复制操作

```
#include <stdio.h>
main()
{ FILE *fp1,*fp2;
  fp1=fopen("f.txt","r");
  fp2=fopen("fx.txt","w");
  while(!feof(fp1))
    putchar(getc(fp1));
  rewind(fp1);
  while(!feof(fp1))
    putc(getc(fp1),fp2);
  fclose(fp1);
  fclose(fp2);
}
```



fseek()

- **用法：** `fseek(文件指针, 位移量, 参照点);`
- **功能：** 将指定文件的位置指针，从参照点开始，移动指定的字节数。
- **说明：**

(1)参照点：0（文件头）、1（当前位置）和 2（文件尾）

在ANSI C标准中，还规定：

SEEK_SET——文件头

SEEK_CUR——当前位置

SEEK_END——文件尾

```
fseek(fp,100L,0);
```

```
fseek(fp,50L,1);
```

```
fseek(fp,-10L,SEEK_END);
```

(2)位移量：以参照点为起点，

向前（当位移量 >0 时）或后（当位移量 <0 时）移动的字节数。



fseek()

例：

```
#include <stdio.h>
main( )
{ FILE *fp; int i,a[4]={1,2,3,4},b;
  fp=fopen("data.dat","wb");
  for(i=0;i<4;i++) fwrite(&a[i],sizeof(int),1,fp);
  fclose(fp);
  fp=fopen("data.dat","rb");
  fseek(fp,-2L*sizeof(int).SEEK_END); //使位置指针从文件尾向前移2*sizeof(int)字节)
  fread(&b,sizeof(int),1,fp);         //从文件中读取sizeof(int)字节的数据到变量b中
  fclose(fp);
  printf("%d\n",b);
}
```

执行后输出结果是

A) 2

B) 1

C) 4

☒ D) 3



ftell()

返回文件当前位置的函数ftell()

由于文件的位置指针可以任意移动，也经常移动，往往容易迷失当前位置，ftell()就可以解决这个问题。

➤ **函数原型：** long ftell(FILE *stream)

➤ **用法：** ftell(文件指针)

➤ **功能：** 返回文件位置指针的当前位置（用相对于文件头的位移量表示）。

如果返回值为-1L，则表明调用出错。

例如：

```
offset=ftell(fp);
```

```
if(offset==-1L)printf("ftell() error\n");
```



出错的检测函数ferror()

在调用输入输出库函数时，如果出错，除了函数返回值有所反映外，也可利用ferror()函数来检测。

➤ **用法：** ferror(文件指针);

➤ **功能：** 如果函数返回值为0，表示未出错；如果返回一个非0值，表示出错。

➤ **说明：**

(1)对同一文件，每次调用输入输出函数均产生一个新的ferror()函数值。

因此在调用了输入输出函数后，应立即检测，否则出错信息会丢失。

(2)在执行fopen()函数时，系统将ferror()的值自动置为0。

例： ferror(fp);



clearerr()

- **函数原型：** void clearerr(FILE *stream);
- **用法：** clearerr(文件指针);
- **功能：** 将文件错误标志（即ferror()函数的值）
文件结束标志（即feof()函数的值）置为0



THANKYOU

