第十一章 文件系统

郎大鹏

第十一章 文件系统

11.1　文件概述

11.2　文件的打开与关闭

11.3　文件的读写

11.4　编程举例

11.1 文件概述

- 广义的“文件”
  公文书信或指有关政策、理论等方面的文章。
- 狭义的“文件”
  是指档案，范畴很广泛，电脑上运行的程序、杀毒等等等等都叫文件。
- 文件可以是一篇文章、一幅图像、一段声音、一个程序等等。文件通常具有三个字母的文件扩展名，用于指示文件类型（例如，文档文件的扩展名为 .doc）。
- 分类：
  数据文件：文件中存放的是数据。
  程序文件：文件中存放的是源程序清单或者是编译连接后生成的可执行程序。

11.1　文件概述

1. 文件名

每个文件被分配一个一个标识，这样能够区分不同的文件。能够唯一标识某个文件的就是文件名。

- 磁盘文件的文件标识组成如下：

  盘符：路径\文件主名.扩展名。

  其中：盘符表示文件所在的磁盘，可以是A、B、C等；路径用来表示文件所在的目录，是由目录组成的，目录间用“\”符号分隔；‘盘符”和“路径”都可以省略。
- 例如，“c:\tc\file.c”是一个完整的磁盘文件的文件标识，该文件标识中的文件所在路径为：” c:\tc\”，文件主名为“file”，扩展名为“.c”。

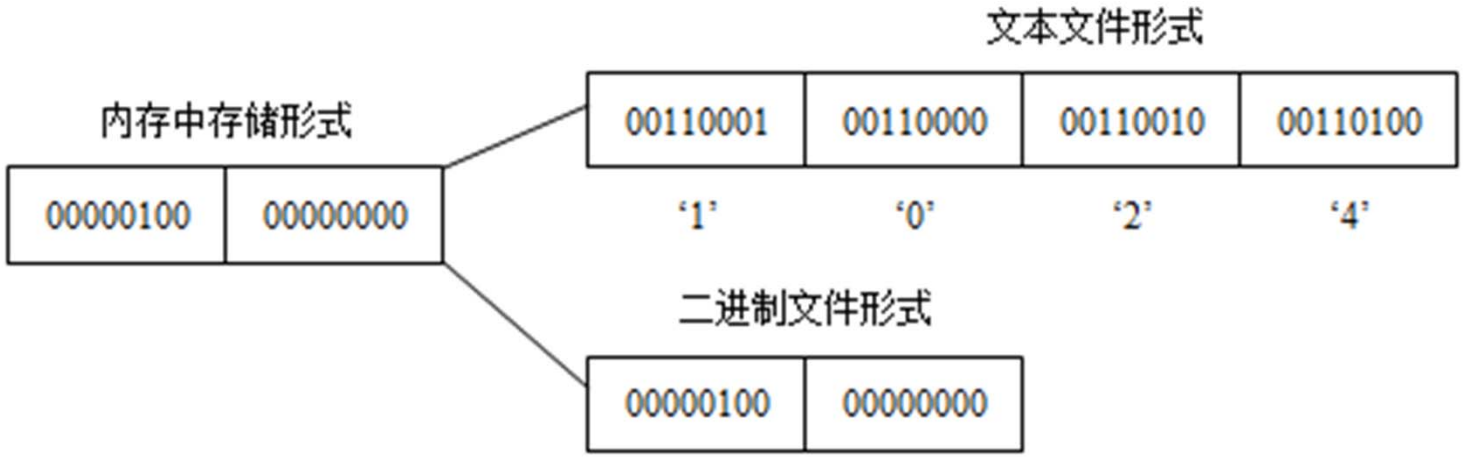
11.1　文件概述

- 2. 文件分类
- （1）按文件用途分类
- 系统文件：存放操作系统主要文件的文件夹。包括操作系统内核、编译程序文件等。系统文件通常都是可执行的二进制文件，直接影响系统的正常运行，只允许用户使用，不允许用户改变。 它对维护计算机系统的稳定具有重要作用。
- 用户文件：用户自己定义的文件，如用户的源程序、可执行程序和文档等。

- （2）按文件数据编码方式分类
- 文本文件在磁盘中存放时每个字符对应一个字节，用于存放对应的ASCII码，因此文本文件也称为ASCII文件。

11.1 文件概述

- 文本文件和二进制文件的区别主要体现在对数值型数据的处理上。
- 要存储一个整型数据1024，按二进制文件的格式存储，在文件中存放的是1024的二进制形式1000000000，占2个字节的存储空间（一个整型数占2个字节）；而以文本文件的格式存储，则在文件中保存的是’1’、’0’、’2’、’4’四个字符的ASCII码，占用4个字节。



11.1　文件概述

- （3）按文件存取方式分类
  按文件的存取方式，可以把文件分为“顺序文件”和“随机文件”。
- 顺序文件的信息是按照顺序排列的，而且只提供第一条记录的存储位置，因此访问每一个数据只能从头开始访问，直到访问的数据是要处理的数据为止。
- 随机文件，既可以从头到尾顺序访问每一个数据，也可以随机访问其中的任一个数据。

- （4）按文件数据的形式分类
- 源文件：由源代码和数据构成的文件。
- 目标文件：源程序经过编译程序编译，但尚未链接成可执行代码的目标代码文件。
- 可执行文件：编译后的目标代码由链接程序连接后形成的可以运行的文件。

11.1 文件概述

- 3. 文件型指针
- 系统在内存中开辟一个缓冲区，用来存放正在运行的文件相关的信息，如文件名、文件状态等，这些信息保存在一个FILE类型的结构体变量之中，以后对文件的操作都可通过这个FILE类型的结构体变量进行。
- FILE类型不需要用户自己定义，它是由系统事先定义的，固定包含在C语言的标准输入输出头文件stdio.h中。

```
typedef struct
{  int _fd; /* 文件位置指针，即当前文件的读写位置 */
  int _cleft; /* 文件缓冲区中剩余的字节数 */
  int _mode; /* 文件操作模式 */
  char *nextc; /* 用于文件读写的下一个字符位置 */
  char *_buff; /* 文件缓冲区位置（指针） */
} FILE;
```

11.2　文件的打开和关闭

- 文件的打开是指从磁盘文件中读取数据到内存。由于程序只能处理内存中的数据，因此必须把存放在磁盘上的数据读取到内存。因此，C语言规定文件必须先打开，后使用。

- 文件的关闭是指内存中的数据存回到磁盘文件。修改文件中的数据后，还需要将内存中的数据保存到磁盘上，才能保证文件中的数据被修改。因此，C语言规定文件使用完后必须将其关闭。

11.2　文件的打开和关闭

使用文件的一般步骤是：

- 打开文件--操作文件--关闭文件
- 操作文件：是指对文件的读、写、追加和定位操作。
- 读操作：从文件中读出数据，即将文件中的数据读入计算机内存。
- 写操作：向文件中写入数据，即将计算机内存中的数据向文件输出数据。
- 追加操作：将新的数据写到原有数据的后面。
- 定位操作：移动文件读写位置指针。

C语言中提供了三个标准设备文件的指针，它们是：

- stdin 标准输入文件（键盘）
- stdout 标准输出文件（显示器）
- stderr 标准错误输出文件（显示器）

11.2.1　文件的打开

函数原型 FILE * fopen(char *filename, char *mode)

参数　　filename　　字符串，要打开的文件的文件名；

mode　　字符串，打开文件的具体方式。mode的含义如表11.1所示。

功能　　按字符串mode的“打开方式”，打开字符串filename指定的文件，同时自动给该文件分配一个内存缓冲区。

返回值　　能正确打开指定的文件，则返回一个指向filename 的地址，该地址为打开文件所分配缓冲区的首地址。如果打开文件出现错误，返回值为“NULL”（NULL在stdio.h文件中已被定义为0），表示打开文件出错。

11.2.1　文件的打开

```
FILE *fp;                                                    /*定义文件型指针*/
char filename[16];
……
printf("Input a filename:");
scanf("%s",filename);
if ((fp=fopen(filename,"r"))==NULL)                          /*只读方式打开一个文件*/
{    printf("\n Can not open this file.");                   /*打开文件出错的提示*/
     exit(0);                                               /*关闭文件，中止程序运行*/
}
……                                                          /*文件正确打开，可对文件操作*/
```

由系统打开的三个标准文件 stdin、stdout 和 stderr，在使用的时候不需要调用 fopen 函数打开，可以直接使用它们的文件指针进行操作。

11.2.2　文件的关闭

函数原型　int fclose(FILE *fp)

参数　　fp 文件型指针，指向某个通过 fopen()函数打开的文件。

功能　　关闭 fp 所指向的文件。

返回值　能正确关闭指定的文件，则返回 0；否则返回非 0。

例如：fclose(fp);

11.3.1　文件尾测试函数

读取文件中数据时，需要判断是否到达文件尾。系统提供的文件尾测试函数可以帮助用户判断是否到达文件尾。

函数原型 int feof(FILE *fp)

参数　　fp 文件型指针。

功能　　测试 fp 所指向的文件是否到达文件尾。

返回值　若当前是文件尾，返回非 0，否则返回 0。

在对文件中数据进行读操作时，都要事先利用该函数来判断。

11.3.2　文件的字符读/写函数

1．读字符函数

函数原型　int fgetc(FILE *fp)

参数　fp 文件型指针。

功能　从fp所指向的文件当前位置读取单个字符。

返回值　读字符成功，返回读取的单个字符；否则返回EOF 。

2．写字符函数

函数原型　int fputc(char ch, FILE *fp)

参数　ch　写到文件中的字符，可以是字符常量、字符变量等。

fp　文件型指针。

功能　将ch中的字符写到fp所指向的文件的当前位置。

返回值　成功，则返回刚写到文件中的字符；否则返回EOF。

11.3.3　文件的字符串读/写函数

1. 读字符串函数

函数原型　char *fgets(char *str, int num, FILE *fp)

参数　　str　　字符型指针。

　　num　　整型。

　　fp　　文件型指针。

功能　　从fp所指向的文件当前位置读取n-1个字符。

返回值　成功，返回str对应的地址；否则返回NULL（其值为0）。

2. 写字符串函数

函数原型　int *fputs(char *str, FILE *fp)

参数　　str　字符型指针。

　　fp　　文件型指针。

功能　　将str指向的一个字符串，舍去结束标记’\0’后写入fp所指向的文件中。

返回值　成功，返回写入文件的实际字符数；否则返回EOF。

11.3.4　文件的数据块读写函数

1．读数据块函数

函数原型 int fread(char *buffer, unsigned size, unsigned count, FILE *fp)

参数　buffer　字符型指针，读入数据的存储首地址。

size　读取的数据所占用的字节总数。

Count　读取的数据（注意是size个字节）的个数。

fp　文件型指针。

功能　从fp所指向的文件的当前位置读取count次数据，每次数据的字节数为size，存入buf指定的内存区。

返回值　成功，返回count值；否则返回NULL(其值为0)。

11.3.4　文件的数据块读写函数

2. 写数据块函数

函数原型 int fwrite(char *buf, unsigned size, unsigned count, FILE *fp)

参数　　buf　字符型指针，输出数据的首地址。

size　代表写入文件的每个数据所占用的字节总数。

Count　代表写入文件的数据的个数（注意每个数据是size个字节）。

fp　文件型指针，通过fopen()函数获得的、已指向打开的可写文件。

功能　从buf为首地址的内存中取出count个数据（每个数据的字节数为size）写入fp指向的文件。

返回值　成功，返回count的值；否则，返回NULL(其值为0)。

11.3.5　文件的格式读写函数

1．格式读数据函数

函数原型 int fscanf(FILE *fp,格式控制,输入列表)

参数　　fp　　文件型指针。

　　格式控制　　常量字符串。

　　输入列表　　指向变量的指针变量。

功能　　从fp指向的文件中，按照format指定的格式，读取n个数据依次存入输入列表中

返回值　　成功，返回读取数据的数目；否则，返回EOF。

11.3.5　文件的格式读写函数

2．格式写函数

函数原型　int fprintf(FILE *fp,格式控制,输出列表)

参数　　fp　　文件型指针，通过fopen()函数获得的、已指向打开的可写文件。

　　格式控制　常量字符串。

　　输出列表

功能　　将输出列表中的各个变量或常量，依次按格式控制符说明的格式写入fp指向的文件。该函数调用的返回值是实际输出的字符

返回值　成功，返回写入文件的表达式数目；否则，返回EOF。

11.4　编程举例

1．编写一个统计文本文件中字符个数。

```
#include "stdio.h"
void main()
{
        FILE *fp;
        int n=0;
        char ch;
        if((fp=fopen("f:\\string.txt","r"))==NULL)
        /* 设文本文件为string.txt */
        {printf("cannot open infile\n");
   exit(0);}
        do{ch=fgetc(fp);
        n++;
        }while(!feof(fp));
        printf("the num of char = %d\n",n);
        fclose(fp);
}
```

11.4　编程举例

2．编写一个统计文本文件中行数的程序。

```
#include "stdio.h"
void main()
{
        FILE *fp;
        int n=0;
        char ch;
        if((fp=fopen("f:\\char.txt","r"))==NULL)
                        /* 设文本文件为char.txt */
        {printf("cannot open infile\n");
   exit(0);}
        do{ch=fgetc(fp);
        if(ch=='\n') n++;
        }while(!feof(fp));
        printf("the num of rows = %d\n",n);
        fclose(fp);
}
```