# 文　件

# 10.1　C语言文件概述

在程序运行时，程序本身和数据一般都存放在内存中。当程序运行结束后，存放在内存中的数据被释放。

如果需要长期保存程序运行所需的原始数据，或程序运行产生的结果，就必须以文件形式存储到外部存储介质上。

1．文件与文件名

文件：是指存放在外部存储介质上的数据集合。

说明：每个文件都必须有一个文件名。

格式：主文件名[.扩展名]

文件命名规则：遵循操作系统的约定。

2．文件分类

（1）根据文件的内容分

.程序文件　　..源文件

..目标文件

..可执行文件

.数据文件

（2）根据文件的组织形式分

.顺序存取文件　　.随机存取文件。

（3）根据文件的存储形式分

.ASCII码文件

说明：ASCII码文件的每1个字节存储1个字符，对字符进行逐个处理。占用存储空间较多，而且要花费转换时间（二进制与ASCII码之间的转换）。

.二进制文件

说明：二进制文件是把内存中的数据，原样输出到磁盘文件中。可以节省存储空间和转换时间，但1个字节并不对应1个字符，不能直接输出字符形式。

3．读文件与写文件

读文件：将磁盘文件中的数据传送到计算机内存的操作。

写文件：从计算机内存向磁盘文件中传送数据的操作。

4．构成文件的基本单元与流式文件

在其它高级语言中，组成文件的基本单位是记录，对文件操作的基本单位也是记录。

C语言将文件看作是由一个一个的字符（ASCII码文件）或字节（二进制文件）组成的。将这种文件称为流式文件。

5. 文件类型FILE

系统给每个打开的文件都在内存中开辟一个区域，用于存放文件的有关信息（如文件名、文件位置等）。这些信息保存在一个结构类型变量中，该结构类型由系统定义、取名为FILE。

注意：结构类型名“FILE”必须大写。

6. ANSI C的缓冲文件系统

缓冲文件系统：指系统自动地在内存区为每个正在使用的文件开辟一个缓冲区。

说明：

①从内存向磁盘输出数据时，必须首先输出到缓冲区中。待缓冲区装满后，再一起输出到磁盘文件中。

②从磁盘文件向内存读入数据时，首先将数据读入到缓冲区中，再从缓冲区中将数据逐个送到程序数据区。

# 10.2 文件的打开与关闭

对文件进行操作之前，必须先打开该文件；使用结束后，应立即关闭，以免数据丢失。

C语言规定了标准输入输出函数库，用fopen()函数打开一个文件，用fclose()函数关闭一个文件。

## 10.2.1 文件的打开——fopen()函数

格式： FILE *fopen("文件名", "操作方式");

功能：返回一个指向指定文件的指针。

函数原型：stdio.h

注：对文件操作的库函数，函数原型均在头文件stdio.h中。后续函数不再赘述。

说明：

（1）“文件名”是指要打开（或创建）的文件名。若使用字符数组(或字符指针)，则不使用双引号。

（2）“操作方式”如下表所示。

| 文件使用方式 | 意 义 |
| --- | --- |
| “rt” | 只读打开一个文本文件，只允许读数据 |
| “wt” | 只写打开或建立一个文本文件，只允许写数据 |
| “at” | 追加打开一个文本文件，并在文件末尾写数据 |
| “rb” | 只读打开一个二进制文件，只允许读数据 |
| “wb” | 只写打开或建立一个二进制文件，只允许写数据 |
| “ab” | 追加打开一个二进制文件，并在文件末尾写数据 |
| “rt+” | 读写打开一个文本文件，允许读和写 |
| “wt+” | 读写打开或建立一个文本文件，允许读写 |
| “at+” | 读写打开一个文本文件，允许读，或在文件末追加数据 |
| “rb+” | 读写打开一个二进制文件，允许读和写 |
| “wb+” | 读写打开或建立一个二进制文件，允许读和写 |
| “ab+” | 读写打开一个二进制文件，允许读，或在文件末追加数据 |

例如：FILE　*fp;

　　fp＝fopen(″data.99″, ″r″);

（3）如果不能实现打开指定文件的操作，则fopen()函数返回一个空指针NULL(其值在头文件stdio.h中被定义为0)。

（4）“r（b）+”与“a（b）+”的区别：使用前者打开文件时，读写位置指针指向文件头；使用后者时，读写指针指向文件尾。

（5）使用文本文件向计算机系统输入数据时，系统自动将回车换行符转换成一个换行符；在输出时，将换行符转换成回车和换行两个字符。使用二进制文件时，内存中的数据形式与数据文件中的形式完全一样，就不再进行转换。

（6）有些C编译系统，可能并不完全提供上述对文件的操作方式，或采用的表示符号不同，请注意所使用系统的规定。

（7）在程序开始运行时，系统自动打开三个标准文件，并分别定义了文件指针：

- 标准输入文件——stdin：
  指向终端输入(一般为键盘)。如果程序中指定要从stdin所指的文件输入数据，就是从终端键盘上输入数据。
- 标准输出文件——stdout：
  指向终端输出(一般为显示器)。
- 标准错误文件——stderr：
  指向终端标准错误输出（一般为显示器）。

为增强程序的可靠性，常用下面的方法打开一个文件：

```
if((fp=fopen("文件名","操作方式"))==NULL)
{ printf("can not open this file\n");
  exit(0); }
```

exit()函数

格式：void  exit([程序状态值]);

功能：关闭已打开的所有文件，结束程序运行，返回操作系统，并将“程序状态值”返回给操作系统。

说明：当“程序状态值”为0时，表示程序正常退出；非0值时，表示程序出错退出。

## 10.2.2　文件的关闭——fcolse()函数

格式：　int　fclose(FILE　*文件指针);

功能：关闭“文件指针”所指向的文件。

说明：如果正常關閉了文件如果正常关闭了文件，则函数返回值为0；否则，返回值为非0。

例如，fclose(fp)；/*关闭fp所指向的文件*/

# 10.3 文件的读写操作

文件打开之后，就可以对它进行读与写的操作了。

10.3.1 读/写文件中的一个字符
10.3.2 读/写一个字符串
10.3.3 读/写一个数据块
10.3.4 对文件进行格式化读/写
10.3.5 读/写函数的选用原则

# 10.3.1 读/写文件中的一个字符

1. 将一个字符写到文件中——fputc()函数

例：将键盘上输入的一个字符串（以“@”作为结束字符），以ASCII码形式存储到一个磁盘文件中。

```
#include “stdio.h”
void main(int argc, char *argv[])
{ FILE *fp;
  char ch;
  if(argc!=2)                                  /*参数个数不对*/
  { printf("the number of arguments not correct\n\n");
    exit(0);
  }
  if((fp=fopen(argv[1],"w"))==NULL) /*打开文件失败*/
  {  printf("can not open this file\n");
     exit(0);
  }
```

```
/*输入字符，并存储到指定文件中*/
   for(;(ch=getchar())!='@';)
       fputc(ch,fp);  /*输入字符并存储到文件中*/
   fclose(fp);        /*关闭文件*/
}
```

程序运行情况：

abcdefg1234567@←┘

库函数fputc()：

格式：int fputc(字符数据,文件指针);

功能：将字符数据输出到“文件指针”所指向的文件中去，同时将读写位置指针向前移动1个字节（即指向下一个写入位置）。

说明：

① 其中“字符数据”，既可以是字符常量，也可以是字符变量。

② 如果输出成功，则函数返回值就是输出的字符数据；否则，返回一个符号常量EOF(其值在头文件stdio.h中，被定义为-1)。

## 2. 从文件中读入一个字符——fgetc()函数和feof()函数

[案例12.2] 顺序显示[案例12.1]创建的磁盘ASCII码文件。

```
/*案例代码文件名：AL12_2.C*/
/*程序功能：顺序显示一个磁盘ASCII码文件*/
/*参数：带参主函数，使用格式：可执行文件名  源文件名*/
#include "stdio.h"
void main(int argc, char *argv[])
{ FILE *fp;
  char ch;
  if(argc!=2)                /*参数个数不对*/
  {  printf("the number of arguments not correct\n");
     printf("\n Usage:  可执行文件名  源文件名");
     exit(0);
  }
```