

这个程序有bug：最后一个数字出现了两遍，根本的原因在于对feof的理解有问题

我只有先“读”了一下才知道这个文件有没有结束，feof是对当前文件有没有结束做出判断，所以先要读才能做判断，所以最后的数据会被输出两次。

怎么改：先fscanf，再while；while中先printf, i++, if(), 再fscanf

修改：while (fscanf(fp, “%d”, &n) != EOF)

或者这么写：while (fscanf(fp, “%d”, &n) == 1)

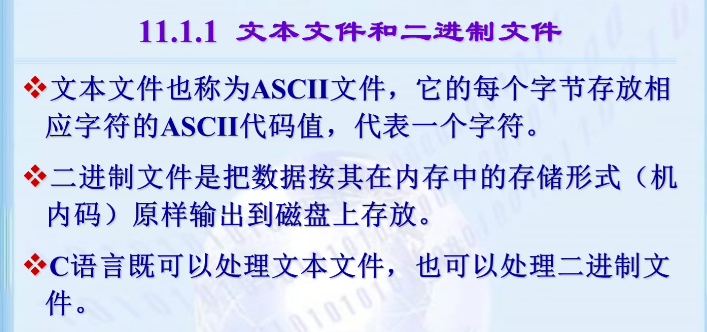
1、定义一个文件指针变量：FILE \*fp

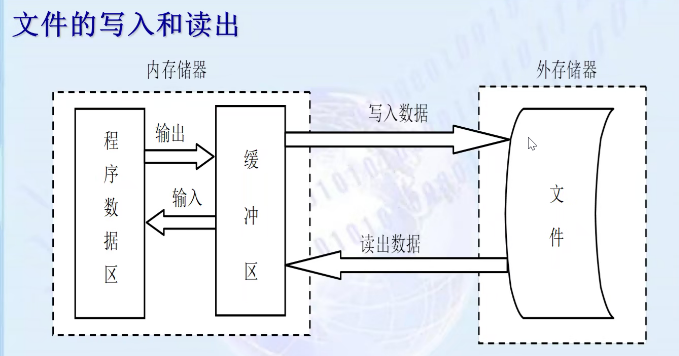
2、文件路径的分隔符是[\\（因为一个\](file:///\\（因为一个\)表示的是转义字符，例如\r）

3、打开文件的目的是创建一个文件缓冲区（有两种方式，有缓冲区和无缓冲区；默认的是有缓冲区的方式打开文件，这个缓冲区便于数据的交互）

4、feof()函数判断这个文件当前是否结束，返回的是一个整数

5、fscanf()函数用于读取文件

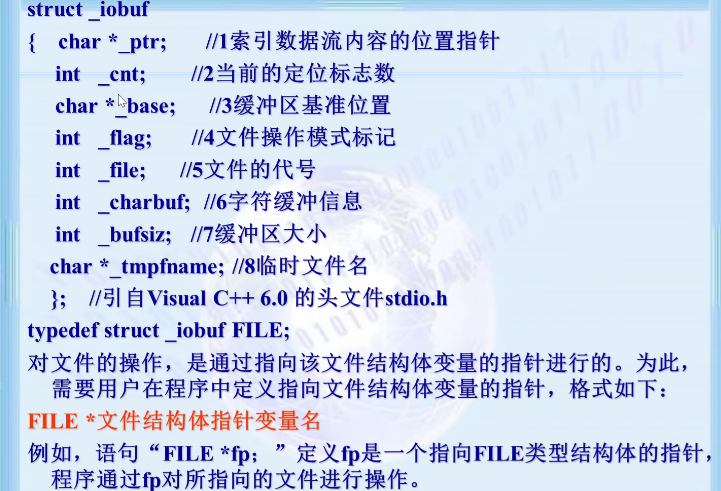




文件的类别：

存放123：

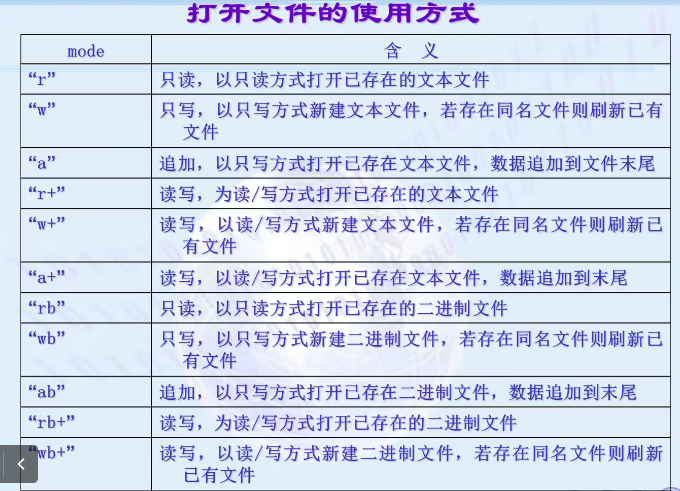
1. 文本文件：三个ascii字符1 2 3
2. 二进制文件：将三个ascii字符转换成二进制的形式（int类型四个字节）

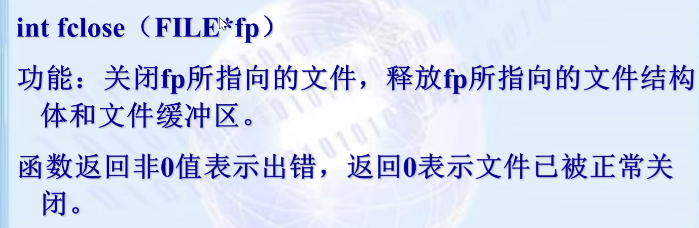


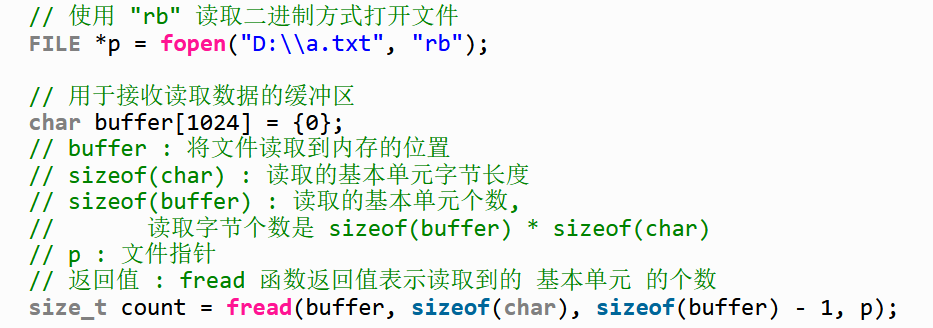
文件缓冲区的结构体类型

--->通过文件指针访问文件缓冲区









二进制文件：fread和fwrite

**文件的定位操作：**

**1、fseek函数用来移动文件位置指针到指定的位置上，接着的读或写操作将从此位置开始。函数的调用形式如下：**

**fseek（pf，offset，origin）；**

**此处，pf是文件指针；offset是以字节为单位的位移量，为长整型数；origin是起始点，用以指定位移量是以哪个位置为基准，起始点既可用标识符来表示，也可用数字来表示。下表给出了代表起始点的标识符和对应的数字。**



对于二进制文件，当位移量为正整数时，表示位置指针从指定的起始点向文件尾部方向移动；当位移量为负整数时，表示位置指针从指定的起始点向文件首部方向移动。

假设pf已指向一个二进制文件，以下函数调用将使文件位置指针从文件的开头后移30个字节：

fseek（pf，30L，SEEK\_SET）；

若pf已指向一个二进制文件，以下函数调用将使文件位置指针从文件尾部前移10个sizeof（int），即20个字节：

fseek（pf，-10L\*sizeof（int），SEEK\_END）；

对于文本文件，位移量必须是0。假设pf已指向一个文本文件，以下函数调用将使文件位置指针移到文件的开始：

fseek（pf，0L，SEEK\_SET）；

假设pf已指向一个文本文件，以下函数调用将使文件位置指针移到文件的末尾：

fseek（pf，0L，SEEK\_END）；

2、ftell函数用以获得文件当前位置指针的位置，函数给出当前位置指针相对于文件开头的字节数。若fp已指向一个正确打开的文件，函数的调用形式如下：

long t；

t=ftell（fp）；

当函数调用出错时，函数返回-1L。

当打开一个文件时，通常并不知道该文件的长度，通过以下函数调用可以求出文件的字节数：

fseek（fp，0L，SEEK\_END）；//把位置指针移到文件末尾

t=ftell（fp）；//求出文件中的字节数

若二进制文件中存放的是struct st结构体类型数据，则通过以下语句可求出该文件中的以该结构体为单位的数据块的个数：

fseek（fp，0L，SEEK\_END）；

t=ftell（fp）；

n=t/sizeof（struct st）；

3、rewind函数又称“反绕”函数，调用形式如下：

rewind（pf）；此处pf为文件指针。此函数没有返回值，函数的功能是使文件的位置指针回到文件的开头。

