C语言程序设计

计算机科学与技术学院

C程序输入输出

计算机的输入输出

- 输入输出是以计算机主机为主体而言
- 从外部设备向计算机输入数据称为“输入”
  - 如键盘、磁盘、光盘、扫描仪等
- 从计算机向外部设备输出数据称为“输出”
  - 如显示屏、打印机、磁盘等

C语言中的输入输出

- 实现C程序与用户的交互
- C语言本身不提供输入输出语句
- 输入和输出操作是由函数来实现
  - printf()函数和scanf()函数
  - printf和scanf不是C语言的关键字，只是函数的名字

标准输入输出函数

- c语言函数库中已定义一批“标准输入输出函数”，
- 以标准的输入输出设备(如键盘、显示器)为输入输出对象
- 在调用标准输入输出库函数时，文件开头应有以下预编译命令：

  #include <stdio.h>

  或

  #include "stdio.h"

  - < > 标准路径
  - " "当前（可执行文件）所在路径

常用的标准输入输出函数

- printf( ) /格式输出函数/
- scanf( ) /格式输入函数/
- putchar( ) /单个字符输出函数/
- getchar( ) /单个字符输入函数/
- puts( ) /字符串输出函数/
- gets( ) /字符串输入函数/

printf() 格式输出函数

- 作用：向终端（显示器或指定的输出设备）输出信息
  - 可输出内容包括文本、数值、符号等信息
  - 信息可以是“静态”的，也可以是“动态”的
  - 可按照指定的要求，对数据进行格式控制
- 一般格式：printf(格式控制字符串，输出表列);
  - 格式控制字符串: 双引号("")括起的字符串, 可包括：
    - 普通字符：printf()将其原样输出
    - 格式说明符（占位符）：%+格式字符，printf()将用输出表列中取得实际内容，并按指定格式输出
  - 输出表列：一组变量名或表达式

printf() 格式输出函数

- 一般格式：printf(格式控制字符串，输出表列);
  - 格式控制字符串: 双引号("")括起的字符串, 可包括：
    - 普通字符：printf()将其原样输出
    - 格式说明符（占位符）：%+格式字符，printf()将用输出表列中取得实际内容，并按指定格式输出
  - 输出表列：一组变量名或表达式

- 格式控制字符串为普通字符时：
  - 格式控制字符串为**字符串常量**
  - printf ("Hello, world.");
  - printf ("您好！欢迎你来到程序王国! ");
  - 换行符、制表符等特殊符号用**转义字符**表示
  - printf ("您好！\n欢迎你来到程序王国!");
  - printf ("您好！\t欢迎你来到程序王国!");

printf() 格式输出函数

- 一般格式：printf(格式控制字符串，输出表列);
  - 格式控制字符串: 双引号("")括起的字符串, 可包括:
    - 普通字符：printf()将其原样输出
    - 格式说明符（占位符）：%+格式字符，printf()将用输出表列中取得实际内容，并按指定格式输出
  - 输出表列：一组变量名或表达式

- 格式说明符包括格式控制符时:
  - 常用的格式说明符:
  - %d:十进制整数（带符号）
  - %f:浮点数（单、双精度数）
  - %c:单个字符
  - %s:字符串
  - %e:科学计数法
  - %o:八进制
  - %x:十六进制

变量

- 在程序运行过程中可以发生变化的量称为变量
- 变量名 对应 存储数据的内存地址
- 在程序中如果要使用变量，必须先确定其数据类型和名称，即变量定义
- 变量需要“先定义，再使用”
- 变量定义语句的形式如下:
  数据类型名 变量名1[,变量名2,...];
- 举例:

```
int a=3,b=4;
float pi=3.14159;
char c='A';
```

printf() 格式输出函数示例

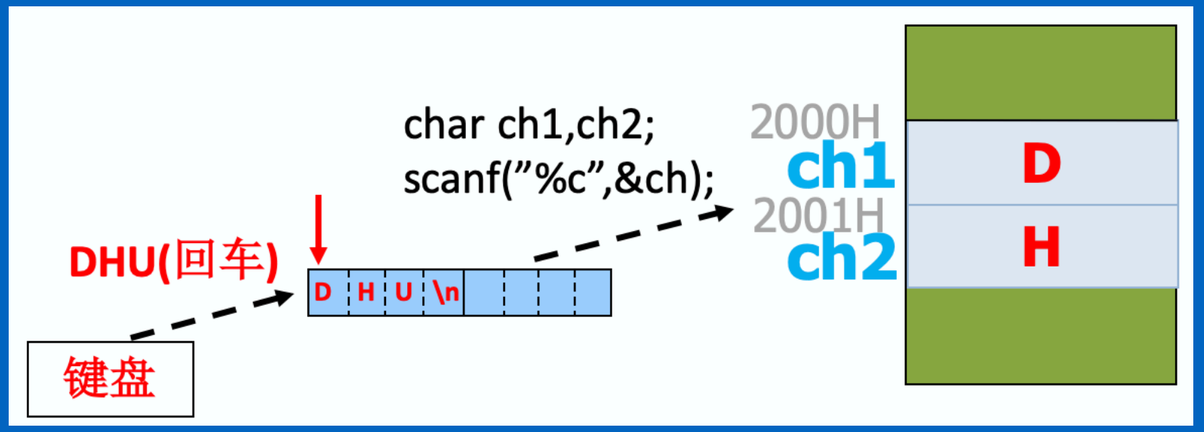
- 若int a=3,b=4; float pi=3.14159; char c='A'; 则：
  - printf("Hello, world!");
  - printf("%d,%d",a,b); 输出: 3,4
  - printf("a=%d,b=%d\n",a,b); 输出: a=3,b=4(换行符）
  - printf("a=%3d,b=%-3d\n",a,b); 输出: a=␣␣3,b=4␣␣
  - printf("%7.2f",pi); 输出: ␣␣␣3.14
  - printf("%d",pi); 出错！
  - printf("%-3c,%d",c,c); 输出: A␣␣,65
  - printf("%s","Hello, world!");
  - printf("\101abc\101\n\0\n"); 输出???

scanf() 格式输入函数

- 作用：从输入设备获取信息，保存到变量所在的内存地址中去
- 一般格式：scanf（格式控制字符串，地址表列）
- 常用的格式说明符：
  - %d:十进制整数（带符号）
  - %f:浮点数（单精度数）
  - %lf:浮点数（双精度数）（与printf不同！）
  - %c:单个字符
  - %s:字符串
- 格式控制字符串除了格式说明符之外的其他字符，则在相应位置原样输入
- scanf() 函数中的“格式控制”后面应当是变量地址，而不应是变量名（与printf不同！）
- 默认的间隔标志：空格、“制表”（Tab）或 “回车”；

scanf() 格式输入函数示例

- 若int a,b; float d,e; double f; char c1,c2; 则：
  - scanf("%d, %d",&a, &b); 须输入：3,5（输入：3␣5 不行 ）
  - scanf("%d%d", &a, &b); 须输入：3␣5（输入：3,5 不行 ）
  - scanf("%3d",&a)； 12345 ⇒ 123 (按指定的宽度结束)
  - scanf("%d",&a)； 45a ⇒ 45 （遇非法输入时结束）
  - scanf("a=%d",&a); 不行
  - scanf("%f,%f",&d, &e); 输入：3.14,5.28
  - scanf("%7.2f",&d); 不行(输入浮点数时不能规定精度）
  - scanf("%d",&e); 不行
  - scanf("%f",&f); 不行(输入双精度浮点数时应用%lf）
  - scanf("%c",&c1);
  - scanf("%c%c",&c1，&c2);
    - 输入a␣b时：‘a’⇒ c1, ‘␣’⇒ c2（而不是‘b’⇒ c2)



getchar() 与putchar()

- 单个字符的输入与输出

```
#include<stdio.h>
int main()
{char c;
 c=getchar();
 putchar(c);
 putchar('\n');
 return 0;
}
```

- 运行程序：
  - 从键盘输入字符‘a’
  - 按Enter键
  - 屏幕上将显示输出的字符‘a’
  - 再输出一个换行符

示例：字母大小写转换

- 从键盘输入一个大写字母，再用小写字母输出

```
#include <stdio.h>
int main()
{ char c1,c2;
  c1=getchar();
  printf("%c,%d\n",c1,c1);//putchar(c1);
  c2=c1+32;//c2=c1-'A'+'a'
  printf("%c,%d\n",c2,c2);//putchar(c2);
}
```

- 运行程序：
  - 从键盘：A
  - 按Enter键
  - 输出

    A,65

    a,97

ASCII 码		字符	ASCII 码		字符	ASCII 码		字符	ASCII 码		字符
十进制	十六进制		十进制	十六进制		十进制	十六进制		十进制	十六进制	
032	20		056	38	8	080	50	P	104	68	h
033	21	!	057	39	9	081	51	Q	105	69	i
034	22	"	058	3A	:	082	52	R	106	6A	j
035	23	#	059	3B	;	083	53	S	107	6B	k
036	24	$	060	3C	<	084	54	T	108	6C	l
037	25	%	061	3D	=	085	55	U	109	6D	m
038	26	&	062	3E	>	086	56	V	110	6E	n
039	27	'	063	3F	?	087	57	W	111	6F	o
040	28	(	064	40	@	088	58	X	112	70	p
041	29	)	065	41	A	089	59	Y	113	71	q
042	2A	*	066	42	B	090	5A	Z	114	72	r
043	2B	+	067	43	C	091	5B	[	115	73	s
044	2C	,	068	44	D	092	5C	\	116	74	t
045	2D	-	069	45	E	093	5D	]	117	75	u
046	2E	.	070	46	F	094	5E	^	118	76	v
047	2F	/	071	47	G	095	5F	_	119	77	w
048	30	0	072	48	H	096	60	`	120	78	x
049	31	1	073	49	I	097	61	a	121	79	y
050	32	2	074	4A	J	098	62	b	122	7A	z
051	33	3	075	4B	K	099	63	c	123	7B	{
052	34	4	076	4C	L	100	64	d	124	7C	|
053	35	5	077	4D	M	101	65	e	125	7D	}
054	36	6	078	4E	N	102	66	f	126	7E	~
055	37	7	079	4F	O	103	67	g	127	7F	⌂

红包程序示例（一）

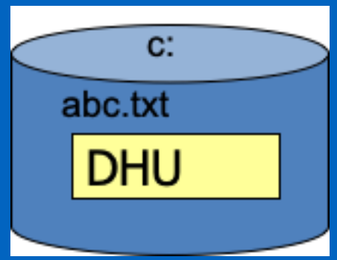
```c
#include <stdio.h>
int main()
{
    int number,money;
    char confirm;

    printf("\n\t\t请输入您的红包个数: \t");
    scanf("%d",&number);
    printf("\n\t\t请输入您单个红包金额（元）: \t");
    scanf("%d",&money);
    getchar();
    printf("\n\n\t\t确认(Y)\t\t取消(C)\n");
    scanf("%c",&confirm);
    printf("您出了%d个红包，单个红包金额%d（元）钱\n",number,money);

    return 0;
}
```

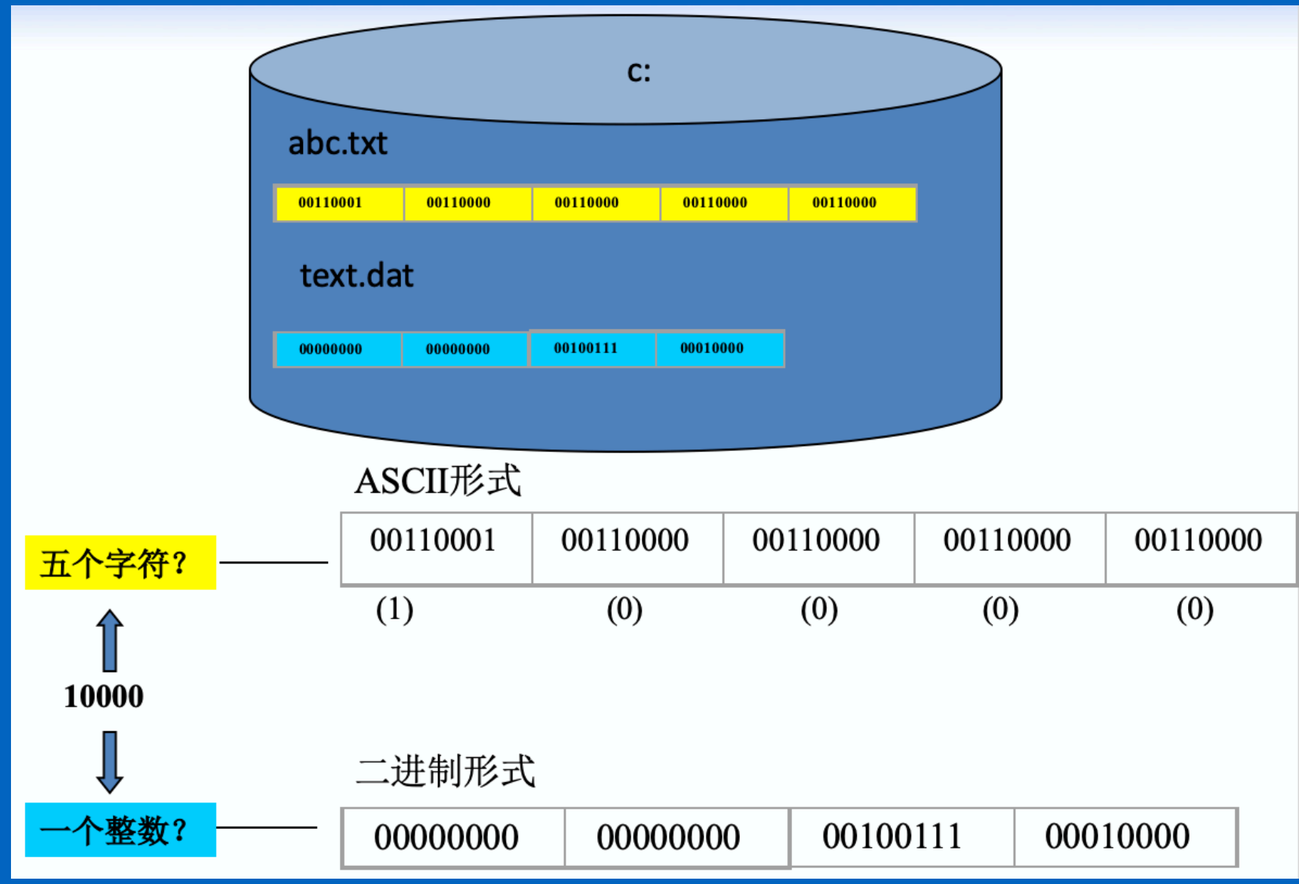
其他方式的输入与输出

- 是不是数据只有通过键盘输入?
- 是否结果只能输出在显示器上?
- 文件:
  - 持久化存储
  - 有序的数据的集合
  - 程序文件是若干程序代码的集合
  - 数据文件是若干数据项的有序集合



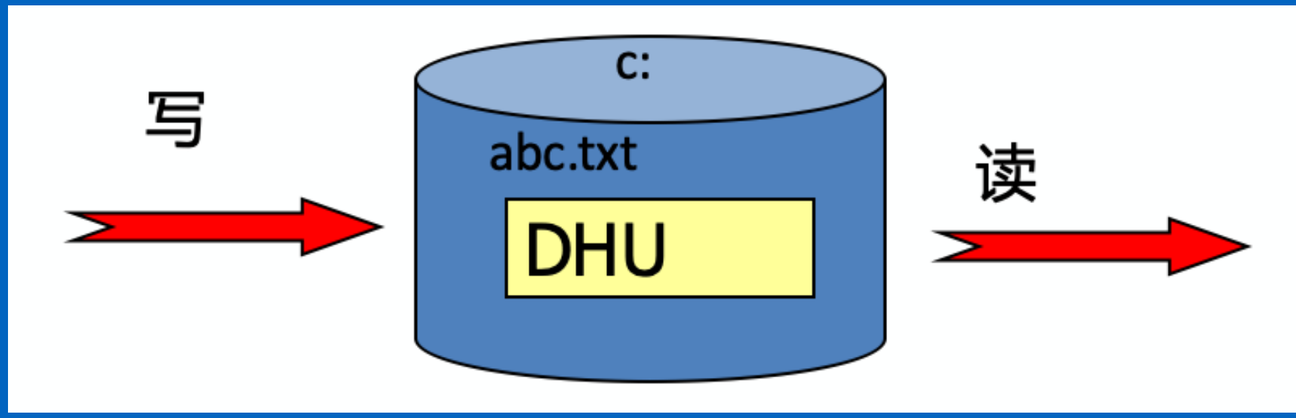
文件的编码

- 文本文件
  - 字符组成
  - ASCII码
  - .txt
- 二进制码文件
  - 二进制数据
  - 特定格式
  - .jpg .mkv .doc



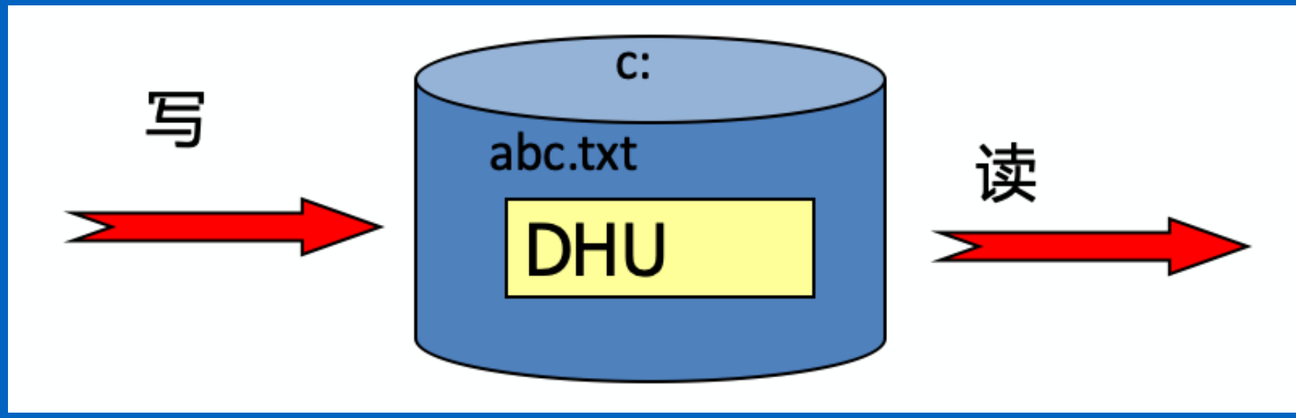
文件的读写

- 文件的读操作
  - 从文件中获取信息,即指从某个文件中读出信息
- 文件的写操作
  - 指向某个文件中写入信息，即向文件中存放信息



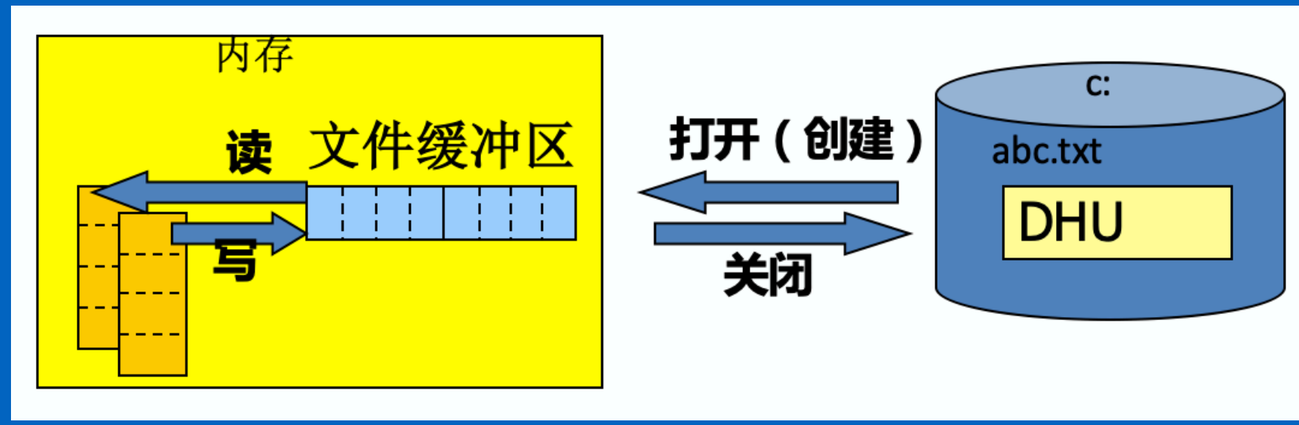
流操作

- C语言的文件又称为“流”，对文件操作就是对这种“流”进行操作
  - 字符文件被看成为字符“流”
  - 二进制文件被看成为二进制数码“流”
  - 从某个输入流的输入信息中使用读函数提取信息,存放在变量中。
  - 使用写函数将某种信息输入到输出信息流中,存放到某个设备中。



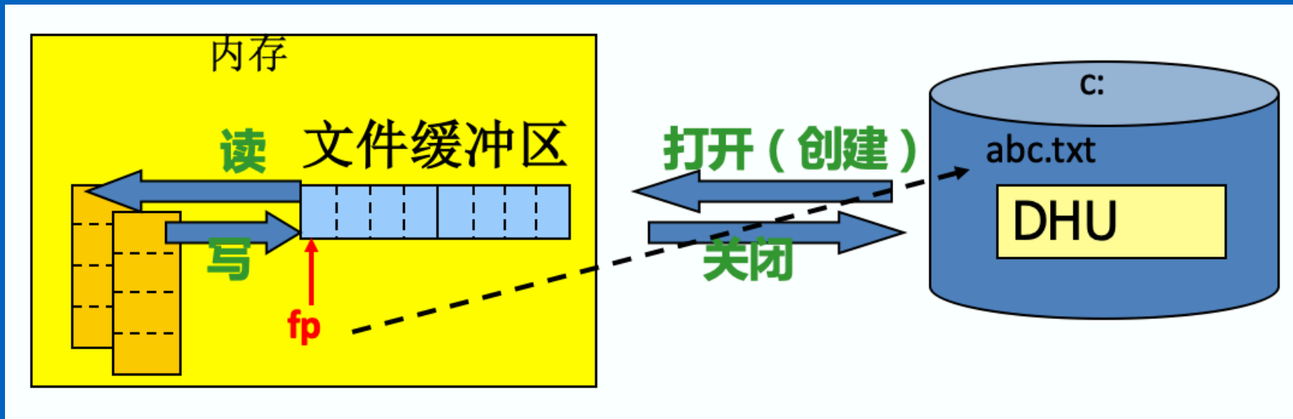
C语言文件的操作

- 打开（创建）文件:将文件从磁盘上读到内存缓冲区中，以备对其操作
- 用系统函数进行文件的读、写操作
- 关闭文件：将被打开的文件返回磁盘中去，并清除所占据的内存空间



使用文件一般流程

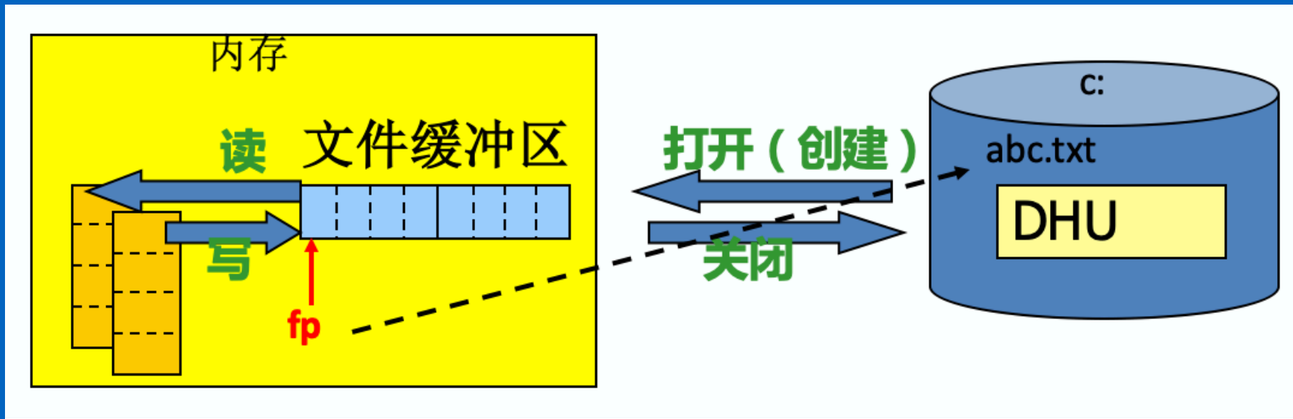
- 定义文件指针
  - 示例：FILE * fp;
- 打开
  - fopen(“<文件名>”,“<打开方式>”);
  - 指明打开文件路径和文件名
  - 指明打开文件方式r
- 读、写信息
  - fscanf()
  - fprintf()
- 关闭
  - fclose(fp)



文件指针

- 指向文件的指针，即指针指向的对象是某个文件。
- FILE *fp;
  - 文件类型说明符
  - 存放在stdio.h文件中
  - 被定义成具有5个成员的结构类型
  - 用来存放有关文件的信息

```
Typedef struct
{
    int _fd;
    int _cleft;
    int _mode;
    char *_nextc;
    char *_buff;
}FILE;
```



fopen()打开文件函数

- 给出文件名（完整的标识符）
- 给出打开方式(文本或二进制文件、读或写等)
- 获取文件指针（用此指针对文件操作）
- 函数格式：fopen(“<文件名>”,“<打开方式>”);
  - 文件名：
    - 要打开文件存放在磁盘中的全名，包括扩展名，
    - 必要时应加路径名用双撇号括起
- 打开方式：
  - r 读方式
  - w 写方式
  - a 追加写方式
  - r+ 可读可写方式
  - w+ 可写可读方式
  - rb 二进制文件读方式
  - wb 二进制文件写方式
  - ab 二进制文件追加写方式
  - rb+ 二进制可读可写方式
  - wb+ 二进制可写可读方式

fopen() 打开文件

- fopen()函数返回一个地址值
- 将该函数返回的地址值赋给一个文件指针，表示让这个文件指针指向被打开的文件（缓冲区）
- 若函数值非零，文件被正常打开，该函数值便是存放被打开文件的内存缓冲区的首地址；
- 若函数值为零（NULL)，文件打开失败
- exit() 须使用头文件 stdlib.h

例：

```
FILE *fp;
fp=fopen(“abc.txt”, “r”);
if (fp==NULL)
{
    printf(“file can’t open!\n”);
    exit(1);
}
```

```
FILE *fp;
fp=fopen(“c:\\abc.dat”, “wb”);
if (fp==NULL)
{
    printf(“file can’t open!\n”);
    exit(1);
}
```

fclose()关闭文件函数

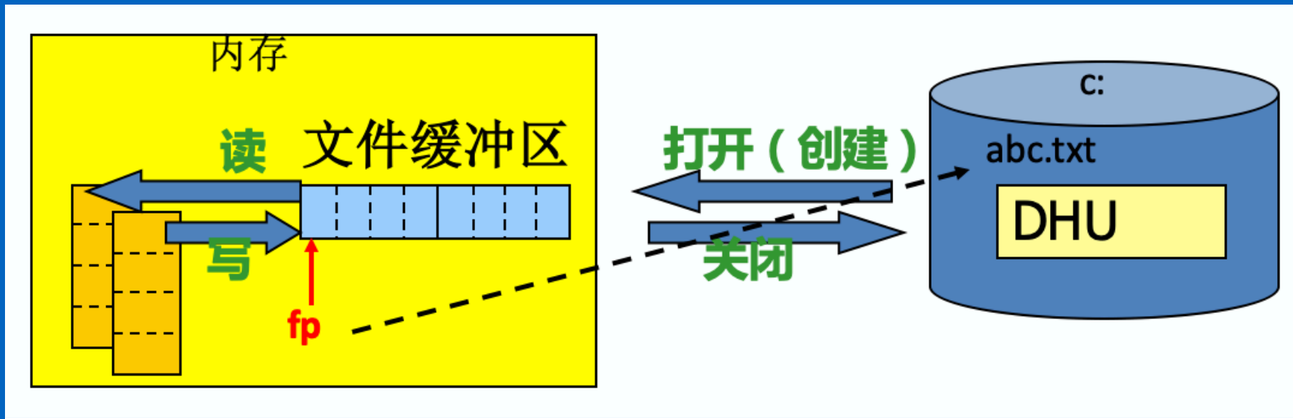
- 函数格式：fclose(<文件指针>);
- 文件指针:待关闭文件的文件指针
- fclose函数一次只能关闭一个被打开的文件，多个被打开的文件要使用多次关闭函数。
- 关闭文件不是删除文件，关闭文件的好处：
  - 及时释放被占据的内存空间，提高机器的运行效率；
  - 保证文件内容的安全

文件的读写操作

- 针对一个字符的读写函数
- 针对一个字符串的读写函数
- 针对一个数据块的读写函数
- **带格式的读写函数**
  - fprintf(**文件指针**，格式控制字符串，输出表列);
  - fscanf(**文件指针**，格式控制字符串，地址表列)

回顾：使用文件一般流程

- 定义文件指针
  - 示例：FILE * fp;
- 打开
  - fopen(“<文件名>”,“<打开方式>”);
  - 指明打开文件路径和文件名
  - 指明打开文件方式r
- 读、写信息
  - fscanf()
  - fprintf()
- 关闭
  - fclose(fp)



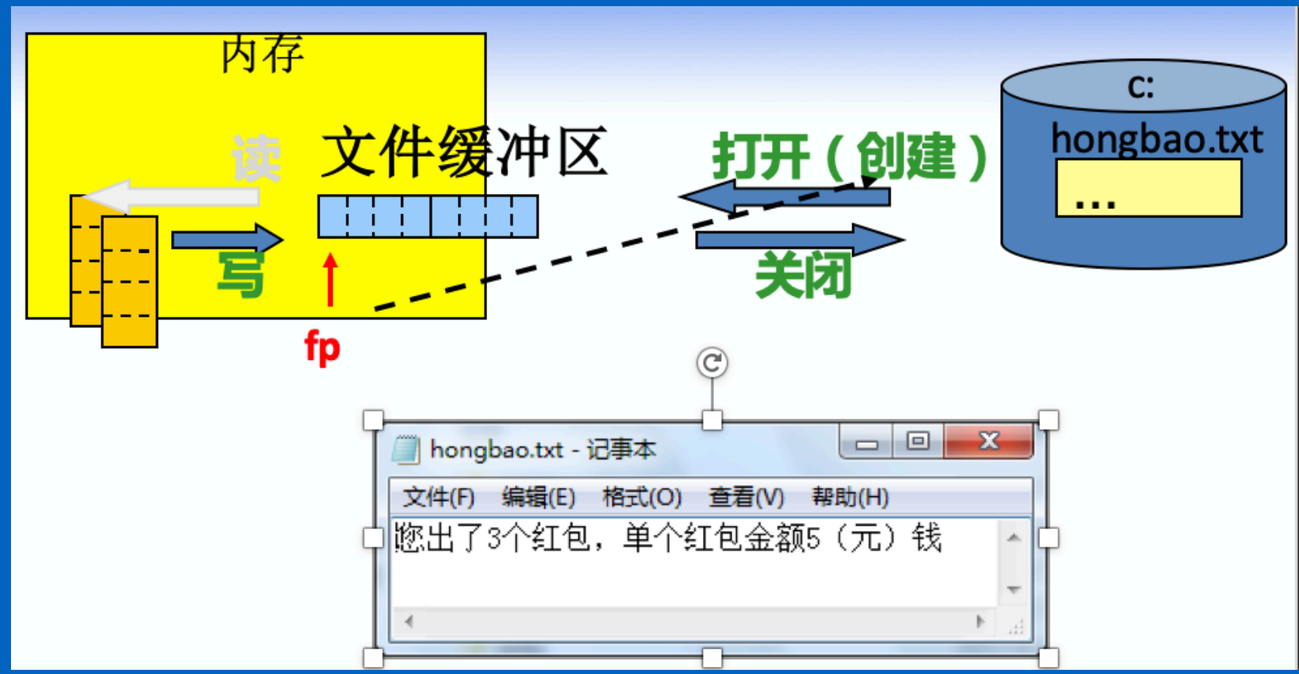
红包程序示例（二）

```c
#include <stdio.h>
int main()
{
    int number,money;
    char confirm;

    FILE *fp;
    fp=fopen("hongbao.txt","w");

    printf("\n\t\t请输入您的红包个数：\t");
    scanf("%d",&number);
    printf("\n\t\t请输入您单个红包金额（元）：\t");
    scanf("%d",&money);
    getchar();
    printf("\n\n\t\t确认(Y)\t\t取消(C)\n");
    scanf("%c",&confirm);
    printf("您出了%d个红包，单个红包金额%d（元）钱\n",number,money);

    fprintf(fp,"您出了%d个红包，单个红包金额%d（元）钱\n",number,money);
    fclose(fp);
    return 0;
}
```



成绩信息处理示例

```c
#include <stdio.h>
int main()
{
    float comp_s,eng_s,maths_s,sum;
    FILE *fp;
    fp=fopen("list.txt","r");
    printf("\t\t\t成绩报表\t\n\n");
    printf(" |       计算机|         英语|         高数|        总分|\n");
    printf("     ------------------------------------------------\n");
    if(fscanf(fp,"%f,%f,%f",&comp_s,&eng_s,&maths_s) != NULL)
    {
        sum=comp_s+eng_s+maths_s;
        printf(" |%12.1f|%12.1f|%12.1f|%12.1f\n",comp_s,eng_s,maths_s,sum);
        printf("     ------------------------------------------------\n");
    }
    fclose(fp);
    return 0;
}
```