IO 进程

scanf\printf:终端

IO：input/output，文件

标准 IO

文件 IO

文件属性获取：ls -l 文件类型 文件权限 链接数 用户名 组名 大小 时间 文件名

目录操作：ls

库

进程

进程：创建进程

线程：创建线程、同步和互斥

进程间通信：7->6 种

7 左右

函数：主要用函数，50-60 个函数

学习函数的方法

上课认真、积极回应、有问题及时问

标准 IO

文件：7 种文件类型

b(块设备) c(字符设备) d(目录) -(普通文件) l(链接文件) s(套接字) p(有名管道)

1. 概念

1.1 定义

在 C 库中定义的一组专门用于输入输出的函数

1.2 特点

1）有缓冲机制，通过缓冲机制减少系统调用的次数，提高效率

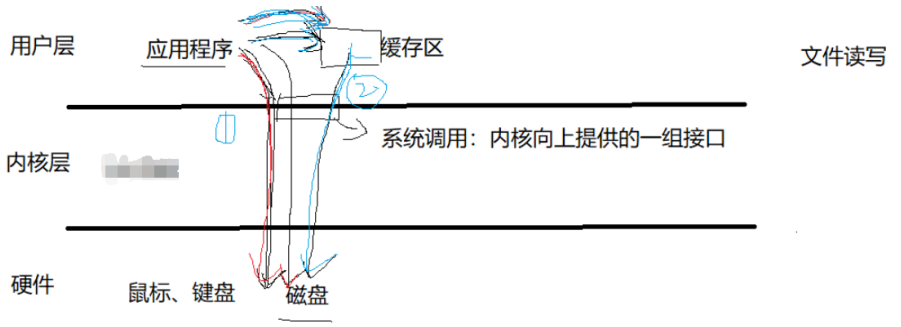
系统调用：内核向上提供的一组接口

2）围绕流进行操作，流用 FILE *描述，FILE 是一个结构体，描述的是文件的相关信息

typedef struct _IO_FILE FILE;

3）默认打开了三个流，stdin(标准输入)、stdout(标准输出)、stderr(标准错误)

struct _IO_FILE *stdin; --> FILE *stdin ;



1.写1000个字符到文件，每次写100个，写10次，每次经过系统调用
2.写1000个字符到文件，每次写100个写10次，先将字符写入缓存区，写满缓存区再刷新到文件，只有一次系统调用，效率高

补充：ctags 的使用(可以追代码)

vi -t FILE(typedef 定义数据类型、宏定义、结构体等)

选择合适的编号

将光标定位在目标位置，ctrl+] :向下追代码

ctrl+t：回退

q：退出

1.3 缓存区

1）全缓存：和文件相关

刷新缓存区的条件：

1-程序正常结束

2-缓存区满刷新

3-fflush 强制刷新

2）行缓存：和终端相关

刷新缓存区的条件：

1-程序正常结束

2-\n 刷新缓存

3-缓存区满刷新

4-fflush 强制刷新

3）不缓存：没有缓存区，标准错误

练习：计算行缓存中标准输出的缓存区大小

2. 函数接口

2.1 打开文件

```
FILE *fopen(const char *path, const char *mode);
参数：
    path：打开文件
    mode：打开方式
        r：只读，流被定位到文件开头
        r+：可读可写，流被定位到文件开头
        w：只写，文件不存在创建，文件存在清空，流被定位到文件开头
        w+：可读可写，文件不存在创建，文件存在清空，流被定位到文件开头
        a：追加，文件不存在创建，存在追加，流被定位到文件末尾
        a+：可读可写，文件不存在创建，存在追加，开始进行读从头读，进行写流被定位到文件末尾
返回值：成功：文件流
       失败：NULL，并且设置 errno(错误码)
```

2.2 读写文件

2.2.1 每次一个字符的读写

```
int fgetc(FILE *stream);
功能：从文件中读一个字符
参数：stream：文件流
返回值：成功：读到字符的 ASCII
        失败或读到文件末尾：EOF（-1）
int fputc(int c, FILE * stream)
功能：向文件中写入一个字符
参数：c：要写的字符
   stream：文件流
返回值：成功：写的字符的 ASCII
       失败：EOF
```

练习：编程实现 cat 功能

思路：打开文件，循环读文件(fgetc)，当读到文件末尾(fgetc 函数返回值为 EOF)循环结束，打印读到的内容

补充：

2.2.2 每次一行的读写

```
char *fgets(char *s, int size, FILE *stream);
功能：从文件中读取一串字符
参数：s：存放读取的字符串的首地址
      size：读取的大小
      stream：文件流
返回值：成功：读取的字符串的首地址
       失败或读到文件末尾：NULL
特性：1.实际读取 size-1 个字符,在末尾添加\0
      2.读到\n 结束读取
int  fputs(const char *s,  FILE *stream);
功能：向文件中写字符串
参数：s：要写的内容
     stream：文件流
返回值：成功：非负整数
        失败：EOF
```

练习：编程实现计算一个文件行数的功能(wc -l 文件名)。

要求：使用 fgets 实现

思路：打开文件，循环读文件，当读到文件末尾(fgets 返回值为 NULL)循环结束，在循环中判断字符串中是否有\n，如果是\n，则变量 n++即可

2.3 关闭文件

```
int fclose(FILE* stream);
功能：关闭文件
参数：stream：文件流
```

作业：

1. 梳理今天内容
2. 题目要求：编程读写一个文件 test.txt，每隔 1 秒向文件中写入一行数据，类似这样：

1，2007-7-30 15:16:42

2，2007-7-30 15:16:43

该程序应该无限循环，直到按 Ctrl-C 中断程序。

再次启动程序写文件时可以追加到原文件之后，并且序号能够接续上次的序号，比如：

1，2007-7-30 15:16:42

2，2007-7-30 15:16:43

3，2007-7-30 15:19:02

4，2007-7-30 15:19:03

5，2007-7-30 15:19:04

sleep(1);

fprintf/sprintf();

time(); //计算时间，秒

localtime(); //秒转换成年月日时分秒

思路：打开文件，计算行数，循环向文件中写字符串，每隔一秒写入一行

注意：全缓存