需要包含头文件<fcntl.h><sys/stat.h><sys/types.h>

int open(const char \*pathname, int flags);

int open(const char \*pathname, int flags, mode\_t mode);

当oflag中不包括O\_CREAT时，open只需要两个参数;常用oflag(O\_RDONLY、O\_WRONLY、O\_RDWR、O\_TRUNC、O\_APPEND、O\_CREAT);成功返回文件描述符，失败返回-1;

O\_APPEND在**每次写**时都追加到文件的尾部，是一个原子操作;

int creat(const char \*pathname, mode\_t mode);

若成功返回文件描述符，若失败返回-1;

int close(int fd);

若成功返回0,若失败返回-1;关闭一个文件，同时释放该进程加在该文件上的所有记录锁;

需要包含头文件<unistd.h>

ssize\_t read(int fd, void \*buf, size\_t count);

返回读到的字节数，若已到达文件尾返回0,若出错返回-1;

ssize\_t write(int fd, const void \*buf, size\_t count);

返回写入的字节数,若出错返回-1;

需要包含头文件<unistd.h><sys/types.h>

off\_t lseek(int fd, off\_t offset, int whence);

若成功返回新的文件偏移量，若失败返回-1;是fseek和ftell的综合;64位系统的off\_t至少为64位;

int truncate(const char \*path, off\_t length);

int ftruncate(int fd, off\_t length);

将文件或已打开文件截断到length长度;成功返回0,失败返回-1;read和write在读写后偏移量自动增加;

int dup(int oldfd);

int dup2(int oldfd, int newfd);

若成功返回新的文件描述符，若出错返回-1;

dup会返回当前可用最小的文件描述符;等价于fcntl(fd, F\_DUPFD, 0);

dup2如果newfd已打开则先将其关闭，若fd等于fd2则dup2返回fd2而不关闭;

void sync(void);

将内核的buffer和cache写入硬盘，用于解除设备挂载时;

/dev/fd

当前进程的文件描述符信息，如ls访问就是ls这个进程的描述符信息;