FILE \*fopen(const char \*pathname, const char \*mode);

pathname为路径加文件名组成的字符串，mode为r、w、a或r+、w+、a+的字符串;

若成功返回文件指针，若出错返回NULL并设置errorno;

int fclose(FILE \*stream);

若成功返回0，若出错返回EOF;

int fgetc(FILE \*stream);

若成功返回该字符由unsigned char转换为int后的值，若到达文件尾或出错返回EOF;

int fputc(int c, FILE \*stream);

若成功返回c，若出错返回EOF;

char \*fgets(char \*buf, int size, FILE \*stream);

buf为缓冲区是一个char型数组，size是这个缓冲区的大小，通常宏定义为BUFFERSIZE，stream为FILE\*格式的流;一次读入size-1个字符，最后一位填充’\0’;若成功返回buf，若到达文件尾或出错返回NULL;

int fputs(const char \*s, FILE \*stream);

s为要写入的字符串，stream为FILE\*格式的流；若成功返回非负值，若出错返回EOF;

size\_t fread(void \*ptr, size\_t size, size\_t nmemb, FILE \*stream);

读二进制文件，ptr为缓冲区大小为size\*nmemb，size为数据结构的大小（建议为1作为fgetc来用），nmemb一次读取的对象数量，stream为读取文件的流;成功返回读取到的对象个数，若到达文件尾或出错返回低于nmemb的数量或0;

size\_t fwrite(const void \*ptr, size\_t size, size\_t nmemb,FILE \*stream);

写二进制文件，ptr为缓冲区，size为数据结构的大小，nmemb写入对象数量（fread的返回值），stream为写入文件的流;成功返回写入的对象个数，若到达文件尾或出错返回低于nmemb的数量或0;

int printf(const char \*format, ...);//写入到标准输出

int fprintf(FILE \*stream, const char \*format, ...);//写入指定流

int dprintf(int fd, const char \*format, ...);//写入指定文件描述符

若成功则返回输出字符数，若输出出错，则返回负值;

int sprintf(char \*str, const char \*format, ...);

实现atoi()相反的功能，将多种格式转化为字符串;成功则返回存入数组的字符数，出错则返回负值;可能造成缓冲区溢出;

int snprintf(char \*str, size\_t size, const char \*format, ...);

超过缓冲区大小size-1的所有字符将被丢弃，若缓冲区足够大，返回存入数组的字符数，出错则返回负值;

int scanf(const char \*format, ...);//从标准输入读取

int fscanf(FILE \*stream, const char \*format, ...);//从指定流读取

int sscanf(const char \*str, const char \*format, ...);//从指定文件描述符读取

仅有POSIX.1具有fdopen和sscanf等读写文件描述符的函数

定位流

int fseek(FILE \*stream, long offset, int whence);

offset的大小为正负2g，若成功返回0,若出错返回-1;offset为以字节为单位的偏移量，whence有SEEK\_SET、SEEK\_CUR、SEEK\_END三种;

long ftell(FILE \*stream);

文件大小必须小于2g，若成功返回当前文件位置指示，若出错返回-1L;

void rewind(FILE \*stream);

可以将流设置到文件的起始位置;

int fgetpos(FILE \*stream, fpos\_t \*pos);

int fsetpos(FILE \*stream, const fpos\_t \*pos);

若成功返回0,若失败返回非0;由IOS C引入，fpos\_t可以突破2g文件的限制;

行缓冲：换行、满了、强制时刷新（如stdout，因为是终端）;

全缓冲：满了、强制时刷新（系统默认缓冲模式，终端设备除外）;

无缓冲：如stderr（需要立刻输出）;

int fflush(FILE \*stream);

强制刷新对应文件流的缓冲区;

setvbuf可以修改缓冲模式

临时文件

char \*tmpnam(char \*s);

若s为NULL则返回文件放在静态区，会被重写，若不是NULL则ptr是长度至少为L\_tmpnam个字符的数组，返回指向唯一路径名（通常位于tmp目录下，如/tmp/filej2s3Hm）的指针;

FILE \*tmpfile(void);

创建一个类型为w+b的二进制临时文件，在关闭该文件或程序结束时删除这种文件;