头文件：#include <sys/stat.h>　　#include <unistd.h>

定义函数：int stat(const char * file_name, struct stat *buf);

函数说明：stat()用来将参数file_name 所指的文件状态，复制到参数buf 所指的结构中。

下面是struct stat 内各参数的说明：

```
struct stat
{
    dev_t st_dev; //device 文件的设备编号
    ino_t st_ino; //inode 文件的i-node
    mode_t st_mode; //protection 文件的类型和存取的权限
    nlink_t st_nlink; //number of hard links 连到该文件的硬连接数目，刚建立的文件值为1.
    uid_t st_uid; //user ID of owner 文件所有者的用户识别码
    gid_t st_gid; //group ID of owner 文件所有者的组识别码
    dev_t st_rdev; //device type 若此文件为装置设备文件，则为其设备编号
    off_t st_size; //total size, in bytes 文件大小，以字节计算
    unsigned long st_blksize; //blocksize for filesystem I/O 文件系统的I/O 缓冲区大小.
    unsigned long st_blocks; //number of blocks allocated 占用文件区块的个数，每一区块大小为512 个字节.
    time_t st_atime; //time of lastaccess 文件最近一次被存取或被执行的时间，一般只有在用mknod、utime、read、write 与tructate 时改变.
    time_t st_mtime; //time of last modification 文件最后一次被修改的时间，一般只有在用mknod、utime 和write 时才会改变
    time_t st_ctime; //time of last change i-node 最近一次被更改的时间，此参数会在文件所有者、组、权限被更改时更新
};
```

先前所描述的st_mode 则定义了下列数种情况：

1、S_IFMT 0170000 文件类型的位遮罩

2、S_IFSOCK 0140000 scoket

3、S_IFLNK 0120000 符号连接

4、S_IFREG 0100000 一般文件

5、S_IFBLK 0060000 区块装置

6、S_IFDIR 0040000 目录

7、S_IFCHR 0020000 字符装置

8、S_IFIFO 0010000 先进先出

9、S_ISUID 04000 文件的 (set user-id on execution)位

10、S_ISGID 02000 文件的 (set group-id on execution)位

11、S_ISVTX 01000 文件的sticky 位

12、S_IRUSR (S_IREAD) 00400 文件所有者具可读取权限

13、S_IWUSR (S_IWRITE)00200 文件所有者具可写入权限

14、S_IXUSR (S_IEXEC) 00100 文件所有者具可执行权限

15、S_IRGRP 00040 用户组具可读取权限

16、S_IWGRP 00020 用户组具可写入权限

17、S_IXGRP 00010 用户组具可执行权限

18、S_IROTH 00004 其他用户具可读取权限

19、S_IWOTH 00002 其他用户具可写入权限

20、S_IXOTH 00001 其他用户具可执行权限上述的文件类型在 POSIX 中定义了检查这些类型的宏定义

21、S_ISLNK (st_mode) 判断是否为符号连接

22、S_ISREG (st_mode) 是否为一般文件

23、S_ISDIR (st_mode) 是否为目录

24、S_ISCHR (st_mode) 是否为字符装置文件

25、S_ISBLK (s3e) 是否为先进先出

26、S_ISSOCK (st_mode) 是否为socket 若一目录具有sticky 位 (S_ISVTX), 则表示在此目录下的文件只能被该文件所有者、此目录所有者或root 来删除或改名.

返回值：执行成功则返回0，失败返回-1，错误代码存于errno。

错误代码：

1、ENOENT 参数file_name 指定的文件不存在

2、ENOTDIR 路径中的目录存在但却非真正的目录

3、ELOOP 欲打开的文件有过多符号连接问题, 上限为16 符号连接

4、EFAULT 参数buf 为无效指针, 指向无法存在的内存空间

5、EACCESS 存取文件时被拒绝

6、ENOMEM 核心内存不足

7、ENAMETOOLONG 参数file_name 的路径名称太长

范例

```
#include <sys/stat.h>
#include <unistd.h>
main()
{
      struct stat buf;
      stat("/etc/passwd", &buf);
      printf("/etc/passwd file size = %d \n", buf.st_size);
}
```

执行：

```
/etc/passwd file size = 705
```

说明： 获取文件大小 我们可以使用 fseek ， ftell ， rewind 来实现，也可以使用 stat ，
使用 fseek 方式 移植性更好，但是性能 比不上 stat，stat 是 linux 函数