**Важно! В программе не должно быть Break и Continue**

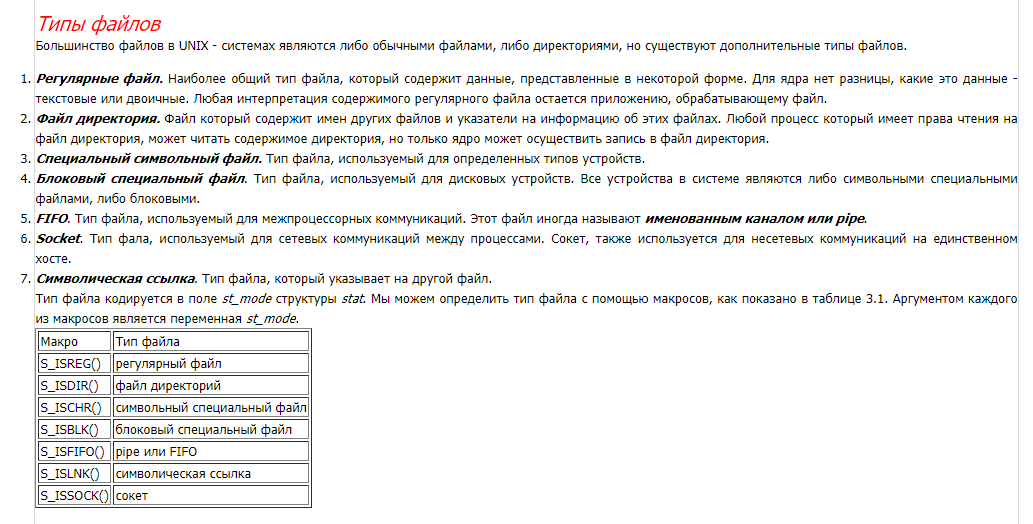
Opendir => dir

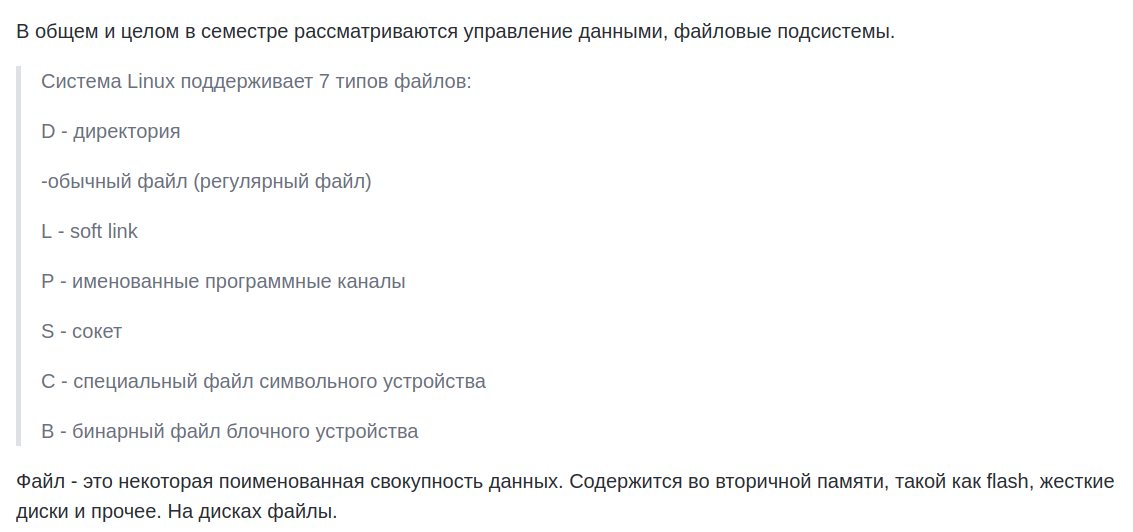
Readdir => dirent

* **Какие есть точки выхода из программы?**

**+глубина рекурсии**

* **Условие выхода рекурсии**
* **Типы файлов unix**







* **Stat, fstat, lstat + поля структуры stat**

int \_stat(

const char \*path,

struct \_stat \*buffer

);

Функция **\_stat** получает сведения о файле или каталоге, указанные по *пути* , и сохраняет их в структуре, на которую указывает *buffer*.

***путь***  
Указатель на строку, содержащую путь к существующему файлу или каталогу.

***буффер***  
Указатель на структуру, в которой хранятся результаты.

* **Что делает функция lstat?**
* **Чем отличаются stat и lstat? Почему мы используем именно lstat?**

Результатом вызова **stat() для символической ссылки будет информация о файле, на который эта ссылка указывает**. Это соответствует привычным для пользователя приемам работы, так как права на доступ к целевому файлу, а не к самой символической ссылке, управляют взаимодействием с этим файлом.

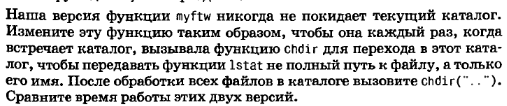
Некоторые приложения, такие как ls и программы для резервного копирования, должны быть способны отобразить информацию о самой символической ссылке, например, на какой файл она указывает. Подобного результата можно добиться, используя **lstat()** вместо stat(); этот способ пригодится на случай, **если надо работать с самой символической ссылкой, а не с файлом, на который она указывает.**

* **Для чего мы используем chdir?**

Функция chdir() устанавливает в качестве текущего каталог, на который указывает параметр path. Путь может включать в себя и спецификацию диска. Каталог должен существовать. В случае успеха функция chdir() возвращает 0.

При неудаче возвращается значение —1, а переменная errno устанавливается в ENOENT

В некоторых системах эта команда используется в качестве псевдонима для команды оболочки [**cd**](http://espressocode.top/cd-command-in-linux-with-examples/) .



* **Используете ли вы макрос S\_ISDIR? для чего?**

проверка на то, является ли файл каталогом

* **Что делает функция opendir?**

Функция opendir() открывает поток каталога и возвращает указатель на структуру типа DIR, которая содержит информацию о каталоге.

Функции opendir() и readdir() возвращают NULL, если возникла какая-то проблема, а в глобальную переменную errno записывается причина проблемы. Если readdir() возвращает NULL и errno равняется 0 (или, по-другому, EOK или ENOERROR), это значит, что в каталоге больше нет записей.

**Функция readdir() возвращает NULL, когда достигается конец каталога.**

Стандарт определяет только один необходимый элемент структуры struct dirent – массив элементов типа char с именем d\_name. Это имя элемента каталога в виде стандартной NUL-завершенной строки.

* **Что делает функция readdir? В какой момент мы выходим из цикла?**

Функция readdir() возвращает название следующего файла в каталоге. Иными словами, функция readdir() читает оглавление каталога по одному файлу за раз. Параметр ptr должен указывать на поток каталога, открытый с помощью opendir().

Структура dirent определена следующим образом:

struct dirent  
{  
char d\_name[13];

ino\_t d\_fileno

size\_t d\_namlen  
};

**Компонента имени файла** с нулевым символом в конце.

**Серийный номер файла**

**Длина имени файла**, исключая пустой символ завершения.

Таким образом, после вызова функции readdir() параметр d\_name содержит имя следующего файла в каталоге.

* **Почему пропускаем "." ".."?**

Нужно отметить, что каждый каталог содержит записи "." (указатель на сам каталог) и ".." (указатель на родительский каталог). В зависимости от поставленных задач обработку этих двух записей, возможно, придется пропустить.

Функция **readdir** возвращает также "." и "..". Если этого делать не нужно, то исключить эти значения можно следующим образом:

if (!(strcmp(dirp->d\_name, ".") == 0 || strcmp(dirp->d\_name, "..") == 0))

Также следует использовать strncmp(), чтобы игнорировать элементы ." и "..". Если нет необходимости пропускать обработку этих элементов, то **обработка каталогов будет выполняться бесконечно** (например, "theDir/./././././././././." и так далее).

stat, fstat, lstat - считывает статус файла

Вызовы **stat**() и **fstatat**() возвращают информацию о файле, указанном в *pathname*;

Вызов **lstat**() идентичен **stat**(), но в случае, если *pathname* является символьной ссылкой, то возвращается информация о самой ссылке, а не о файле, на который она указывает.

**stat** возвращает информацию о файле *file\_name* и заполняет буфер *buf*.

**lstat** идентична **stat**, но в случае символьных ссылок она возвращает информацию о самой ссылке, а не о файле, на который она указывает.

**fstat** идентична **stat**, только возвращается информация об открытом файле, на который указывает *filedes* (возвращаемый [**open**](https://www.opennet.ru/cgi-bin/opennet/man.cgi?topic=open&category=2)(2)), а не о *file\_name*.

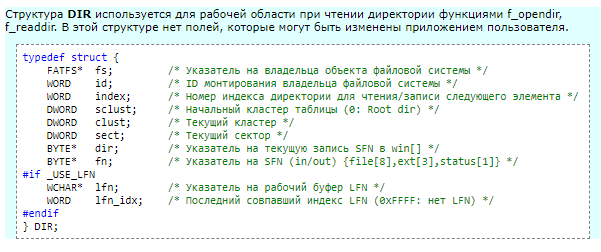
Функции из заголовочного файла dirent.h: opendir(), readdir() и closedir() – это то, что надо в подобной ситуации. Их применение очень схоже с использованием функций open/read/close при работе с файлами, но с одним исключением:

**функция readdir()**возвращает указатель на специальную структуру (тип **struct dirent**) для каждого элемента каталога.

Тип **данных DIR** представляет поток каталога.

Вы не должны когда-либо резервировать объекты struct dirent или типов данных DIR, так как функции доступа каталога делают это для Вас. Вместо этого, Вы обращаетесь к этим объектам, используя указатели, возвращенные следующими функциями.

Тип DIR обычно реализован, используя дескриптор файла



**-Тип данных: struct dirent**

Это - тип структуры, используемый, чтобы возвратить информацию относительно входов в каталог. Она содержит следующие поля:

char \*d\_name

Это - компонента имени файла с нулевым символом в конце. Это единственное поле, на которое Вы можете рассчитывать во всех POSIX системах.

ino\_t d\_fileno

Это - серийный номер файла. Для BSD совместимости, Вы можете также обратиться к этому элементу как к d\_ino.

size\_t d\_namlen

Это - длина имени файла, исключая пустой символ завершения.

Эта структура в будущем может содержать дополнительные элементы.

Когда файл имеет многочисленные имена, каждое имя имеет собственный вход в каталог. Единственный способ, которым Вы можете сообщать, что входы каталога принадлежат одиночному файлу - это, если они имеют то же самое значение для d\_fileno поля.

Стандарт POSIX 1003.1 определяет только один необходимый элемент структуры struct dirent – массив элементов типа char с именем d\_name. Это имя элемента каталога в виде стандартной NUL-завершенной строки.

Функция chdir() устанавливает в качестве текущего каталог, на который указывает параметр path. Путь может включать в себя и спецификацию диска. Каталог должен существовать. В случае успеха функция chdir() возвращает 0.

При неудаче возвращается значение —1

Функция opendir() открывает поток каталога и возвращает указатель на структуру типа DIR, которая содержит информацию о каталоге.

Их применение очень схоже с использованием функций open/read/close при работе с файлами, но с одним исключением: функция readdir() возвращает указатель на специальную структуру (тип struct dirent) для каждого элемента каталога.

Функция stat() вносит в структуру, на которую указывает statbuf, информацию, содержащуюся в файле, связанном с указателем filename. Структура stat определена в sys\stat.h.

При успешном заполнении структуры stat возвращается 0. В случае неудачи возвращается —1, а errno устанавливается в ENOENT.

**Структура stat{}**

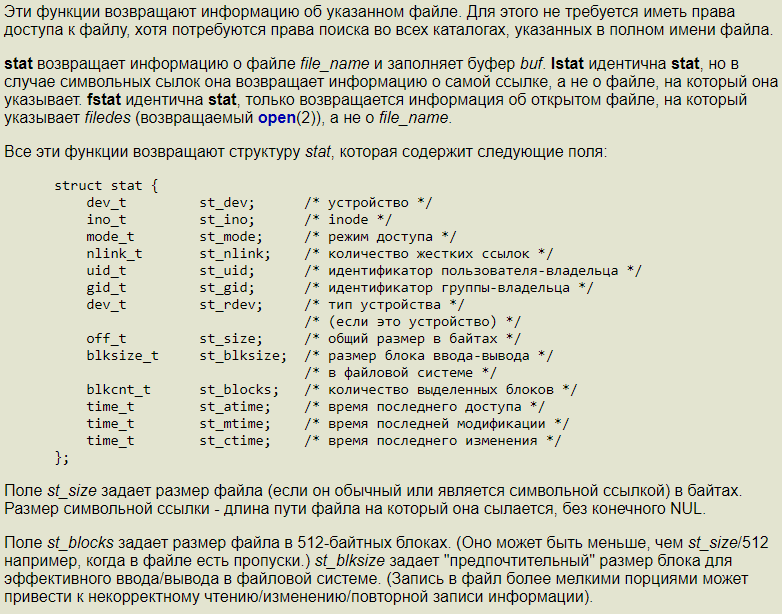
Функция stat() заполняет структуру struct stat информацией об определенном файле; если вместо имени файла имеется файловый дескриптор, то можно использовать его совместно с fstat(). Если также необходимо обнаруживать символические ссылки, то вместе с именем файла следует использовать lstat().

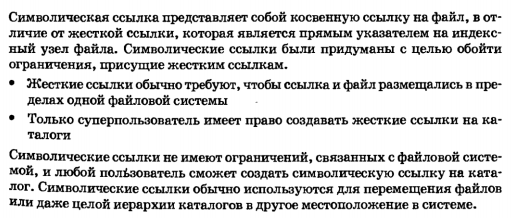
В отличие от struct dirent, которую возвращает readdir(), struct stat имеет довольно много обязательных стандартных полей:

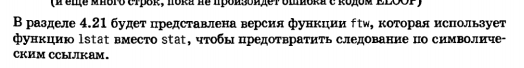
Используя макрос S\_\*() для поля st\_mode, можно определить тип файла:

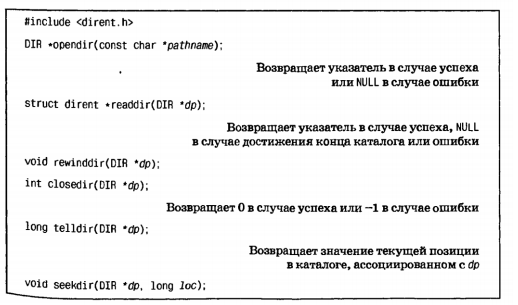
* S\_ISBLK(mode) – специальный блочный файл? (обычно это блочное устройство).
* S\_ISCHR(mode) – специальный символьный файл? (обычно это символьное устройство).
* S\_ISDIR(mode) – каталог?
* S\_ISFIFO(mode) – UNIX-канал (pipe) или файл типа FIFO?
* S\_ISLNK(mode) – символическая ссылка?
* S\_ISREG(mode) – обычный файл?

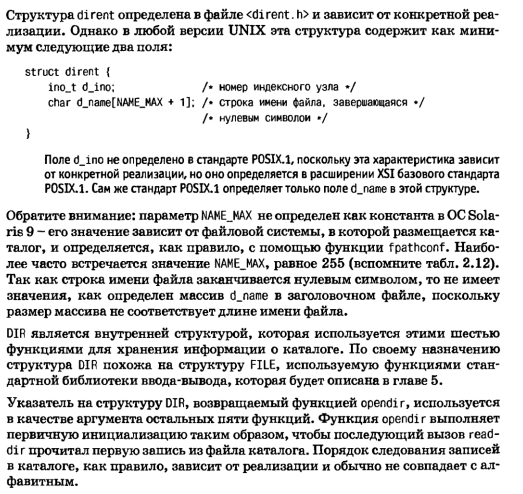
Функция stat() выполняется достаточно медленно на большинстве файловых систем, поэтому лучше будет хранить эту информацию в памяти на случай, если она понадобится позже.











Функция process\_directory() (которая начинает свою работу в [листинге 3](https://www.ibm.com/developerworks/ru/library/au-unix-readdir/#listing3) и заканчивает в [листинге 6](https://www.ibm.com/developerworks/ru/library/au-unix-readdir/#listing6)) просматривает заданный каталог и выводит некоторую информацию о каждом элементе содержимого. Указатель DIR, возвращаемый функцией opendir(), идентичен указателю на файл (FILE), который возвращает функция fopen(); это зависимый от типа ОС объект, который используется для прослеживания потока каталога, содержимое указателя DIR следует игнорировать.

Теперь, когда получено имя элемента каталога, для него надо собрать абсолютный путь ([листинг 5](https://www.ibm.com/developerworks/ru/library/au-unix-readdir/#listing5)), а затем вызвать функцию lstat() для получения информации об этом элементе. Поскольку символическим ссылкам требуется особая обработка, в данном случае следует использовать lstat(). Целевой файл символической ссылки можно узнать при помощи функции readlink().

Если элемент является каталогом, то для него выполняется рекурсивный вызов process\_directory() и к общему числу обнаруженных элементов добавляются вновь найденные. Если элемент является файлом, выводятся его имя и размер (размер берется из поля st\_size структуры struct stat).

В конце цикла while выполняются чтение и обработка еще одного элемента каталога ([листинг 6](https://www.ibm.com/developerworks/ru/library/au-unix-readdir/#listing5)). Если обработка элементов каталога закончена, то текущий рабочий каталог закрывается и возвращается число элементов, которые были обработаны.

