

1. Цель лабораторной работы

Цель лабораторной работы – закрепить знания о базовых командах ОС Unix и получить навыки работы с файлами и каталогами.

2. Основные действия пользователя при работе в ОС

"Обычному" пользователю для работы с любой реализацией операционной системы Unix из командной строки необходимо знать чуть более десятка команд.

Как правило, эти команды предназначены для обслуживания файлов и каталогов, а также отладки программ, написанных на основном языке высокого уровня операционной системы Unix – языке Си.

Перечислим те действия, которые необходимы «обычному» пользователю при работе с операционной системой:

- переход из каталога в каталог;
- просмотр содержимого каталогов;
- копирование, перемещение и уничтожение файлы;
- создание и уничтожение каталогов;
- просмотр содержимого файлов;
- создание новых файлов с помощью текстовых редакторов;
- просмотр файлов инструкций (файлов справочной системы Unix);
- поиск файлов;
- установка и изменение прав доступа к файлам;
- создание исполняемых файлов из исходных модулей на языке Си.

Таким образом, практически вся работа пользователя по созданию новых программ и обслуживанию своих каталогов сводится к работе с файловой системой. Эта работа может быть выполнена при помощи небольшого (базового) количества команд (10-15).

Функции и формат базового набора команд не очень сильно отличаются от аналогичных команд в других операционных системах.

3. Формат команд ОС Unix

Команды операционной системы Unix имеют следующий формат:

command [-f] [-a flag_parameter] [parameter]

Каждая команда состоит из одного или нескольких слов и начинается с названия – (**command**).

За названием могут следовать флаги, параметры флагов и параметры, которые называются аргументами.

Аргумент, состоящий из знака минус '-', за которым следует одна буква, называется **флагом**. Флаги обычно задают режим работы команды.

За некоторыми флагами могут следовать аргументы, относящиеся только к этому флагу (flag_parameter). Такие аргументы называются **параметрами флагов**.

Аргументы, задаваемые после последнего флага и параметры флага, называются **параметрами**. Обычно они задают объекты для обработки.

Все команды ОС Unix подчиняются следующим правилам (им следуют разработчики новых команд):

- 1) длина имени команды должна составлять от двух до девяти символов;
- 2) имя команды записывается только строчными (малыми) буквами и цифрами;
- 3) флаги должны состоять из одного символа;

- 4) флаги разделяются символом '-';
- 5) флаги без параметров флагов могут группироваться за одним разделителем '-';
- 6) все флаги должны быть перечислены до параметров в командной строке;
- 7) порядок флагов не является существенным.

4. Порождение имён файлов

Интерпретатор командного языка производит поиск метасимволов '*', '?' и '[' в каждом слове команды. Если в слове найден хотя бы один из этих метасимволов, то слово считается шаблоном имени файла. Интерпретатор командного языка ищет в указанном в команде каталоге имена файлов, удовлетворяющие этому шаблону, и заменяет исходное слово лексикографически упорядоченной последовательностью найденных имен, разделенных пробелами.

Правила подстановки:

- * любая последовательность символов, в том числе, пустая;
- ? любой одиночный символ;
- [abc] соответствует любому одиночному символу из тех, которые указаны в квадратных скобках;
- [a-d] соответствует одиночному символу, код которого попадает в диапазон между кодами указанных символов, включая их самих.

5. Базовый пользовательский набор команд

5.1. Получение справки о команде (команда **man**)

Получить справку о любой команде операционной системы Unix или найти нужную команду можно с помощью команды **man** (от слова manual - руководство, инструкция).

Синтаксис:

```
man <имя_команды>
man [-k] <контекст>
```

Первый вариант команды **man** используется для получения **справки** о команде, имя которой указывается в качестве параметра:

Пример_1: man cd

Эта команда выводит инструкцию по команде **cd**.

Управление просмотром инструкции осуществляется с помощью односимвольных команд, которые не отображаются на экране дисплея:

- **d** – вперед на половину экрана;
- **Пробел** – вперед на экран;
- **b** – возврат на один экран назад;

Примечание. Для получения справки о работе команды **man** используйте строку: **man man**.

Второй вариант команды **man** используется тогда, когда имя нужной команды не известно. Для поиска команды (команд), предназначенных для выполнения нужных функций

необходимо после ключа `-k` указать слово (контекст), характеризующее команду и которое будет отыскиваться в кратких описаниях команд.

Примечание. Единственное неудобство этого, безусловно, полезного варианта команды `man` состоит в том, что ключевое слово необходимо указывать на английском языке.

Пример 2: `man -k file`

Эта команда позволяет найти команды, в кратком описании которых встречается ключевое слово `file`, т.е. команды, предназначенные для разнообразной работе с файлами (рис. 1).

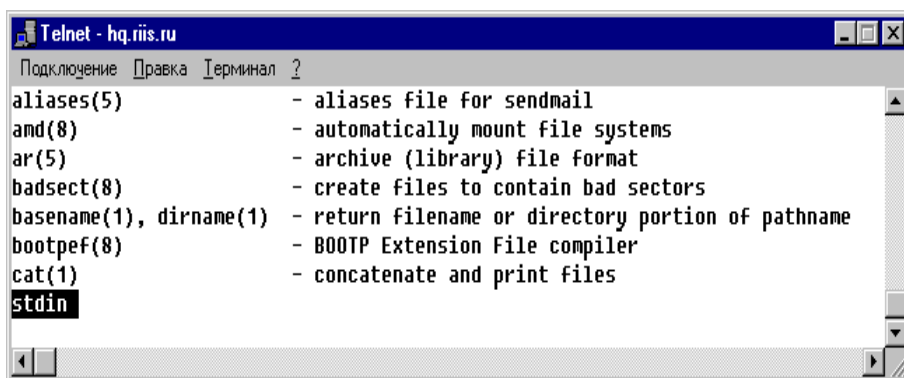


Рис. 1. Фрагмент вывода команды “`man -k file`”.

Вы видите, что полученный список команд содержит информацию о группе объектов, в названии которых (правая колонка) встречается сочетание символов “file”.

Этот режим работы команды `man` весьма эффективен и для изучения набора команд Unix.

Упражнение. Попробуйте получить сведения о командах для обслуживания каталогов (ключевое слово “catalog”) и терминала (“terminal”).

5.2. Переход из каталога в каталог (команда `cd`)

Для перехода в другой каталог служит команда `cd` (change directory – сменить каталог).

Синтаксис:

`cd [имя_каталога]`

где: `имя_каталога` – путь к новому каталогу.

Команда `cd` объявляет **текущим** каталог, указанный аргументом `имя_каталога`.

Примечание. Без аргумента команда `cd` назначает текущим **домашний** (начальный) каталог пользователя.

На рисунке 2 приведен фрагмент файловой системы, который будем использовать далее в примерах. Звездочкой (*) отмечен начальный (домашний) каталог.

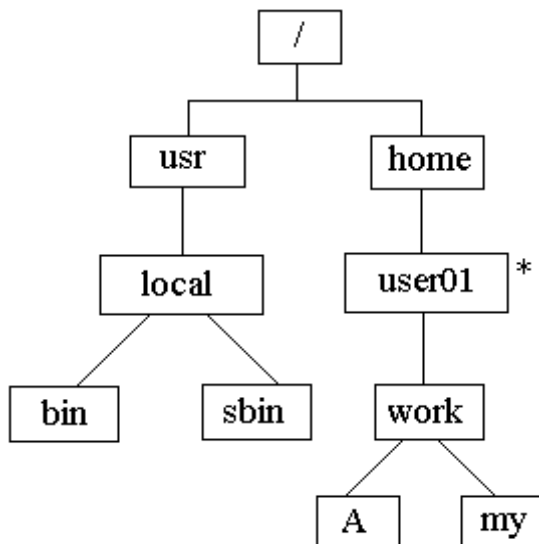


Рис. 2. Фрагмент файловой системы.

Пример_1: `cd /usr/local/bin`

Переход из **текущего** каталога (помечен звездочкой) в каталог `/usr/local/bin`. В качестве аргумента `dir` указано **полное имя каталога** (путь от корня файловой системы к нужному каталогу).

Пример_2: `% cd`

Возврат в домашний каталог (**user01**) из произвольного каталога.

Пример_3: `cd work`

Переход к каталогу **work**, находящемуся в текущем каталоге.

В каждом каталоге существуют две специальные ссылки (псевдонимы) на текущий и родительский (каталог, в составе которого находится текущий каталог) каталоги. Эти ссылки заданы, соответственно, именами `.` и `..`.

Для перехода в родительский каталог можно ввести команду `cd ..`, а для перехода «на два этажа выше» - `cd ../../`.

Пример_4: `cd my`
`cd ../A`

Переход из каталога `work` в каталог `my` (первая команда), а затем – переход из каталога `my` в каталог `A`, находящийся на том же уровне, что и каталог `my`, с помощью псевдонима родительского каталога (`..`).

5.3. Определение имени текущего каталога (команда `pwd`)

При работе с операционной системой из командной строки достаточно трудно ориентироваться в файловой системе. В ориентировке помогает команда **pwd**, выводящая в стандартный вывод абсолютное (полное) имя текущего каталога.

Синтаксис:

`pwd`

Пример_1: `cd work`
`pwd`
`/home/user01/work` *#результат работы команды pwd*

Первая команда осуществляет переход из **текущего** каталога (user01) в каталог work и делает этот каталог текущим. Вторая команда выводит **полный** путь к новому текущему каталогу.

5.4. Просмотр содержимого каталогов (команда ls)

Синтаксис:

`ls [ключи] [имя_файла | имя_каталога] ...`

Для каждого заданного аргумента команда ls выводит:

- содержимое (оглавление) указанного каталога, если аргумент является именем каталога;
- имя файла и другую требуемую информацию, если аргумент является именем файла.

По умолчанию результат работы команды ls сортируется в алфавитном порядке.

Если аргумент не задан, то выводится содержимое **текущего каталога**.

Если задано несколько аргументов, то аргументы сначала сортируются соответствующим образом, причем файлы выводятся перед содержимым каталогов.

Примечание. В простейшем случае (без ключей и аргументов) командой ls выводятся только **имена** файлов и каталогов текущего каталога без каких-либо пометок и дополнительной информации (рис. 5).

Для управления форматом вывода служат многочисленные ключи команды.

Таблица 1. Параметры команды ls.

№	Ключ	Назначение
1.	-l (цифра 1)	Вывод информации о каждом следующем файле или каталоге в отдельной строке.
2.	-a	Вывод списка всех файлов и подкаталогов в каталоге, включая скрытые файлы (их имена начинаются с точки).
3.	-c	Сортировка по дате создания.
4.	-F	Пометить исполняемые файлы звездочкой (*), каталоги – наклонной чертой (/) и символические ссылки – символом @.
5.	-r	Вывод в обратном порядке.
6.	-R	Рекурсивная работа.
7.	-l (“эль малое”)	Длинный (наиболее полный) формат вывода.
8.	-u	Сортировка по дате последнего доступа к файлу.

Наиболее полный формат вывода обеспечивается при указании ключа **-l** (английская буква “эль малое”).

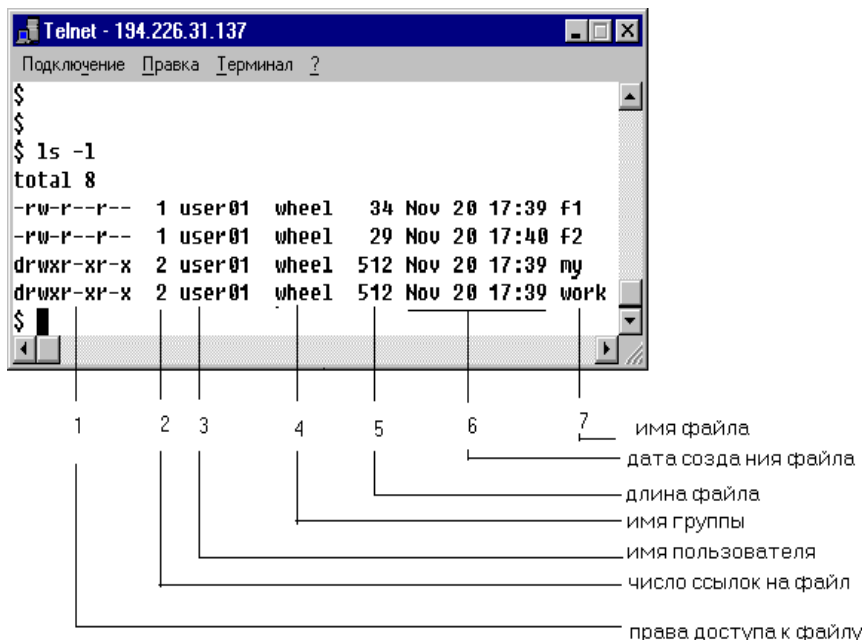


Рис. 3. Результат работы команды ls с ключом -l.

Каждый элемент вывода в этом формате занимает одну строку. Рассмотрим поля строк в “длинном” выводе команды ls.

- 1) В первом поле в виде односимвольных индикаторов указаны **тип файла** и **права доступа** к нему для трех категорий пользователей, определенных в операционной системе Unix.

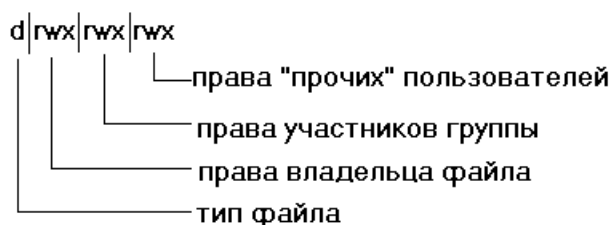


Рис.4. Поле №1 – тип файла и права доступа к файлу.

Примечание. Для удобства рассмотрения этого поля группы индикаторов разделены вертикальными линиями.

Тип файла (табл. 2) указывается с помощью следующих односимвольных индикаторов, размещаемых в **первой** позиции поля:

Таблица 2. Типы файлов в ОС Unix.

-	обычный файл;
d	каталог;
b	блок-ориентированный специальный файл; предназначен для работы с устройствами, обмен с которыми производится блоками, например, жесткие и гибкие диски;
c	символьно-ориентированный специальный файл;

	предназначен для работы с устройствами, обмен с которыми производится по-символьно блоками, например, терминал, печатающее устройство;
p	FIFO-файл или именованный программный канал
l	связь (link) – указатель на другой файл;
s	контакт (socket) – точка доступа процессов к сервису транспортного протокола TCP/IP.

Права доступа к файлу в ОС Unix определяются для трех категорий пользователей:

- владельца файла;
- пользователя, входящего в какую-либо группу владельцев файла;
- для всех остальных пользователей.

Права доступа к файлу для всех категорий пользователей указываются с помощью следующих односимвольных индикаторов:

r	файл доступен для чтения
w	файл доступен для записи
x	файл доступен для выполнения
-	право доступа отсутствует

Каждый из индикаторов указывается строго на своем месте – сначала r, затем w и x:

|**rw**x|

Если какой-либо из режимов доступа отсутствует, на его месте проставляется индикатор '-':

|**r**-**x**|

Этот файл можно читать и выполнять; запись в него запрещена.

- 2). Во втором поле «длинного» вывода команды ls указано число ссылок на данный файл из других каталогов.
- 3). В третьем поле задано регистрационное имя владельца файла.
- 4). Четвертое поле хранит имя группы владельцев файла.
- 5). Пятое поле – длина файла в байтах.
- 6). В шестом поле содержится дата создания файла (месяц, день, минуты, секунды).
- 7). Седьмое поле содержит имя файла.

Приведем несколько примеров вывода оглавления каталога с помощью команды ls.

Пример 1: ls

Вывод только имен объектов каталога без указания типов файлов. Вывод имен производится в 6 колонок.

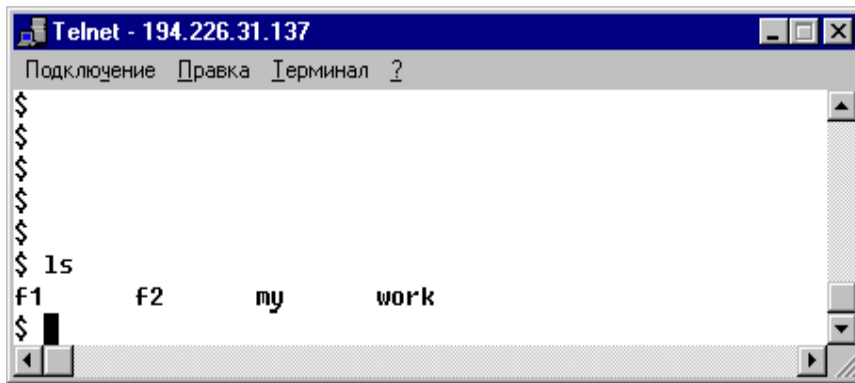


Рис. 5. Результат работы команды ls без ключей.

Пример_2: ls -la

Ключ -a позволяет вывести все имена, в том числе и начинающиеся с **точки** ('.').

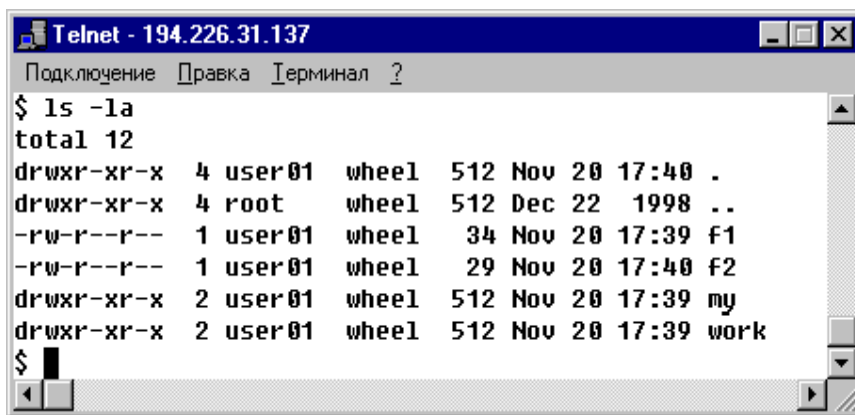


Рис. 6. Результат работы команды ls -la.

Примечание. Кроме имен текущего ('.') и родительского ('..') каталогов с точки начинаются имена служебных (настроечных) файлов (например, '.profile').

Пример_3: ls -F

Вывод в сокращенном формате (к имени каталога добавляется символ '/').

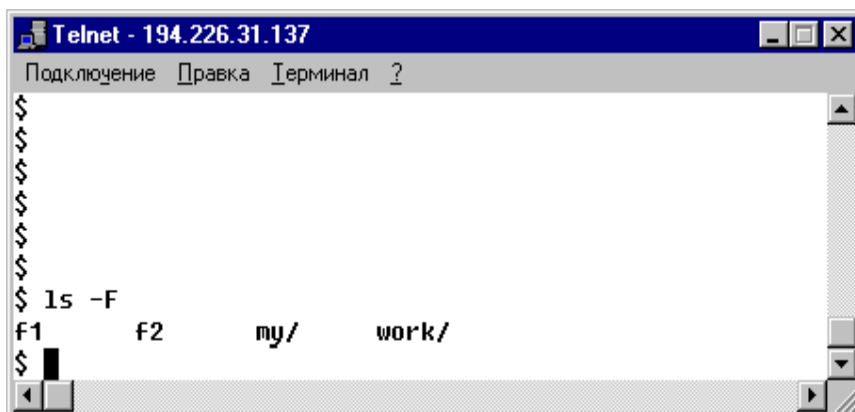


Рис. 7. Результат работы команды ls -F.

Упражнения. Выведите оглавление текущего каталога и каталога /bin в следующих вариантах:

- сокращенный вариант;
- сокращенный вариант (помечены типы файлов);
- полный (длинный) вариант;
- полный вариант и имена, начинающиеся с точки;
- полный вид с сортировкой: по именам, по дате создания файлов, по дате последнего доступа к файлу.

5.5. Создание каталога (команда **mkdir**)

Синтаксис:

mkdir *имя_каталога...*

Команда **mkdir** создает новый каталог с именем *имя_каталога*. Вновь созданный каталог будет автоматически содержать элемент *'.'* для созданного каталога и элемент *'..'* для каталога, являющегося родительским данного каталога.

Примечание. Команда **mkdir** требует наличие у пользователя, ее выполняющего, права записи в родительский каталог создаваемого каталога.

Упражнение. Находясь в текущем каталоге (user01), создать в каталоге work каталог **lab1**.

5.6. Удаление каталога (команда **rmdir**)

Синтаксис:

rmdir *имя_каталога*

Команда **rmdir** предназначена для удаления **пустых** (содержащих только имена *'.'* и *'..'*) каталогов.

Примечание. Для удаления **не пустых** каталогов используйте команду **rm** с ключом **-r**.

5.7. Копирование файлов и каталогов (команда **cp**)

Синтаксис:

cp [-R] [-f | -i] [-pv] *исходный_файл* *целевой_файл*

cp [-R] [-f | -i] [-pv] *исходный_файл* *целевой_каталог*

Существуют два вида команды **cp**.

В **первом** случае *исходный_файл* копируется в *целевой_файл*.

Во **втором** случае *целевой_каталог* - это имя **каталога**, в который будет скопирован файл, заданный аргументом *исходный_файл*.

Таблица 3. Основные ключи команды **cp**.

№	Ключ	Назначение
1.	-R	Рекурсивная работа. Если <i>исходный_файл</i> является каталогом, то команда cp копирует каталог и все поддерево.
2.	-f	Этот ключ заставляет команду cp удалять существующий файл без информирования пользователя.
3.	-i	Ключ -i задает интерактивный режим работы команды и требует от пользователя подтверждения на удаление уже

		существующего целевого файла.
4.	-p	Сохранить файл с имеющимися у него атрибутами (права доступа, время создания, идентификатор пользователя и т.п.).
5.	-v	Выводить в стандартный вывод имена копируемых файлов.

Пример_1: `cp /home/work/p.c .`

Файл `p.c` из каталога **/home/work** копируется в текущий каталог (`.`) под существующим именем.

Упражнение. Скопировать файлы `a.out` и `p.c` из каталога `/home/work` в каталог `my` и затем скопировать каталог `my` в каталог `lab2` (см. рис. 8.)

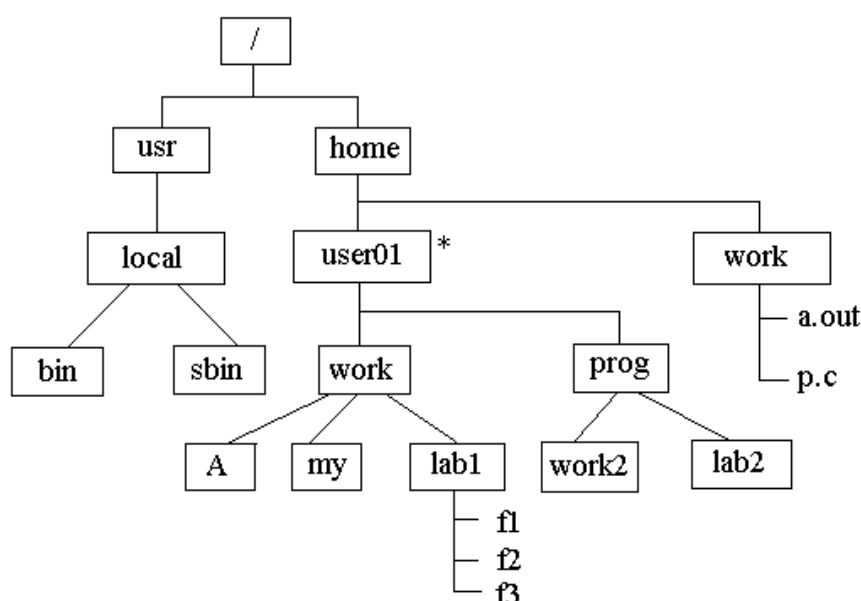


Рис. 8. Фрагмент файловой системы.

5.8. Перемещение (или переименование) файлов (команда `mv`)

Синтаксис:

`mv [-f | -i] [-v] исходный_файл целевой_файл`

`mv [-f | -i] [-v] исходный_файл целевой_каталог`

Команда **mv** переименовывает *исходный_файл* в *целевой_файл*. Если *целевой_файл* уже существует, то он уничтожается перед тем, как будет переименован *исходный_файл*.

Второй вид команды **mv** позволяет переместить один или более *исходных_файлов* в каталог с именем *целевой_каталог* с сохранением их локальных имен.

Таблица 4. Основные ключи команды **mv**.

№	Ключ	Назначение
1.	-f	Этот ключ заставляет команду mv удалять существующий файл без информирования пользователя.

2.	-i	Ключ <code>-i</code> задает интерактивный режим работы команды и требует от пользователя подтверждения на удаление уже существующего целевого файла.
3.	-v	Выводить в стандартный вывод имена копируемых файлов.

Пример_1: `cd`
`mv work/my/p.c .`

Переход в домашний каталог и перемещение файла `p.c` из каталога `my` в домашний каталог (рис. 9).

Упражнение. 1). Переименовать файл `p.c` в файл `prog.c` в каталоге `lab2`. 2). Переместить каталог `my` в каталог `work2`.

5.9. Удаление файлов (команда **rm**)

Синтаксис:

`rm [-dfiPRrvW] файл1 файл2 ...`

Команда **rm** удаляет из каталога файлы, имена которых заданы аргументами *файл1* ... *файл2*.

Внимание! Использовать флаг `-r` и метасимволы (шаблоны) имен файлов необходимо очень осторожно. Например, под шаблон `.*` подходит каталог `“..”` (родительский каталог). Поэтому **никогда** не следует выдавать команду `rm -r .*`

Таблица 5. Основные ключи команды **rm**.

№	Ключ	Назначение
1.	-d	
2.	-f	Этот ключ заставляет команду rm удалять существующий файл без информирования пользователя.
3.	-i	Ключ <code>-i</code> задает интерактивный режим работы команды и требует от пользователя подтверждения на удаление уже существующего целевого файла.
4.	-R	Рекурсивное удаление файлов и каталогов, начиная с указанной точки.
5.	-v	Выводить в стандартный вывод имена копируемых файлов.

5.10. Просмотр содержимого файлов (команды **cat** и **more**)

1) команда **cat** - конкатенация (склейка) файлов

Синтаксис:

`cat файл1 файл2 ...`

Команда **cat** последовательно считывает содержимое файлов, имена которых задано аргументами *файл1*, *файл2*, ... и выводит их содержимое в стандартный вывод.

Если имя не указано или в качестве одного из аргументов задан минус '-', команда **cat** считывает данные из стандартного ввода.

Пример_1: cat f1 f2 f3

В стандартный вывод последовательно и без остановки выводятся файлы f1, f2, f3, находящиеся в текущем каталоге.

2) команда **more** - вывести файл на экран порциями

Синтаксис:

more *файл1 файл2 ...*

Команда **more** (в отличие от команды **cat**) позволяет просматривать файлы, имена которых заданы аргументами *файл1 файл2 ...*, по страницам.

Работой команды **more** можно управлять с помощью односимвольных команд, которые **не** отображаются на экране дисплея (Таблица 6).

Таблица 6. Команды управления просмотром команды **more**.

Команда	Действие
ПРОБЕЛ (клавиша)	показать следующую страницу
ENTER (клавиша)	сдвинуть просматриваемый файл на экране на 1 строку вверх
i<ПРОБЕЛ>	вывод i строк (Ввести цифру и нажать клавишу ПРОБЕЛ)
d	вывести 11 строк (половина экрана)
=	показать № строки
b	назад на половину экрана
q	завершить просмотр

Упражнение. Создать в каталоге lab1 три текстовых файла: f1, f2, f3. Вывести в стандартный вывод созданные файлы с помощью команд **cat** и **more**.

5.11. Установка и изменение прав доступа к файлам

Синтаксис:

chmod *атрибуты_защиты файл ...*

Команда **chmod** изменяет **атрибуты защиты** (права доступа, режим доступа к файлу) заданного файла (файлов) в соответствии с вновь заданными в команде.

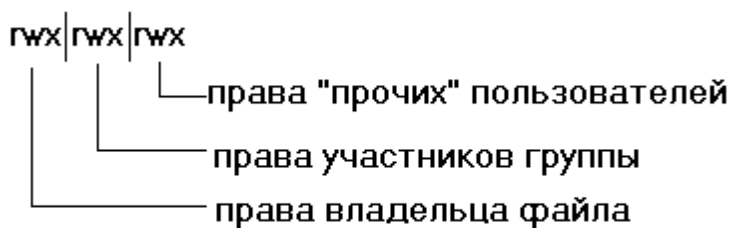


Рис. 9. Права доступа к файлу.

Атрибуты защиты могут быть заданы в **числовом** или **символьном** виде.

а). В **числовом виде** атрибуты защиты задаются в виде восьмеричного числа, схема формирования которого приведена на рисунке 10:

rw-	r--	r--	устанавливаемые права доступа
110	100	100	двоичная маска прав доступа
6	4	4	восьмеричное представление маски (параметр mode)

Рис. 10. Схема формирования атрибутов защиты файла в числовом виде.

Сначала записываются права доступа в символьном виде (строка 1), затем формируется двоичная маска (строка 2), указывающая наличие права на чтение, запись или выполнение файла (для каждой категории пользователей). Наличие права обозначается цифрой '1', отсутствие - цифрой '0'. Далее для каждой категории пользователей полученное двоичное число записывается в виде 8-ричной цифры (строка 3).

Итоговое 8-ричное число используется в качестве параметра mode:

```
chmod 644 fl
```

В результате выполнения этой команды будут установлены следующие права доступа к файлу fl: "rw-r--r--".

б) В **символическом виде** атрибуты защиты файла задаются в следующем формате:

[<пользователь>] <операция> [<атрибуты_защиты>]

Пользователь обозначается одним из следующих символов:

u	владелец файла
g	участник группы
o	прочие (остальные) пользователи
a	все категории пользователей

Если пользователь **не указан**, по умолчанию используется значение **a** (все пользователи).

Операция:

- = определение (назначение) перечисленных прав доступа и отмена всех прочих для данного Пользователя. Если новые права не определены, отменяются существующие.
- отмена прав доступа
- + добавление прав доступа

Атрибуты защиты:

- r разрешение чтения
- w разрешение записи
- x разрешение выполнения

Пример_1: `chmod a=r,u+w f1`

Примечание. Обратите внимание на отсутствие пробелов внутри поля «операция».

Всем пользователям разрешено читать файл (`a=r`), **владельцу** разрешена запись в файл (`u+w`). Эта команда эквивалентна команде `chmod 644 f1`.

Пример_2: `chmod u=rw,go=r f1`

Ещё один вариант команды `chmod` из Примера_1.

Пример_3: `chmod o-r f1`

Прочим (остальным) пользователям запрещается читать файл `f1`.

Упражнение. Установить для файлов из каталога `lab1` следующие атрибуты защиты: для файла `f1` – всем категориям пользователей разрешается читать и писать; для файла `f2` – владельцу и прочим пользователям разрешается писать и читать; для файла `f3` – разрешается читать файл только владельцу.

5.12. Поиск файлов (команда `find`)

Синтаксис:

`find список_имен выражение`

Команда `find` рекурсивно сканирует все каталоги и подкаталоги, определенные "`список_имен`" (одно и более имен каталогов), в поисках файлов, соответствующих булевскому выражению "`выражение`".

В последующих описаниях аргумент **n** используется как десятичное целое, причем:

- +n означает "больше чем n";
- n означает "меньше чем n".

Аргумент "**выражение**" конструируется из следующих атомарных выражений:

- name file** задает файл для поиска. Могут использоваться метасимволы командного языка.
- type c** истинно, если тип текущего файла совпадает с "c";
"c" может принимать значения:
b - блок-ориентированный файл;
c - байт-ориентированный файл;
d - каталог;
f - обычный файл;
p - FIFO-файл (именованный программный канал);

Примеры:

-type f
-type d

-size n[c] истинно, если размер текущего файла равен n блокам (блок - 512 байтов). Если после числа указано "с", размер задается в байтах.

Пример: -size +500 файл размером больше чем 500*512 байтов

-atime n истинно, если был доступ к текущему файлу в последние n дней;

Примечание. Время доступа к каталогу изменяется в ходе выполнения команды find.

-mtime n истинно, если в последние n дней осуществлялась модификация текущего файла;

Пример:

-mtime -3 файл модифицировался **менее** 3-х дней назад

-ctime n истинно, если в последние n дней осуществлялась модификация атрибутов файла;

-newer file истинно, если текущий файл модифицировался позднее файла, указанного в аргументе file.

В команде find используются также следующие **ключи-действия** (другой тип атомарных выражений):

-print распечатать полное имя файла;

-exec command {} \; выполнить команду над файлом

Здесь: command - команда Unix;

{ } - заменяется именем найденного файла;

подстрока command всегда завершается конструкцией "\;"

Атомарные выражения могут комбинироваться с использованием следующих логических операторов:

! отрицание;

-o логическое сложение ИЛИ (операция OR);

-a логическое умножение И (операция AND).

Примечание. Примеры применения команды **find** приведены в **Приложении 1**.

Упражнения:

1). Найти в поддереве домашнего каталога:

- файлы, начинающиеся с 'f';
- файлы, имеющие суффикс 'с'.

2). Найти в поддереве домашнего каталога файл a.out и удалить его.

3). В каталоге /bin найти файлы объемом более 500 Кбайт.

6. Дополнительные возможности интерпретатора команд

Перенаправление ввода-вывода. Конвейеры. Фоновые команды. Группирование команд.

6.1. Перенаправление ввода-вывода

Когда в Unix начинает работать команда или программа, автоматически открываются три потока ввода-вывода:

- стандартный ввод;
- стандартный вывод;
- стандартный протокол (стандартное устройство для вывода сообщений об ошибках).

Многие команды берут исходные данные из стандартного ввода и печатают результаты в стандартный вывод.

Первоначально в качестве устройства стандартного ввода используется клавиатура терминала, а стандартный вывод и стандартный протокол назначены на экран терминала.

Стандартный интерпретатор команд (sh) позволяет изменять первоначально установленные направления ввода-вывода.

Для изменения направления вывода используется операция ‘>’ (знак «больше»):

команда > имя_файла.

Конструкция ‘>>’ позволяет дописать в существующий файл (иначе файл усекается до 0 и запись в него ведется с начала).

Для изменения направления ввода следует использовать операцию ‘<’:

команда < имя_файла

И, наконец, для изменения направления вывода для стандартного протокола применяют конструкцию ‘2>’:

команда 2> имя_файла

Здесь цифра 2 указывает дескриптор файла стандартного протокола.

Примечание. Файловая система Unix позволяет унифицировать передачу данных в файлы, на устройства и при обмене данными между процессами. Поэтому выражения «стандартный ввод», «устройство стандартного ввода» «файл стандартного ввода» можно считать синонимами.

Пример_1: ls -l > dir1

Результат работы команды ls -l перенаправляется в файл dir1 (в «обычном» режиме результаты выводятся в стандартный вывод – на экран дисплея).

Пример_2: cat file1 file2 > file3

Файлы file1 и file2 «склеиваются» и результат записывается в file3.

Пример_3: Создание файла с помощью команды cat.

cat > newfile

текст

текст

.....

текст

<Ctrl/D>

Команда `cat`, если не указан аргумент (входной файл), вводит данные со стандартного ввода. Введенные данные перенаправляются в файл `newfile`. Завершается работа команды `cat` при вводе управляющего кода EOF (End of File – конец файла) – одновременное нажатие клавиш `Ctrl` и `D`.

Пример_4: `run 2> /dev/null`

Запущена команда `run`. Диагностические сообщения (выводимые в стандартный протокол) перенаправляются на «нулевое» устройство с целью их уничтожения. Специальный файл (устройство) `/dev/null` представляет собой своего рода «черную дыру», способную поглотить любое количество данных, направляемых на него.

Пример_5: `find / -name *r* -print 2> diag`

Диагностические сообщения команды `find` (сообщение о том, что просматриваемые каталоги закрыты для чтения) будут направлены в файл `diag`, а не на экран дисплея.

Используя цифровые дескрипторы (0, 1, 2, ...) открытых потоков данных можно конструировать более сложные перенаправления потоков данных.

Пример_6: `run_program > /dev/null 2>&1`

Выполняется команда `run_program`. Стандартный вывод направлен в нулевое устройство (конструкция `> /dev/null`). Следующая конструкция `2>&1` является указанием интерпретатору, что стандартный поток диагностики (2>) нужно поместить в тот же поток, что и стандартный выводной (обозначен с помощью дескриптора 1).

Амперсанд (&) не содержит какого-либо мнемонического смысла; это просто идиома (для того чтобы отличить операнд справа от `>` от имени файла).

Для добавления стандартного выводного потока к стандартному потоку диагностики можно использовать конструкцию `1>&2`.

6.2. Конвейеры

Интерпретатор командного языка Unix позволяет перенаправлять стандартный вывод одной команды на стандартный ввод другой. Для этого служит операция `|` (вертикальная черта).

Примечания. 1). Команды или программы, которые вводят данные со *стандартного ввода*, а выводят на *стандартный вывод* – называются фильтрами. 2). Синхронизация работы фильтров в конвейере производится ядром ОС Unix.

Пример_1: Подсчитать количество файлов в текущем каталоге.

а) «обычный» вариант:

ls -l > file	направляет результаты работы команды <code>ls</code> в файл <code>file</code>
wc < file	команда <code>wc</code> (word count – счетчик слов) подсчитывает число строк, слов и символов во входном потоке. Данные во входной поток команды <code>wc</code> перенаправлены из файла <code>file</code>
rm file	удаляется рабочий файл <code>file</code> .

б) вариант с конвейером:

`ls -l | wc`

Как только команда `ls` начинает выдавать результаты, они сразу поступают на вход фильтра `wc`. Никакого рабочего промежуточного файла не требуется.

Пример_2: `ls -l | more`

применение фильтра `more` позволяет просматривать большие каталоги.

Пример_3: `ls | grep old`

распечатать из вывода команды `ls` только те строки, которые содержат цепочку символов “old” (контекстный поиск этой цепочки осуществляет фильтр `grep`).

Пример_4: `ls | grep old | wc`

подсчитать количество файлов в текущем каталоге, имена которых содержат цепочку символов “old”.

6.3. Фоновые команды

При работе с операционной системой в режиме командной строки пользователь после подсказки операционной системы (`%` или `$`) вводит очередную команду и дожидается завершения работы команды. Затем после подсказки операционной системы вводится следующая команда.

В ОС Unix можно указать выполнение команды в «фоновом» режиме. Для этого необходимо после команды через пробел набрать символ ‘&’.

«Попросим» компилятор языка Си изготовить исполняемый файл в фоновом режиме.

<code>cc prog.c &</code>	запуск программы <code>cc</code> в фоновом режиме
<code>308</code>	№ процесса, который выполняет процесс <code>cc</code> (выводится для контроля за его работой)
<code>%</code>	подсказка интерпретатора команд

Теперь можно вводить новую команду (в то время как программа выполняется в фоновом режиме).

Увидеть информацию о выполняющихся процессах можно по команде `ps`.

	<code>ps</code>	
<code>PID</code>	<code>TTY</code>	<code>CMD</code> # заголовок таблицы выполняющихся процессов
<code>308</code>		<code>cc prog.c</code>

В колонке `PID` указан № процесса, а в колонке `CMD` – выполняемая команда.

Если фоновая команда выполняется очень долго (например, компиляция небольших программ на языке Си длится меньше минуты), остановить выполнение фоновой команды можно с помощью команды `kill`:

`kill 308` параметр `308` задает № процесса, который должен быть завершен.

6.4. Группирование команд

Если последовательность команд, которую вы собираетесь ввести, вам хорошо известна, можно ввести их в одной строке, разделяя точкой с запятой:

`ls -l > file; wc < file; rm file`

Для группирования команд используются и другие способы. Один из них – применение круглых скобок:

```
(cd /usr/a; cat a.c)
```

Здесь запускается субинтерпретатор `sh`, затем выполняются команды в скобках и по закрывающей круглой скобке происходит возврат в текущий каталог.

7. Создание файлов с помощью текстовых редакторов

7.1. Редактор vi

Программа `vi`, разработанная в Калифорнийском Университете для BSD Unix, входит сегодня в стандартный пакет поставки практически всех версий Unix.

Редактор `vi` не поддерживает работу с графикой и использование мыши и на фоне современных средств выглядит достаточно архаично. Выполнение таких функций, как редактирование, сохранение и просмотр содержимого файлов возможно только с применением клавиатуры. Однако, поскольку клавиши, используемые при работе с этой программой, имеются на любой клавиатуре, он может работать с терминалом практически любого типа и является одним из самых популярных редакторов среди программистов и пользователей Unix.

При работе с редактором используются лишь клавиши с буквами, цифрами и знаками пунктуации, а также клавиша `<Esc>`. Клавиши управления курсором поддерживаются только некоторыми терминалами.

Режимы работы редактора `vi`. Существует по меньшей мере пять режимов работы программы `vi`. Чаще всего используются два из них - **командный режим** и **режим ввода** (рис. 10).

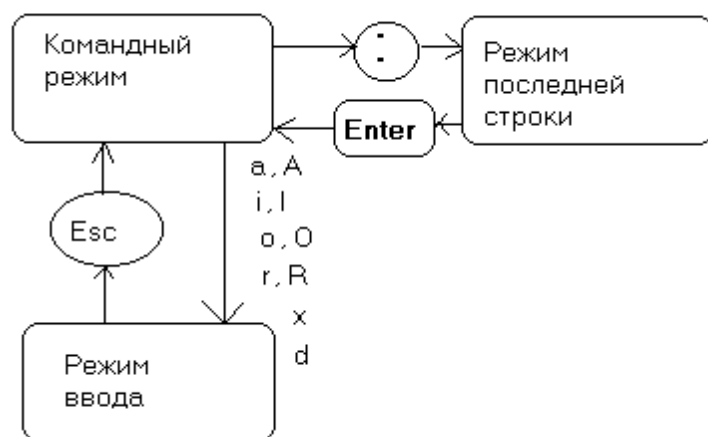


Рис. 10. Основные режимы работы редактора `vi`

В **командном режиме** редактор `vi` интерпретирует нажатие клавиш как команды. Можно указать редактору, например, переместить курсор, удалить текст и т.д.

Приведем часть команд, доступных в **командном режиме**, достаточных для редактирования текста.

Примечания:

- Если Вы не уверены, в каком режиме работает редактор `vi`, нажмите клавишу `<Esc>`.

- Команда, заданная в режиме команд, не отображается на экране. Почти каждое нажатие клавиши приводит к выполнению какой-либо команды.

(!) Будьте очень осторожны.

Основные команды редактора vi приведены в Табл. 7.

Таблица 7. Основные команды редактора vi.

l или <sp> или → h или <BS> или ← j или ↓ k или ↑	Перемещение курсора. sp клавиша ПРОБЕЛ BS клавиша Backspace Примечание: односимвольные команды (l,h,j,k) работают на любых терминалах. Некоторые терминалы поддерживают также работу с клавишами управления курсором.
x	удаляет символ, расположенный справа от курсора (для удаления нескольких символов укажите перед командой x соответствующее числовое значение, например, 5x для удаления 5 символов)
dw	удалить целое слово справа от курсора
dd	удалить всю строку (для удаления нескольких строк необходимо указать их число перед командой dd)
a<текст><Esc>	добавить текст <u>за</u> символом, на который указывает курсор
i<текст><Esc>	добавить текст <u>перед</u> символом, на который указывает курсор
o<текст><Esc>	вставить новую строку. Курсор помещается в начало пустой строки.
rc	замена <u>одного</u> символа, на который указывает курсор, на <u>один</u> символ c.
R	редактор переходит в режим замены и можно заменять <u>один</u> символ на <u>произвольное</u> число символов. Возврат в командный режим по <Esc>.

Приведенных команд достаточно для редактирования любого текста. Однако необходимо иметь в виду, что возможности редактора vi не ограничиваются приведенным списком команд. Редактор vi - это очень гибкий и мощный текстовый редактор.

Программа vi имеет еще так называемый **режим последней строки**. Все команды, вводимые в этом режиме, начинаются с двоеточия (:). Ввод двоеточия заставляет курсор переместиться в нижнюю часть экрана, где следует ввести оставшуюся часть команды.

Примечание. В режиме последней строки ввод команды завершается нажатием клавиши <Enter>.

Основные команды режима последней строки:

:wq запись изменений в редактируемый файл и выход;
:w запись изменений в редактируемый файл;
:w file запись в новый файл;
:q выход из редактора;
:q! выход без сохранения изменений.

7.2. Редактор joe

Более удобным для работы является экранный редактор joe. В отличие от редактора vi он прост в применении и позволяет редактировать текст по всему полю экрана с помощью функциональных и управляющих клавиш.

Основные функции редактирования текста поддержаны "горячими" клавишами, совпадающими с аналогичными клавишами известного редактора Word Perfect.

Копия экрана редактора joe с размещенной в верхней части справочной информацией по командам редактора, приведена на рис. 11.

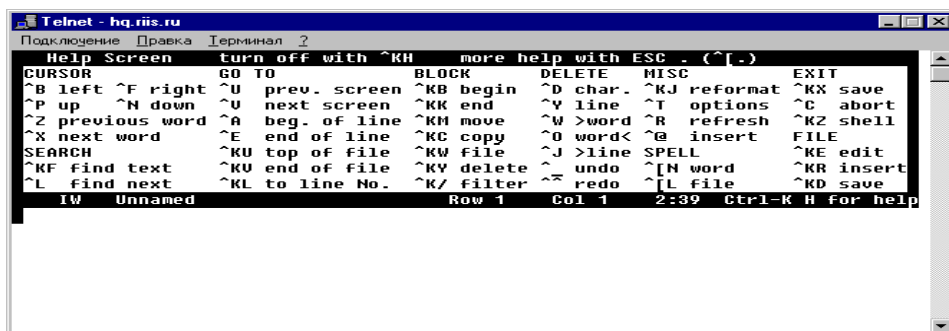


Рис. 11. Справочная информация по командам редактора joe.

Все функции редактора разделены на следующие группы:

CURSOR	управление курсором
GO TO	переходы по тексту
BLOCK	работа с блоками символов
SEARCH	поиск в файле
DELETE	операции удаления
FILE	работа с файлами
MISC	прочие операции
EXIT	выход из редактора

Примечания. 1) Запись ^B означает одновременное нажатие клавиш Ctrl и B.

2) Запись ^KH означает одновременное нажатие клавиш Ctrl и K, а в след за этим - нажатие клавиши H. 3) Вызов справочной таблицы и её отключение производится по команде ^KH.

Редактор прост в работе и основные действия по редактированию текста не требуют пояснения. Остановимся лишь на командах работы с блоками символов.

Для выделения блока необходимо:

- установить курсор на начало выделенного блока;
- выполнить команду ^KB (отметить начало блока);
- установить курсор на конец блока;
- выполнить команду ^KK (отметить конец блока).

Теперь можно с выделенным блоком выполнять следующие действия:

- копировать - ^KC
- перемещать - ^KM
- записывать в файл - ^KW.

Вызов редактора joe осуществляется по команде:

joe file

где file - имя редактируемого файла.

Выход из редактора с сохранением результатов редактирования - ^KX.

Задание по лабораторной работе

1. Освоить следующие команды UNIX:
 - определение текущего каталога - **pwd**;
 - переход в новый текущий каталог - **cd**;
 - просмотр файлов - **cat, more**;
 - получение справки по команде (команда **man**);
 - просмотр содержимого каталогов - **ls** (ключи **laiRF...**); научиться распознавать права доступа к файлу;
 - создание/удаление каталогов - **mkdir/rmdir**;
 - копирование/перемещение файлов - **cp/mv**;
 - изменение прав доступа к файлам - **chmod**.
2. Освоить следующие информационные команды:
 - получение справки о группе команд - **man -k <ключевое_слово>**;
 - поиск файлов – команда **find**.
3. Просмотреть содержимое основных каталогов: **/, /bin, /usr, /usr/bin, /usr/include, ...**
4. Освоить работу с текстовым редактором **vi** в основных режимах (ввод текста; вставка, замена и удаление строк и символов; выход из редактора с сохранением и без сохранения текста).
5. Освоить работу с текстовым редактором **joe**.
7. Выполнить упражнения, приведённые в тексте данных методических указаний.

Примечание. Для освоения указанных команд выполнить Задание из Приложения 2.

Отчет о работе

Выполните задания лабораторного практикума и продемонстрируйте результаты работы преподавателю.

Приложение 1. Примеры использования команды **find**

1. Найти в поддереве каталогов, начинающемся с текущего каталога, файл 'checklist'

```
find . -name checklist -print
```

2. Напечатать список файлов текущего поддерева

```
find . -print
```

3. Найти в текущем поддереве все файлы с суффиксом ".c"

```
find . -name "*.c" -print
```

4. Напечатать имена всех обыкновенных файлов, содержащих программы на языке Си и изменившихся за последние 4 дня, в текущем каталоге и его подкаталогах.

```
find . -name \*.c -type f -mtime -4 -print
```

5. Уничтожить все файлы **core** и файлы с расширением **"*.out"**, которые не менялись больше месяца

```
find / \( -name core -o -name "*.out" \) -atime +7 -exec rm{ } \;
```

6. Вывести сведения об обычных файлах из корневой файловой системы, длина которых в блоках превышает 2

```
find / -type f -size +2 -exec ls -sla{ } \;
```

7. Найти все файлы в поддереве, которыми владеют petrov и ivanov

```
find /home \( -user petrov -o -user ivanov \) -print
```


1. Основной набор команд

1.1. Команды **cd**, **pwd**, **mkdir**

- 1.1.1. Создайте в домашнем каталоге каталог **Catalog1**. В нем создайте каталог **Subdir1**, содержащий каталог **Subdir2**.
- 1.1.2. Из каталога **Subdir2** перейдите на 2 уровня вверх; вернитесь в каталог **Subdir2**.
- 1.1.3. Из каталога **Subdir2** перейдите на 1 уровень вверх.
- 1.1.4. Перейдите в домашний каталог и определите полный путь к этому каталогу.

1.2. Команды **cat-more**

- 1.1. С помощью команды **cat** создайте файл *file1.txt*, содержащий не менее 15-ти строк и файл *file2.txt*, содержащий не менее 30-ти строк.
- 1.2. Выведите на экран содержимое файлов *file1.txt* и *file2.txt* последовательно и без остановок.
- 1.3. Выведите на экран содержимое файлов *file1.txt* и *file2.txt* порциями, используя команды управления просмотром:
 - перейти в начало файла;
 - показать следующую страницу;
 - вернуться на 10 строк вверх по тексту;
 - вывести на экран следующие 16 строк;
 - вывести 5-ю строку;
 - переместиться вниз по тексту на половину экрана (11 строк);
 - завершить просмотр.

1.3. Команда **chmod**

- 1.1. Создайте командой **touch** пустые файлы *f1*, *f2*, *f3*, *f4* и *f5*.
- 1.2. Для файла *f1* установить следующие права:
 - *владелец файла* – запись и чтение;
 - *группа* – чтение;
 - *остальные пользователи* – чтение.
- 1.3. Для файла *f2* разрешите всем выполнять любые действия, а потом:
 - для *владельца* - оставить права на все действия;
 - для *группы* - разрешить только чтение и запись;
 - *остальным пользователям* - разрешить только чтение.
- 1.4. Для файла *f3* установить следующие права:
 - *всем* - разрешить чтение файла;
 - добавить *владельцу* право на исполнение;
 - добавить *группе* право на запись.
- 1.5. Для файла *f4* **всем** установить права на чтение и запись.
- 1.6. Для файла *f5* установить следующие права:
 - *владельцу* – исполнение;
 - *группе* – чтение;
 - *остальным* – чтение.

1.4. Команда **ls** (ключи **laiRF**)

- 1.1. В каталоге **Catalog1**, расположенном в домашнем каталоге, должны содержаться файлы *file1.txt*, *file2.txt*, *f1*, *f2*, *f3*, *f4*, *f5*, а также каталог **Subdir1**, в котором расположен файл *file3.txt*. Находясь в домашнем каталоге, прочитайте оглавление каталога **Catalog1**.

- 1.2. Перейдите в каталог **Catalog1** и выведите на экран его содержимое в самом полном формате. Объясните содержимое полей при "длинном" выводе команды **ls**.
- 1.3. Выведите имена всех файлов и каталогов, содержащихся в каталоге **Catalog1**, в том числе и те, которые начинаются с **.** и **..**. Поясните смысл псевдонимов **.** и **..**.
- 1.4. Выведите имена всех файлов и каталогов, содержащихся в каталоге **Catalog1** вместе с номерами файловых дескрипторов.
- 1.5. Выведите полное содержимое каталога **Catalog1**, включая содержимое всех подкаталогов.
- 1.6. Выведите содержимое каталога **Catalog1** в сокращенном формате.

1.5. Команды **cp**, **mv**

- 1.1. В каталоге **Catalog1**, расположенном в домашнем каталоге, должны содержаться файлы *file1.txt*, *file2.txt*, *f1*, *f2*, *f3*, *f4*, *f5*, а также каталоги **Subdir1** и **Mydir**. Скопируйте файл *file1.txt* в файл *file4.txt* того же каталога.
- 1.2. Скопируйте файл *file1.txt* в файл *file4.txt* того же каталога с запросом на подтверждение копирования.
- 1.3. Скопируйте файл *file1.txt* в каталог **Subdir1** с сохранением атрибутов защиты исходного файла. Для проверки результата данной операции выведите на экран атрибуты защиты исходного файла *file1.txt* и его копии.
- 1.4. В каталоге **Mydir** создайте пустой файл *file.txt*. Скопируйте весь каталог **Mydir** в каталог **Subdir1**.
- 1.5. Находясь в каталоге **Subdir1**, скопируйте в него файл *file2.txt*, расположенный в каталоге **Catalog1**.
- 1.6. Переименуйте файл *file4.txt*, расположенный в каталоге **Catalog1**, в *file1.txt* с запросом на подтверждение копирования;
- 1.7. Переместите файл *file2.txt* из каталога **Subdir1** в **Catalog1** с запросом на подтверждение копирования.

1.6. Команда **man**

- 1.1. Выведите инструкцию по пользованию командой **df**. С помощью команд управления просмотром:
 - перейдите вперед на половину экрана;
 - перейдите вперед на один экран вниз;
 - перейдите назад на половину экрана;
 - выйдите из команды.
- 1.2. Выведите справку о командах для обслуживания каталогов (ключевое слово "**catalog**") и файлов ("**file**").

2. Поиск файлов (команда **find**)

- 2.1. В каталоге **Catalog1**, расположенном в домашнем каталоге, должны содержаться файлы *file1.txt*, *file2.txt*, *f1*, *f2*, *f3*, *f4*, *f5*, а также каталог **Subdir1**, в котором находится файл *file3.doc*, содержащий не менее трех строк.
- 2.2. Перейдите в домашний каталог. Найдите в поддереве каталогов, начинающемся с текущего каталога, файл *file3.txt* и выведите на экран его имя.
- 2.3. Напечатайте список файлов текущего поддерева каталогов.
- 2.4. Найдите в текущем поддереве каталогов все файлы с расширением *".txt"*.
- 2.5. Напечатайте имена всех обыкновенных файлов файловой системы, содержащих программы на командном языке (shell) (расширение **.sh*) и изменившихся в последние 4 дня.

- 2.6. Из поддерева домашнего каталога выведите на экран содержимое файла *file3.doc* и всех файлов с расширением *".txt"*, к которым осуществлялся доступ в последние 3 дня.
- 2.7. Вывести сведения об обычных файлах, расположенных в *поддереве* домашнего каталога, длина которых в блоках превышает 2.

3. Дополнительные возможности интерпретатора команд

3.1. Команды **wc**, **grep**, **sort**

- 3.1.1. В каталоге **Catalog1**, расположенном в домашнем каталоге, должны содержаться файлы *file1.txt*, *file2.txt*, *f1*, *f2*, *f3*, *f4*, *f5*, а также каталог **Subdir1**.
- 3.1.2. Выведите оглавление каталога **Catalog1** в файл *dir.txt*.
- 3.1.3. Объедините файлы *file1.txt* и *dir.txt* в файл *res.txt*.
- 3.1.4. Используя возможность перенаправления стандартного ввода, создайте текстовый файл *text.txt* в каталоге **Subdir1**, содержащий не менее трех строк.
- 3.1.5. Используя возможность перенаправления стандартного ввода, дополните файл *text.txt* еще двумя строками.
- 3.1.6. Избавьтесь от диагностических сообщений при выполнении поиска файла с именем, начинающемся с буквы **f**; поиск начните с корневого каталога.
- 3.1.7. Запретите отображение сообщений стандартного вывода при выполнении поиска файла с именем, содержащим букву **b**, в поддереве корневого каталога. Диагностические сообщения перенаправьте туда же, куда и направлены сообщения стандартного вывода.
- 3.1.8. Подсчитайте количество слов в файле *file1.txt*.
- 3.1.9. Используя конвейер, выведите на экран количество файлов (в том числе и каталогов), содержащихся в каталоге **Catalog1**.
- 3.1.10. Находясь в каталоге **Catalog1**, распечатайте из вывода команды **ls** только те строки, которые содержат в имени цепочку символов *"txt"*.
- 3.1.11. Используя конвейер, выведите на экран количество файлов каталога **Catalog1**, которые содержат в имени цепочку символов *"txt"*.
- 3.1.12. Без применения конвейера, используя группирование команд, выведите на экран количество файлов (в том числе и каталогов), содержащихся в каталоге **Catalog1**.
- 3.1.13. Используя группирование команд, перейдите в каталог **Subdir1**, удалите файл *file.txt* и вернитесь в текущий каталог.
- 3.1.14. Выполните команду вывода содержимого дерева корневого каталога в фоновом режиме. Отобразите номер этого процесса (его PID). Остановите выполнение этого процесса.

4. Редакторы текстов

4.1. Редактор текстов **vi**

- 4.1.1. С помощью редактора **vi** создайте файл *test.txt*, содержащей не менее пяти строк. В каждой строке должно быть не менее трех слов. Закройте файл, сохранив его.
- 4.1.2. Откройте файл *test.txt* с помощью редактора **vi**. Переместите курсор вниз/вправо/вверх/влево, используя символьные клавиши и клавиши управления курсором. Выйдите из редактора **vi**.
- 4.1.3. Откройте файл *test.txt* с помощью редактора **vi**. Удалите первый символ второго слова первой строки. Вернитесь в командный режим и переместитесь на начало второй строки, используя символьные клавиши. Удалите 4 символа справа от курсора одной командой. Закройте файл, не сохраняя его.
- 4.1.4. Откройте файл *test.txt* с помощью редактора **vi**. Удалите второе слово третьей строки одной командой. Удалите две последние строки одной командой. Сохраните файл как *test2.txt* и закройте его.

- 4.1.5. Откройте файл *test.txt* с помощью редактора **vi**. Перейдите на начало второго слова первой строки. Вставьте символ слева от курсора. Вставьте символ справа от курсора. Вставьте пустую строку, и наберите в ней два слова. Удалите ранее вставленную строку. Сохраните изменения, не закрывая файл. В первом слове второй строки замените первый символ на "z". Во втором слове текущей строки замените символы, начиная с первого, на "test". Между первым и вторым словами первой строки вставьте слово "file". Закройте файл, сохранив его.

5. Завершение работы. Команды **rmdir**, **rm**, **exit**

- 5.1. В каталоге **Catalog1**, расположенном в домашнем каталоге, должны содержаться файлы *file1.txt*, *file2.txt*, *f1*, *f2*, *f3*, *f4*, *f5*, а также каталог **Subdir1**, в котором расположен файл *file3.txt*. Удалите файл *file3.txt* из **Subdir1** без запроса на подтверждение;
- 5.2. Удалите каталог **Subdir1**;
- 5.3. Удалите файл *file2.txt* с запросом на подтверждение.

Внимание! Покажите преподавателю результаты выполненных заданий.

Оглавление

1. Цель лабораторной работы.....	1
2. Основные действия пользователя при работе в ОС.....	1
3. Формат команд ОС UNIX	1
4. Порождение имен файлов	2
5. Базовый пользовательский набор команд.....	2
5.1. Получение справки о команде (команда man)	2
5.2. Переход из каталога в каталог (команда cd)	3
5.3. Определение имени текущего каталога (команда pwd).....	4
5.4. Просмотр содержимого каталогов (команда ls)	5
5.5. Создание каталога (команда mkdir)	9
5.6. Удаление каталога (команда rmdir)	9
5.7. Копирование файлов и каталогов (команда cp)	9
5.8. Перемещение (или переименование) файлов (команда mv).....	10
5.9. Удаление файлов (команда rm).....	11
5.10. Просмотр содержимого файлов (команды cat и more).....	11
5.11. Установка и изменение прав доступа к файлам	12
5.12. Поиск файлов (команда find).....	14
6. Дополнительные возможности интерпретатора команд	16
6.1. Перенаправление ввода-вывода	16
6.2. Конвейеры.....	17
6.3. Фоновые команды.....	18
6.4. Группирование команд	18
7. Создание файлов с помощью текстовых редакторов.....	19
7.1. Редактор vi	19
7.2. Редактор joe	21
Задание по лабораторной работе.....	23
Отчет о работе	23
Приложение 1. Примеры использования команды find	24
Литература	30

Литература

- 1) С. Немнюгин, О. Стесик, А. Робачевский. Операционная система UNIX. БХВ-Петербург, 2014, 656с.
- 2) Майкл Лукас. FreeBSD. Подробное руководство. – М.: Символ-Плюс, 2009. – 535 с.
- 3) Д. Колисниченко. FreeBSD. От новичка к профессионалу. БХВ-Петербург, 2011. – 456 с.
- 4) 2). П. Кью. Использование Unix. Специальное издание. 3-е изд.: Пер. с англ. - М., СПб.,К.: Издат. дом "Вильямс", 1999. - 624 с.
- 5) 3). К. Рейчард, Э. Фостер-Джонсон. Unix: справочник - СПб.: Петер Ком, 1998. - 384 с.