**Лабораторная работа 1. Работа с интерфейсом командной строкой в ОС Linux**

**Цель:** изучить интерфейс командной строки ОС Linux, приобрести основные навыки по работе с терминалом командной строки оболочки bash.

**Теория:**

Командная оболочка — это программа, взаимодействующая с пользователем с помощью текстового интерфейса. Он называется интерфейсом командной строки (CLI). Оболочка позволяет пользователю запускать программы и выполнять команды операционной системы.

Оболочка интерпретирует введенные пользователем команды, преобразуя их в инструкции операционной системы.

Существует несколько разновидностей командной оболочки в ОС Linux:

1. Bourne shell (sh) — оригинальная командная оболочка, является самой ранней оболочкой UNIX. sh является стандартной и доступна почти в любом дистрибутиве \*nix. Существует много командных оболочек, основанных (идейно или напрямую) на Bourne shell:

* ksh (KornShell) — клон шелла Борна, разработанный Дэвидом Корном из AT&T Labs. Синтаксис совместим, функциональность интерактивности увеличена.
* pdksh (public domain ksh) — открытая реализация ksh.
* bash (bourne again shell) (эмуляция совместимости POSIX) расширенная свободная оболочка ash, сходная с pdksh, стандартная оболочка в Linux.

2. C shell — (несовместима с POSIX shell) оболочка, с синтаксисом на основе Си.

* csh (C-Shell) — оболочка из состава дистрибутива BSD, имеет Си-образный синтаксис и не является POSIX-совместимой.
* tcsh (csh) — реализация csh с интерактивными возможностями, не уступающими bash. Удобна для интерактивной работы, cовместима с csh.

3. ash (Almquist shell) — одна из самых маленьких оболочек, доступных для UNIX. ash используется при загрузке Linux в однопользовательском режиме, в защищённом режиме или при загрузке дискетных версий Linux.

Оболочки обладают различной функциональностью и даже различными командами. При написании сценария оболочки (скрипта) необходимо проверять, в какой оболочке он запускается. Команда /etc/shells позволяет посмотреть список доступных оболочек.

Встроенные системные команды Linux-систем состоят из процедур оболочки и исполняемых файлов. Встроенные команды в разных оболочках могут выполняться по-разному, однако они выполняются быстрее, чем системные. Для многих встроенных команд есть системные аналоги.

Места расположения системных команд pwd и /bin/pwd:

* /bin;
* /sbin;
* /usr/bin;
* /usr/sbin;
* /usr/local/bin;
* /usr/local/sbin.

Запустить терминал можно сочетание клавиш Ctrl+Alt+T. Закрыть терминал можно введя команду exit или сочетанием клавиш Ctrl+D.

Переключиться в виртуальный терминал можно нажав комбинацию клавиш Ctrl+Alt+F1, выйти из виртуального терминала: Ctrl+Alt+F7.

Другие клавиатурные сочетания bash:

* <Ctrl> + <B> курсор влево;
* <Ctrl> + <F> курсор вправо;
* <Alt> + <B> курсор на слово влево;
* <Alt> + <F> курсор на слово вправо;
* <Ctrl> + <A> курсор в начало строки;
* <Ctrl> + <E> курсор в конец строки;
* <Ctrl> + <H> удаление символа перед курсором;
* <Ctrl> + <D> удаление символа в позиции курсора;
* <Alt> + <D> удаление слова;
* <Ctrl> + <L> очистка экрана;
* <Alt> + <T> перемена мест аргументов;
* <Alt> + <U> перевод слова в верхний регистр;
* <Alt> + <L> перевод слова в нижний регистр;
* <Ctrl> + <С> остановка выполнения команды;
* <Ctrl> + <Z> приостановка выполнения задания (bg, kill);
* <Ctrl> + <R> поиск команды в истории.

Выполнение множества команд доступны только **root** пользователям. Root - это специальный аккаунт в UNIX-подобных системах с UID (User IDentifier) 0, владелец, которого имеет право на выполнение всех без исключения операций. Для того, что бы войти под root сначала его необходимо активировать, а затем в командной строке ввести команду su, после прохождения авторизации ваша работа не будет ограничивается в правах доступа.

В основном, команды, запускаемые из командной строки, имеют следующий формат:

**command -options <filename>** .

options и <filename> являются необязательными параметрами. Существуют команды, не требующие ввода ни одного из них, и команды, требующие ввода нескольких опций и имен файлов. Если используются несколько опций одновременно, их можно сгруппировать. Например, для просмотра подробного списка (-l) всех файлов текущего каталога, включая скрытые файлы (-a), воспользуйтесь командой: ls –al.

Для ввода длинной команды используется \ (перевод строки). Для ввода нескольких команд в одной строки используется «;». При использовании синтаксиса Команда1 && Команда2 — Команда2 выполняется только в случае удачного выполнения Команды1. Команда1 || Команда2 — Команда2 выполняется только в случае неудачного выполнения Команды1.

Получить помощь по командам в Unix-системе можно несколькими способами.

1. help – встроенная помощь оболочки. Большинство команд Linux могут быть запущены с параметром --help . Например, эта команда даст Вам краткую помощь по команде cp (копирование): cp --help | less.

2. man — система помощи в любой Unix системе. Более расширенная информация доступна из командной строки с использованием так называемых страниц руководства (manual pages). Например: man cp выведет на экран страницу руководства для команды cp (копирование).

Страницы руководства являются стандартными системами помощи в Linux, и содержат много детальной технической информации. Система man не занимается отображением страниц, а только находит их и форматирует, передавая программе просмотра (по умолчанию less):

PgUp,PgDn – перемещение по тексту, <пробел> следующая страница, </> строка – поиск подстроки вниз, <?>строка – поиск подстроки вверх, <n> следующее вхождение искомой строки, <q> выход. Команда manpath позволяет узнать путь поиска страниц man. Страницы man состоят из стандартных разделов:

* NAME – информация, которая будет использована при поиске по ключевому слову;
* SYNOPSIS – формат вызова, опции и аргументы;
* DESCRIPTION – описание объекта (программы, файла, библиотеки);
* OPTIONS – подробное описание опций;
* FILES – файлы, связанные с командой;
* AUTHOR – имя автора с указанием электронной почты;
* SEEALSO– указатели на другие страницы man;
* COPYRIGHT – права собственности, политика распространения.

3. info— гипертекстовая иерархическая система Gnu TexInfo.

Например: info cp выведет Вам информацию о команде "cp" (копирование).

Часто info содержит информацию схожую с man, но более свежую.

Команды: <n> следующий узел, <p> предыдущий узел, <u> родительский узел, <l> предыдущая страница, <s> поиск строки на странице, <q> выход.

4. /usr/share/doc — документация программ.

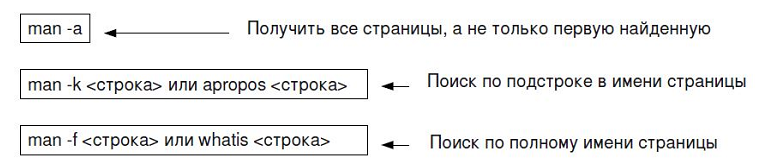


Рисунок 1 - Работа с системой man

Страницы руководства man в Linux делятся на следующие секции:

1. Команды пользовательского уровня и приложения;

2. Системные вызовы и коды ошибок ядра;

3. Библиотечные функции;

4. Информация о файлах устройств и других специальных файлах;

5. Форматы конфигурационных файлов;

6. Помощь по играм;

7. Макросы, кодировки, интерфейсы;

8. Команды системного администрирования;

9. Внутренние интерфейсы и спецификации ядра.

Порядок перечисления секций в этой таблице не случаен. Дело в том, что файлы с информацией расположены в подкаталогах каталога /usr/man и команда man ищет нужную информацию, просматривая эти подкаталоги именно в том порядке, который приведен. Если вы, например, дадите команду «[user]$ man swapon» то получите справку о команде swapon из секции 8. Поэтому если вы хотите получить справку по системному вызову swapon, надо дать команду «[user]$ man 2 swapon» указывая номер секции, в которой надо искать информацию.

В Linux формирование файловой системы и каталогов осуществляется по таким правилам:

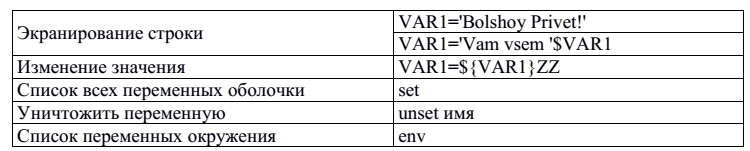
* логически файловая структура организована в виде иерархии;
* каждый каталог может иметь множество подкаталогов, но у каждого подкаталога имеется только один родительский каталог;
* имя корневого каталога - / (он сам для себя является родительским);
* прописные и строчные буквы в именах файлов различаются: TheFile и thefile — разные файлы;
* абсолютные имена файлов показывают путь к файлу от корневого каталога (имена начинаются с /) /home/user1/homework/lab1.html;
* относительные имена показывают путь к файлу от текущего каталога;
* имена файлов могут содержать точки: archive.tar.gz;
* имя файла не может содержать символов / и \0 (null).

Типы файлов в Linux:

* - - обычные файлы;
* d – каталоги;
* l – символические ссылки (указатель на другой файл);
* b – блочные устройства (специальные файлы для обращения к устройствам, например, жесткому диску);
* с – символьные устройства (специальные файлы, предназначенные для ввода/вывода с таких устройств, как терминал или мышь);
* p – именованный канал (один из вариантов организации взаимодействия между процессами);
* s – сокеты (предназначены для организации сетевого межпроцессного взаимодействия).

Понятие параметра в оболочке bash подобно понятию переменной в обычных языках программирования. Именем (или идентификатором) параметра может быть слово, состоящее из алфавитных символов, цифр и знаков подчеркивания (только первый символ этого слова не может быть цифрой), а также число или один из следующих специальных символов: \*, @, #, ?, - (дефис), $, !, 0, \_ (подчеркивание). Чтобы отличать команды от переменных, переменные лучше обозначать большими буквами (пример - HOSTNAME).

Таблица 1 Переменные оболочки и окружения



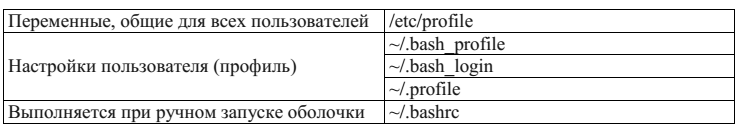
Говорят, что параметр задан или установлен, если ему присвоено значение. Значением может быть и пустая строка. Чтобы вывести значение параметра, используют символ $ перед его именем. Так, команда «[user]$ echo name» выдаст на экран слово name, а команда «[user]$ echo $name» выдаст значение переменной name (если таковое, конечно, задано).

Переменные оболочки доступны только в той оболочке, в которой были описаны! Перевод переменной оболочки в переменную окружения (доступна для дочерних процессов): export VAR1.

В ОС Linux важнейшими переменными окружения являются:

* HOME — путь к домашнему каталогу;
* LOGNAME и USER — имя пользователя;
* MAIL — путь к почтовому ящику;
* PATH — путь поиска исполняемых файлов;
* PS1 — вид приглашения оболочки;
* PWD — имя текущего каталога;
* OLDPWD — имя предыдущего каталога;
* SHELL — имя исполняемого файла оболочки;
* TERM — тип терминала;
* HOSTNAME — имя хоста;
* SHLVL — номер загруженной оболочки.

Таблица 2 Файлы настроек, хранящие переменные окружения



В командных оболочках, используемых в Linux, есть масса способов экономии усилий (нажатий на клавиши) при выполнении наиболее распространённых действий: автоматическое дополнение длинных названий команд или имён файлов, поиск и повторное выполнение команды, уже когда-то исполнявшейся раньше, подстановка списков имён файлов по некоторому шаблону и многое другое.

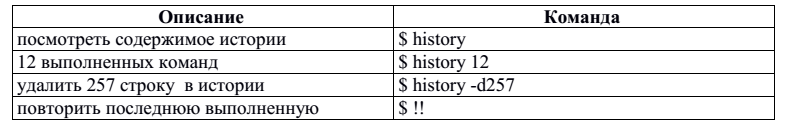
Хорошую возможность не тратить время на набор одних и тех же команд подарили нам разработчики оболочки bash. Они предоставили возможность использовать историю команд, чтобы сократить время набора команд и сделать работу в командной строке более эффективной. По умолчанию, история команд включена, и все команды, которые вы выполняете в командной строке, могут быть использованы вами повторно без особых затрат времени на их набор.

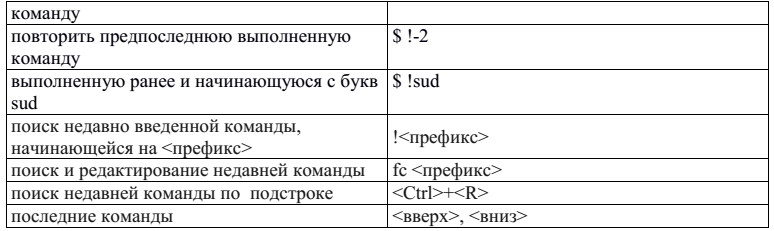
Историю команд в Linux можно отключить, выполнив в командной строке команду: $ set +o history

Если, после отключения истории команд, вы вновь захотите использовать ее в своей работе, просто выполните команду: $ set -o history

Одна из переменных окружения, имеющая название HISTSIZE, хранит в себе количество выполненных команд. Посмотреть значение переменной HISTSIZE можно выполнив в командной строке следующую команду: $ echo $HISTSIZE.

Таблица 3. Команды для работы с историей

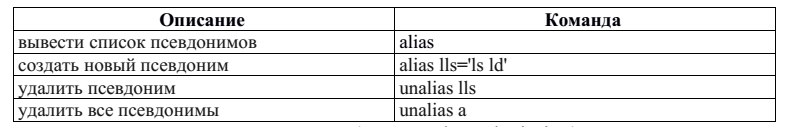




Автоматически попытаться дополнить командную строку именами файлов или команд можно использовав клавишу <Tab>, если вариантов несколько то повторным нажатием <Tab> можно вывести список. Если строка начинается с $ - дополняется имя переменной оболочки, ~ дополняется имя пользователя, @ дополняется имя хоста.

Псевдонимы команд служат для ускорения набора длинных, часто используемых команд.

Таблица 4. Псевдонимы команд



Командная подстановка (command substituion) — результат выполнения одной команды автоматически передается в качестве аргументов другой команде. Синтаксис: внешняя\_команда `внутренняя\_команда` и внешняя\_команда $(внутренняя\_команда).

Пример: *ls l `which rpm` ls l $(cat /etc/shells).*

Можно присвоить результат выполнения переменной оболочки: ID=`id';echo $ID.

В командной строке можно вычислять выражения, заключенные в квадратные скобки или в двойные круглые скобки. Перед скобками должен стоять символ $, а результаты выражений можно передавать как аргумент какой-либо команде или назначать переменной. Арифметика, выполняющаяся в командной строке является целочисленной.

Пример: echo $((1+2)),echo $((7%3)),echo $((5\*6)),echo $((7/3)) .

**Шаблоны подстановки и перечисление.**

Символ «звездочка» \* является шаблоном для любого количества любых символов в именах файлов, и даже для их отсутсвия. echo \* (выведет все имена файлов в данной директории);

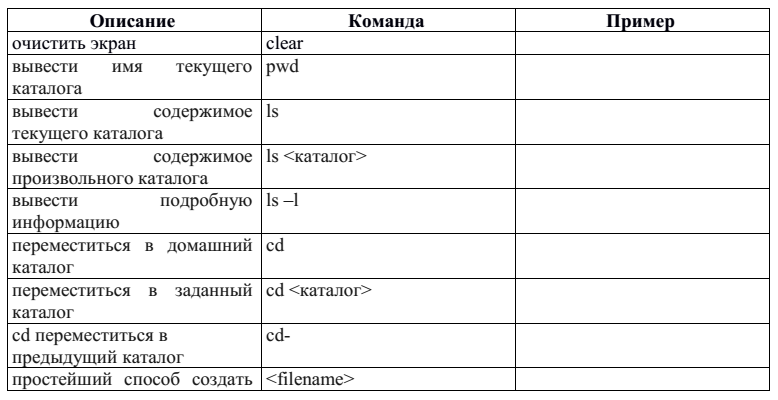
Единственный символ, который не удовлетворяет этому шаблону - лидирующая точка в именах скрытых файлов echo .\* (выведет все имена файлов в данной директории, включая те, которые начинаются с точки);

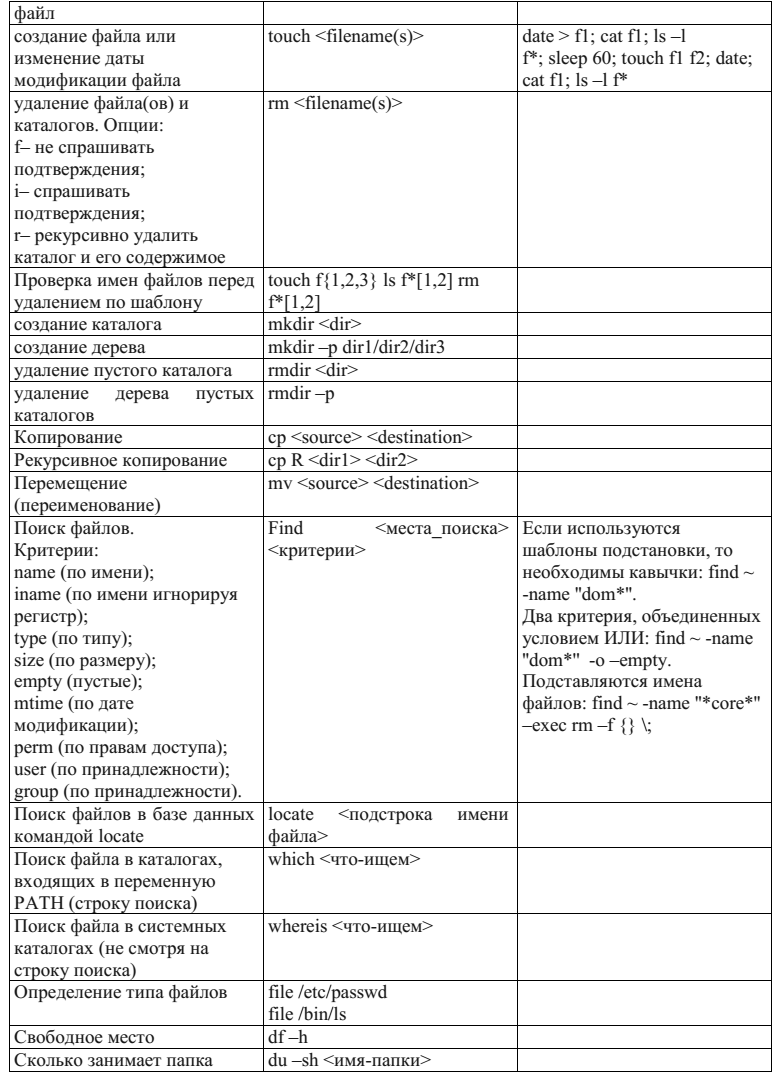
Символ ? заменяет один символ в имени файла, который должен находится в той позиции, где находится знак вопроса;

Шаблон диапазона - квадратные скобки [0-9] для любых цифр [a-zA-Z] для букв английского [!abc] множество любых символов кроме a, b, c;

Механизм перечисления - фигурные скобки {rc,\_profile} echo .bash{rc,\_profile} обращение к двум файлам .bash rc и .bash\_profile. Имена этих файлов имеют общую подстроку .bash, которая вынесена за фигурные скобки. В фигурных скобках через запятую перечислены варианты продолжения: rc и \_profile.

Таблица 5. Базовые команды





**Практическая часть:**

1. Перейти в корневую директорию (папку). Проверьте, в какой директории находитесь.

2. Вывести пронумерованный список директорий далее работать с директорией # (ваш –номер по списку);

3. Вывести содержимое директории:

* в формате по умолчанию;
* в обратном порядке;
* содержимое поддиректорий;
* вывести все файлы включая скрытые;
* вывести файлы с указанием их размера в КБ/МБ/ГБ;
* вывести файлы отсортированные по размеру, с указанием размера в КБ/МБ/ГБ;
* только имена вложенных директорий, расположенных в текущей директории;
* отсортированное по дате создания файла;
* отсортированное по дате обращения к файлу;
* только файлы, вторая буква имени которых – гласная англ. алфавита
* записать список файлов и папок в текущей директории (с полной информацией о них) в файл dirlist.txt , который лежит в домашней директории.

4. Перейти в домашнюю директорию с помощью короткой команды;

5. Вернуться в предыдущую директорию;

6. Вернуться обратно в домашнюю;

7. Перейти на уровень выше;

8. Перейти в каталог /tmp. С помощью одной команды перейти в подкаталог local/bin каталога /usr.

9. Вывести содержимое файла dirlist.txt:

* просто;
* в обратном порядке;
* с нумерацией непустых строк;
* с нумерацией всех строк;
* схлопывая подряд идущие пустые строки в одну;

14. Создать в домашней директории папку linux\_lab1

15. Войти в директорию linux\_lab1

16. Скопировать в нее файл dirlist.txt из домашней директории

17. Удалить файл dirlist.txt из домашней директории

18. Создать директорию manyfiles

19. Создать в ней 100 файлов с именами a1, a2, a3, …. a100.

20. Удалить только файлы с четными номерами.

21. Вывести строки файла dirlist.txt, содержащие файлы с определенным месяцем (в зависимости от номер варианта 1-январь,..12 – декабрь, 13 – опять январь) и записать их в файл grep\_month\_name.txt.

22. Записать строки, не содержащие этот месяц, в файл grep\_other\_monthes.txt.

23. Создать папку grep, переместить в нее файлы, созданные в пунктах 21 и 22.

24. Находясь в папке linux\_lab1 найти все файлы в этой директории и ее поддиректориях в которых встречается подстрока root, вывести строки с указанием их номеров.

25. Найти все файлы в системе, содержащие в имени bash;

26. Найти файлы, измененные за последний час.

27. Найти символические ссылки в каталоге / (но не глубже чем на 2), вывести, на что они указывают.

28. Просмотреть, какие переменные окружения заданы в вашей системе;

29. Поменять приглашение командной строки, добавить текущее время.

30. Удалить весь каталог manyfiles со всеми файлами;

31. Создайте текстовый файл следующего содержания:

1+2

6\*4

97%12

43215/43\*100

Посчитайте все примеры из файла с помощью одной команды.