Министерство образования Российской Федерации
УФИМСКИЙ ГОСУДАРСТВЕННЫЙ АВИАЦИОННЫЙ
ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ

РАБОТА ПОЛЬЗОВАТЕЛЯ В ОПЕРАЦИОННОЙ СИСТЕМЕ LINUX

МЕТОДИЧЕСКИЕ УКАЗАНИЯ

к лабораторному практикуму по курсу
"Операционные системы"
для студентов специальностей 220200 –
Автоматизированные системы обработки
информации и управления и 351400 –
Прикладная информатика в экономике

Уфа 2005

Составители: О.Д. Лянцев, Р.Р. Еникеев, А.А. Колесников, П.И. Тараракo

УДК 681.3

Работа пользователя в операционной системе LINUX: Методические указания к лабораторному практикуму по курсу "Операционные системы" для студентов специальностей 220200 – Автоматизированные системы обработки информации и управления и 351400 – Прикладная информатика в экономике /Уфимск. гос. авиац. техн. ун-т; Сост.: О.Д. Лянцев, Р.Р. Еникеев, А.А. Колесников, П.И. Тараракo - Уфа, 2005. - 56 с.

Приведены основные сведения о принципах организации и функционирования многопользовательской операционной системы LINUX. Рассматриваются структура файловой системы, функции по обработке и управлению данными, создание и выполнение командных файлов. Изучаются принципы взаимодействия LINUX с внешними устройствами и формирование системных и инсталляционных дисков. Изучаются многозадачный режим выполнения процессов, а также пользовательский и программный интерфейсы операционной системы. Практическое изучение функциональных особенностей системы иллюстрируется примерами формирования простых и сложных команд по обработке данных. Приводится методика и порядок выполнения лабораторных работ, в приложении содержатся вспомогательные материалы.

Библиогр.: 7 назв.
Рецензенты: Ю.Б. Головкин, Р.В. Насыров

СОДЕРЖАНИЕ

Введение

ОС Linux - это многопользовательская, многозадачная, многотерминальная операционная система (ОС) из семейства UNIX, под управлением которой могут одновременно выполняться несколько задач. Она предназначена для работы на серверах и рабочих станциях, обеспечивает подключение дополнительных терминалов и допускает этом режиме использование графических оболочек.

UNIX-серверы предназначены для хранения и обработки больших объемов информации. Особенно эффективно использование UNIX-серверов при распределенной обработке данных. Для этого разработаны системы распределенных вычислений в соответствии со стандартом CORBA. К таким системам относятся системы управления базами данных (СУБД типа Oracle, Informix), файл-серверы, FTP-серверы, WWW-серверы и др., которые поддерживаются ОС Linux. В распределенных системах информация может находиться на различных рабочих станциях, различных дисках, программные модули могут функционировать на различных компьютерах, но система работает таким образом, что это составляет единое целое. При обработке больших объемов информации используется технология клиент - сервер, при которой пользователь работает только с той информацией, которая ему необходима. Развитием технологии клиент - сервер является технология интеллектуальных агентов.

ОС Linux является сетевой операционной системой для 32-х или 64-х разрядных платформ. Она обеспечивает маштабируемость в диапазоне от игровых приставок (Sony Play Station) до кластерных серверов Internet.

ОС Linux не связана с конкретной моделью компьютеров. Её ядро реализовано на языке высокого уровня (языке СИ), что позволяет достаточно легко переносить эту систему с одной платформы на другую. Система распространяется по лицензии GNU либо подобным свободным лицензиям, обеспечивается как коммерческое, так и свободное сопровождение через Internet. Поставка исходных модулей системы обеспечивает возможность адаптации прикладных программ в случае перехода на другую платформу и дает возможность контроля кодов, реализующих несанкционированный доступ. В разработке системы приняло участие большое количество специалистов, зарегистрировавших свои авторские права, что дает гарантии ее немонополизации.

Подключение персональных компьютеров (ПК) в вычислительную сеть с UNIX - серверами может осуществляться по протоколу TCP/IP, при этом пользователи получают следующие возможности:

1) использование UNIX-сервера, как файл - сервера;

2) эмуляция на ПК удаленного терминала (режим TELNET);

3) организация системы клиент - сервер (рабочая станция формирует SQL - запросы, сервер их обрабатывает);

4) непосредственный обмен файлами между ПК по протоколу FTP;

5) организация распределенных вычислений по стандарту CORBA.

Все действия в ОС UNIX оформлены как процессы. Процесс представляет собой совокупность выполняемых программ или одну выполняемую программу, которые вызываются при исполнении системной команды. Процесс может породить один или несколько других процессов, которые могут выполняться параллельно. ОС Linux поддерживает многопроцессорную архитектуру для параллельного выполнения процессов.

ЛАБОРАТОРНАЯ РАБОТА № 1

ОСНОВНЫЕ ПРИНЦИПЫ ФУНКЦИОНИРОВАНИЯ ОПЕРАЦИОННОЙ СИСТЕМЫ LINUX

1. Цель работы

Целью работы является изучение архитектуры и принципов функционирования многопользовательской многозадачной операционной системы Linux, особенности ее использования в качестве сервера и рабочей станции.

2. Теоретическая часть

Система включает следующие основные компоненты.

Ядро. Выполняет функции управления памятью, процессорами. Осуществляет диспетчеризацию выполнения всех программ и обслуживание внешних устройств. Все действия, связанные с вводом/выводом и выполнением системных операций, выполняются с помощью системных вызовов. Системные вызовы реализуют программный интерфейс между программами и ядром. Имеется возмож-

ность динамического конфигурирования ядра.

Диспетчер процессов Init. Активизирует процессы, необходимые для нормальной работы системы и производит их начальную инициализацию. Обеспечивает завершение работы системы, организует сеансы работы пользователей, в том числе, для удаленных терминалов.

Интерпретатор команд Shell. Анализирует команды, вводимые с терминала либо из командного файла, и передает их для выполнения в ядро системы. Команды обычно имеют аргументы и параметры, которые обеспечивают модернизацию выполняемых действий. Shell является также языком программирования, на котором можно создавать командные файлы (shell-файлы). При входе в ОС пользователь получает копию интерпретатора shell в качестве родительского процесса. Далее, после ввода команды пользователем создается порожденный процесс, называемый процессом-потомком. Т.е. после запуска ОС каждый новый процесс функционирует только как процесс - потомок уже существующего процесса. В ОС Linux имеется возможность динамического порождения и управления процессами.

Shell - интерпретатор в соответствии с требованиями стандарта POSIX поддерживает графический экранный интерфейс, реализованный средствами языка программирования Tcl/Tk.

Обязательным в системе является интерпретатор Bash, полностью соответствующий стандарту POSIX. В качестве Shell может быть использована оболочка **mc** с интерфейсом, подобным Norton Commander.

Сетевой графический интерфейс X-сервер (X-Windows). Обеспечивает поддержку графических оболочек.

Графические оболочки KDE, Gnome. Отличительными свойствами KDE являются: минимальные требования к аппаратуре, высокая надежность, интернационализация. Базовые библиотеки KDE (qt, kde-libs) признаны одними из лучших продуктов по созданию графического интерфейса, обеспечивают простое написание программ с использованием передовых технологий. Gnome имеет развитые графические возможности, но более требователен к аппаратным средствам.

Сетевая поддержка NFS, SMB, TCP/IP. NFS - программный комплекс PC-NFS (Network File System) для выполнения сетевых функций. PC-NFS ориентирован для конкретной ОС персонального

компьютера (PC) и включает драйверы для работы в сети и дополнительные утилиты. SMB - сетевая файловая система, совместимая с Windows NT. TCP/IP - протокол контроля передачи данных (Transfer Control Protocol/Internet Protocol). Сеть по протоколам TCP/IP является неотъемлемой частью ОС семейства UNIX. Поддерживаются любые сети, от локальных до Internet, с использованием только встроенных сетевых средств.

Инструментальные средства программирования. Основой средств программирования является компилятор GCC или его экспериментальные версии EGCS и PGCC для языков C и C++; модули поддержки других языков программирования (Obective C, Фортран, Паскаль, Modula-3, Ада, Java и др.); интегрированные среды и средства визуального проектирования: Kdevelop, Xwpe; средства адаптации привязки программ AUTOCONFIG, AUTOMAKE.

2.1. Регистрация пользователя в системе

Для входа пользователя с терминала в многопользовательскую операционную систему LINUX необходимо зарегистрироваться в качестве пользователя. Для этого нужно после сообщения

Login:

ввести системное имя пользователя, например, "student". Если имя задано верно, выводится запрос на ввод пароля:

Password:

Наберите пароль "student" и нажмите клавишу *Enter*.

Если имя или пароль указаны неверно, сообщение *login* повторяется. Значение пароля проверяется в системном файле *password*, где приводятся и другие сведения о пользователях. После правильного ответа появляется приветствие LINUX и приглашение:

student@linux:>

Вы получили доступ к ресурсам ОС LINUX.

2.2. Выход из системы

exit - окончание сеанса пользователя.

2.3. Выполнение простых команд

Формат команд в ОС LINUX следующий:

имя команды [аргументы] [параметры] [метасимволы]

Имя команды может содержать любое допустимое имя файла; аргументы - одна или несколько букв со знаком минус (-); параметры

- передаваемые значения для обработки; метасимволы интерпретируются как специальные операции. В квадратных скобках указываются необязательные части команд.

Введите команду **echo**, которая выдает на экран свои аргументы:

echo good morning

и нажмите клавишу *Enter*. На экране появится приветствие "*good morning*" – аргумент команды **echo.** Командный интерпретатор *shell* вызвал команду **echo**, реализованную в виде программы на языке СИ, и передал ей аргументы. После этого интерпретатор команд вывел знак-приглашение. Синтаксис команды **echo**:

echo [-n] [arg1] [arg2] [arg3]...

Команда помещает в стандартный вывод свои аргументы, разделенные пробелами и завершаемые символом перевода строки. При наличии флага *-n* символ перевода строки исключается.

who [am i] - получение информации о работающих пользователях.

В квадратных скобках указываются аргументы команды, которые можно опустить. Ответ представляется в виде таблицы, которая содержит следующую информацию:

- идентификатор пользователя;
- идентификатор терминала;
- дата подключения;
- время подключения.

date - вывод на экран текущей даты и текущего времени.

cal [[месяц]год] - календарь; если календарь не помещается на одном экране, то используется команда **cal год | more** и клавишей пробела производится постраничный вывод информации.

man <название команды> - вызов электронного справочника об указанной команде. Выход из справочника - нажатие клавиши Q. Команда **man man** сообщает информацию о том, как пользоваться справочником.

tty - сообщение имени специального файла стандартного вывода, соответствующего терминалу пользователя.

cat <имя файла> - вывод содержимого файла на экран. Команда **cat > text.1** создает новый файл с именем text.1, который можно заполнить символьными строками, вводя их с клавиатуры. Нажатие клавиши *Enter* создает новую строку. Завершение ввода - нажатие *Ctrl - d*. Команда **cat text.1 > text.2** пересылает содержимое файла

text.1 в файл text.2. Слияние файлов осуществляется командой **cat text.1 text.2 > text.3**.

ls [-alrstu] [имя] - вывод содержимого каталога на экран. Если аргумент не указан, выдается содержимое текущего каталога.

Аргументы команды:

-a - выводит список всех файлов и каталогов, в том числе и скрытых;

-l - выводит список файлов в расширенном формате, показывая тип каждого элемента, полномочия, владельца, размер и дату последней модификации;

- r - выводит список в порядке, обратном заданному;

- s - выводит размеры каждого файла;

- t - перечисляет файлы и каталоги в соответствии с датой их последней модификации;

- u - перечисляет файлы и каталоги в порядке, обратном их последней модификации.

rm <имя файла> - удаление файла (файлов). Команда **rm text.1 text.2 text.3** удаляет файлы text.1, text.2, text.3. Другие варианты этой команды - **rm text.[123]** или **rm text.[1-3].**

wc [имя файла] - вывод числа строк, слов и символов в файле.

clear - очистка экрана.

2.4. Группирование команд

Группы команд или сложные команды могут формироваться с помощью специальных символов (метасимволов):

& - процесс выполняется в фоновом режиме, не дожидаясь окончания предыдущих процессов;

? - шаблон, распространяется только на один символ;

* - шаблон, распространяется на все оставшиеся символы;

| - программный канал - стандартный вывод одного процесса является стандартным вводом другого;

> - переадресация вывода в файл;

< - переадресация ввода из файла;

; - если в списке команд команды отделяются друг от друга точкой с запятой, то они выполняются друг за другом;

&& - эта конструкция между командами означает, что последующая команда выполняется только при нормальном завершении предыдущей команды (код возврата 0);

|| - последующая команда выполняется только, если не выполнилась предыдущая команда (код возврата 1);

() - группирование команд в скобки;

{ } - группирование команд с объединенным выводом;

[] - указание диапазона или явное перечисление (без запятых);

>> - добавление содержимого файла в конец другого файла.

Примеры.

who | wc - подсчет количества работающих пользователей командой **wc** (word count - счет слов);

cat text.1 > text.2 - содержимое файла text.1 пересылается в файл text.2;

mail student < file.txt - электронная почта передает файл file.txt всем пользователям, перечисленным в командной строке;

cat text.1,text.2 - просматриваются файлы text.1 и text.2;

cat text.1 >> text.2 - добавление файла text.1 в конец файла text.2;

cc primer.c & - трансляция СИ - программы в фоновом режиме. Имя выполняемой программы по умолчанию a.out.

cc -o primer.o primer.c - трансляция СИ-программы с образованием файла выполняемой программы с именем primer.o;

rm text.* - удаление всех файлов с именем text;

{cat text.1; cat text.2} | lpr - просмотр файлов text.1 и text.2 и вывод их на печать;

ps [al] [number] - команда для вывода информации о процессах:

-a - вывод информации обо всех активных процессах, запущенных с вашего терминала;

-l - полная информация о процессах;

number - номер процесса.

Команда **ps** без параметров выводит информацию только об активных процессах, запущенных с данного терминала, в том числе и фоновых. На экран выводится подробная информация обо всех активных процессах в следующей форме:

```
F  S  UID  PID  PPID  C  PRI  NI  ADDR  SZ  WCHAN  TTY  TIME  CMD

1  S  200  210  7     0  2    20  80    30  703a   03   0:07  cc
1  R  12   419  7     11 5    20  56    20         03   0:12  ps
```

F - флаг процесса (1 - в оперативной памяти, 2 - системный процесс, 4 - заблокирован в ОЗУ, 20 - находится под управлением другого процесса, 10 - подвергнут свопингу);

S - состояние процесса (O - выполняется процессором , S - задержан, R - готов к выполнению, I - создается);

UID - идентификатор пользователя;

PID - идентификатор процесса;

PPID - номер родительского процесса;

C - степень загруженности процессора;

PRI - приоритет процесса, вычисляется по значению переменной NICE и чем больше число, тем меньше его приоритет;

NI - значение переменной NICE для вычисления динамического приоритета, принимает величины от 0 до 39;

ADDR - адрес процесса в памяти;

SZ - объем ОЗУ, занимаемый процессом;

WCHAN - имя события, до которого процесс задержан, для активного процесса - пробел;

TTY - номер управляющего терминала для процесса;

TIME - время выполнения процесса;

CMD - команда, которая породила процесс.

nice [-приращение приоритета] команда[аргументы] - команда изменения приоритета. Каждое запущенное задание (процесс) имеет номер приоритета в диапазоне от 0 до 39, на основе которого ядро вычисляет фактический приоритет, используемый для планирования процесса. Значение 0 представляет наивысший приоритет, а 39 - самый низший. Увеличение номера приоритета приводит к понижению приоритета, присвоенного процессу. Команда **nice -10 ls -l** увеличивает номер приоритета, присвоенный процессу **ls -l** на 10.

renice 5 1836 - команда устанавливает значение номера приоритета процесса с идентификатором 1836 равным 5. Увеличить приоритет процесса может только администратор системы.

kill [-sig] <идентификатор процесса> - прекращение процесса до его программного завершения. sig - номер сигнала. Sig = -15 означает программное (нормальное) завершение процесса, номер сигнала = -9 - уничтожение процесса. По умолчанию sig= -9. Вывести себя из системы можно командой kill -9 0. Пользователь с низким приоритетом может прервать процессы, связанные только с его терминалом.

mc - вызов файлового менеджера (программы - оболочки) *Mid-*

night Commander, аналогичного *Norton Commander*.

sort [-dr] - сортировка входных файлов и вывод результата на экран.

3. Порядок выполнения работы

1. Ознакомиться с теоретической частью к лабораторной работе.
2. Зарегистрироваться в системе LINUX.
3. Определить день недели, в который Вы родились.
4. Получить подробную информацию обо всех активных процессах.
5. Используя редактор VI (см. приложение), создать два текстовых файла (с расширением TXT) и командой CAT просмотреть их на экране.
6. Получить информацию о работающих пользователях, подсчитать их количество и запомнить в файле.
7. Объединить текстовые файлы в единый файл и посмотреть его на экране.
8. Посмотреть приоритет своего процесса и уменьшить скорость его выполнение за счет повышения номера приоритета.
9. Используя редактор VI, написать программу на языке СИ и запустить ее на трансляцию в фоновом режиме.
10. Показать преподавателю исходный текст программы на языке СИ, текстовый файл, файл с сохранением количества пользователей.
11. Продемонстрировать выполнение СИ - программы.
12. Удалить свои файлы и выйти из системы.

4. Контрольные вопросы

1. Перечислите основные функции и назначение многопользовательской многозадачной операционной системы LINUX и ее отличительные особенности от однопрограммной системы DOS.
2. Какое назначение имеет ядро системы и интерпретатор команд?
3. В чем заключается понятие "процесс" и какие операции можно выполнить над процессами?
4. Как задаются и выполняются простые и сложные команды?
5. Какие функции выполняет командный интерпретатор *Shell*?

ЛАБОРАТОРНАЯ РАБОТА № 2

ИЗУЧЕНИЕ ФАЙЛОВОЙ СИСТЕМЫ И ФУНКЦИЙ ПО ОБРАБОТКЕ И УПРАВЛЕНИЮ ДАННЫМИ

1. Цель работы

Целью работы является изучение структуры файловой системы ОС LINUX, изучение команд создания, удаления, модификации файлов и каталогов, функций манипулирования данными.

2. Теоретическая часть

2.1. Файловая структура системы LINUX

В операционной системе LINUX файлами считаются обычные файлы, каталоги, а также специальные файлы, соответствующие периферийным устройствам (каждое устройство представляется в виде файла). Доступ ко всем файлам однотипный, в том числе, и к файлам периферийных устройств. Такой подход обеспечивает независимость программы пользователя от особенностей ввода/вывода на конкретное внешнее устройство.

Файловая структура LINUX имеет иерархическую древовидную структуру. В корневом каталоге размещаются другие каталоги и файлы, включая 5 основных каталогов:

bin - большинство выполняемых командных программ и *shell* - процедур;

tmp - временные файлы;

usr - каталоги пользователей (условное обозначение);

etc - преимущественно административные утилиты и файлы;

dev - специальные файлы, представляющие периферийные устройства; при добавлении периферийного устройства в каталог /dev должен быть добавлен соответствующий файл (черта / означает принадлежность корневому каталогу).

Текущий каталог - это каталог, в котором в данный момент находится пользователь. При наличии прав доступа, пользователь может перейти после входа в систему в другой каталог. Текущий каталог обозначается точкой (**.**); родительский каталог, которому принадлежит текущий, обозначается двумя точками (**..**).

Полное имя файла может включать имена каталогов, включая

корневой, разделенных косой чертой, например: /home/student/file.txt. Первая косая черта обозначает корневой каталог, и поиск файла будет начинаться с него, а затем в каталоге home, затем в каталоге student.

Один файл можно сделать принадлежащим нескольким каталогам. Для этого используется команда **ln (link)**:

ln <имя файла 1> <имя файла 2>

Имя 1-го файла - это полное составное имя файла, с которым устанавливается связь; имя 2-го файла - это полное имя файла в новом каталоге, где будет использоваться эта связь. Новое имя может не отличаться от старого. Каждый файл может иметь несколько связей, т.е. он может использоваться в разных каталогах под разными именами. Команда **ln** с аргументом -s создает символическую связь:

ln -s <имя файла 1> <имя файла 2>

Здесь имя 2-го файла является именем символической связи. Символическая связь является особым видом файла, в котором хранится имя файла, на который символическая связь ссылается. LINUX работает с символической связью не так, как с обычным файлом - например, при выводе на экран содержимого символической связи появятся данные файла, на который эта символическая связь ссылается.

В LINUX различаются 3 уровня доступа к файлам и каталогам:

1) доступ владельца файла;

2) доступ группы пользователей, к которой принадлежит владелец файла;

3) остальные пользователи.

Для каждого уровня существуют свои байты атрибутов, значение которых расшифровывается следующим образом:

r – разрешение на чтение;

w – разрешение на запись;

x – разрешение на выполнение;

- – отсутствие разрешения.

Первый символ байта атрибутов определяет тип файла и может интерпретироваться со следующими значениями:

- – обычный файл;

d – каталог;

l – символическая связь;

в – блок-ориентированный специальный файл, который соответствует таким периферийным устройствам, как накопители на магнитных дисках;

c – байт-ориентированный специальный файл, который может соответствовать таким периферийным устройствам как принтер, терминал.

В домашнем каталоге пользователь имеет полный доступ к файлам (READ, WRITE, EXECUTE; r, w, x).

Атрибуты файла можно просмотреть командой **ls -l** и они представляются в следующем формате:

```
d          rwx          rwx          rwx
|           |            |            |
|           |            |            | Доступ для остальных пользователей
|           |            | Доступ к файлу для членов группы
|           | Доступ к файлу владельца
| Тип файла (директория)
```

Пример. Командой **ls -l** получим листинг содержимого текущей директории student:

```
- rwx --- --- 2 student 100  Mar  10 10:30 file_1
- rwx --- r-- 1 adm     200  May 20 11:15 file_2
- rwx --- r-- 1 student 100  May 20 12:50 file_3
```

После байтов атрибутов на экран выводится следующая информация о файле:

- число связей файла;
- имя владельца файла;
- размер файла в байтах;
- дата создания файла (или модификации);
- время;
- имя файла.

Атрибуты файла и доступ к нему, можно изменить командой:

chmod <коды защиты> <имя файла>

Коды защиты могут быть заданы в числовом или символьном виде. Для символьного кода используются:

знак плюс (+) - добавить права доступа;

знак минус (-) - отменить права доступа;

r,w,x - доступ на чтение, запись, выполнение;

u,g,o - владельца, группы, остальных.

Коды защиты в числовом виде могут быть заданы в восьмеричной форме. Для контроля установленного доступа к своему файлу после каждого изменения кода защиты нужно проверять свои действия с помощью команды **ls -l.**

Примеры:

chmod g+rw,o+r file.1 - установка атрибутов чтения и записи для группы и чтения для всех остальных пользователей;

ls -l file.1 - чтение атрибутов файла;

chmod o-w file.1 - отмена атрибута записи у остальных пользователей;

>letter - создание файла letter. Символ > используется как для переадресации, так и для создания файла;

cat - вывод содержимого файла;

cat file.1 file.2 > file.12 - конкатенация файлов (объединение);

mv file.1 file.2 - переименование файла file.1 в file.2;

mv file.1 file.2 file.3 directory - перемещение файлов file.1, file.2, file.3 в указанную директорию;

rm file.1 file.2 file.3 - удаление файлов file.1, file.2, file.3;.

cp file.1 file.2 - копирование файла с переименованием;

mkdir namedir - создание каталога;

rm dir_1 dir_2 - удаление каталогов dir_1 dir_2;

ls [acdfgilqrstv CFR] namedir - вывод содержимого каталога; если в качестве namedir указано имя файла, то выдается вся информация об этом файле. Значения аргументов:

- l –- список включает всю информацию о файлах;
- t – сортировка по времени модификации файлов;
- a – в список включаются все файлы, в том числе и те, которые начинаются с точки;
- s – размеры файлов указываются в блоках;
- d – вывести имя самого каталога, но не содержимое;
- r – сортировка строк вывода;
- i – указать идентификационный номер каждого файла;
- v – сортировка файлов по времени последнего доступа;
- q – непечатаемые символы заменить на знак ?;
- c – использовать время создания файла при сортировке;
- g – то же что -l, но с указанием имени группы пользователей;
- f – вывод содержимого всех указанных каталогов, отменяет

флаги -l, -t, -s, -r и активизирует флаг -a;

- C – вывод элементов каталога в несколько столбцов;

- F – добавление к имени каталога символа / и символа * к имени файла, для которых разрешено выполнение;

- R – рекурсивный вывод содержимого подкаталогов заданного каталога.

cd <namedir> - переход в другой каталог. Если параметры не указаны, то происходит переход в домашний каталог пользователя.

pwd - вывод имени текущего каталога;

grep [-vcilns] [шаблон поиска] <имя файла> - поиск файлов с указанием или без указания контекста (шаблона поиска).

Значение ключей:

- v – выводятся строки, не содержащие шаблон поиска;

- c – выводится только число строк, содержащих или не содержащих шаблон;

- i – при поиске не различаются прописные и строчные буквы;

- l – выводятся только имена файлов, содержащие указанный шаблон;

- n – перенумеровать выводимые строки;

- s – формируется только код завершения.

Примеры.

1. Напечатать имена всех файлов текущего каталога, содержащих последовательность "student" и имеющих расширение .txt:

grep -l student *.txt

2. Определить имя пользователя, входящего в ОС LINUX с терминала tty23:

who | grep tty23

3. Порядок выполнения работы

1. Ознакомиться с файловой структурой ОС LINUX. Изучить команды работы с файлами.

2. Используя команды ОС LINUX, создать два текстовых файла.

3. Полученные файлы объединить в один файл и его содержимое просмотреть на экране.

4. Создать новую директорию и переместить в нее полученные файлы.

5. Вывести полную информацию обо всех файлах и проанализи-

ровать уровни доступа.

6. Добавить для всех трех файлов право выполнения членам группы и остальным пользователям.

7. Просмотреть атрибуты файлов.

8. Создать еще один каталог.

9. Установить дополнительную связь объединенного файла с новым каталогом, но под другим именем.

10. Создать символическую связь.

11. Сделать текущим новый каталог и вывести на экран расширенный список информации о его файлах.

12. Произвести поиск заданной последовательности символов в файлах текущей директории и получить перечень соответствующих файлов.

13. Получить информацию об активных процессах и имена других пользователей.

14. Сдать отчет о работе и удалить свои файлы и каталоги.

15. Выйти из системы.

4. Контрольные вопросы

1. Что считается файлами в ОС LINUX?

2. Объясните назначение связей с файлами и способы их создания.

3. Что определяет атрибуты файлов и каким образом их можно просмотреть и изменить?

4. Какие методы создания и удаления файлов, каталогов Вы знаете?

5. В чем заключается поиск по шаблону?

6. Какой командой можно получить список работающих пользователей и сохранить его в файле?

ЛАБОРАТОРНАЯ РАБОТА № 3

СОЗДАНИЕ И ВЫПОЛНЕНИЕ КОМАНДНЫХ ФАЙЛОВ В СРЕДЕ ОС LINUX

1. Цель работы

Целью работы является изучение методов создания и выполнения командных файлов на языке Shell - интерпретатора.

2. Теоретическая часть

В предыдущих лабораторных работах взаимодействие с командным интерпретатором Shell осуществлялось с помощью командной строки. Однако, Shell является также и языком программирования, который применяется для написания командных файлов (shell - файлов). Командные файлы также называются скриптами и сценариями. Shell - файл содержит одну или несколько выполняемых команд (процедур), а имя файла в этом случае используется как имя команды.

2.1. Переменные командного интерпретатора

Для обозначения переменных Shell используется последовательность букв, цифр и символов подчеркивания; переменные не могут начинаться с цифры. Присваивание значений переменным проводится с использованием знака = , например, PS2 = '<' . Для обращения к значению переменной перед ее именем ставится знак $. Их можно разделить на следующие группы:

- позиционные переменные вида $n, где n - целое число;
- простые переменные, значения которых может задавать пользователь или они могут устанавливаться интерпретатором;
- специальные переменные # ? - ! $ устанавливаются интерпретатором и позволяют получить информацию о числе позиционных переменных, коде завершения последней команды, идентификационном номере текущего и фонового процессов, о текущих флагах интерпретатора Shell.

Простые переменные. Shell присваивает значения переменным:

```
z=1000
x= $z
echo $x
1000
```

Здесь переменной x присвоено значение z.

Позиционные переменные. Переменные вида $n, где n - целое число, используются для идентификации позиций элементов в командной строке с помощью номеров, начиная с нуля. Например, в командной строке

cat text_1 text_2...text_9

аргументы идентифицируются параметрами $1...$9. Для имени команды всегда используется S0. В данном случае $0 - это cat, $1 -

text_1, $2 - text_2 и т.д. Для присваивания значений позиционным переменным используется команда **set**, например:

set arg_1 arg_2... arg_9

здесь $1 присваивается значение аргумента arg_1, $2 - arg_2 и т.д.
Для доступа к аргументам используется команда **echo**, например:

echo $1 $2 $9

arg_1 arg_2 arg_9

Для получения информации обо всех аргументах (включая последний) используют метасимвол *. Пример:

echo $*

arg_2 arg_3 ... arg_10 arg_11 arg_12

С помощью позиционных переменных Shell можно сохранить имя команды и ее аргументы. При выполнении команды интерпретатор Shell должен передать ей аргументы, порядок которых может регулироваться также с помощью позиционных переменных.

Специальные переменные. Переменные - ? # $! устанавливаются только Shell. Они позволяют с помощью команды **echo** получить следующую информацию:

- – текущие флаги интерпретатора (установка флагов может быть изменена командой **set**);

– число аргументов, которое было сохранено интерпретатором при выполнении какой-либо команды;

? – код возврата последней выполняемой команды;

$ – числовой идентификатор текущего процесса PID;

! – PID последнего фонового процесса.

2.2. Арифметические операции

Команда **expr** (express -- выражать) вычисляет выражение expression и записывает результат в стандартный вывод. Элементы выражения разделяются пробелами; символы, имеющие специальный смысл в командном языке, нужно экранировать. Строки, содержащие специальные символы, заключают в апострофы. Используя команду **expr**, можно выполнять сложение, вычитание, умножение, деление, взятие остатка, сопоставление символов и т. д.

Пример. Сложение, вычитание:

b=190

a=` expr 200 - $b`

где ` - обратная кавычка (левая верхняя клавиша). Умножение *, де-

ление /, взятие остатка %:

d=` expr $a + 125 "*" 10`

c=` expr $d % 13`

Здесь знак умножения заключается в двойные кавычки, чтобы интерпретатор не воспринимал его как метасимвол. Во второй строке переменной *c* присваивается значение остатка от деления переменной *d* на 13.

Сопоставление символов с указанием числа совпадающих символов:

concur=` expr "abcdefgh" : "abcde"`

echo $concur

ответ 5.

Операция сопоставления обозначается двоеточием (:). Результат - переменная concur.

Подсчет числа символов в цепочках символов. Операция выполняется с использованием функции *length* в команде **expr**:

chain="The program is written in Assembler"

str=` expr length "$chain"`

Echo $str

ответ 35. Здесь результат подсчета обозначен переменной *str*.

2.3. Встроенные команды

Встроенные команды являются частью интерпретатора и не требуют для своего выполнения проведения последовательного поиска файла команды и создания новых процессов. Встроенные команды:

cd [dir] - назначение текущего каталога;

exec [cmd [arg...]] <имя файла> - выполнение команды, заданной аргументами cmd и arg, путем вызова соответствующего выполняемого файла.

umask [-o | -s] [nnn] - устанавливает маску создания файла (маску режимов доступа создаваемого файла, равную восьмеричному числу nnn: 3 восьмеричных цифры для пользователя, группы и других). Если аргумент nnn отсутствует, то команда сообщает текущее значение маски. При наличии флага -o маска выводится в восьмеричном виде, при наличии флага -s - в символьном представлении;

set, unset - режим работы интерпретатора, присваивание значений параметрам;

eval [-arg] - вычисление и выполнение команды;

sh <filename.sh> выполнение командного файла filename.sh;

exit [n] - приводит к прекращению выполнения программы, возвращает код возврата, равный нулю, в вызывающую программу;

trap [cmd] [cond] - перехват сигналов прерывания, где: cmd - выполняемая команда; cond=0 или EXIT - в этом случае команда cmd выполняется при завершении интерпретатора; cond=ERR - команда cmd выполняется при обнаружении ошибки; cond - символьное или числовое обозначение сигнала, в этом случае команда cmd выполняется при приходе этого сигнала;

export [name [=word]...] - включение в среду. Команда **export** объявляет, что переменные name будут включаться в среду всех вызываемых впоследствии команд;

wait [n] - ожидание завершения процесса. Команда без аргументов ожидает завершения процессов, запущенных синхронно. Если указан числовой аргумент n, то **wait** ожидает фоновый процесс с номером n;

read name - команда вводит строку со стандартного ввода и присваивает прочитанные слова переменным, заданным аргументами name.

Пример. Пусть имеется shell-файл *data*, содержащий две команды:

```
echo -n "Please write down your name:"
read name
```

Если вызвать файл на выполнение, введя его имя, то на экране появится сообщение:

Please write down your name:

Программа ожидает ввода с клавиатуры (в данном случае - фамилии пользователя). После ввода фамилии и нажатия клавиши *Enter* команда выполнится и на следующей строке появится знак - приглашение.

2.4. Управление программами

Команды **true** и **false** служат для установления требуемого кода завершения процесса: true - успешное завершение, код завершения 0; false - неуспешное завершение, код может иметь несколько значений, с помощью которых определяется причина неуспешного завершения. Коды завершения команд используются для принятия решения о дальнейших действиях в операторах цикла **while** и **until** и в условном операторе **if**. Многие команды LINUX вырабатывают код завершения

только для поддержки этих операторов.

Условный оператор if проверяет значение выражения. Если оно равно true, Shell выполняет следующий за **if** оператор, если false, то следующий оператор пропускается. Формат оператора **if**:

if <условие>
then
list1
else
list2
fi

Команда **test** (проверить) используется с условным оператором **if** и операторами циклов. Действия при этом зависят от кода возврата **test**. **Test** проводит анализ файлов, числовых значений, цепочек символов. Нулевой код выдается, если при проверке результат положителен, ненулевой код при отрицательном результате проверки.

В случае анализа файлов синтаксис команды следующий:

test [-rwfds] file

где

-r – файл существует и его можно прочитать (код завершения 0);
-w – файл существует и в него можно записывать;
-f – файл существует и не является каталогом;
-d – файл существует и является каталогом;
-s – размер файла отличен от нуля.

При анализе числовых значений команда **test** проверяет, истинно ли данное отношение, например, равны ли A и B . Сравнение выполняется в формате:

-eq		A = B
-ne		A <> B
test A -ge B	эквивалентно	A >= B
-le		A <= B
-gt		A > B
-lt		A < B

Отношения слева используются для числовых данных, справа – для символов.

Кроме команды **test** имеются еще некоторые средства для проверки:

! - операция отрицания инвертирует значение выражения, например, выражение **if test true** эквивалентно выражению **if test ! false**;

o - двуместная операция "ИЛИ" (or) дает значение true, если один из операндов имеет значение true;

a - двуместная операция "И" (and) дает значение true, если оба операнда имеют значение true.

2.5. Циклы

Оператор цикла с условием **while** true и **while** false. Команда **while** (пока) формирует циклы, которые выполняются до тех пор, пока команда **while** определяет значение следующего за ним выражения как true или false. Формат оператора цикла с условием **while** true:

```
while       list1
      do
            list2
      done
```

Здесь list1 и list2 - списки команд. **While** проверяет код возврата списка команд, стоящих после **while**, и если его значение равно 0, то выполняются команды, стоящие между **do** и **done**. Оператор цикла с условием **while** false имеет формат:

```
until       list1
      do
            list2
      done
```

В отличие от предыдущего случая условием выполнения команд между **do** и **done** является ненулевое значение возврата. Программный цикл может быть размещен внутри другого цикла (вложенный цикл). Оператор **break** прерывает ближайший к нему цикл. Если в программу ввести оператор **break** с уровнем 2 (**break 2**), то это обеспечит выход за пределы двух циклов и завершение программы.

Оператор **continue** передает управление ближайшему в цикле оператору **while**.

Оператор цикла с перечислением **for**:

```
for name    in    [wordlist]
      do
            list
      done
```

где name - переменная; wordlist - последовательность слов; list - список команд. Переменная name получает значение первого слова по-

следовательности wordlist, после этого выполняется список команд, стоящий между **do** и **done**. Затем name получает значение второго слова wordlist и снова выполняется список list. Выполнение прекращается после того, как кончится список wordlist.

Ветвление по многим направлениям **case**. Команда **case** обеспечивает ветвление по многим направлениям в зависимости от значений аргументов команды. Формат:

```
case <string> in
s1) <list1>;;
s2) <list2>;;
 .
 .
 .
sn) <listn>;;
*) <list>
esac
```

3десь list1, list2 ... listn - список команд. Производится сравнение шаблона string с шаблонами s1, s2 ... sk ... sn. При совпадении выполняется список команд, стоящий между текущим шаблоном sk и соответствующими знаками ;;. Пример:

```
echo -n 'Please, write down your age'
read age
case $age in
test $age -le 20) echo 'you are so young' ;;
test $age -le 40) echo 'you are still young' ;;
test $age -le 70) echo 'you are too young' ;;
*)echo 'Please, write down once more'
esac
```

В конце текста помещена звездочка * на случай неправильного ввода числа.

3. Порядок выполнения работы

Составьте и выполните shell - программы, включающей следующие действия:

1. Вывод на экран списка параметров командной строки с указанием номера каждого параметра.

2. Присвоение переменным A, B и C значений 10, 100 и 200, вы-

числение и вывод результатов по формуле D=(A*2 + B/3)*C.

3. Формирование файла со списком файлов в домашнем каталоге, вывод на экран этого списка в алфавитном порядке и общего количества файлов.

4. Переход в другой каталог, формирование файла с листингом каталога и возвращение в исходный каталог.

5. Запрос и ввод имени пользователя, сравнение с текущим логическим именем пользователя и вывод сообщения: верно/неверно.

6. Запрос и ввод имени файла в текущем каталоге и вывод сообщения о типе файла.

7. Циклическое чтение системного времени и очистка экрана в заданный момент.

8. Циклический просмотр списка файлов и выдача сообщения при появлении заданного имени в списке.

4. Контрольные вопросы

1. Какое назначение имеют shell - файлы?
2. Как создать shell - файл и сделать его выполняемым?
3. Какие типы переменных используются в shell - файлах?
4. В чем заключается анализ цепочки символов?
5. Какие встроенные команды используются в shell - файлах?
6. Как производится управление программами?
7. Назовите операторы создания циклов.

ЛАБОРАТОРНАЯ РАБОТА № 4

ФОРМИРОВАНИЕ СИСТЕМНОГО ГИБКОГО ДИСКА ОС LINUX

1. Цель работы

Целью работы является изучение принципов взаимодействия LINUX- системы с внешними устройствами и формирования системных и инсталляционных дисков.

2. Теоретическая часть

2.1. Монтирование и демонтирование файловой системы

Файловое дерево формируется из отдельных частей, называемых файловыми системами. Каждая файловая система состоит из од-

ного корневого каталога, его подкаталогов и файлов. Файловые системы прикрепляются к файловому дереву с помощью команды **mount**. Эта команда берет из существующего файлового дерева каталог (называется точкой монтирования) и делает его корневым каталогом присоединяемой файловой системы. Например, команда

mount /dev/sd0a /users

монтирует файловую систему, находящуюся на устройстве /dev/sd0a, под именем users. После монтирования с помощью команды **ls /users** можно посмотреть, что содержит эта файловая система.

Таким образом, в системе LINUX вся файловая система представлена как единое дерево каталогов. Аналогично монтируется сетевая файловая система. Например:

mount host.asu.ugatu.ac.ru:/users /husers

где сетевой диск /users на машине host.asu.ugatu.ac.ru монтируется как каталог /husers. Монтирование внешних устройств при их использовании необходимо выполнять, если в системе нет соответствующих настроек для их автоматического монтирования.

Демонтирование файловой системы можно выполнить командой **umount**. Для этого в файловой системе не должно быть открытых файлов и процессов, ее использующих. То есть она должна быть незанятой. Пример размонтирования файловой системы /users

umount /users

Подключение диска CD-ROM к системе выполняется командой

mount /cdrom

Для снятия диска его необходимо размонтировать командой

umount /cdrom

Формат функции **mount** может включать ключ t со знаком минус (-t), за которым следует параметр, определяющий тип файловой системы и может принимать следующие значения:

vfat либо msdos - файловая система на основе FAT;

ext2 – файловая система типа UNIX;

minix – файловая система, соответствующая стандарту POSIX для взаимодействия между различными платформами;

qnx – тип файловой системы QNX, поддерживается только для чтения;

ufs – файловая система BSD, только для чтения;

ntfs – файловая система Windows NT, но только для чтения.

Поддерживается также ряд экспериментальных файловых сис-

тем, например, ext3 - журнальная файловая система, extfs - файловая система с криптографической защитой и т.д. Например, монтирование дискеты, отформатированной в DOS, к каталогу /media/floppy:

mount -t vfat /dev/fd0 /media/floppy

или

mount /media/floppy

где /dev/fd0 - системное имя файла-устройства на гибких дисках.

Дискета, отформатированная в LINUX-системе, может быть смонтирована следующим образом

mount -t ext2 /dev/fd0 /mnt/a

Командой **df** или **mount** (без ключей) можно посмотреть, какие файловые системы смонтированы и какой объем они имеют.

2.2. Форматирование дисков

Форматирование дисков выполняется в два этапа. На первом этапе непосредственно производится форматирование (несмонтированного) диска. Форматирование можно выполнить командой **fdformat** (на 1,44 Мбайт) с проверкой

fdformat /dev/fd0u1440

На втором этапе создается корневая файловая система с использованием системной функции **mke2fs (mkfs)** с аргументами:

mke2fs -m ext2 /dev/fd0

Создание файловой системы в стиле MS DOS можно выполнить командой **mformat** без проверки (приставка m позволяет выполнять DOS - команды в среде Linux: **mdir**, **mcopy** и др.)

mformat a:

При форматировании и создании файловой системы диск должен быть размонтирован.

2.3. Запись системных образов на дискеты

Можно выполнить загрузку ОС Linux на трех дискетах. Для этого необходимо скопировать на дискеты из каталога /home/bootdisk двоичные образы файлов bootdisk, modules1 и rescuefloppy из существующей системы следующими командами:

а) запись образа bootdisk на первую дискету с помощью команды, которая может работать с любой UNIX - системой

dd if=bootdisk of=/dev/fd0

или запись образа командой, предназначенной для работы с ОС Linux (не на Linux может работать некорректно)

cat bootdisk > /dev/fd0

б) запись образа modules1 на вторую дискету

cat modules1 > /dev/fd0

в) запись образа modules1 на третью дискету

cat rescuefloppy > /dev/fd0

Эти двоичные образы файлов при их загрузке создают ядро операционной системы, корневую файловую систему и электронный диск в оперативной памяти.

2.4. Загрузка системы с дискет

Сформированная на дискетах система не поддерживает многопользовательский режим работы. Загрузка системы начинается с диска, содержащего образ bootdisk. На сообщение системы о загрузке Linux выбрать в меню пункт manual installation, нажать клавишу *Enter* для инсталляции системы с последующей загрузкой. При формировании запросов в возникающих диалоговых окнах последовательно заменять дискеты в дисководе. Загрузка завершается появлением значка-приглашения системы #.

3. Порядок выполнения работы

1. Отформатировать в системе Linux три гибких диска.
2. Создать на дискете файловую систему MS DOS.
3. Смонтировать дискету на каталог /media/floppy.
4. Проверить результат монтирования командой df.
5. Создать на дискете рабочий каталог и в него записать текстовый файл (создать новый, либо скопировать с жесткого диска).
6. Просмотреть содержимое корневой файловой системы гибких дисков и рабочего каталога.
7. Запретить изменение данных на гибких дисках и отдельно в рабочем каталоге методом изменения атрибутов соответствующего файла.
8. Проверить установленный уровень доступа к гибким дискам и рабочему каталогу.
9. Скопировать рабочий каталог с гибкого диска на жесткий диск и убедиться, что функция копирования выполнена успешно.
10. Размонтировать устройство на гибких дисках и убедиться,

что эта операция выполнена успешно с использованием команды df.

11. На первую дискету скопировать образ bootdisk, на вторую – modules1, и на третью – rescuefloppy.

12. Выполнить загрузку компьютера с гибких дисков.

16. Завершить работу системы командой *halt*.

4. Контрольные вопросы

1. Каким образом можно отформатировать диск для поддержки файловой системы на основе FAT, NTFS и для взаимодействия разных типов файловых систем.

2. В чем заключаются функции монтирования и размонтирования файловой системы и какими командами они выполняются в ОС Linux.

3. Какие образы требуются для создания гибкого системного диска ОС Linux, где они размещаются.

4. Каким способом можно перенести образы на формируемый системный диск.

5. Как ограничить доступ к устройствам на гибких дисках.

6. Какие режимы работы может поддерживать ОС Linux на гибких дисках и какие режимы работы нельзя подключить.

ЛАБОРАТОРНАЯ РАБОТА № 5
Изучение графической оболочки KDE

1. Цель работы

Целью работы является изучение работы с основными функциональными частями графической оболочки KDE, получение навыков по настройке KDE и создании простейших текстовых и графических документов в KWord и Paint.

2. Общие теоретические сведения

K Desktop Environment (Среда рабочего стола K) KDE предназначена для поддержания тех же функциональных возможностей графического интерфейса, какие предоставляют и другие популярные системы, например MacOS и Windows. Кроме выполнения стандартных функций, KDE обладает рядом специфических характеристик, которые расширяют возможности графической среды. Для Linux разработано несколько диспетчеров окон, таких, как *olwm, fvwm, after-*

step и другие. Однако, их возможности не идут ни в какое сравнение с возможностями KDE.

2.1 Оконная среда KDE

Как и большинство оконных менеджеров, KDE представляет собой интегрированную среду, содержащую базовые средства для решения ряда повседневных задач. Например, с помощью **KDE** можно выполнять ряд операций:

• Размещение на рабочем столе ярлыков гибких дисков для их монтирования, размонтирования и работы с ними.

• Отображение в графическом виде файловой структуры и перемещение по ней.

• Сопоставление приложений с файлами определенных типов. При этом если щелкнуть на выбранном файле, автоматически будет загружаться нужное приложение.

• Создание на рабочем столе ярлыков принтеров. Если мышью перетащить к такому ярлыку файл, он будет распечатан.

В состав KDE входит не только рабочий стол, но и целый набор приложений и утилит для работы с ним. В стандартном дистрибутиве KDE имеется более сотни программ — от игр и системных утилит до целых блоков офисных программ. Кроме того, приложения KDE могут взаимодействовать друг с другом для упрощения выполнения всевозможных операций.

2.2 Компоненты рабочего стола KDE.

Рабочий стол KDE разделен на три основные части - "поверхность" рабочего стола, панель и линейку задач. Основная рабочая область среды KDE называется рабочим столом. Это тот фон, на котором отображаются все другие компоненты. На рабочем столе можно размещать ярлыки программ, документов и устройств, к которым чаще всего приходится обращаться. Это позволяет легко получать доступ к соответствующим объектам для работы с ними. Кроме той области, что отображается на экране, KDE предоставляет дополнительное виртуальное рабочее пространство для выполнения программ. По умолчанию поддерживается четыре виртуальных рабочих стола. Виртуальный рабочий стол — это, по сути, другой экран, на который можно переключиться для того, чтобы запустить приложение или выполнить еще какую-то работу. Программы и окна легко переме-

щаются между различными виртуальными рабочими столами. Дополнительные возможности, предоставляемые за счет использования виртуальных рабочих столов, могут быть использованы самыми разными программами. При этом нет необходимости сворачивать и разворачивать окна выполняемых приложений. Можно просто отложить выполняемое приложение в таком виде, как есть, а затем вернуться к нему по завершении выполнения.

2.2.1 Панель

Панель располагается в нижней части экрана. На панели размещаются кнопки, позволяющие выполнять основные процедуры KDE, а также ярлыки наиболее часто используемых программ. Одним из особо важных элементов на панели является кнопка Application Starter (Запуск Приложений), которая расположена (по умолчанию) в левой части панели. Это кнопка с литерой "К" над изображением зубчатого колеса. С ее помощью можно открыть меню, в котором представлены все приложения, установленные на данную систему. Кроме того, это же меню может быть использовано для доступа к некоторым другим разделам KDE, таким, как диалоговая справка и *Панель Управления (Control Panel).*

На панели размещен переключатель виртуальных рабочих столов *Пейджер, Панель Задач (Taskbar)* и *Часы (Clock).* Панель задач отображает открытые на текущем рабочем столе окна. Чтобы получить немедленный доступ к программе, нужно просто щелкнуть в соответствующем месте на панели задач.

Запустить на выполнение программу можно одним из перечисленных ниже способов.

• **Щелкнуть кнопкой на панели.** Некоторые программы представлены по умолчанию на панели в виде ярлыков или кнопок, например эмулятор виртуальных рабочих столов, панель управления, вызов справки и текстовой редактор.

• **Щелкнуть на элементе рабочего стола.** По умолчанию на рабочем столе размещается только два объекта. Это *Корзина* и ярлык рабочего каталога. Пользователи сами размещают на рабочем столе наиболее нужные и часто используемые программы.

• **Выбрать программу из меню запуска приложений.** Достаточно щелкнуть на литере "К" и выбрать тот пункт меню, который соответствует запускаемому приложению.

• **Использовать диспетчер файлов.** В окне диспетчера файлов нужно выбрать соответствующий файл и щелкнуть на нем мышью.

Можно, конечно, запустить программу на выполнение в командной строке окна терминала – задать название программы. Можно также нажатием клавиш <Alt+F2> вызвать окно запуска программ и ввести туда название программы.

Ряд полезных программ облегчает работу пользователя.

В первую очередь, это программа эмуляции терминала konsole, позволяющая открывать окна и получать доступ к стандартной командной строке. На панели имеется соответствующая кнопка с изображением маленького монитора и ракушки.

Справку в диалоговом режиме можно получить, если щелкнуть на кнопке панели с изображением спасательного круга. Справка включает в себя разные темы, как, например, программа-гид для начинающих пользователей и система контекстного поиска для используемых в KDE приложений.

Просмотреть файловую систему или получить доступ к ресурсам World Wide Web можно, используя окно диспетчера файлов. Для того чтобы диспетчер файлов отобразил в своем окне содержимое рабочего каталога, нужно щелкнуть на папке панели с изображением домика.

Щелканье по кнопке ► удаляет панель с экрана. Эта кнопка остается при этом на экране, так что можно вернуть панель обратно. Это свойство действует только на открытый в данный момент рабочий стол; другие рабочие столы сохраняют вид мини - или главной панели.

Список задач - кнопка, расположенная справа от меню приложений (обозначена пиктограммой монитора), несет меню, содержащее все активные на данный момент окна, отсортированные по имени. Это позволяет легко и быстро найти необходимое окно и уменьшает захламленность экрана при работе с несколькими окнами.

2.3.2 Настройка KDE

Центр управления (Control Center) (кнопка с изображением гаечного ключа) составляет основу всей системы настроек KDE. В ее входит множество панелей для всевозможных компонентов рабочей среды и даже некоторых приложений KDE.

В центре управления используется деление на группы, щелкнув

на знаке "плюс" в углу группы, можно увидеть список входящих в группу компонентов. Щелкнув на знаке "минус" в том же углу группы, этот список можно свернуть. Доступ к любому диалогу с раскрытым деревом меню можно получить при помощи Preferences (Предпочтения) и из меню запуска программ Start Application.

Большинство диалоговых окон имеют кнопку вызова справки. В самом простом случае это контекстная справка. Для ее получения нужно щелкнуть мышью на знаке вопроса на рамке окна. Курсор мыши при этом изменит свой вид на стрелку с большим знаком вопроса. Если теперь щелкнуть на том элементе диалогового окна, с которым возникли трудности, появится прямоугольник желтого цвета с текстом справки. Для получения более детальной справки можно воспользоваться опцией Help (Справка) на левой панели. Наконец, если возникли проблемы с поиском необходимого диалогового окна, на этой же панели нужно выбрать опцию Search (Поиск). Затем нужно ввести ключевое слово, по которому и будет осуществляться поиск.

Control Center

KDE предоставляет широкие возможности по модифицированию внешнего вида окон и рабочей области, включая отображение фона, ярлыков, шрифтов и тому подобное. Не представляет труда и управление работой отдельных компонентов, вроде того же рабочего стола или окна. Например, можно управлять реакцией элемента на щелчок мышью, процессом загрузки и отображения выбранных окон, выбирать хранитель экрана. Все эти и многие другие возможности может предоставить рассматриваемая группа Control Center.

Для настройки параметров работы рабочего стола и окон следует выбрать опцию нужного диалога настройки под названием Desktop на дереве центра управления.

Изменение схемы цветов

Диалоговое окно выбора цвета Appearance&Themes $\Rightarrow$ Colors (Цвета) предназначено для изменения используемой цветовой схемы для окон KDE и других графических приложений.

Цветовая схема включает в себя 18 пунктов выбора цвета для различных элементов окна программы и установки контрастов. В области предварительного просмотра отображаются все элементы окна, реагирующие на изменение цветовой схемы. Как только пользователь меняет параметры или установки, в области просмотра отражаются

внесенные изменения. Можно выбрать уже готовую цветовую схему из списка Color Scheme (Схема Цветов).

Для изменения какой-то конкретной установки нужно выбрать соответствующий элемент из выпадающего меню области цветов Widget Color (Декорация) или щелкнуть в нужной части окна предварительного просмотра. После того как элемент выбран, можно изменить его цвет. Для этого достаточно щелкнуть на кнопке и выбрать понравившийся цвет из появившегося диалогового окна выбора цвета.

Контраст изменяется при помощи позиционирования специального рычажка контраста, который может размещаться в диапазоне от Low (Низкий) до High (Высокий). Эти установки применяются при отображении трехмерных рамок вокруг элементов интерфейса приложений KDE.

Для подтверждения выбора следует щелкнуть на кнопке Apply (Применить). Если приходится часто менять цветовые установки, бывает полезно внести изменения в список цветовых схем. Для этого нужно щелкнуть на кнопке Save Scheme и задать название для своей схемы. Для удаления схемы из списка нужно выделить ее и щелкнуть на кнопке Remove (Удалить).

Изменение фона

Для изменения цвета фона или фонового узора рабочего стола нужно на дереве опций центра управления последовательно выбрать Control Center=>Appearance&Themes=>Background. В результате появится диалоговое окно, имеющее три основные области:

- список виртуальных рабочих столов;
- окно предварительного просмотра;
- окно настройки параметров.

Каждый виртуальный стол в KDE имеет собственные настройки фона. Для каждого такого стола можно выбрать фон с одноцветной или двухцветной палитрой, а также фоновый узор. Если используется фоновый узор, можно задать способ его отображения. Можно также выбрать несколько узоров и автоматически переключаться между ними. Доступны и более усовершенствованные опции, позволяющие сочетать цвета и узоры, а также поддерживать динамические настройки фона.

В процессе внесения изменений в установки они отображаются в окне предварительного просмотра.

Виртуальные рабочие столы

Производить настройку параметров виртуальных рабочих столов в KDE можно в диалоговом окне Control Center ⇒Desktop ⇒Multiple Desktops.

Указатель Number of Desktops (Количество рабочих столов) показывает, сколько виртуальных рабочих столов доступно. Их число может изменяться в диапазоне от одного до шестнадцати. Здесь же можно задать название для рабочего стола, которое потом будет отображено в списке окон (Window List) или использовано в настройках панели.

Хранитель экрана

Диалоговое окно выбора хранителя экрана Appearance&Themes ⇒ Screensaver позволяет выбрать хранитель экрана и осуществить настройку его параметров. Опции настройки бывают глобальные, как, например, опция установки времени запуска хранителя экрана, и индивидуальные — для каждого отдельного хранителя. Диалоговое окно выбора хранителя экрана имеет три основные секции:

- окно предварительного просмотра;
- список программ — хранителей экрана;
- опции настройки.

Необходимо выбрать название нужной программы из предложенного списка. Для настройки параметров необходимо щелкнуть на кнопке Setup (Настройка) и в появившемся диалоговом окне произвести установку нужных характеристик.

Для установки интервала времени, через который будет запускаться хранитель экрана, нужно ввести в поле опции Settings (Установки) величину данного интервала в минутах.

Параметр Priority (Приоритет) позволяет определить распределение процессором времени на работу хранителя экрана. Это пример того, как в Linux организована многозадачность. Если нужно, чтобы у хранителя был наивысший приоритет (например, для качественного вывода анимации), следует передвинуть рычажок в позицию High (Высокий). Если же, наоборот, необходимо обеспечить высокий приоритет других процессов, нужная позиция для рычажка приоритетности хранителя — Low (Низкий).

Для того чтобы проверить выполненные установки, следует щелкнуть на кнопке Test (Просмотр). Для подтверждения сделанного выбора нажмите кнопку OK или Apply.

Настройка диспетчера окон

С помощью опций Control Center Appearance&Themes ⇒ Desktop/Window behavior (Поведение Окон) центра управления можно устанавливать поведение диспетчера окон. Эти настройки определяют способ отображения окон в случае их перемещения и изменения размера, а также управляют процессом разворачивания, размещения и выделения окон при работе с диспетчером окон. Опции в верхней части диалогового окна позволяют выполнить настройку параметров, задающих режим перемещения окна и изменения его размеров, а также определяют функциональность команды Maximize (Развернуть). Можно задать такой режим отображения окна при перемещении или изменении его размера, что окно будет отображаться вместе со всем своим содержимым или же в виде прозрачной рамки. Если выбран режим отображения всего содержимого окна, процесс перемещения или изменения размеров окна будет требовать дополнительного времени для обновления отображаемых на экране элементов.

Если используются окна с изменяемыми размерами, можно выбрать режим обновления содержимого окна при каждом изменении его размера. Для этого следует воспользоваться установками Resize (Изменение размера). Для выбора частоты обновления можно воспользоваться специальным рычажком. Если сделан выбор, отличный от None (Никакой), то каждый раз, при изменении размеров окна, его содержимое будет обновляться. Это дает возможность отслеживать процесс заполнения окна программой и позволяет выбрать оптимальные размеры последнего.

Window behavior/Moving – меню установок размещения окна на экране Placement (Расположение) позволяет определить место на экране, где будет отображаться окно. Поддерживаются такие методы.

• **Smart (Умный) —** минимизируется перекрытие между окнами.

• **Cascade (Каскад) —** первое окно отображается в левом верхнем углу. Следующее окно отображается сдвинутым немного вправо и вниз, так что окна практически полностью перекрываются. И так далее. Окна расположены, как карты в руке при игре в преферанс.

• **Random (Произвольный) —** окна располагаются на экране в произвольном порядке.

Метод получения фокуса (т.е. метод выделения отдельных окон или элементов) является, пожалуй, индивидуальным методом настро-

ек KDE. С помощью этого метода определяется, какое из открытых окон активно и какие следует выполнить действия при активизации окна. Более детально это выглядит так.

• **Click to focus (Передача фокуса щелчком).** Окно получает фокус (т.е. становится активным) при щелчке на нем мышью. При этом окно автоматически выводится на первый план по отношению к другим окнам. Такой метод используется по умолчанию.

• **Focus follows mouse (Фокус за мышью).** Окно получает фокус при непосредственном обращении к нему (это можно сделать с помощью указателя мыши, используя комбинацию клавиш <Alt+Tab> и тому подобное). При этом окно может подниматься поверх других окон, а может и не подниматься. Перемещение указателя мыши на рабочую область за пределы окна не означает потерю последним фокуса. При выборе опции Auto Raise (Всплывать автоматически) окно будет всплывать на экране при перемещении в его область курсора в течение нескольких миллисекунд. Число этих миллисекунд устанавливается с помощью рычажка Delay (Задержка). Если выбрана опция Click Raise (Всплывать при щелчке), окно будет подниматься поверх других окон при щелчке в любой части окна. В противном случае такая реакция окна будет наблюдаться только при щелчке на его заголовке. Это исключительно полезный метод передачи фокуса, поскольку позволяет набирать текст в одном окне и одновременно читать содержимое другого окна, расположенного частично поверх указанного.

• **Focus Under Mouse (Фокус под мышью).** Окно получает фокус при любом перемещении на него указателя мыши. При этом комбинация клавиш <Alt+Tab> может и не помочь.

• **Focus Strictly Under Mouse (Фокус только под мышью).** Окно получает фокус, только если указатель мыши находится внутри окна. Если указатель мыши находится в рабочей области, где нет окон, ни одно из окон не получит фокус.

Использование окон

Открытое окно состоит из следующих элементов.

Window menu (Меню управления окном) - В левом верхнем углу каждого окна находится пиктограмма манипулирования окном. При щелчке на ней появляется меню, содержащее команды с помощью которых можно манипулировать данным окном. Maximize (Максимизировать) увеличит окно до максимально возможного размера. Mini-

mize (Минимизировать) сделает ваше окно невидимым. Move (Переместить) позволяет передвигать окно с помощью мыши. Size (Изменить размер) позволит вам увеличить или уменьшить окно. Shade – свернет окно до заголовка. To desktop...(На рабочий стол) позволит перевести окно на другой рабочий стол. Выберите рабочий стол, на который вы хотите переместить это окно. Окно при этом исчезнет. Для того чтобы увидеть его снова, выберите имя на Линейке задач, или щелкните на соответствующую кнопку рабочего стола на панели KDE. Close (Закрыть) закроет данное окно. Always on Top – оставляет окно поверх всех открытых окон.

Использование панели меню каждого окна в KDE очень просто. Щелкните на команду, и она будет исполнена. При нажатии на правую кнопку мыши появится контекстное меню, позволяющее вывести на экран панель меню. Можете отсоединить меню от окна и оставить его "плавать" по экрану.

Ниже панели меню находятся пиктограммы инструментов, которые позволяют исполнять различные команды. Можно передвинуть инструментальную панель - влево, вправо, вверх, вниз, и, конечно, она тоже может "плавать".

3. Порядок выполнения работы

1. Запустите Центр управлений.

2. Поменяйте Фон, сначала на одноцветный, а затем вставьте фоновое изображение.

3. Установите хранитель экрана, на своё усмотрение, и режим ожидания равный минуте.

4. Сделайте так, чтобы окна передвигались вместе со всем их содержимым.

5. Задайте звуковой щелчок, подтверждающий нажатие каждой клавиши.

6. Измените ширину линейки панели.

7. Запустите диспетчер приложений. И запустите программу текстового процессора KWord.

8. В другом рабочем столе откройте программу растрового редактора Paint.

9. Откройте KWord и наберите следующий текст:

`The Quick Brown Fox Jumps Over The Lazy Dog`, используя два разных стиля по вашему выбору. Сохраните этот файл в

домашнем каталоге пользователя, закройте KWord.

10. Откройте ваш домашний каталог пользователя Konqueror'ом, создайте в нем каталог, скопируйте ваш текстовой файл в этот каталог.

11. Ознакомьтесь с содержанием домашнего каталога, скопируйте с дискеты файлы.

12. Получите справку об интересующем вас объекте.

13. Создайте любой рисунок с помощью Paint, чтобы в нем были ВСЕ фигуры (1. эллипс, 2. окружность, 3. линия, 4. прямоугольник, 5. круг) хотя бы по одному разу и присутствовало не менее четырех цветов.

14. Сохраните файл с рисунком в домашнем каталоге, закройте Paint.

15. Скопируйте файл с рисунком в тот же созданный вами каталог.

16. Измените атрибуты доступа к созданным файлам.

17. Покажите преподавателю ваши файлы, затем удалите их.

4. Контрольные вопросы

1. Перечислите стандартные функции KDE.
2. Что является компонентом рабочего стола KDE?
3. Назовите функции панели рабочего стола.
4. Как получить справку в диалоговом режиме?
5. Какие функции предоставляет центр управления KDE?

ЛАБОРАТОРНАЯ РАБОТА №6
Файловый менеджер Konqueror Web Browser

1. Цель работы

Целью работы является получение основных навыков работы с файловом менеджером Konqueror.

2. Общие теоретические сведения

Будучи программой для просмотра и управления файловыми системами, Konqueror также является клиентом протокола FTP, броузером World Wide Web, работает с архивными файлами, изображениями и может делать многое другое. Диспетчер файлов Konqueror является универсальным и позволяет просматривать не только файлы

и содержимое каталогов, но и отображать многие другие объекты. Реализовать такие широкие возможности удалось за счет использования в Konqueror универсальных указателей ресурсов (URL) для всех просматриваемых путей, а также за счет использования подключаемых модулей для отображения информации самого разного типа.

2.1. Возможность обработки объектов

Диспетчер файлов Konqueror предназначен не только для просмотра документов. Он может также быть использован для настройки просматриваемых объектов и активной работы с ними. Например, он использует систему типизации файлов KDE для запуска программ и загрузки документов. Распознавание типа файла производится как для локальных файлов, так и для файлов на удаленных узлах. Поэтому, таких проблем, как, скажем, выбор отображаемых ярлыков или контекстных меню для данного элемента, не возникает. При запуске файла на удаленной системе KDE загрузит этот файл и запустит нужное для работы с ним приложение. Можно перетаскивать элементы из одного окна диспетчера файлов в другое, создавая, таким образом, копию элемента или связь с этим элементом. Наконец, при работе с локальной файловой системой, Konqueror может быть использован для изменения атрибутов этой самой файловой системы (как, например, владелец файла и права доступа) с помощью простого графического диалогового окна.

Прежде чем начинать работать с диспетчером файлов, полезно сначала познакомиться с некоторыми элементами интерфейса пользователя. Ниже описаны те задачи, которые можно решать с помощью Konqueror.

2.2. Работа с файлами и каталогами.

Чтобы начать работу с диспетчером файлов, достаточно щелкнуть на папке любого каталога или щелкнуть кнопкой с изображением глобуса на панели. В результате запустится диспетчер файлов, и в главном окне будет выведено содержимое рабочего каталога. Эта область называется областью просмотра. Как правило, в основном окне такая область просмотра единственная. Тем не менее, там также есть область, где отображается дерево каталогов, которая вместе с областью просмотра занимает место в основном окне при его отображении на экране. Наконец, там еще можно увидеть и область эмуляции

окна терминала.

2.2.1. Область просмотра

В области просмотра, как правило, отображается содержимое выбранного каталога. Для обозначения элементов в каталоге используются ярлыки, которые определяются типом файлов. Вид окна просмотра можно изменить. Для этого нужно воспользоваться опциями меню View (Вид)⇒View Mode (Способ отображения). Существует пять отображаемых типов (согласно опциям меню View).

• **Icon View (ярлыки).** Содержимое каталога отображается в виде больших ярлыков, помещенных в рамку.

• **Text View (Текст).** Выводится детальный листинг файлов и каталогов со всеми их атрибутами.

• **MultiColumn View (Колонки).** Выводится в виде колонок только название файлов и их мини-ярлыки.

• **Detailed List View (Список).** Выводится та же информация, что и в режиме Text View, только включая еще и мини-ярлыки для идентификации типа файлов.

• **Tree View (Дерево).** Все точно так же, как в предыдущем случае, только теперь каждый ярлык может быть развернут в дерево подкаталогов.

Обычно скрытые файлы (те, имена которых начинаются с точки) в описанных выше списках файлов не отображаются. Для того чтобы включить такие файлы в листинги, нужно выбрать опции меню View(Вид)⇒Show Hidden Files (Показывать скрытые файлы). Переходить между каталогами с помощью диспетчера файлов можно несколькими способами. Для перехода в подкаталог нужно щелкнуть на названии папки или каталога в области просмотра. Для перехода в каталог высшего уровня можно щелкнуть на кнопке со стрелкой вверх на панели инструментов диспетчера файлов. Для переключения между каталогами, которые уже посещались, можно использовать кнопки со стрелками влево и вправо на той же панели инструментов. Каждая из этих кнопок имеет маленькую стрелочку, направленную вниз. Это значит, что если удерживать кнопку нажатой, можно будет увидеть список адресов, куда можно перейти. Для кнопки со стрелкой вверх такой список будет состоять из каталогов вышестоящих уровней – первого, второго и так далее. Для кнопок со стрелками влево и вправо это будут каталоги, которые последовательно посещались. Их хроно-

логический порядок записан в кэш-памяти.

Для переключения на элемент, помеченный закладкой, нужно выбрать соответствующий пункт из меню закладок Bookmarks (Закладки).

Наконец, для перехода в нужное место файловой системы можно просто ввести адрес перехода в поле Location (Адрес), размещенном в верхней части окна диспетчера файлов. Это же можно сделать и при помощи выпадающего диалогового окна Open Location (Открыть Адрес). Для этого достаточно выбрать опции меню Location ⇒Open Location. К тому же результату приведет нажатие комбинации клавиш <Ctrl+O>. При вводе адреса его можно указывать как обычный путь к каталогу или как URL. При работе с локальной файловой системой следует использовать префикс file.

2.2.2. Дерево каталогов

Левое подокно окна диспетчера файлов, как правило, используется для отображения дерева каталогов. По умолчанию оно скрыто, но при помощи опций в меню диспетчера файлов Window (Окно) ⇒Show Navigation Panel (Показывать дерево директорий) его можно вывести на экран. Подокно Tree View (Просмотр дерева) имеет три каталога наивысшего уровня, соответствующих тем областям, в которых производится просмотр файловой системы.

• **The Home Directory (Рабочий каталог).** Соответствует рабочему каталогу пользователя.

• **The Network (Сетевая папка).** Эта папка содержит еще три папки: FTP Archives (Архивы FTP) для работы с FTP узлами; Web Sites (Страницы Web), в которой хранятся используемые закладки; Windows Shares (Сетевые ресурсы Windows), используемая для доступа к совместно используемым ресурсам с помощью SMB.

• **The Root Directory (Корневой каталог).** Соответствует корневому каталогу файловой системы.

В подокне просмотра дерева каталогов отображаются только каталоги. Файлы и связи там не показаны. Для того чтобы развернуть или свернуть каталог, нужно щелкнуть на квадрате со знаком плюс или минус соответственно, размещенном слева от названия каталога. Когда каталог свернут, в квадрате отображается плюс, когда же каталог раскрыт, в квадрате появляется минус. Для отображения в окне просмотра содержимого каталога нужно в подокне просмотра дерева

каталогов щелкнуть мышью на названии этого каталога (при этом должна быть установлена связь между окнами, подробнее об этом рассказано в разделе "Установка связи между окнами").

2.2.3. Окно эмуляции терминала

В нижней части окна диспетчера файлов можно вывести эмуляцию окна терминала (что-то вроде консольного устройства). Для этого достаточно выбрать опции меню Window(Окно)⇒Show Terminal emulator (Показать окно эмуляции терминала). Это позволит получить доступ к командной строке, где можно обычным способом вводить команды LINUX. Когда с помощью дерева каталогов или в окне просмотра пользователь переходит в другой каталог, изменение текущего каталога будет автоматически отображаться и в этом окне (при установленной связи между окнами). А вот перемещения, выполняемые в окне эмуляции терминала с помощью команды **cd**, не изменят содержимого области просмотра и подокна дерева каталогов.

2.2.4. Установка связи между окнами

Изменения, внесенные в области просмотра диспетчера файлов или подокна дерева каталогов, могут отражаться и в других окнах. По умолчанию все окна связаны друг с другом, так что они будут отражать одни и те же каталоги. Иногда бывает полезно убрать такую связь для отдельного окна, чтобы оно отражало какой-то один каталог.

Для установки или снятия связи между окнами, нужно воспользоваться опцией View ⇒ Link View (Связать). Между всеми выделенными таким способом окнами устанавливается связь, поэтому они будут отображать одинаковые каталоги. Единственным исключением является эмулятор окна терминала. При использовании команды **cd** эмулятор терминала с другими окнами работать синхронно не будет.

2.2.5. Создание окон

Три окна, используемые в диспетчере файлов, — это только начало. Пользователь может создать несколько копий окна просмотра или эмулятора терминала. Причем каждая копия может работать с разными каталогами или узлами Web.

Для создания нового окна нужно выбрать существующее окно того типа, который нужно создать. Затем следует выбрать одну из перечисленных ниже опций меню.

• Window (Окно)⇒Split View Left/Right (Разделить по вертикали) – текущее окно разбивается по вертикали на два окна того же типа. Это же можно сделать, используя комбинацию клавиш <Ctrl+Shift+L>.

• Window (Окно)⇒Split View Top/Bottom (Разделить по горизонтали) — текущее окно разбивается по горизонтали на два окна того же типа. Комбинация клавиш в данном случае <Ctrl+Shift+T>.

Для создаваемых окон по умолчанию связи с другими окнами не устанавливаются. Это удобно для того, чтобы просматривать разные каталоги. Для изменения просматриваемой области нужно сначала просто щелкнуть мышью на окне. Маленький зеленый индикатор указывает активное в данный момент окно. Затем следует ввести новый адрес для просмотра в поле Location в верхней части окна Konqueror (Location ⇒ Duplicate Windows).

Для того чтобы убрать существующее окно просмотра или эмулятор терминала, нужно выделить его щелчком мыши, а затем выбрать опции контекстного меню Window⇒Remove Active View (Убрать окно) или нажать комбинацию клавиш <Ctrl+Shift+R>. Для изменения размера нужно при помощи мыши переместить границу между двумя соседними окнами.

2.2.6. Сохранение формата

Для использования созданного пользователем формата для отображения на экране диспетчера файлов, этот формат нужно сохранить. Для этого следует воспользоваться опциями Settings(Установки) => Save View Profile (Сохранить профиль).

Сначала для профиля нужно задать название или воспользоваться уже существующим. Затем следует определить, нужно ли сохранять в профиле URL (опция Save URLs in profile). Если соответствующая опция выбрана, то каждый раз при загрузке профиля будет происходить обращение к URL.

2.3. Задачи управления

В этом разделе описываются те задачи по управлению файловой системой, которые можно решать с помощью диспетчера файлов. Сюда следует включить получение информации о файлах, копирование, перемещение и удаление файлов, изменение таких атрибутов файлов, как название, владельцы и права доступа к ним.

2.3.1. Получение информации о файле

Одна из самых основных процедур заключается в получении информации о файле. Получить ее можно самыми разнообразными способами.

Например, панель состояния отображает данные о размере и типе объекта, который выделен при помощи курсора мыши. Поэтому для получения такой информации достаточно навести указатель мыши на интересующий объект.

Более детальную информацию о файле можно получить, если в меню View/View Mode (Вид) выбрать опции Text View (Текст) или Detailed List (Список). В этом случае о каждом элементе в выведенном на экран списке можно узнать такие подробности, как тип, размер, название, время изменения, права доступа, владелец, группа и наличие связей. Ярлыки в окне просмотра, устанавливающиеся KDE автоматически, соответствуют типу каждого элемента. В KDE для отображения файлов различных типов используется большое число ярлыков. По негласному соглашению для каталогов используется ярлык в виде папки, для документов — ярлыки с изображением листа бумаги, а для программ — ярлык с изображением зубчатого колеса.

2.3.2. Выбор элемента

Некоторые действия с объектами можно выполнять прямо в окне диспетчера файлов. Существует много способов, с помощью которых можно выделять группы объектов и производить с ними разнообразные процедуры.

Для выбора объекта без его запуска нужно щелкнуть на нем мышью, удерживая при этом нажатой клавишу <Ctrl>. Объект будет при этом затемнен. Это означает, что он выделен. Для того чтобы к выделенному объекту добавить еще один, или убрать объект из группы выделенных объектов, используется та же описанная процедура. Можно выделить группу объектов, захватив их в рамку при помощи курсора мыши.

Большую группу объектов, которую неудобно или просто невозможно выделить с помощью мыши, можно выделить, используя названия и спецификацию шаблона. Для этого нужно в меню Edit (Правка) выбрать опцию Select (Выделить) или нажать комбинацию клавиш цифровой панели <Ctrl+Плюс>. Затем в диалоговом окне Se-

lect files (Выделить файлы) следует ввести названия файлов или спецификацию шаблона. Затем нужно щелкнуть на кнопке ОК, после чего будут выделены все файлы, отвечающие заданному шаблону. Аналогично, из группы выделенных файлов можно и убирать файлы. Для этого в меню Edit следует выбрать опцию Unselect (Отменить выделение) или нажать комбинацию клавиш цифровой панели < Ctrl+Минус>. Если воспользоваться опцией Edit⇒UnselectAll (Отменить для всех), будет отменено выделение для всех выделенных до этого объектов. К тому же результату приводит нажатие комбинации <Ctrl+U>. Опция Edit⇒Invert (Наоборот) приведет к выделению невыделенных файлов и отмене выделения для выделенных. То же можно сделать, нажав <Ctrl+*>.

2.3.3. Перемещение и копирование файлов

Наиболее простой способ переместить или скопировать файл из одного места файловой системы в другое или создать связь с файлом заключается в том, чтобы выделить его и просто перетащить мышью в нужное место. Таким способом можно перетаскивать файлы из одного открытого окна диспетчера файлов в другое, между окном диспетчера файлов и рабочим столом.

Иногда бывает трудно определить, какая из операций — копирование (Copy), или перемещение (Move) – является наиболее приемлемой. Это особенно актуально для тех, кто начинает работу с особыми файлами и каталогами KDE, вроде рабочего стола (Desktop). Ниже приведены некоторые соображения по этому поводу.

• *Если действительно необходимо создать копию объекта*, следует выбирать Copy и только Copy. Для программ такая процедура копирования используется редко. Что касается документов и других подобных файлов, то все зависит от конкретных обстоятельств. Здесь следует помнить, что если используется несколько копий одного файла, то внесение изменений в одну из этих копий на других копиях не отражается. Поэтому, хотя и можно размещать документы прямо на рабочем столе, желательно размещать там только связи с документом, или перемещать туда файлы только на время.

• *Если нужно изменить место хранения файла,* следует выбирать Move. Как и в предыдущем случае, эта процедура редко используется для программ и командных файлов. Программы обычно хранятся в специальных каталогах, что отражено в указании используе-

мых в приложениях путей. Перемещение программы в другое место может привести к тому, что она станет недоступной для использования при вызове из командной строки.

• *Для файлов конфигурации рабочего стола* обычно используется команда копирования или создается связь. Иногда смещение файла рабочего стола с привычного места приводит к некорректной его работе. Например, файл рабочего стола MimeType может быть использован, только если он находится в каталоге mimelink. Поэтому если нет уверенности в правильности предпринимаемых действий, перемещать файлы рабочего стола из их специальных каталогов в другие не стоит.

2.3.4. Удаление файлов

Для того чтобы удалить файл из файловой системы, **можно** воспользоваться одним из трех способов: переместить файл в корзину (Trash), непосредственно удалить файл, или вытереть его. Помещение файла в корзину означает его перемещение в каталог Trash, т.е. файл будет продолжать занимать место в системе и его можно будет в будущем, если потребуется, восстановить. Непосредственное удаление файла означает, что файл из системы полностью удаляется с освобождением места. В этом случае восстановление файла невозможно. Вытирание подразумевает предварительную запись в файл набора специальных данных перед удалением. Это делается для того, чтобы быть уверенным, что даже самая совершенная технология восстановления файлов не сможет восстановить его содержимое.

Для перемещения файла в корзину нужно в меню Edit или контекстном меню выбрать опцию Move to Trash (Поместить в Корзину). Можно также просто перетащить нужный объект из окна диспетчера файлов на пиктограмму с изображением корзины на рабочем столе. Аналогично, для удаления файла (или файлов) нужно выделить этот файл (или файлы) и затем выбрать опцию Delete (Удалить) из одного из упоминавшихся выше меню.

2.3.5. Запуск файлов

Запускать файлы из окна диспетчера файлов можно так же, как это делается при использовании рабочего стола. Можно либо просто щелкнуть на выбранном объекте, либо перетащить объект к нужной программе, либо войти в контекстное меню документа и с помощью

опции Open With (Открыть с помощью) выбрать программу из списка.

Если щелкнуть один раз мышью на объекте, KDE выберет необходимый способ действий, исходя из типа файла. Если KDE не в состоянии определить программу, используемую по умолчанию для работы с файлом, KDE предложит пользователю самостоятельно выбрать такую программу.

2.3.6. Изменение файлов и каталогов

В KDE, за счет использования простого и понятного графического диалога для работы с объектами, изменение атрибутов объектов файловой системы является очень простой задачей. Для получения доступа к этому диалоговому окну нужно выбрать опцию Properties (Свойства) из контекстного меню объекта. После этого появится диалоговое окно со вкладками, разными для объектов разного типа. Но первые две вкладки одинаковы для объектов всех типов. Это вкладка General (Общие) и Permissions (Доступ).

2.3.6.1. Изменение названия файла

Для того, чтобы изменить название файла, нужно выбрать из контекстного меню этого файла опцию Properties. После того, как появится окно диалога, на вкладке General нужно отредактировать название файла в поле Name (Название), затем нажать кнопку ОК.

2.3.6.2. Замена владельца и изменение прав доступа

Для замены владельца файла и прав доступа к нему в диалоговом окне Properties нужно перейти на закладку Properties/Permissions (Права доступа). Для изменения прав доступа к объекту в секции Access permissions (Права доступа) диалогового окна следует напротив нужных опций поставить флажки. Чтобы изменить владельца файла или его группы, нужно воспользоваться элементами управления в секции Ownership (Принадлежность).

3. Порядок выполнения работы

1. Откройте окно диспетчера файлов Konqueror, щелкнув на кнопке Home на панели.
2. Разверните окно.
3. Выберите опцию Window⇒Show Terminal Emulator.

4. Выберите опцию Window⇒Split View Left/Right.

5. Переместите панель кнопок Button и панель адреса Location с помощью мыши.

6. Измените размеры панелей, используя специальные метки изменения размера на границах окон.

7. Запустите на выполнение в правой панели команду vi.

8. На левой панели просмотра откройте аплет, нажав клавиши <Ctrl+O>.

9. На нижней панели воспользуйтесь компилятором и другими средствами командной строки.

10. Выберите опцию Setting⇒Save View Profile для сохранения профиля.

11. Введите название профиля, а затем выберите опцию Save window size in profile. После этого нужно для сохранения установок щелкнуть на кнопке Save.

4. Контрольные вопросы

1. Какие программы называются файловыми менеджерами?

2. Какая информация отражается в области просмотра программы Konqueror?

3. Как создать новое окно с помощью программы Konqueror?

4. Перечислите задачи по управлению файловой системой, которые можно решать с помощью диспетчера файлов?

Список литературы

1. Скловская А.М. Команды LINUX. Справочник. Изд-во Диасофт. 2004. – 848 с.

2. Моли Б. Unix/Linux: Теория и практика программирования. Изд-во КУДИЦ-ОБРАЗ, 2004. – 576 с.

3. Бендел Д., Нейпир Р. Использование Linux. /Пер.с англ. - М.: издательский дом "Вильямс", 2004. - 784 с.

4. Немет Э., Снайдер Г., Сибас С., Хейн Т.Р. UNIX: руководство системного администратора. Для профессионалов / Пер. с англ. – СП.: Питер; К.: Издательская группа BHV, 2002. – 928 с.

5. http://www.linuxjournal.com.

6. http://pluto.xTech.RU/Russian/Unix-Doc/ - сервер Новосибирского института систем информатики. Содержит книги и документацию по UNIX на русском языке.

Приложение

ПРИНЦИПЫ РАБОТЫ И ОСНОВНЫЕ КОМАНДЫ ТЕКСТОВОГО РЕДАКТОРА VI

В составе ОС LINUX обычно поставляются текстовые редакторы: **ed** - интерактивный строковый редактор, **vi** и **ex** - его расширенные версии. Под именем **vi** (**visual interpretator** - визуальный интерпретатор) эта программа работает как экранно-ориентированный редактор, а под именем **ex** - как строчно-ориентированный.

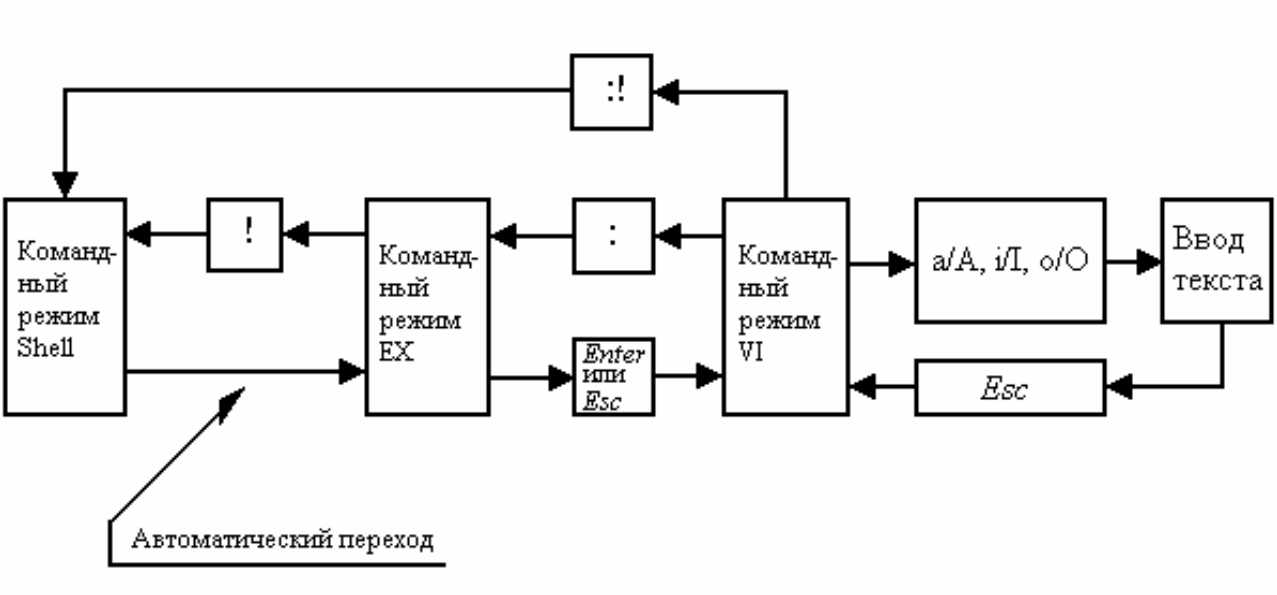
Для вызова редактора **vi** используется команда **vi**:

vi [+line] [-R] [-x] [-r] [-t] file...

где +line - номер строки, с которой Вы хотите начать редактирование; R - читать; это означает, что файл можно только просматривать, но не модифицировать; x - расшифровывающее чтение т.е. просмотр файла, зашифрованного командой **crypt**, или редактирование обычного текста с последующим шифрованием по мере записи на диск; r - восстановление файла после системного или программного крахов; t - вызов для редактирования файла, который содержит названный (в поле file команды **vi**) тег (tag). Тег - это список символов, с которых начинается раздел в текстовом файле. Теги разных файлов объединяют в один файл - файл тегов с именем tags. Опцией -t обеспечивается вызов файла tags, который содержит названный тег и имя редактируемого файла, в котором тег находится. Команду вызова редактора можно использовать в форме **vi +/word/file** - начало редактирования файла file с первой строки, которая содержит слово word, или в форме **vi +file** - начало редактирования файла с последней строки.

Структура редактора

Работая с редактором, пользователь находится или в одном из его командных режимов, или в режиме ввода текста. Ниже приведенная схема иллюстрирует взаимодействие этих режимов и способы перехода редактора между ними.



В простейшем случае для вызова редактора нужно ввести команду **vi** text и нажать клавишу *Enter*. На экране появится:

$ vi text

_

~

.

.

"text"

Строка начинается знаком ~, знак _ определяет положение курсора. В данный момент пользователь находится в командном режиме **vi**. Перейти в режим ввода текста можно с помощью команд добавления текста, которые не отображаются на экране после их ввода:

a/A - ввод текста после курсора/после конца строки (append - присоединение);

i/I - вставка текста перед курсором/с 1-й позиции данной строки (insert - вставить);

o/O - образовать пустую строку ниже имеющейся / выше имеющейся.

Для выполнения команд (например, записи в файл, перемещения курсора) после введения текста или его части нужно перейти снова в командный режим **vi**, нажав клавишу *Esc*. После вызова **vi** нажмите клавишу a (ввод текста после курсора), не нажимая после этого клавишу *Enter*, и Вы попадете в режим ввода текста. Вводите текст, нажимая клавишу *Enter* в конце каждой строки (курсор в режиме ввода текста можно перемещать вправо, используя клавишу "*пробел*", и влево, используя клавишу *BackSpace*.

Переход в командный режим vi. Для перехода в командный режим vi нужно нажать клавишу *Esc*. Теперь редактор находится в командном режиме vi. В этом режиме выполняются следующие команды:

. - повторение последней команды;

u - аннулирование действия последней команды;

Изучение других многочисленных команд этого командного режима целесообразно проводить, разбив их на тематические группы. Они приведены в разделе 2.2.

Переход в режим ex. Чтобы перейти к группе команд редактора ex (под именем ex редактор работает как строчно-ориентированный), нужно ввести символ : (двоеточие), команду и нажать <Enter> или Esc. Команды редактора ex начинаются с символа : и отображаются в нижней части экрана. После нажатия клавиши Esc или <Enter> происходит возврат (назад) в командный режим. Команды режима ex:

:w - запись текста в файл;

:r - чтение файла;

:e - редактирование нового файла;

:e! - выход без сохранения данного файла и редактирование нового;

:n - авторедактирование;

:wq - запись текста и выход из редактора;

:x - запись текста только при наличии в нем изменений;

:q! - оставить текст в рабочей области и закончить редактирование;

:ab - присвоение сокращений;

:map - определение ключей;

:set - изменение установочных режимов;

:s - выполнение замещений.

Переход в Shell. Редактор позволяет в процессе работы с ним выполнять команды ОС LINUX. Для этого нужно перейти в командный режим Shell с помощью команды !.

Рассмотрим пример. Определите текущее время командой date (вывод и установка даты) :! date. Здесь символ : означает переход в командный режим ex, а символ ! дает доступ к Shell. Для продолжительной работы с командами Shell можно вызвать командой :bash и после окончания работы вернуться в редактор vi, набрав CTRL-D.

Для возврата в командный режим vi нажмите клавишу *Enter*.

Команды, выполняемые в командном режиме VI

Изучим группу команд режима vi: перемещения курсора, добавления текста, поиска (частично), изменения и смещения текста, удаления,

замены букв. Команды vi не отображаются на экране, кроме команд поиска, начинающихся со знаков / ? перемещение курсора, управление экраном дисплея, добавление текста.

Многие команды редактора выполняются только при определенном положении курсора, и нужно уметь пользоваться клавишами управления курсором (клавиши со стрелками <- , -> и т.д.). Кроме клавиши со стрелками для перемещения курсора можно использовать клавиши: CTRL-H - влево; CTRL-N - вниз; CTRL-P - вверх; SPACE - вправо.

Команды перемещения курсора:

h - на одну позицию влево;

l - на одну позицию вправо;

j - на одну позицию вниз;

k - на одну позицию вверх;

b - к первому символу предыдущего слова;

B - то же самое, что b, но игнорируются знаки пунктуации;

w - к первому символу следующего слова;

W - то же самое, что w, но игнорируются знаки пунктуации;

e - к последнему символу следующего слова;

E - то же самое, что e, но игнорируются знаки пунктуации;

(- к началу текущего предложения (предложение считается законченным, если после него есть два пробела или пустая строка);

) - к концу текущего предложения;

{ - к началу текущего раздела (разделителем раздела является пустая строка);

} - к концу текущего раздела;

[- к началу текущей секции;

] - к концу текущей секции;

^ - к первому отображаемому символу на текущей строке;

O - к началу текущей строки;

$ - к концу текущей строки;

H - к началу экрана;

M - на середину экрана;

L - к концу экрана;

nG - к строке с номером n (на последнюю строку, если номера n нет); % - к символу парной скобки, если курсор находится под одной из них.

Команды управления экраном:

^U - смещение текста на одну строку вверх (CTRL-U);

^D - смещение текста на одну строку вниз (CTRL-D);

^B - смещение текста на один кадр назад (CTRL-B);

^F - смещение текста на один кадр вперед (CTRL-F).

Чтобы переместить текущую строку:

• в верхнюю часть экрана нужно ввести команду z и нажать клавишу *Enter*;

• в середину экрана z;

• в нижнюю часть экрана z- .

Для очистки экрана от сообщений нужно использовать команды CTRL-R и CTRL-L; тексты в рабочей области при этом сохраняются.

Команды изменения текста:

cw - изменение слова;

cW - то же самое, что и cw, но игнорируются знаки пунктуации;

cO - от начала текущей строки;

c$ - до конца текущей строки;

cc - изменение всей строки;

c(- от начала текущего предложения;

c) - до конца текущего предложения;

c{ - от начала текущего раздела;

c} - до конца текущего раздела.

Для внесения изменений в текст необходимо: переместить курсор в нужную позицию; ввести команду изменения; без пробела набрать новый текст; нажать клавишу ESC.

Во всех командах можно использовать множители n, например для изменения пяти слов используется команда c5w.

Команды поиска начинаются косой чертой / (поиск вперед по тексту) или знаком ? (поиск назад); далее следует номер строки или ключевое слово. Команда заканчивается нажатием клавиши *Enter*.

Команды смещения текста:

<(или>(- к началу текущего предложения;

<)или>) - к концу текущего предложения;

<{или>{ - к началу текущего раздела;

<}или>} - к концу текущего раздела.

В командах смещения текста можно использовать множители, например может использоваться команда 2>> (сдвиг вправо). Смещение устанавливается командой: set sw=m. По умолчанию m=8. После того как курсор подведен к требуемой строке, нужно набрать символы << или >>.

Удаление, замена строчных букв на прописные и наоборот. Для удаления текста/фрагмента нужно переместить курсор в требуемую

позицию и ввести команду удаления.

dw - до конца текущего слова;

dW - то же, что и dw, но игнорируются знаки пунктуации;

d^ - до 1-го видимого символа текущей строки;

dO - удаление начала строки;

d$ - удаление конца строки;

d(- до начала текущего предложения;

d) - до конца текущего предложения;

d{ - до начала текущего раздела;

d} - до конца текущего раздела;

dd - удаление всей строки;

dkw - удаление k слов;

dk)/dk} - удаление k предложений, k разделов;

kdd - удаление k строк.

Для удаления одиночного символа нужно подвести к нему курсор и набрать x (не d), а для удаления нескольких символов подряд набрать команду nx.

Для удаления текста от начала строки до определенного места и от определенного места до конца строки используются команды d^ и d$ соответственно.

Символ ~ используется для замены строчных букв на прописные и наоборот. Замена 1-й буквы в последней строке текста:

- Введите символ ((к началу текущего предложения).
- Наберите команду .~
- Восстановите текст командой u.

Определение текущей рабочей позиции в файле. После ввода пользователем в командном режиме CTRL-G в нижней части экрана появится статусная информация в соответствии с положением курсора в тексте, включающая: имя файла; сведения о проведенной ранее модификации; номер текущей строки; общее число строк; расстояние курсора от начала файла (в процентах).

Для окончания работы с редактором введите в командном режиме :wq (запись текста из рабочей области в файл и окончание редактирования) и нажмите клавишу *Enter*. На экране появится сообщение о том, что Вы вышли из редактора и находитесь в Shell:

:wq <*Enter*>

/home/student >

Составители: Лянцев Олег Дмитриевич
Еникеев Рустем Радомирович
Колесников Андрей Александрович
Тараракo Павел Иванович

РАБОТА ПОЛЬЗОВАТЕЛЯ В ОПЕРАЦИОННОЙ СИСТЕМЕ LINUX

МЕТОДИЧЕСКИЕ УКАЗАНИЯ

к лабораторному практикуму по курсу

"Операционные системы" для студентов специальностей
220200 – Автоматизированные системы обработки информации и управления и
351400 – Прикладная информатика в экономике

Подписано к печати 30.12.2004 Формат 60x84 1/16.
Бумага писчая. Печать плоская. Усл. печ. л.2,25.
Усл. кр. –отт. 2,0. Уч. –изд. л. 2,0. Тираж 100 экз.
Заказ №
Уфимский государственный авиационный технический университет
Центр оперативной полиграфии УГАТУ
450000, Уфа-центр, ул. К. Маркса, 12