Министерство образования и науки Российской Федерации (РФ)

ТОМСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ СИСТЕМ
УПРАВЛЕНИЯ И РАДИОЭЛЕКТРОНИКИ (ТУСУР)

Кафедра Электронных приборов (ЭП)

УТВЕРЖДАЮ
Заведующий кафедрой ЭП
_____ С.М. Шандаров

Архитектура вычислительных систем

Учебно-методическое пособие
Методические указания к лабораторным работам

Разработчик:
ст. преподаватель каф. ЭП
_____ Е.С. Шандаров

Томск 2011

Шандаров Е.С.

Архитектура вычислительных систем: Учебно-методическое пособие. – Томск: Томский государственный университет систем управления и радиоэлектроники, 2011. - 52 с.

Оглавление

Лабораторная работа №1. Определение технических параметров компьютера

Цель работы

Научиться определять технические параметры компьютера с помощью служебных программ операционной системы Linux.

Теоретическая часть

Linux — свободно распространяемая многозадачная, многопользовательская операционная система.

Ядро Linux разработано Линусом Торвальдсом в 1991 г. Файлы первая версия Linux (версия 0.01) были опубликованы в Интернете 17 сентября 1991 года.

Если быть более точными, то Linux — это только ядро, когда же речь заходит об операционной системе, то более правильно говорить «Операционная система на основе ядра Linux». Ядро ОС Linux разрабатывается под общим руководством Линуса Торвальдса и распространяется свободно (на основе лицензии GPL)

К основным характеристикам Linux можно отнести многозадачность, многопользовательский доступ и разграничение прав доступа к файлам , поддержка различных форматов файловых систем .

В силу того, что исходные коды Linux распространяются свободно и общедоступны, к развитию системы с самого начала подключилось большое число независимых разработчиков. Благодаря этому на сегодняшний момент Linux — самая современная, устойчивая и быстроразвивающаяся система, почти мгновенно вбирающая в себя самые последние технологические новшества. Она обладает всеми возможностями, которые присущи современным полнофункциональным операционным системам типа UNIX.

Реальная многозадачность

Все процессы независимы; ни один из них не должен мешать выполнению других задач. Для этого ядро осуществляет режим разделения времени центрального процессора, поочередно выделяя каждому процессу интервалы времени для выполнения.

Многопользовательский доступ

Linux — не только многозадачная ОС, она поддерживает возможность одновременной работы многих пользователей. При этом Linux может предоставлять все системные ресурсы пользователям, работающим с хостом через различные удаленные терминалы.

Свопирование оперативной памяти на диск

Свопирование оперативной памяти на диск позволяет работать при ограниченном объеме физической оперативной памяти; для этого содержимое некоторых частей (страниц) оперативной памяти записываются в выделенную область на жестком диске, которая трактуется как дополнительная оперативная память. Это несколько снижает скорость работы, но позволяет организовать работу программ, требующих большего объема ОЗУ, чем фактически имеется в компьютере.

Страничная организация памяти

Системная память Linux организована в виде страниц объемом 4K. Если оперативная память полностью исчерпана, ОС будет искать давно не использованные страницы памяти для их перемещения из памяти на жесткий диск. Если какие-либо из этих страниц становятся нужны, Linux восстанавливает их с диска. Некоторые старые Unix-системы и некоторые современные платформы (включая Microsoft Windows) переносят на диск все содержимое ОП, относящееся к неработающему в данный момент приложению, (т.е. ВСЕ страницы памяти, относящиеся к приложению, сохраняются на диске при нехватке памяти) что менее эффективно.

Загрузка выполняемых модулей "по требованию"

Ядро Linux поддерживает выделение страниц памяти по требованию, при котором только необходимая часть кода исполняемой программы находится в оперативной памяти, а не используемые в данный момент части остаются на диске.

Совместное использование исполняемых программ

Если необходимо запустить одновременно несколько копий какого-то приложения (либо один пользователь запускает несколько идентичных задач, либо

разные пользователи запускают одну и ту же задачу), то в память загружается только одна копия исполняемого кода этого приложения, которая используется всеми одновременно исполняющимися идентичными задачами.

Общие библиотеки

Библиотеки — наборы процедур, используемых программами для обработки данных. Существует некоторое количество стандартных библиотек, используемых одновременно более чем одним процессом. В старых системах такие библиотеки включались в каждый исполняемый файл, одновременное выполнение которых приводило к непродуктивному использованию памяти. В новых системах (в частности, в Linux), обеспечивается работа с динамически и статически разделяемыми библиотеками, что позволяет сократить размер отдельных приложений.

Динамическое кеширование диска

Кеширование диска — это использование части оперативной памяти для хранения часто используемых данных с диска, что существенно ускоряет доступ к часто используемым программам и задачам. Пользователи MS-DOS работают со SmartDrive, который резервирует фиксированные области системной памяти для кеширования диска. Linux использует более динамичную систему кеширования: память, зарезервированная под кеш, увеличивается, когда память не используется, и уменьшается, если системе или процессу пользователя требуется больше памяти.

Поддержка различных форматов файловых систем

Linux поддерживает большое число форматов файловых систем, включая файловые системы DOS и OS/2, а также современные журналируемые файловые системы. При этом и собственная файловая система Linux, которая называется Second Extended File System (ext2fs), позволяет эффективно использовать дисковое пространство.

Сетевые возможности

Linux можно интегрировать в любую локальную сеть. Поддерживаются все службы Unix, включая Networked File System (NFS), удаленный доступ (telnet, rlogin), работа в TCP/IP сетях, dial-up-доступ по протоколам SLIP и PPP, и т. д.. Также

поддерживается включение Linux-машины как сервера или клиента для другой сети, в частности, работает общее использование (sharing) файлов и удаленная печать в Macintosh, NetWare и Windows.

Работа на разных аппаратных платформах

Хотя ОС Linux первоначально была разработана для ПК на базе Intel 386/486, сейчас она может работать на всех версиях Intel-овских микропроцессоров, начиная с 386 и кончая многопроцессорными системами на Pentium. Так же успешно Linux работает на различных клонах Intel от других производителей. Кроме того, разработаны версии для других типов процессоров — ARM, DEC Alpha, SUN Sparc, M68000 (Atari и Amiga), MIPS, PowerPC и других.

Дистрибутивы Linux

В любой операционной системе можно выделить 4 основных части: ядро, файловую структуру, интерпретатор команд пользователя и утилиты. Ядро — это основная, определяющая часть ОС, которая управляет аппаратными средствами и выполнением программ. Файловая структура — это система хранения файлов на запоминающих устройствах. Интерпретатор команд или оболочка — это программа, организующая взаимодействие пользователя с компьютером. И, наконец, утилиты — это просто отдельные программы, которые, вообще говоря, ничем принципиально не отличаются от других программ, запускаемых пользователем, разве только своим основным назначением — они выполняют служебные функции.

Как уже говорилось выше, если быть точным, то слово "Linux" обозначает только ядро. Поэтому, когда речь идет об операционной системе, правильнее было бы говорить "операционная система, основанная на ядре Linux". Ядро ОС Linux разрабатывается под общим руководством Линуса Торвальдса и распространяется свободно (на основе лицензии GPL), как и огромное количество другого программного обеспечения, утилит и прикладных программ. Одним из следствий свободного распространения ПО для Linux явилось то, что большое число разных фирм и компаний, а также просто независимых групп разработчиков стали выпускать так называемые дистрибутивы Linux.

Дистрибутив — это набор программного обеспечения, включающий все 4 основные составные части ОС, т. е. ядро, файловую систему, оболочку и совокупность

утилит, а также некоторую совокупность прикладных программ. Обычно все программы, включаемые в дистрибутив Linux, распространяются на условиях GPL, так что может сложиться впечатление, что дистрибутив может выпустить кто угодно, точнее любой, кто не поленится собрать коллекцию свободного ПО. И какая-то степень правдоподобия в таком утверждении есть. Однако разработчик дистрибутива должен по крайней мере создать программу инсталляции, которая будет устанавливать ОС на компьютер, на котором никакой ОС еще нет. Кроме того, необходимо обеспечить разрешение взаимозависимостей и противоречий между разными пакетами (и версиями пакетов), что, как мы увидим позже, тоже является нетривиальной задачей.

Тем не менее, в мире существует уже более сотни различных дистрибутивов Linux, и все время появляются новые. Более-менее полный список их можно найти на сервере http://www.linuxhq.com, где даны краткие характеристики каждому дистрибутиву (упоминаются и некоторые локализованные версии). Кроме того, там же есть ссылки на другие списки дистрибутивов, так что при желании можно найти все, что вообще существует в мире.

Файловая система

Файловая система — это структура, с помощью которой ядро операционной системы предоставляет пользователям (и процессам) ресурсы долговременной памяти системы, т.е. памяти на различного вида долговременных носителях информации — жестких дисках, магнитных лентах. CD-ROM и т. п.

Информация в любой ОС хранится на носителях в виде файлов. Файлы группируются в каталоги, которые, в свою очередь, могут быть включены в другие каталоги. В результате получается иерархическая структура каталогов, начинающаяся с корневого каталога. Каждый (под)каталог может содержать как отдельные файлы, так и подкаталоги.

Иерархическую структуру каталогов обычно иллюстрируют рисунком "дерева каталогов", в котором каждый каталог изображается узлом "дерева", а файлы — "листьями". В MS Windows или DOS каталоговая структура строится отдельно для каждого физического носителя (т. е., имеем не отдельное "дерево", а целый "лес") и корневой каталог каждой каталоговой структуры обозначается какой-нибудь буквой

латинского алфавита (отсюда уже возникает некоторое ограничение). В Linux (и UNIX вообще) строится единая каталоговая структура для всех носителей, и единственный корневой каталог этой структуры обозначается символом "/". В эту единую каталоговую структуру можно подключить любое число каталогов, физически расположенных на разных носителях (как говорят, "смонтировать файловую систему" или "смонтировать носитель").

Имена каталогов строятся по тем же правилам, что и имена файлов. И, вообще, каталоги в принципе ничем, кроме своей внутренней структуры (до которой ОС уже есть дело) не отличаются от "обычных" файлов, например, текстовых.

Полным именем файла (или путем к файлу) называется список имен вложенных друг в друга подкаталогов, начинающийся с корневого каталога и оканчивающийся собственно именем файла. При этом имена подкаталогов в этом списке разделяются тем же символом "/", который служит для обозначения корневого каталога.

В каждый момент времени пользователь работает с одним экземпляром оболочки shell и эта оболочка хранит значение так называемого "текущего" каталога, т. е. того каталога, в котором пользователь сейчас работает. Имеется специальная команда, которая сообщает вам значение текущего каталога — pwd.

В Linux типовая структура каталогов выдерживается, пожалуй, даже более строго, чем в Windows. Более того, существует даже стандарт на структуру каталогов для UNIX-подобных ОС, так называемый Filesystem Hierarchy Standart (FHS).

Стандарт FHS предлагает создать в корневом каталоге следующие подкаталоги:

bin - Этот каталог содержит в основном готовые к исполнению программы, большинство из которых необходимы во время старта системы (или в однопользовательском системном режиме, используемом для отладки). Здесь хранится значительное количество общеупотребительных команд Linux .

boot - неизменяемые файлы, необходимые для загрузки системы;

dev - файлы устройств;

etc -этот каталог и его подкаталоги содержат большинство данных, необходимых для начальной загрузки системы и основные конфигурационные файлы. В /etc находятся, например, файл inittab, определяющий загружаемую конфигурацию, и файл паролей пользователей passwd. Часть конфигурационных файлов может

находится и в /usr/etc. Каталог /etc не должен содержать двоичных файлов (их следует перенести в /bin или /sbin)

home - домашние каталоги пользователей;

lib - основные разделяемые библиотеки и модули ядра; Этот каталог содержит разделяемые библиотеки функций, необходимых компилятору языка C и модули (драйверы устройств). Даже если в системе не установлен компилятор языка C, разделяемые библиотеки необходимы, поскольку они используются многими прикладными программами. Они загружаются в память по мере необходимости выполнения каких-то функций, что позволяет уменьшить объем кода программ — в противном случае один и тот же код многократно повторялся бы в различных программах

mnt - это точка монтирования для временно монтируемых файловых систем. Если на компьютере запускается поочередно Linux и MS DOS, то этот каталог обычно используется, чтобы монтировать файловую систему MS DOS. Если вы имеете привычку монтировать несколько дополнительных носителей, например, дискеты, CD-ROM, дополнительный жесткий диск и т. д., то можно создать в нем соответственно дополнительные подкаталоги для каждого носителя;

root - домашний каталог пользователя суперпользователя root

opt - дополнительные пакеты программного обеспечения;

sbin - основные системные исполняемые файлы;

tmp - временные файлы;

usr - Этот каталог огромен и его структура в основном повторяет структуру корневого каталога. В его подкаталогах находятся все основные приложения. В соответствии со стандартом FHS рекомендуется выделять для этого каталога отдельный раздел диска или вообще располагать его на сетевом диске, общем для всех компьютеров в сети. Такой раздел или диск монтируют только для чтения и располагают в нем общие конфигурационные и исполняемые файлы, документацию, системные утилиты и библиотеки, а также включаемые файлы (файлы типа include);

var - переменные данные.

В соответствии с требованиями стандарта приложения не должны создавать файлы и каталоги или требовать наличия каких-то специальных файлов и каталогов (помимо перечисленных) в корневом каталоге. Во-первых, размер корневой файловой

системы желательно сохранять по возможности малым, а во-вторых, стандарт FHS обеспечивает достаточную гибкость и удобство размещения файлов, не попавших в корневую систему, в других файловых системах и подкаталогах. Некоторые подкаталоги корневого каталога факультативны. Но уж если они существуют, то должны размещаться в корневом каталоге, но не обязательно в корневой файловой системе.

Права доступа к файлам и каталогам

Поскольку Linux — система многопользовательская, вопрос об организации разграничения доступа к файлам и каталогам является одним из существенных вопросов, которые должна решать операционная система. Механизмы разграничения доступа, разработанные для системы UNIX в 70-х годах (возможно, впрочем, они предлагались кем-то и раньше), очень просты, но они оказались настолько эффективными, что просуществовали уже более 30 лет и по сей день успешно выполняют стоящие перед ними задачи.

В основе механизмов разграничения доступа лежат имена пользователей и имена групп пользователей. В Linux каждый пользователь имеет уникальное имя, под которым он входит в систему (логируется). Кроме того, в системе создается некоторое число групп пользователей, причем каждый пользователь может быть включен в одну или несколько групп. Создает и удаляет группы суперпользователь, он же может изменять состав участников той или иной группы. Члены разных групп могут иметь разные права по доступу к файлам, например, группа администраторов может иметь больше прав, чем группа программистов.

В индексном дескрипторе каждого файла записаны имя так называемого владельца файла и группы, которая имеет права на этот файл. Первоначально, при создании файла его владельцем объявляется тот пользователь, который этот файл создал. Точнее — тот пользователь, от чьего имени запущен процесс, создающий файл. Группа тоже назначается при создании файла — по идентификатору группы процесса, создающего файл. Владельца и группу файла можно поменять в ходе дальнейшей работы с помощью команд chown и chgrp .

Вообще говоря, права доступа и информация о типе файла в UNIX-системах хранятся в индексных дескрипторах в отдельной структуре, состоящей из двух

байтов, т. е. из 16 бит (это естественно, ведь компьютер оперирует битами, а не символами r, w, x). Четыре бита из этих 16-ти отведены для кодированной записи о типе файла. Следующие три бита задают особые свойства исполняемых файлов, о которых мы скажем чуть позже. И, наконец, оставшиеся 9 бит определяют права доступа к файлу. Эти 9 бит разделяются на 3 группы по три бита. Первые три бита задают права пользователя, следующие три бита — права группы, последние 3 бита определяют права всех остальных пользователей (т. е. всех пользователей, за исключением владельца файла и группы файла).

При этом, если соответствующий бит имеет значение 1, то право предоставляется, а если он равен 0, то право не предоставляется. В символьной форме записи прав единица заменяется соответствующим символом (r, w или x), а 0 представляется прочерком.

Право на чтение (r) файла означает, что пользователь может просматривать содержимое файла с помощью различных команд просмотра, например, командой more или с помощью любого текстового редактора. Но, подредактировав содержимое файла в текстовом редакторе, вы не сможете сохранить изменения в файле на диске, если не имеете права на запись (w) в этот файл. Право на выполнение (x) означает, что вы можете загрузить файл в память и попытаться запустить его на выполнение как исполняемую программу. Конечно, если в действительности файл не является программой (или скриптом shell), то запустить этот файл на выполнение не удастся, но, с другой стороны, даже если файл действительно является программой, но право на выполнение для него не установлено, то он тоже не запустится.

Команды Linux

man (от англ. manual — руководство) — команда Unix, предназначенная для форматирования и вывода справочных страниц.

top — консольная команда UNIX-совместимых операционных систем, список работающих в данный момент процессов и информацию о них. Команда top показывает список работающих в данный момент процессов и информацию о них, включая использование ими памяти и процессора. Список интерактивно формируется в реальном времени. Чтобы выйти из программы top, нажмите клавишу [q].

free - Показывает количество свободной и используемой памяти в системе.

ps (от англ. process status) — консольная команда UNIX-совместимых операционных систем, выдающая отчёт о работающих процессах.

ls - выдача информации о файлах или каталогах

Синтаксис команды:

ls [флаги] [имя ...]

Команда ls для каждого имени каталога распечатывает список входящих в этот каталог файлов; для файлов - повторяется имя файла и выводится дополнительная информация в соответствии с указанными флагами. По умолчанию имена файлов выводятся в алфавитном порядке. Если имена не заданы, выдается содержимое текущего каталога. Если заданы несколько аргументов, то они сортируются по алфавиту, однако сначала всегда идут файлы, а потом каталоги с их содержимым.

cat <имя_файла> -вывод содержимого файла на стандартный вывод (по умолчанию - на экран).

Можно записать вводимый на экран текст с помощью следующей последовательности действий:

cat > <имя_файла>

more <имя_файла> -просмотр содержимого длинного текстового файла по страницам.

less <имя_файла> -просмотр содержимого текстового файла с возможностью вернуться к предыдущим страницам. Нажмите q, когда захотите выйти из программы. "less" - аналог команды DOS "more", хотя очень часто "less" бывает более удобной чем "more".

pico <имя_файла> -редактировать текстовый файл с помощью текстового редактора pico.

whoami - вывести имя под которым Вы зарегистрированы.

date - вывести дату и время.

time <имя программы> -выполнить программу и получить информацию о времени, нужном для ее выполнения. Не путайте эту команду с date. Например: Я могу определить выполнить команду ls и узнать, как много времени требуется для вывода списка файлов в каталоге, набрав последовательность: time ls

who - определить кто из пользователей работает на машине.

finger <имя_пользователя> - системная информация о зарегистрированном

пользователе.

uptime -количество времени, прошедшего с последней перезагрузки операционной системы.

uname -a - вывести информацию о версии операционной системы.

free - вывести информацию по использованию памяти.

df - вывести информацию о свободном и используемом месте на дисках.

login запрос от пользователя имени и пароля (запрос от системы к пользователю) для входа в систему(по умолчанию, при наборе пароля, он не отображается).

logout выход из текущего сеанса оболочки.

startx команда для запуска графического интерфейса X Window.

shutdown останавливает систему и предотвращает повреждение файловой системы при этом, но,используется только при работе в консольном режиме.При работе в режиме X Window, не используйте.

halt быстрое и корректное выключение системы.

poweroff корректное выключение системы.

reboot корректное выключение с последущей загрузкой.Перезагрузка.

vmstat выдаёт сведения о процессах, памяти и загруженности центрального процессора.

cal форматированный календарь на текущий месяц (добавить у и будет календарь на весь текущий год).

oclock простые часы, которые висят на рабочем столе (много дополнительных параметров).

hostname команда отображает идентификатор данного узла сети (его имя). root может изменить имя узла на новое.

hwclock встроенные часы Вашего компьютера.Для изменения даты и времени и синхронизации с системными часами, необходимы привелегии root.

users отображает краткий список пользователей работающих в системе в данный момент.

w подробная информация о всех пользователях, работающих в данный момент и также простой, вход в систему и др.Если нужен один пользователь, то указать имя в параметре.

whatis поиск по базе данных страниц руководства и отображение краткого описания.

which показывает полный путь к исполняемому файлу команды.

write отправляет сообщение другому пользователю, находящемуся в системе, путём копирования строк с терминала отправителя на терминал получателя.

wall отправляет сообщение на терминал каждого пользователя находящегося в системе в данный момент.

history показывает пронумерованный список команд, которые Вы выполняли в этом и предыдущем сеансе. Если в списке историии их довольно много, то увидите последние.

jobs выводит список всех выполняемых и приостановленных задач.

kill завершить процесс (необходимо указать какой).

killall позволят управлять процессами используя их имена или имена файлов, а не идентификаторы как в kill. Завершаются все указанные процессы.

nice позволяет отобразить или настроить приоритет задачи.

pstree показывает иерархию процессов системы, что хорошо показывает их взаимозависимость.

renice задаёт приоритет для указанной задачи.

script позволяет записывать весь вывод с терминала в файл.Что бы остановить запись нажмите Ctrl+d.Если имя файла не указано то записывается в typescript.

times показывает полное время выполнения процессов для всей системы и данного пользователя.

file показывает тип содержимого указанного файла(текст, выполняемый, данные).

last показывает список пользователей, которые заходили в систему с момента создания файла /var/log/wtmp.

lastlog проверяет историю входа в систему зарегистрированных пользователей.Форматирует и выводит на печать файл /var/log/lastlog.

logger посылает запрос демону syslogd с просьбой поместить сообщение в системный журнал.

lpr отправляет документ на печать демону печати.

chfn изменяет сведения о пользователе в файле /etc/passwd из которого берёт

информацию команда finger .

chgrp команда для администратора,для изменения группы владельцев файла.

clear очищает экран терминала (если это возможно).

crontab обеспечивает возможность выполнения определённых задач по расписанию. Чаще используется администратором, хотя свои задачи могут быть и у пользователей.

csplit разбивает файл на несколько частей.Надо задать метод разбивки (строки и т.д.).

dd копирование файла с одновременным выполнением различных, дополнительных преобразований.

dc калькулятор.

debugfs применяется для восстановления файловой системы (ext2,ext3) если недостаточно команды fsck.

du показывает количество блоков диска, занятых каждым из файлов каталога.

mc запускает программу Midnight Commander диспетчер файлов в текстовой консоли. Напоминает MSDOS менеджеры и довольно проста и удобна в использовании.Очень много нужных и удобных функций.

mcat копирует необработанные данные на дискету.

mcopy использует отформатированную дискету MSDOS для копирования файлов в Линукс и из Линукс без предварительного подключения дискеты к файловой системе.

mdel удаляет файл на отформатированной дискете MSDOS.

mdir отображает содержимое каталога на дискете MSDOS.

mdu показывает дисковое пространство занятое каталогом MSDOS.

mesg контролирует доступ к Вашему терминалу, что бы коллеги не могли засыпать Вас сообщениями с помощью команды write

mformat создаёт на дискете файловую систему MSDOS.

mkbootdisk применяется в некоторых дистрибутивах, для создания загрузочной дискеты, содержащей всё необходимое для аварийной загрузки.

mktemp создаёт уникальное имя файла для временной работы.

mlabel создаёт метку тома на MSDOS на отформатированной дискете.

mmd создаёт подкаталог MSDOS на отформатированной дискете.

mmount подключает к файловой системе отформатированное устройство MSDOS.

mmove перемещает или переименовывает файл на дискете MSDOS.

stat отображение всей доступной информации об указанном файле.

touch изменяет время последнего доступа или изменения файла на текущее время.

wc показывает число строк, слов и символов в файле.

bunzip2 распаковывает указанный файл на 30%быстрей чем gzip.

bzip2 сжимает указанный файл по ускоренному алгоритму.

bzip2recover делает попытку восстановить данные из повреждённого файла сжатого bzip2.

compress сжимает указанный файл по другому алгоритму.

uncompress распаковывает файл сжатый предыдущей командой.

gzip сжимает указанный файл.

gunzip распаковывает указанный файл (расширения .Z,.gz,.tgz,.zip).

gzexe позволяет сжать исполняемый файл с указанным именем так, что бы он автоматически распаковывался и выполнялся, когда пользователь даёт команду на выполнение сжатого файла.

gpasswd устанавливает пароль группы.

mcrypt Шифрует указанный файл.Создаётся новый файл в рабочем каталоге с расширением .enc. Вам будет предложено ввести пароль.Не забудьте его.

mdecrypt расшифровывает это же файл.Если этих утилит нет, скачайте http://mcrypt/hellug.grl

tar помещает два и более файлов в новый или существующий архив или извлекает их из архива.При задании каталога, заархивирует все файлы в каталоге и подкаталоге.

echo выводит строку текста на стандартное устройство вывода.

fdformat форматирование гибкого диска.Дополнительно вводится имя устройства и необходимый вид форматирования.

fg переводит процесс выполняемый в фоновом режиме в приоритетный режим.

fgconsole показывает количество активных виртуальных консолей.

fsck проверяет и восстанавливает файловую систему.

mount монтирование файловой системы.

umount отмонтирование файловой системы (в обоих командах необходимо указать, что именно).

rdev при вызове без параметров выводит информацию о текущей файловой системе.

rcp применяется для копирования файлов с одного компьютера на другой.

rdate получает значение даты и времени от другого узла сети.Используется для синхронизации системного времени узлов.

rename переименовывает файлы.Очень удобно, когда много файлов.

sleep приостанавливает начало выполнения процесса на заданное количество секунд.

usleep приостанавливает на микросекунды.

sync очищает буферы файловой системы.

cmp производит быстрое сравнение двух указанных файлов.Если они идентичны, то никакие сообщения не выводятся.

column форматирует входной текст из указанного файла в список из пяти колонок.

diff сравнивает два указанных текстовых файла.Каждое отличие выводится в контексте. Позволяет сравнивать каталоги.

id отображает действующие значения идентификаторов пользователя и группы для текущего пользователя.

ifconfig отображает состояние текущей конфигурации сети или настраивает сетевой интерфейс.

nl команда нумерует строки в указанном файле.

sort команда позволяет отсортировать строки файла в алфавитном порядке.

groupadd создание группы пользователей с указанным именем.

groupdel удаляет группу с указанным именем.

groupmod изменяет параметры группы с указанным именем.

mkpasswd создаёт высококачественный пароль, состоящий по умолчанию из девяти символов и содержащий по крайней мере буквы в разном регистре и цифры.

useradd создание нового пользователя с указанным именем.

userdel удаляет пользователя с указанным именем.

usermod изменяет параметры пользователя с указанным именем.

netstat вывод информации о сетевой подсистеме.Очень много настроек и параметров.

ping отправка на указанный адрес пакетов для проверки возможности соединения с этим узлом.

telnet открывает окно терминала на удалённом узле и запускает интерактивный сеанс.

passwd - смена входного пароля

Синтаксис команды:

passwd [входное_имя]

Команда passwd меняет (или устанавливает) пароль, связанный с входным_именем пользователя.

Обычный пользователь может менять только пароль, связанный с его собственным входным_именем.

Команда запрашивает у обычных пользователей старый пароль (если он был), а затем дважды запрашивает новый. После первого запроса проверяется, достаточен ли "возраст" старого пароля. Возраст - это промежуток времени (обычно несколько дней), который должен пройти между сменами пароля. Если возраст недостаточен, новый пароль отвергается и passwd завершается.

Если возраст достаточен, делается проверка на соответствие нового пароля техническим требованиям. Когда новый пароль вводится во второй раз, две копии нового пароля сравниваются. Если они не совпали, цикл запроса нового пароля повторяется, но не более двух раз.

Технические требования к паролям:

1. Каждый пароль должен содержать не менее 6 символов. Значащими являются только первые 8.

2. Каждый пароль должен содержать как минимум две буквы (большие или малые) и хотя бы одну цифру или знак.

3. Каждый пароль должен отличаться от входного_имени, прочитанного слева направо или задом наперед, и от его циклических сдвигов. При сравнении не делается различий между большими и малыми буквами.

4. Новый пароль должен отличаться от старого хотя бы тремя символами. При

сравнении не делается различий между большими и малыми буквами.

Суперпользователь (root) имеет право изменять любые пароли, поэтому у него старый пароль не запрашивается. Суперпользователь не связан ограничениями на возраст пароля и соответствие техническим требованиям. Суперпользователь может создать пустой пароль, нажимая возврат каретки в ответ на запрос нового пароля.

su (сокр. от англ. Substitute User) — команда Unix-подобных операционных систем, позволяющая пользователю войти в систему под другим именем, не завершая текущий сеанс. Обычно используется для временного входа суперпользователем для выполнения административных работ.

Синтаксис команды:

su [-] [имя_пользователя [аргумент ...]]

Команда su позволяет пользователю выполнять команды от имени другого пользователя, не завершая текущий сеанс, или получить роль. По умолчанию предполагается работа от имени пользователя root (суперпользователя).

Для использования su необходимо ввести соответствующий пароль (если только команду не вызывает пользователь root). Если введен правильный пароль, su создает новый процесс командного интерпретатора, с такими же реальными и эффективными идентификаторами пользователя и группы, а также списком дополнительных групп, что и у указанного пользователя.

sudo - выполнить команду от имени другого пользователя

Синтаксис команды:

sudo -V | -h | -l | -L | -v | -k | -K | -s | [-H] [-P] [-S] [-b] | [-p запрос] [-c класс|-] [-a тип_аутентификации] [-u имя_пользователя/#uid] команда

sudo позволяет разрешенному пользователю выполнять команду как суперпользователь или другой пользователь, как определено в файле sudoers. Реальный и эффективный uid и gid при этом устанавливаются так, чтобы соответствовать таковым целевого пользователя, как определено в файле passwd (также инициализируется вектор группы, если целевой пользователь - не root). По умолчанию sudo требует, что бы пользователи аутентифицировали себя при помощи пароля (ЗАПОМНИТЕ: это пароль пользователя, не пароль root). Как только пользователь аутентифицировал себя происходит обновление временной метки и пользователь может использовать sudo некоторый период времени без пароля (по

умолчанию пять минут, если в sudoers не указано другое).

Работа с файлами и каталогами

pwd - выдача имени текущего каталога. Бывает, что при ее изучении, вы попадаете в какой-то каталог, про который уже не помните, как он называется и как вы в него попали. Узнать его полное имя позволяет команда pwd.

cd - смена текущего каталога.

Синтаксис команды:

cd [каталог]

Команда cd применяется для того, чтобы сделать заданный каталог текущим. Если каталог не указан, используется значение переменной окружения $HOME (обычно это каталог, в который Вы попадаете сразу после входа в систему). Если каталог задан полным маршрутным именем, он становится текущим. По отношению к новому каталогу нужно иметь право на выполнение, которое в данном случае трактуется как разрешение на поиск.

chmod - изменение режима доступа к файлам

Синтаксис команды:

chmod режим файл

Права доступа к указанным файлам (среди которых могут быть каталоги) изменяются в соответствии с указанным режимом. Режим может быть задан в абсолютном или символьном виде.

Использование символьного вида основано на однобуквенных обозначениях, которые определяют класс доступа и права доступа для членов данного класса. Права доступа к файлу зависят от идентификатора пользователя и идентификатора группы, в которую он входит. Режим в целом описывается в терминах трех последовательностей, по три буквы в каждой:

Владелец	Группа	Прочие
(u)	(g)	(o)
rwx	rwx	rwx

Здесь владелец, члены группы и все прочие пользователи обладают правами чтения файла, записи в него и его выполнения. В примере показаны обозначения как для класса доступа, так и для прав доступа внутри класса.

Для задания режима доступа в символьном виде используется следующий

синтаксис:

[кому] операция права

Часть [кому] есть комбинация букв u, g и o (владелец, члены группы и прочие пользователи соответственно). Если часть кому опущена или указано a, то это эквивалентно ugo.

Операция может быть: + (добавить право), - (лишить права), = (в пределах данного класса присвоить права абсолютно, то есть добавить указанные права и отнять неуказанные).

Права - любая осмысленная комбинация следующих букв:

r Право на чтение.

w Право на запись.

x Право на выполнение (поиск в каталоге).

s При выполнении переустанавливать действующий идентификатор пользователя или группы.

t После выполнения программы сохранять сегмент команд (бит навязчивости).

l Учет блокировки доступа.

Опустить часть права можно только если операция есть = (для лишения всех прав).

Если надо сделать более одного указания об изменении прав, то при использовании символьного вида в правах не должно быть пробелов, а указания должны разделяться запятыми. Например, команда chmod u+w,go+x f1 добавит для владельца право писать в файл f1, а для членов группы и прочих пользователей - право выполнять файл. Права устанавливаются в указанном порядке. Право s можно добавлять только для пользователя и группы, право t - только для пользователя.

Чтобы установить права, позволяющие владельцу читать и писать в файл, а членам группы и прочим пользователям только читать, надо использовать следующую запись:

chmod u=rw,go=r f1

Позволить всем выполнять файл f2

chmod +x f2

chown —изменить владельца файла

Только суперпользователь может изменять владельца файла. Владелец файла

может изменять группу файла на любую группу, к которой он принадлежит. Суперпользователь может произвольно изменять группу.

cp - копирование файлов

cp файл1 [файл2 ...] целевой_файл

Команда cp копирует файл1 в целевой_файл. Файл1 не должен совпадать с целевым_файлом (будьте внимательны при использовании метасимволов shell'a). Если целевой_файл является каталогом, то файл1, файл2, ..., копируются в него под своими именами. Только в этом случае можно указывать несколько исходных файлов.

Если целевой_файл существует и не является каталогом, его старое содержимое теряется.

Режим, владелец и группа целевого_файла при этом не меняются.

Если целевой_файл не существует или является каталогом, новые файлы создаются с

теми же режимами, что и исходные (кроме бита навязчивости, если Вы не суперпользователь). Время последней модификации целевого_файла (и последнего доступа,

если он не существовал), а также время последнего доступа к исходным файлам устанавливается равным времени, когда выполняется копирование. Если целевой_файл был ссылкой на другой файл, все ссылки сохраняются, а содержимое файла изменяется.

mv - перемещение (переименование) файлов

Синтаксис команды:

mv [-f] файл1 [файл2 ...] целевой_файл

Команда mv перемещает (переименовывает) файл1 в целевой_файл. Файл1 не должен совпадать с целевым_файлом (будьте внимательны при использовании метасимволов shell'a). Если целевой_файл является каталогом, то файл1, файл2, ..., перемещаются в него под своими именами. Только в этом случае можно указывать несколько исходных файлов.

Если целевой_файл существует и не является каталогом, его старое содержимое теряется. Если при этом обнаруживается, что в целевой_файл не разрешена запись, то выводится режим этого файла [см. chmod] и запрашивается строка со стандартного ввода. Если эта строка начинается с символа y, то требуемые действия все же

выполняются, при условии, что у пользователя достаточно прав для удаления целевого_файла. Если была указана опция -f или стандартный ввод назначен не на терминал, то требуемые действия выполняются без всяких запросов. Вместе с содержимым целевой_файл наследует режим файла1.

Если файл1 является каталогом, то он переименовывается в целевой_файл, только если у этих двух каталогов общий надкаталог; при этом все файлы, находившиеся в файле1, перемещаются под своими именами в целевой_файл. Если файл1 является файлом, а целевой_файл - ссылкой, причем не единственной, на другой файл, то все остальные ссылки сохраняются, а целевой_файл становится новым независимым файлом.

rm - удаление файлов

Синтаксис команды:

rm [-f] [-i] файл ...

rm -r [-f] [-i] каталог ... [файл ...]

Команда rm служит для удаления указанных имен файлов из каталога. Если заданное имя было последней ссылкой на файл, то файл уничтожается. Для удаления пользователь должен обладать правом записи в каталог; иметь право на чтение или запись файла не обязательно. Следует заметить, что при удалении файла в Linux, он удаляется навсегда. Здесь нет возможностей вроде "мусорной корзины" в windows 95/98/NT или команды undelete в DOS. Так что, если файл удален, то он удален!

Если нет права на запись в файл и стандартный ввод назначен на терминал, то выдается (в восьмеричном виде) режим доступа к файлу и запрашивается подтверждение; если оно начинается с буквы у, то файл удаляется, иначе - нет. Если стандартный ввод назначен не на терминал, команда rm ведет себя так же, как при наличии опции -f.

Допускаются следующие три опции:

-f Команда не выдает сообщений, когда удаляемый файл не существует, не запрашивает подтверждения при удалении файлов, на запись в которые нет прав. Если нет права и на запись в каталог, файлы не удаляются. Сообщение об ошибке выдается лишь при попытке удалить каталог, на запись в который нет прав (см. опцию -r).

-r Происходит рекурсивное удаление всех каталогов и подкаталогов, перечисленных в списке аргументов. Сначала каталоги опустошаются, затем

удаляются. Подтверждение при удалении файлов, на запись в которые нет прав, не запрашивается, если задана опция -f или стандартный ввод не назначен на терминал и не задана опция -i. При удалении непустых каталогов команда rm -r предпочтительнее команды rmdir, так как последняя способна удалить только пустой каталог. Но команда rm -r может доставить немало острых впечатлений при ошибочном указании каталога!

-i Перед удалением каждого файла запрашивается подтверждение. Опция -i устраняет действие опции -f; она действует даже тогда, когда стандартный ввод не назначен на терминал.

ПРИМЕРЫ Опция -i часто используется совместно с -r. По команде:

rm -ir dirname

запрашивается подтверждение:

directory dirname: ?

При положительном ответе запрашиваются подтверждения на удаление всех содержащихся в каталоге файлов (для подкаталогов выполняются те же действия), а затем подтверждение на удаление самого каталога.

rmdir - удаление каталогов

Синтаксис команды:

rmdir [-p] [-s] каталог ...

Команда rmdir удаляет указанные каталоги, которые должны быть пустыми. Для удаления каталога вместе с содержимым следует воспользоваться командой rm с опцией -r. Текущий каталог [см. pwd] не должен принадлежать поддереву иерархии файлов с корнем - удаляемым каталогом.

Для удаления каталогов нужно иметь те же права доступа, что и в случае удаления обычных файлов [см. rm].

Командой rmdir обрабатываются следующие опции:

-p Позволяет удалить каталог и вышележащие каталоги, оказавшиеся пустыми. На стандартный вывод выдается сообщение об удалении всех указанных в маршруте каталогов или о сохранении части из них по каким-либо причинам.

-s Подавление сообщения, выдаваемого при действии опции -p.

ln - создание ссылки на файл

Синтаксис команды:

ln [-f] файл1 [файл2 ...] целевой_файл

Команда ln делает целевой_файл ссылкой на файл1. Файл1 не должен совпадать с целевым_файлом (будьте внимательны при использовании метасимволов shell'a). Если целевой_файл является каталогом, то в нем создаются ссылки на файл1, файл2, ... с теми же именами. Только в этом случае можно указывать несколько исходных файлов.

Если целевой_файл существует и не является каталогом, его старое содержимое теряется. Если при этом обнаруживается, что в целевой_файл не разрешена запись, то выводится режим доступа к этому файлу [см. chmod] и запрашивается строка со стандартного ввода. Если эта строка начинается с символа y, то требуемые действия все же выполняются, при условии что у пользователя достаточно прав для удаления целевого_файла. Если была указана опция -f или стандартный ввод назначен не на терминал, то требуемые действия выполняются без всяких запросов. Целевой_файл наследует режим доступа к файлу1.

ОГРАНИЧЕНИЯ

Команда ln не создает ссылок между разными файловыми системами, поскольку они (файловые системы) могут добавляться и удаляться.

mkdir – создание каталога

mkdir [-m режим_доступа] [-p] каталог ...

По команде mkdir создается один или несколько каталогов с режимом доступа 0777 [возможно измененном с учетом umask и опции -m]. Стандартные файлы (. - для самого каталога и .. - для вышележащего) создаются автоматически; их нельзя создать по имени. Для создания каталога необходимо располагать правом записи в вышележащий каталог.

Идентификаторы владельца и группы новых каталогов устанавливаются соответственно равными реальным идентификаторам владельца и группы процесса.

Командой mkdir обрабатываются две опции:

-m режим_доступа - (явное задание режима_доступа для создаваемых каталогов [см. chmod]).

-p (при указании этой опции перед созданием нового каталога предварительно создаются все несуществующие вышележащие каталоги).

ПРИМЕРЫ Чтобы создать поддерево каталогов tmpdir/temp/dir, надо выполнить

команду

mkdir -p tmpdir/temp/dir

grep - поиск образца в файле

Выполняет поиск образца в текстовых файлах и выдает все строки, содержащие этот образец. Она использует компактный недетерминированный алгоритм сопоставления.

find - поиск файлов

Синтаксис команды:

find список_поиска выражение

Рекурсивно просматривает каждый из каталогов, перечисленных в списке_поиска, отыскивая файлы, удовлетворяющие логическому выражению.

lspci – отображает список PCI устройств компьютера.

lsusb – отображает список USB-устройств компьютера

Практическая часть

В рамках выполнения данной лабораторной работы студентам необходимо выяснить системные характеристики компьютера. Для этого необходимо воспользоваться консольными командами Linux. Необходимо выяснить следующие характеристики:

- тип процессора, его тактовую частоту, архитектуру
- размер оперативной памяти компьютера
- размер swap-области (области подкачки)
- размер свободной и занятой оперативной памяти и области подкачки
- размер дисковой подсистемы компьютера
- размер и занятость разделов файловой системы
- список всех выполняемых процессов на компьютере
- список выполняемых процессов упорядоченный по использованию процессорного времени (в обратном порядке)
- список выполняемых процессов упорядоченный по использованию памяти
- список PCI устройств, подключенных к компьютеру
- список USB-устройств, подключенных к компьютеру.

Результаты выполнения работы необходимо представить в отчете. Результаты

должны быть оформлены с указанием выполняемых команд и их параметров. Возможно в качестве результатов приводить как листинги (списки) так и скриншоты (снимки экрана).

Лабораторная работа №2. Обработка событий клавиатуры

Цель работы

Данная лабораторная работа посвящена обработке данных поступающих с клавиатуры. Работа выполняется с использованием системы программирования Free Pascal.

Теоретическая часть

Клавиатура

Клавиатура - это одно из основных устройств ввода информации в ЭВМ, позволяющее вводить различные виды информации. Вид вводимой информации определяется программой, интерпретирующей нажатые или отпущенные клавиши. С помощью клавиатуры можно вводить любые символы - от букв и цифр до иероглифов и знаков музыкальной нотации. Клавиатура позволяет управлять курсором на экране дисплея -устанавливать его в нужную точку экрана, перемещать по экрану “прокручивать” экран в режиме скроллинга, отправлять содержимое экрана на принтер, производить выбор при наличии альтернативных вариантов и т.д.

В последнее время наблюдаются тенденции отказа от клавиатуры в пользу альтернативных устройств: мыши, речевого ввода, сканеров. Но полностью эти устройства клавиатуру не заменяют.

Стандартная клавиатура IBM PC имеет несколько групп клавиш:

Алфавитно-цифровые и знаковые клавиши (с латинскими и русскими буквами, цифрами, знаками пунктуации, математическими знаками).

Специальные клавиши: <Esc>, <Tab>, <Enter>, <BackSpace>.

Функциональные клавиши: <F1>...<F10...>.

Служебные клавиши для управления перемещением курсора (стрелки: <Up>, <Down>, <Left>, <Right>, клавиши <Home>, <End>, <PgUp>, <PgDn> и клавиша, обозначенная значком “[ ]” - в центре дополнительной цифровой клавиатуры).

Служебные клавиши для управления редактированием <Ins> <Del>.

Служебные клавиши для смены регистров и модификации кодов других клавиш <Alt>, <Ctrl>, <Shift>.

Служебные клавиши для фиксации регистров <CapsLock>, <Scroll-Lock>, <NumLock>.

Разные вспомогательные клавиши <PrtSc>, <Break>, <Grey +>, <Grey ->.

Если клавиша первой, четвертой, а иногда и пятой группы оказывается нажатой дольше, чем 0,5 с, начинает генерироваться последовательность ее основных кодов с частотой 10 раз в секунду (в IBM PC XT), что имитирует серию очень быстрых нажатий этой клавиши. Общее число клавиш в основной модификации клавиатуры - 83, в расширенной клавиатуре – до 101. Количество различных сигналов от клавиатуры значительно превышает это число, так как:

1) при нажатии и освобождении клавиши в ЭВМ передаются разные кодовые комбинации: при нажатии - порядковый номер нажатой клавиши на клавиатуре (ее скан-код), а при освобождении - скан-код, увеличенный на 80h;

2) заглавные и строчные буквы первой группы клавиш (алфавитно-цифровых и знаковых) набираются на разных регистрах. Оперативное переключение регистров производится клавишей <Shift>. Если при нажатой (и удерживаемой в нажатом состоянии) клавише <Shift> “клюнуть” (от английского слова “dick”) любую алфавитную клавишу, то в ЭВМ будет отправлен код заглавной буквы, соответствующий нажатой клавише;

3) после однократного нажатия клавиши <CapsLock> (зажигается лампочка на клавиатуре рядом с клавишей) изменяется порядок работы клавиши <Shift>: без нажатия на нее будут набираться заглавные буквы, а при нажатии (совместном) - строчные. После повторного нажатия на <CapsLock> порядок работы клавиши <Shift> восстанавливается, а лампочка гаснет. Такой режим (переключательный) работы клавиши называется триггерным режимом, или flip-flop;

4) аналогично клавише <Shift> действуют <Alt> и <Ctrl> - при одновременном нажатии с ними любой другой клавиши, в ЭВМ передается не scancode, а расширенный код (2 байта). Иногда таким же образом используется клавиша <Esc>;

5) клавиша <NumLock> является триггерным переключателем дополнительной цифровой клавиатуры: при негорящей лампочке она работает как клавиатура для управления курсором; при зажженной - как цифровая;

6) для переключения регистров (или даже групп регистров) иногда используются другие комбинации клавиш: например, программы - русификаторы

клавиатуры переключают РУС-ЛАТ с помощью правой клавиши <Shift> или при одновременном нажатии двух клавиш <Shift> (правой и левой) и т.д. Эти комбинации клавиш обладают триггерным эффектом.

Сигналы, поступающие от клавиатуры, проходят трехуровневую обработку: на физическом, на логическом и на функциональном уровнях.

Физический уровень имеет дело с сигналами, поступающими в вычислительн машину при нажатии и отпускании клавиш.

На логическом уровне, реализуемом BIOS через прерывание 9, скан-код транслируется в специальный 2-байтовый код. Младший байт для клавиш группы 1 содержит ASCII-код, соответствующий изображенному на клавише знаку. Этот байт называют главным. Старший байт (вспомогательный) содержит исходный скан-код нажатой клавиши.

На функциональном уровне отдельным клавишам программным путем приписываются определенные функции. Такое “программирование” клавиш осуществляется с помощью драйвера-программы, обслуживающей клавиатуру в операционной системе.

На IBM PC AT используется клавиатура с большим количеством клавиш. На этих машинах есть возможность управлять некоторыми функциями клавиатуры, например, изменять время ожидания автоповтора, частоту автоповтора, зажигать и гасить светодиоды на панели управления клавиатурой.

Устройство клавиатуры не является простым: в клавиатуре используется свой микропроцессор, работающий по прошитой в ПЗУ программе. Контроллер клавиатуры постоянно опрашивает клавиши, определяет, какие из них нажаты, проводит контроль на “дребезг” и выдает код нажатой или отпущен-_ ной клавиши в системный блок ЭВМ.

Выпускаемые разными производителями клавиатуры различаются также по расстоянию между клавишами, числу специальных клавиш, способу переключения на цифровой регистр для быстрого ввода числовых данных, углу наклона, форме и текстуре поверхности клавиш, усилию нажима и величине хода клавиш, расположению часто используемых клавиш и др.

Система программирования Free Pascal

Free Pascal Compiler (FPC) - это свободно распространяемый компилятор языка Паскаль с открытыми исходными кодами. Он совместим с Borland Pascal 7 и Object Pascal – Delphi, но при этом обладает рядом дополнительных возможностей, например, поддерживает перегрузку операторов. FPC — кроссплатформенный инструмент, поддерживающий огромное количество платформ. Среди них — AmigaOS, DOS, Linux, *BSD, OS/2, MacOS(X) и Win32.

В состав Free Pascal входит большое количество различных библиотек, реализующих в том числе функции работы с периферийными устройствами. Так, в частности, библиотека Crt реализует функции работы с клавиатурой, консолью.

Модуль Crt реализует ряд мощных программ, предоставляющих вам полную возможность управления средствами компьютера PC, такими, как управление режимом экрана, расширенные коды клавиатуры, цвета, окна, и звуковые сигналы.

Одним из основных преимуществ использования модуля Crt является большая скорость и гибкость при выполнении операций работы с экраном. Программы, не работающие с модулем Crt, выводят на экран информацию с помощью средств операционной системы, что связано с дополнительными непроизводительными затратами. При использовании модуля Crt выводимая информация посылается непосредственно в базовую систему ввода-вывода (BIOS), или, для еще более быстрых операций, непосредственно в видеопамять.

Использование модуля CRT

Чтобы использовать модуль Crt, его нужно указать в операторе uses вашей программы:

```
uses Crt;
```

При инициализации модуля Crt для того, чтобы можно было обращаться к CRT, вместо стандартных файлов ввода и вывода DOS назначаются стандартные входные и выходные текстовые файлы. Это соответствует выполнению в начале программы следующих операторов:

```
AssignCrt(Input); Reset(Input);
AssignCrt(Output); Rewrite(Output);
```

Это означает, что переопределение входных и выходных файлов далее не

допускается до тех пор, пока для данных файлов не будет выполнено обратного переназначения и не произойдет переход к стандартному вводу и выводу с помощью выполнения операторов:

Assing(Input,''); Reset(Input);

Assing(Output,''); RewriteOutput);

Окна CRT

Модуль Crt поддерживает простую, но, тем не менее, мощную форму использования окон. Процедура Window позволяет вам определить в каком-либо месте экрана окно. При записи в это окно оно ведет себя точно также, как целый экран. При этом остальная часть экрана остается нетронутой. Другими словами, доступ к экрану вне окна отсутствует. Внутри окна можно добавлять и удалять строки, при этом курсор возвращается к правому краю и при достижении курсором нижней строки текст продвигается вверх.

Все координаты экрана, кроме тех, которые используются для определения окна, относятся к текущему окну. Координата экрана (1,1) соответствует левому верхнему углу экрана. По умолчанию окном считается весь экран.

Специальные символы

При записи в выходной файл или в файл, который назначен для модуля Crt, специальное значение имеют следующие управляющие символы:

#7 Звонок - Вызывает звуковой сигнал, издаваемый с помощью внутреннего динамика.

#8 Обратный пробел - Возврат на одну позицию. Вызывает перемещение курсора влево на одну позицию. Если курсор уже находится у левого края текущего окна, то никаких действий не производится.

#10 Перевод строки - Перемещает курсор на одну строку вниз. Если курсор уже находится на нижней строке окна, то окно пролистывается вверх на одну строку.

#13 Возврат каретки - Возвращает курсор с левому краю текущего окна.

Ввод строк

При чтении из входного файла (Input) или из текстового файла, который назначен для модуля Crt, текст вводится по одной строке. Строка запоминается во

внутреннем буфере текстового файла и когда переменные считываются, то в качестве источника используется этот буфер. Каждый раз когда буфер становится пустым, вводится новая строка. При вводе строк можно использовать следующие клавиши редактирования:

Backspace - Удаляет последний введенный символ.

Esc - Удаляет всю вводимую строку.

Enter - Прекращает ввод строки и записывает метку конца строки (возврат каретки/перевод строки) в буфере.

Ctrl+S - Действует также, как Backspace.

Ctrl+D - Извлекает один символ из последней вводимой строки и выводит его на экран.

Ctrl+F - Восстанавливает на экране последнюю вводимую строку.

Ctrl+Z - Завершает ввод строки и генерирует символ конца файла.

`Для проверки состояния клавиатуры и ввода отдельных символов под управлением программы используйте функции KeyPressed и ReadKey.

Процедуры и функции модуля Crt

AssignCrt - Назначает текстовый файл для устройства CRT.

ClrEol - Очищает все символы, начиная от позиции курсора до конца строки, без перемещения курсора.

ClrScr - Очищает экран и помещает курсор в верхнем левом углу.

Delay - Выполняет задержку на указанное число миллисекунд.

DelLine - Удаляет строку, на которой находится курсор и перемещает все следующие строки на одну строку вверх. Нижняя строка очищается.

GotoXY - Выполняет позиционирование курсора. X – это горизонтальная позиция, Y - вертикальная позиция.

InsLine - Вставляет пустую строку в месте расположения курсора.

KeyPressed - Возвращает значение True, если клавиша на клавиатуре нажата и False - в противном случае.

TextBackground - Выбирает фоновый цвет.

TextColor - Выбирает цвет самого символа.

TextMode - Выбирает конкретный текстовый режим.

Window - Определяет на экране текстовое окно.

Readkey - Считывает символ с клавиатуры.

WhereX - Возвращает координату X для текущей позиции курсора, относящуюся к текущему окну. X представляет собой горизонтальную позицию.

WhereY - Возвращает координату Y для текущей позиции курсора, относящуюся к текущему окну. Y представляет собой вертикальную позицию.

Подробную информацию о модуле Crt и его процедурах и функциях вы можете получить с помощью справочной системы программы Free Pascal.

Модуль keyboard

Весьма эффективным инструментом для низкоуровневой работы с клавиатурой обладает стандартный модуль Free Pascal keyboard. Приведем краткий перечень основных функций и процедур этого модуля:

DoneKeyboard – завершает работу с драйвером клавиатуры

FunctionKeyName – возвращает строку представляющую код функциональной клавиши

GetKeyEvent – возвращает следующее событие клавиатуры

GetKeyEventChar – возвращает символьную часть события клавиатуры

GetKeyEventCode – возвращает функциональную часть события

GetKeyEventFlags – возвращает установленные флаги клавиатурного события

GetKeyEventShiftState – возвращает текущее состояние клавиш Shift

GetKeyEventUniCode – возвращает клавиатурное событие в формате Unicode

InitKeyboard – инициализирует драйвер клавиатуры

IsFunctionKey – возвращает значение Истина если нажатая клавиша – функциональная

KeyEventToString – возвращает строковое описание клавиатурного события

KeyPressed – проверяет наличие клавиатурного события в очереди

Более полную информацию о функциях, процедурах, константах модуля Keyboard смотрите самостоятельно в документации к Free Pascal.

Практическая часть

В рамках выполнения данной лабораторной работы необходимо написать программу на языке Free Pascal, обеспечивающую обработку событий, поступающих с клавиатуры. Описание событий должно выводится в нижней части экрана (это

может быть последняя строка). По результатам работы нужно подготовить отчет с исходным текстом программы и таблицей устанавливающей соответствие нажатым клавишам кодов клавиатуры.

Лабораторная работа №3. Исследование различных систем счисления

Цель работы

Целью данной лабораторной работы является изучение различных систем счисления. Работа выполняется с использованием системы программирования Free Pascal.

Теоретическая часть

Система счисления — символический метод записи чисел, представление чисел с помощью письменных знаков. Для начала проведём границу между числом и цифрой:

- Число — это некоторая абстрактная сущность для описания количества.
- Цифры — это знаки, используемые для записи чисел.

Цифры бывают разные: самыми распространёнными являются арабские цифры, представляемые известными нам знаками от нуля (0) до девяти (9); менее распространены римские цифры, мы их можем иногда встретить на циферблате часов или в обозначении века (XIX век).

Итак:

- число — это абстрактная мера количества;
- цифра — это знак для записи числа.

Поскольку чисел гораздо больше чем цифр, то для записи числа обычно используется набор (комбинация) цифр. Только для небольшого количества чисел — для самых малых по величине — бывает достаточно одной цифры. Существует много способов записи чисел с помощью цифр. Каждый такой способ называется системой счисления. Величина числа может зависеть от порядка цифр в записи, а может и не зависеть. Это свойство определяется системой счисления и служит основанием для простейшей классификации таких систем.

Итак, указанное основание позволяет все системы счисления разделить на три класса (группы):

- позиционные;

- непозиционные;
- смешанные.

Позиционные системы счисления

Позиционные системы счисления — это системы счисления, в которых значение цифры напрямую зависит от её положения в числе. Например, число 01 обозначает единицу, 10 — десять. Позиционные системы счисления позволяют легко производить арифметические расчёты. Представление чисел с помощью арабских цифр — самая распространённая позиционная система счисления, она называется «десятичной системой счисления». Десятичной системой она называется потому, что использует десять цифр: 0, 1, 2, 3, 4, 5, 6, 7, 8 и 9. Заметьте: максимальная цифра (9) на единичку меньше количества цифр (10).

Для составления машинных кодов удобно использовать не десятичную, а двоичную систему счисления, содержащую только две цифры, 0 и 1. Таким образом, в двоичной системе максимальная цифра 1.

Программисты для вычислений также пользуются ещё восьмеричной и шестнадцатеричной системами счисления.

Количество цифр используемых в системе счисления называется её «основанием». В десятичной системе основание равно десяти, в двоичной системе — двум, а в восьмеричной и шестнадцатеричной — соответственно, восьми и шестнадцати. То есть в p-ичной системе счисления количество цифр равно p и используются цифры от 0 до $p-1$.

В общем случае в позиционной системе счисления числа представляются следующим образом: $(a_n a_{n-1} \ldots a_0)_f$, где a_0 , a_1 ,..., a_n — цифры, а f — основание системы счисления. Если используется десятичная система, то f — можно опустить.

Примеры чисел:

- 11001_2 — число в двоичной системе счисления, $a_0=1$, $a_1=0$, $a_2=0$, $a_3=1$, $a_4=1$;
- 31_8 — число в восьмеричной системе счисления, $a_0=1$, $a_1=3$;
- 25_{10} — число в десятичной системе счисления, $a_0=5$, $a_1=2$;

Практическая часть

На языке FreePascal необходимо написать программу, которая производит преобразование вводимых с клавиатуры чисел в следующие системы счисления:

- десятичную;
- двоичную;
- восьмеричную;
- шестнадцатиричную.

При этом необходимо с клавиатуры вводить число в разных системах счисления. Систему счисления вводимого числа можно задавать либо с помощью экранного меню, либо с помощью спецсимволов.

По результатам работы необходимо подготовить отчет с приведением текста программы и скриншотов, демонстрирующих ее выполнение.

Лабораторная работа №4. Создание программы-демона

Цель работы

Целью данной лабораторной работы является создание программы-демона, которая может выполнять поступающие к ней задания пользователя.

Теоретическая часть

Де́мон (англ. daemon) — в системах класса UNIX — служба, работающая в фоновом режиме без прямого общения с пользователем.

Демоны обычно запускаются во время загрузки системы. Типичные задачи демонов: серверы сетевых протоколов (HTTP, FTP, электронная почта и др.), управление оборудованием, поддержка очередей печати, управление выполнением заданий по расписанию и т. д. В техническом смысле демоном считается процесс, который не имеет управляющего терминала. Чаще всего (но не обязательно) предком демона является init — корневой процесс UNIX.

Практическая часть

На языке FreePascal или на языке скриптов Bash необходимо написать программу, которая периодически (раз в 10 секунд, например) просматривает определенную папку, куда помещаются файл (или файлы) с заданиями пользователя. Задания пользователя представляют собой командные строки которые просто необходимо выполнить. Демон должен выполнить задания пользователя и удалить файлы с завершенными заданиями.

По результатам работы необходимо подготовить отчет с приведением текста программы и скриншотов, демонстрирующих ее выполнение.

Лабораторная работа №5. Работа с регулярными выражениями

Цель работы

Целью данной лабораторной работы явялется изучение регулярных выражений и создание программы, определяющей правильность ввода форматированной информации.

Теоретическая часть

Регулярные выражения (англ. regular expressions — система обработки текста, основанная на специальной системе записи образцов для поиска. Образец (англ. pattern), задающий правило поиска, по-русски также иногда называют «шаблоном», «маской».

Сейчас регулярные выражения используются многими текстовыми редакторами и утилитами для поиска и изменения текста на основе выбранных правил. Многие языки программирования уже поддерживают регулярные выражения для работы со строками. Например, Perl и Tcl имеют встроенный в их синтаксис механизм обработки регулярных выражений. Набор утилит (включая редактор sed и фильтр grep), поставляемых в дистрибутивах Unix, одним из первых способствовал популяризации понятия регулярных выражений.

Практическая часть

Изучить синтаксис регулярных выражений на основе документации Linux. Возможно также использование источников сети Интернет.

С помощью языка PHP создать web-приложение со следующим функционалом:

- страница с html-формой со следующими полями:
  - Фамилия Имя;
  - адрес электронной почты;
  - адрес web-ресурса
- по нажатию на кнопку "Отправить" скриптом производится проверка на правильность заполнения полей формы и пользователь получает отчет

По результатам работы необходимо подготовить отчет с приведением текстов скриптов и скриншотов, демонстрирующих ее выполнение.

Лабораторная работа №6. Работа с архивами в Linux

Цель работы

Целью данной лабораторной работы явялется изучение особенностей упаковки и распаковки данных в операционной системе Linux.

Теоретическая часть

Утилита tar предназначена для создания архивов файлов и каталогов. С помощью этой программы можно архивировать файлы, обновлять их в архиве и вводить в этот архив новые файлы. Можно архивировать и целые каталоги со всеми их файлами и подкаталогами. При необходимости все эти файлы и подкаталоги можно восстановить из архива. Программа tar предназначалась для создания архивов на лентах, отсюда и название tar (tape archive, т.е. "архив на ленте"). Архив можно создавать на любом устройстве, например на дискете или в архивном файле на диске. Программа tar - идеальное средство для создания резервных копий файлов или объединения нескольких файлов в один с целью передачи его по сети.

В операционной системе Linux программу tar часто используют для создания архивов на устройствах и в файлах. Ей можно дать указание архивировать файлы на определенном устройстве или в определенном файле, для чего служит опция f с именем устройства или файла. Синтаксис команды tar с опцией f очевиден из нижеследующего примера. Имя устройства или файла часто называют именем архива. При создании файла для tar-архива к имени этого файла обычно добавляется расширение .tar Это условное обозначение; оно не обязательно. В команде можно указать сколько угодно имен файлов. Если указано имя каталога, то в архив включаются и все подкаталоги этого каталога.

```
$ tar опцииf имя_архива.tar имена_файлов_и_каталогов
```

Для создания архива служит опция c. В сочетании с опцией f опция c приводит к созданию архива в файле или на устройстве. Эта опция ставится непосредственно перед опцией f. Обратите внимание, что дефиса перед опцией нет. В следующем примере каталог mydir и все его подкаталоги сохраняются в файле myarch.tar.

```
$ tar cf myarch.tar mydir
```

Потом пользователь может извлекать каталоги из архива, применяя команду tar с опцией x. Опция xf позволяет извлекать файлы из архивного файла или устройства. При извлечении формируются и все подкаталоги. В следующем примере посредством опции xf команде tar дается указание извлечь все файлы и подкаталоги из файла myarch.tar.

```
$ tar xf myarch.tar
```

Для добавления файлов в существующий архив служит опция r. В приведенном ниже примере пользователь добавляет файлы из каталога letters в архив myarch.tar.

```
$ tar rf myarch.tar letters
```

Если нужно изменить какой-либо файл в архивированных ранее каталогах, можно с помощью опции u дать команде указание обновить архив, заменив некоторые файлы их новыми версиями. Программа tar сравнивает время последнего изменения каждого архивированного файла и соответствующего файла в каталоге и копирует в архив все файлы с более поздней датой модификации. В архив будут добавлены и все вновь созданные в этих каталогах файлы. В следующем примере пользователь обновляет файл myarch.tar, вводя в него все измененные и вновь созданные в каталоге mydir файлы.

```
$ tar uf myarch.tar mydir
```

Если вы хотите посмотреть, какие файлы хранятся в архиве, дайте команду tar с опцией t. В следующем примере показано, как с помощью этой команды можно вызвать список всех файлов, хранящихся в архиве myarch.tar.

```
$ tar tf myarch.tar
```

Для создания резервных копий файлов на определенном устройстве укажите имя этого устройства в качестве имени архива. В следующем примере пользователь создает архив на дискете в устройстве /dev/fd0 и копирует в него все файлы из каталога mydir.

```
$ tar cf /dev/tdO mydir
```

Для того чтобы извлечь архивированные таким образом файлы, используйте опцию xf.

```
$ tar xf /dtv/fd0
```

Если архивируемые файлы занимают больше места, чем имеется на носителе, например на дискете, создайте tar-архив, состоящий из нескольких томов (дискет или лент).

Посредством опции M команде tar дается указание выводить сообщение о том, что текущий носитель заполнен. При архивировании файлов на дискете с использованием опции M в случае заполнения дискеты программа tar предложит вам вставить новую дискету. Таким образом вы сможете записать свой архив на нескольких дискетах.

```
$ tar cMf /dev/fd0 mydir
```

Чтобы распаковать архив, записанный на нескольких дискетах, вставьте первую дискету в дисковод и введите команду tar с опциями x и M, как показано ниже. Программа подскажет вам, когда надо вставить следующую дискету.

```
$ tar xMf /dev/fd0
```

При использовании команды tar операция сжатия архивных файлов не выполняется. Если вы хотите сжать файлы, дайте tar указание вызвать утилиту gzip. Если команда tar применяется с опцией z, то сначала программа gzip выполняет сжатие, а затем tar архивирует файлы. Та же опция z обеспечит вызов gzip для распаковки файлов при извлечении их из архива.

```
$ tar czf myarch.tar mydir
```

Помните, что между сжатием отдельных файлов с последующим архивированием и сжатием всего архива есть разница. Во многих случаях архив создается, чтобы переслать по сети несколько файлов в виде одного tar-файла. Для сокращения времени передачи размер этого архива должен быть по возможности небольшим. Чтобы добиться этого, можно с помощью утилиты gzip сжать архивный tar-файл, уменьшив его размер, а затем переслать сжатую версию. Получатель распакует его и восстановит файл. В результате применения утилиты gzip к tar -файлам часто получаются файлы с расши-рением .tar.gz. Расширение .gz добавляется к сжатому gzip-файлу. В следующем примере создается сжатая версия файла myarch.tar под тем же именем, но с расширением .gz.

```
$ gzip myarch.tar
$ la
myarch.tar.gz
```

Если вы хотите создать архив на некотором устройстве, например на ленте или в файле, нужно дать команду tar с опцией f и именем устройства или файла. Такой вариант эффективен при создании резервных копий файлов. Имя устройства по умолчанию хранится в файле /etc/default/tar. Синтаксис команды tar, подразумевающей использование устройства, заданного по умрлчанию (накопителя на магнитной ленте), приведен в показанном ниже примере. Опция f и имя устройства не задаются. Если указано имя каталога, то в архив включаются все его подкаталоги.

```
$ tar опция имена_каталогов_и_файлов
```

В представленном ниже примере каталог mydir со всеми подкаталогами сохраняется на ленте как на носителе по умолчанию.

```
$ tar c mydir
```

А в этом примере каталог mydir со всеми файлами и подкаталогами извлекается из устройства, принятого по умолчанию, и помещается в рабочий каталог пользователя.

```
$ tar x mydir
```

Сжатие файлов: программа gzip

Уменьшать размер файла приходится по разным причинам. Чаще всего это делается для экономии места и, если вы пересылаете файл по сети, для экономии времени передачи. Сжатие и распаковка файлов осуществляются с помощью утилиты gzip. При сжатии в качестве аргумента вводится имя файла. Этот файл заменяется сжатой версией с расширением .gz.

```
$ gzip mydata
$ la
mydata.gz
```

Для распаковки grip-архива введите либо команду gzip с опцией -d, либо команду gunzip . Эти команды приводят к распаковке файла с расширением .gz и замене его распакованной версией с тем же именем, но без расширения .gz . При использовании команды gunzip не нужно даже вводить расширение .gz. Команды gunzip и gzip -d заведомо предполагают его наличие.

```
$ gunzip mydata.gz
$ ls
mydata
```

Пусть, например, вы хотите вывести на экран или напечатать содержимое сжатого файла, не распаковывая его. Команда zcat создает распакованную версию файла и посылает ее на стандартный вывод. Затем этот вывод можно переадресовать в утилиту печати или отображения, например в more. Оригинал файла остается записанным в сжатом виде.

```
$ zcat mydata.gz | more
```

Можно сжимать и архивированные файлы. Эта операция дает в результате файлы с расширением .tar.gz. Сжатые архивированные файлы часто используются для передачи очень больших файлов по сетям.

```
$ gzip myarch.tax
$ ls
myarch.tar.gz
```

Файлы, входящие в архив, можно сжимать и по отдельности, используя команду tar с опцией z, которая вызывает утилиту gzip. В этом случае файл сначала сжимается, а затем помещается в архив. Следует отметить, однако, что архивы с файлами, сжатыми с применением опции z, обновлению не подлежат, и добавлять в них файлы нельзя. Все файлы необходимо сжимать одновременно и добавлять тоже одновременно.

Для создания сжатых файлов можно также пользоваться командами compress и uncompress. В утилите compress используется другой формат сжатия. В результате ее использования образуются файлы с расширением .Z. Команды compress и uncompress применяются не очень широко, но файлы с расширением .Z иногда встречаются. Для распаковки файла с расширением .Z можно использовать не только команду uncompress, но и команду gunzip. Однако gzip является стандартной утилитой сжатия из набора программного обеспечения GNU, поэтому вместо команды compress по возможности следует использовать именно ее.

Практическая часть

Изучить синтаксис использвания утилит tar и gzip с помощью системы man.

Создать архив .gz в который поместить один текстовый файл

Добавить новый текстовый файл в созданный архив

Удалить файл из архива

Создать архив на основе папки с вложенными папками и файлами с помощью утилиты tar

Создать архив на основе папки с вложенными папками и файлами в формате .tgz

По результатам работы необходимо подготовить отчет с приведением команд и результатов работы.

Лабораторная работа №7. Работа с файлами в Linux

Цель работы

Целью данной лабораторной работы является изучение особенностей работы с файлами в операционной системе Linux.

Теоретическая часть

Файловая система — это структура, с помощью которой ядро операционной системы предоставляет пользователям (и процессам) ресурсы долговременной памяти системы, т. е. памяти на различного вида долговременных носителях информации — жестких дисках, магнитных лентах. CD-ROM и т. п.

С точки зрения ОС файл представляет собой непрерывный поток (или последовательность) байтов определенной длины. Внутренний формат файла операционную систему не интересует. Но ОС должна дать файлу какое-то имя, с помощью которого пользователь, а точнее, программы-приложения, будут обращаться к файлу. Как организовать это обращение — дело файловой системы, пользователя это чаще всего не интересует. Поэтому с точки зрения пользователя файловая система выглядит как логическая структура каталогов и файлов.

Имена файлов в Linux могут иметь длину до 255 символов и состоять из любых символов, кроме символа с кодом 0 и символа / (слэша). Однако имеется еще ряд символов, которые имеют в оболочке shell специальное значение и которые поэтому не рекомендуется включать в имена. Это следующие символы:

```
! @ # $ & ~ % * ( ) [ ] { } ' " \ : ; > < ` пробел.
```

Если имя файла содержит один из этих символов (это не рекомендуется, но возможно), то вы должны перед этим символом поставить символ обратного слэша "\" (в том числе и перед самим этим слэшем, т. е. повторить его дважды).

```
[user]$ mkdir \\my\&his
```

Можно также заключить имя файла или каталога с такими символами в

двойные кавычки. Например, для создания каталога с именем "My old files" следует использовать команду:

```
[user]$ mkdir "My old files"
```

так как команда

```
[user]$ mkdir My old files
```

создаст каталог с именем "My".

Аналогичным образом можно поступать и с другими символами, перечисленными выше, т. е. их можно включать в имена файлов, если имя файла взять в двойные кавычки или отменить специальное значение символа с помощью обратного слэша. Но все же предпочтительнее не использовать эти символы, включая пробел, в именах файлов и каталогов, потому что могут возникнуть проблемы при обращении к таким файлам из некоторых приложений, а также при переносе таких файлов в другие файловые системы.

Но к точке сказанное не относится, и в Linux часто ставят более одной точки в именах файлов, например, This_is.a.forth-chapter_of_my_book.about.Linux. При этом теряет смысл такое понятие (принятое в DOS), как расширение имени файла, хотя все же часто последние части имени, отделенные точками, используют для обозначения файлов каких-то особых типов (например, .tar.gz используется для обозначения сжатых архивов). Но исполняемые и неисполняемые файлы в Linux распознаются не по расширениям имен файлов. Для этого существуют другие признаки, о которых мы скажем чуть позже. Точка имеет особое значение в именах файлов. Если она является первым символом имени, то данный файл считается скрытым для некоторых команд, например, он не показывается при выполнении команды ls.

В Linux различаются символы верхнего и нижнего регистра в именах файлов. Поэтому FILENAME.tar.gz и filename.tar.gz вполне могут существовать одновременно и являться именами разных файлов.

Практическая часть

Изучить синтаксис использвания утилит работы с файлами в Linux с помощью системы man.

Получить список файлов в папке

Найти все файлы в файловой системе, которые начинаются с символа "a"

Написать на языке Bash скрипт который выводит на экран содержимое всех файлов в указанной папке

По результатам работы необходимо подготовить отчет с приведением команд и результатов работы.

Приложение 1

Министерство образования и науки РФ

ТОМСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ СИСТЕМ УПРАВЛЕНИЯ И РАДИОЭЛЕКТРОНИКИ (ТУСУР)

Кафедра ЭЛЕКТРОННЫХ ПРИБОРОВ (ЭП)

дисциплина «Глобальные и локальные компьютерные сети»

ОТЧЕТ

по лабораторной работе

«_____»

Выполнил студент

гр. 348

XXXXXXXXXXX

Проверил преподаватель

_____

_____

20___