# 1. ОСНОВНЫЕ КОМАНДЫ ОС UNIX

Операционная система ОС ***Linux*** создана на основе ОС ***UNIX*** и во многом имеет схожую структуру и систему команд. Пользователь может работать в текстовом режиме с помощью командной строки, или с использованием графического интерфейса ***X Window*** и одного из менеджеров рабочего стола (например, ***KDE*** или ***GNOME***). Причем, одновременно в системе могут работать 7 пользователей (6- в текстовом режиме консоли и 1 – в графическом режиме), переключение между пользователями осуществляется по нажатию клавиш:

.

***Ctrl***

***Alt***

***F1***

***Ctrl***

***Alt***

***F7***

В табл. 1 приведены основные команды системы

Таблица 1

|  |  |  |  |
| --- | --- | --- | --- |
| Команда | Аргументы/ключи | Пример | Описание |
| ***dir*** | каталог | ***dir***  ***dir /home*** | Выводит на консоль содержимое каталога |
| ***ls*** | ***-all***  и другие (см. man) | ***ls -all*** | Выводит на консоль содержимое каталога |
| ***ps*** | ***-a***  ***-x*** и другие (см. man) | ***ps -a*** | Выводит на консоль список процессов |
| ***mkdir*** | имя каталога | ***mkdir stud11*** | Создает каталог |
| ***rmdir*** | имя каталога | ***rmdir stud11*** | Удаляет каталог |
| ***rm*** | файл | ***rm myfile1*** | Удаляет файл |
| ***mv*** | файл новое имя | ***mv myfile1 myf1*** | Переименование файла |
| ***cat*** | файл | ***cat 1.txt*** | Вывод файла на консоль |
| ***cd*** | имя каталога | ***cd home*** | Переход по каталогам |
| ***grep*** | (см. man) | ***grep "^a" "words.txt"*** | Поиск строки в файле |
| ***kill*** | ***pid*** процесса | ***kill 12045*** | Уничтожает процесс |
| ***top*** |  |  | Выводит на консоль список процессов |
| ***htop*** |  |  | Выводит на консоль полный список запущенных процессов |
| ***su*** |  |  | Переход в режим root |
| ***chmod*** | права доступа файл | ***chmod 777 1.txt*** | Изменение прав доступа к файлам |
| ***mount*** | устройство каталог | ***mount /dev/cdrom /MyCD*** | Монтирование устройств |
| ***dd*** | ***if=***файл ***of=***файл ***bs=n count=n*** | ***dd if=/dev/hda1***  ***of=/F.bin bs=512 count=1*** | Копирование побайтное |
| ***ln*** | файл1 файл2  ***-l*** | ***ln файл1 файл2***  ***ln –l файл1 файл2*** | Создать жёсткую или символическую ссылку на файл |
| ***uname*** | ***-a*** | ***uname –a*** | Информация о системе |
| ***find*** | dir файл | ***find /home –name a1.txt*** | Поиск файлов |
| ***man*** |  | ***man fgetc*** | Справка по системе |
| ***info*** |  | ***info fgetc*** | Справка по системе |

Linux и Windows используют различные файловые системы для хранения и организации доступа к информации на дисках. В Linux используются файловые системы- ***Ext2/Ext3***, ***RaiserFS, FFS*** и другие. Все файловые системы имеют поддержку ***журналирования***. ***Журналируемая*** файловая система сначала записывает изменения, которые она будет проводить в отдельную часть файловой системы (***журнал***) и только потом вносит необходимые изменения в остальную часть файловой системы. После удачного выполнения всех транзакций, записи удаляются из ***журнала***. Это обеспечивает лучшее сохранение целостности системы и уменьшает вероятность потери данных. Следует отметить, что ***Linux*** поддерживает доступ к ***Windows***-разделам.

Файловая система ***Linux*** имеет лишь один корневой каталог, который обозначается косой чертой ( ***/*** ). В файловой структуре ***Linux*** нет дисков ***A, B, C, D*** …., а есть только каталоги. В ***Linux*** различаются прописные и строчные буквы в командах, именах файлов и каталогов. В ***Windows*** у каждого файла существует лишь одно имя, в ***Linux*** их может быть много. Это – «***жесткие***» ссылки, которые указывают непосредственно на индексный дескриптор файла. Жесткая ссылка – это один из принципов организации файловой системы ***Linux***.

Структура каталогов ОС ***Linux*** представлена в табл. 1. Есть также несколько полезных сокращений для имен каталогов:

* Одиночная точка (**.**) обозначает текущий рабочий каталог.
* Две точки (**..**) обозначают родительский каталог текущего рабочего.
* Тильда (**~**) обозначает домашний каталог пользователя (обычно это каталог, который является текущим рабочим при запуске Bash).

Таблица 1

|  |  |
| --- | --- |
| ***/*** | Корневой каталог |
| ***/bin*** | Содержит исполняемые файлы самых необходимых для работы системы программ. Каталог ***/bin*** не содержит подкаталогов. |
| ***/boot*** | Здесь находятся само ядро системы (файл ***vmlinuz***-...) и файлы, необходимые для его загрузки. |
| ***/dev*** | Каталог ***/dev*** содержит файлы устройств (драйверы). |
| ***/etc*** | Это каталог конфигурационных файлов, т. е. файлов, содержащих информацию о настройках системы (например, настройки программ). |
| ***/home*** | Содержит домашние каталоги пользователей системы. |
| ***/lib*** | Здесь находятся библиотеки (функции, необходимые многим программам). |
| ***/media*** | Содержит подкаталоги, которые используются как точки монтирования для сменных устройств (CD-ROM'ов, floppy-дисков и др.) |
| ***/mnt*** | Данный каталог (или его подкаталоги) может служить точкой монтирования для временно подключаемых файловых систем. |
| ***/proc*** | Содержит файлы с информацией о выполняющихся в системе процессах. |
| ***/root*** | Это домашний каталог администратора системы. |
| ***/sbin*** | Содержит исполняемые программы, как и каталог ***/bin***. Однако использовать программы, находящиеся в этом каталоге может только администратор системы (***root***). |
| ***/tmp*** | Каталог для временных файлов, хранящих промежуточные данные, необходимых для работы тех или иных программ, и удаляющиеся после завершения работы программ. |
| ***/usr*** | Каталог для большинства программ, которые не имеют значения для загрузки системы. Структура этого каталога фактически дублирует структуру корневого каталога. |
| ***/var*** | Содержит данные, которые были получены в процессе работы одних программ и должны быть переданы другим, и файлы журналов со сведениями о работе системы. |

# 2. ЛАБОРАТОРНЫЕ РАБОТЫ

# ЛАБОРАТОРНАЯ РАБОТА №1 УПРАВЛЕНИЕ ОС LINUX, ИНТЕРПРЕТАТОР BASH

Цель работы – изучить основные объекты, команды, типы данных и операторы управления интерпретатора BASH; создать скрипт-файл.

**Теоретическая часть**

***Bash*** - это ***sh***-совместимый интерпретатор командного языка, выполняющий команды, прочитанные со стандартного входного потока или из файла. ***Скрипт-файл*** – это обычный текстовый файл, содержащий последовательность команд ***bash***, для которого установлены права на выполнение. Пример скрипта, выводящего содержимое текущего каталога на консоль и в файл:

***#!/bin/bash***

***dir***

***dir > 1.txt***

Следующие переменные используются командным интерпретатором.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| ***$0,$1,$2,$3…*** | | | Значения аргументов командной строки при запуске скрипта. Где ***$0***-имя самого файла скрипта, ***$1***- первый аргумент, ***$2***- второй аргумент, и т.д. | |  |
| ***$@*** | | | Все аргументы командной строки | |  |
| ***$?*** | | Код возврата последней команды | | | |
|  |  | | |  |  |

Пример простого скрипта, выводящего на консоль и в файл содержимое каталога, где имя каталога передаётся скрипту в качестве аргументов при запуске:

Запуск скрипта: ***>./home/stud 1.sh***

Скрипт:

***#!/bin/bash***

***dir $1***

***dir $1 > 1.txt***

Можно создать собственную переменную и присвоить ей значение:

***A=121***

***A=”121”***

***let A=121***

***let “A=А+1”***

Вывод значения на консоль: ***echo $A***

Проверка условия: ***test[expr]***

где ***expr***: а) для строк: ***S1 = S2*** ***S1***содержит ***S2***

***S1 != S2******S1***не содержит ***S2***

***-n S1*** если длина ***S1 >0***

***-z S1*** если длина ***S1 =0***

б) целые ***i1***и ***i2***

***i1 – ge i2***

***i1 – gt i2***

***i1 – ie i2***

***i1 – et i2***

***i1 – nt i2***

в) файлы

***-d name\_file*** является ли файл каталогом

***-f name\_file*** является ли файл обычным файлом

***-r name\_file*** доступен ли файл для чтения

***-s name\_file*** имеет ли файл ненулевую длину

***-w name\_file*** доступен ли файл для записи

***-x name\_file*** является ли файл исполняемым

г) логически операции

***!exp*** логическое отрицание (не)

***exp1 –a exp2*** умножение условий (и)

***exp1 –o exp2*** сложение условий (или)

Проверка условия: ***if [expr ]***

***then com 1*** Если условие ***expr=true*** то команда

***…***

***(elif expr2***

***…***

***)***

***else***

***…***

***fi***

Проверка нескольких условий: ***case string1 in***

***str 1)***

***…***

***; ;***

***str 2)***

***…***

***; ;***

***str 3)***

***…***

***; ;***

***\*)***  // default

***…***

***; ;***

***esac***

Функция пользователя: ***fname2 (arg1,arg2...argN)***

***{  
commands  
}***

Организация циклов:

1. ***for var1 in list***

***do***

***…***

***done***

2. ***while exp***

***…***

***end***

3. ***until exp // аналог do-while***

***do***

***…***

***done***

**Порядок выполнения работы**

1. Изучить теоретическую часть лабораторной работы.
2. В консольном режиме создать, используя команды из табл.1, в ***домашней папке*** подкаталог: ***/номер\_группы/ФИО\_студента,*** где в дальнейшем будут храниться все файлы студента. Перейти в корневой каталог и вывести его содержимое используя команды ***dir*** и ***ls –all*** , проанализировать различия.
3. Проверить действие команд ***ps***, ***ps –x, top, htop***. Найти в справочной системе используя команду ***man*** справку по функциям ***fprintf***, ***fputc*** и команде ***ls***.
4. В текстовом редакторе ***nano*** (вызов: ***nano hello\_linux.c***) написать программу ***hello\_linux.c***, выводящую на экран фразу ***“Hellor Linux”***. Компилировать полученную программу компилятором ***gcc***: ***gcc hello\_linux.c -o hello\_linux.exe***. Запустить полученный файл ***hello\_linux.exe*** на выполнение: ***./hello\_linux .exe***
5. Написать скрипт, выводящий на консоль и в файл все аргументы командной строки. Выводить одной строкой с пробелами между аргументами.
6. Написать скрипт, выводящий в файл (имя файла задаётся пользователем в качестве первого аргумента командной строки) имена всех файлов с заданным расширением (третий аргумент командной строки) из заданного каталога (имя каталога задаётся пользователем в качестве второго аргумента командной строки). Если не хватает параметров, то вывести подсказку с описанием количества параметров и их значения. Если количество параметров верное, но они не валидные, то вывести сообщение об ошибке в поток ошибок (поток номер 2). Имя каждого файла должно печататься с новой строки. Имена файлов должны быть отсортированы в алфавитном порядке.
7. Написать скрипт, компилирующий и запускающий программу (имя исходного файла и ***exe-*** файла результата задаётся пользователем в качестве аргументов командной строки). В случае ошибок при компиляции вывести на консоль сообщение об ошибках и не запускать программу на выполнение.

## Варианты индивидуальных заданий

1. Написать скрипт для поиска файлов заданного размера в заданном каталоге (имя каталога задаётся пользователем в качестве третьего аргумента командной строки). Диапазон (мин.- мах.) размеров файлов задаётся пользователем в качестве первого и второго аргумента командной строки. Формат вывода: размер\_файла\_в\_байтах относительный\_путь\_к\_файлу. Файлы должны быть отсортированы по возрастанию размера.

Пример:

./var1.sh 0 500 /tmp

1 /tmp/2.out

2 /tmp/1

144 /tmp/323

499 /tmp/54.txt

1. Написать скрипт с использованием цикла ***for***, выводящий на консоль относительные пути к файлам, размеры и права доступа для всех файлов в заданном каталоге и всех его подкаталогах (имя каталога задается пользователем в качестве первого аргумента командной строки). На консоль выводится общее число просмотренных файлов.

Пример:

./var2.sh usr/bin

-rwxr-xr-x 186672 /usr/bin/kcolorschemeeditor

-rwxr-xr-x 39 /usr/bin/7z

-rwxr-xr-x 89488 /usr/bin/zipnote

-rwxr-xr-x 14656 /usr/bin/lsattr

Listed 4 files

1. Написать скрипт для поиска заданной пользователем строки во всех файлах заданного каталога и всех его подкаталогов (строка и имя каталога задаются пользователем в качестве первого и второго аргумента командной строки). На консоль выводятся размер и относительный путь к файлам, в содержимом которых присутствует заданная строка. Файлы должны быть отсортированы по размеру в возврастающем порядке.
2. Написать скрипт поиска одинаковых по содержимому файлов в двух каталогах, например, ***Dir1*** и ***Dir2***. Пользователь задаёт имена ***Dir1*** и ***Dir2*** в качестве первого и второго аргумента командной строки. В результате работы программы файлы, имеющиеся в ***Dir1***, сравниваются с файлами в ***Dir2*** по их содержимому. На экран выводятся число просмотренных файлов и результаты сравнения.

Пример:

./var4 tasks/ tasks/

tasks/1.sh == tasks/1.sh

tasks/hello\_linux.c == tasks/hello\_linux.c

tasks/4.sh == tasks/4.sh

tasks/compile.sh == tasks/compile.sh

tasks/.4.sh.swp == tasks/.4.sh.swp

tasks/2.1.sh == tasks/2.1.sh

tasks/2.sh == tasks/2.sh

tasks/args\_out.sh == tasks/args\_out.sh

Number of viewed files is 18

1. Написать скрипт, находящий в заданном каталоге и всех его подкаталогах все файлы, владельцем которых является заданный пользователь. Имя владельца и каталог задаются пользователем в качестве первого и второго аргумента командной строки. Скрипт выводит результаты в файл (третий аргумент командной строки) в виде полный путь, имя файла, его размер. На консоль выводится общее число просмотренных файлов.
2. Написать скрипт, находящий в заданном каталоге и всех его подкаталогах все файлы заданного размера. Диапазон (мин.- мах.) размеров файлов задаётся пользователем в качестве первого и второго аргумента командной строки. Скрипт выводит результаты поиска в файл (третий аргумент командной строки) в виде: полный путь, имя файла, его размер. На консоль выводится общее число просмотренных файлов.
3. Написать скрипт подсчитывающий суммарный размер файлов в заданном каталоге и всех его подкаталогах (имя каталога задаётся пользователем в качестве аргумента командной строки). Скрипт выводит результаты подсчёта в файл (второй аргумент командной строки) в виде каталог (полный путь), суммарный размер файлов число просмотренных файлов.