Почечура Артемий   
М8О-106Б-20

Лабораторная работа №2

Итак, начнём. Изначально, при входе в терминал, мы находимся в домашней директории. Команда pwd даёт нам понять наше местонахождение в терминале, поэтому с помощью неё мы можем узнать путь к домашней директории:

root@DESKTOP-5HM2HTK:~# pwd

/root

.  
 Написав команду ls, мы можем понять, какие директории или файлы находятся в текущей директории или, если написать ls x/, где x – это путь к директории, какие директории или файлы находятся в указанной директории. 1s ~ позволяет узнать директории или файлы в домашней директории, а ls –a – директории и файлы, начинающиеся с “.” и “..” . Пример :

root@DESKTOP-5HM2HTK:/# ls root/Desktop/Laba/

Quak

root@DESKTOP-5HM2HTK:/# ls -a

. bin dev home lib lib64 media opt root sbin srv tmp var

.. boot etc init lib32 libx32 mnt proc run snap sys usr

.

Благодаря команде cd мы можем делать переходы в те директории, которые можем это сделать. cd .. – переход из данной директории на уровень выше, cd ~ - переход в домашнюю директорию, cd x/ - переход в дирекцию с путём x. Пример:

root@DESKTOP-5HM2HTK:/# cd root/Desktop/Laba

root@DESKTOP-5HM2HTK:~/Desktop/Laba#

.

Кроме того, что смотреть, какие директории и файлы у нас на пути, и перемещаться в директории, мы ещё можем и создавать их! Делается это просто: с помощью команды mkdir мы создаём директории, с помощью touch - файлы(если записать без кавычек через пробел, то элементов создастся несколько). Очевидно, мы их можем создавать не только в текущей директории, а в любой, просто проложив путь в нужную нам директорию. Удалять файлы можно благодаря команде rm, директории – благодаря команде rm –r. Пример:

root@DESKTOP-5HM2HTK:~# ls

Desktop Tests file

root@DESKTOP-5HM2HTK:~# mkdir TopGear

root@DESKTOP-5HM2HTK:~# touch BBC

root@DESKTOP-5HM2HTK:~# ls

BBC Desktop Tests TopGear file

root@DESKTOP-5HM2HTK:~# rm -r BBC TopGear/

root@DESKTOP-5HM2HTK:~# ls

Desktop Tests file

.

Также можно задать путь из несуществующих директорий и после mkdir написать –p. Тогда по пути создадутся все директории, которых не было:

root@DESKTOP-5HM2HTK:~# ls

Desktop Tests file

root@DESKTOP-5HM2HTK:~# mkdir -p lr/Sparta/Greece

root@DESKTOP-5HM2HTK:~# ls lr/Sparta

Greece

.

Двигаемся дальше. У нас также есть возможность создавать копии, перезаписывать и перемещать файлы. Команда cp работает таким образом: написав cp file1 file2, создастся точная копия file1 (c названием file2), если file2 не было. Если же он был, то file2 станет перезаписью file1. Данная команда также работает с директориями, если после cp дописать -r: если при записи cp –r dir1/ dir2/ второй директории не было, то создастся точная копия dir1/, если же она уже было, то в неё поместиться копия dir1/:

root@DESKTOP-5HM2HTK:~# ls

Desctop Tests file

root@DESKTOP-5HM2HTK:~# ls Desctop/

File3

root@DESKTOP-5HM2HTK:~# cp -r Desctop/ Tests/

root@DESKTOP-5HM2HTK:~# ls Tests/Desctop/

File3

.

Благодаря команде mv мы можем перемещать cодержимое файлов в другие файлы или в новые файлы, или директории, или файлы между директориями. При работе этой команды с директориями уже не надо дописывать –r! При записи mv file1 file2, первый файл удалиться, а второй станет его точной копией. При записи mv dir1/ dir2/, dir2/ станет копией dir1/, если dir2/ не было, а если dir2/ был, то в него переместиться dir1/. Нельзя перемещать директории в файлы! При записи команды file1 и dir1/ всегда должны существовать (это правило также относится к команде cp)! Пример:  
 root@DESKTOP-5HM2HTK:~# mkdir -p Duck Giant/Rock/Water

root@DESKTOP-5HM2HTK:~# mv Giant/Rock/Water/ Duck

root@DESKTOP-5HM2HTK:~# ls Duck

Water

.

Если вдруг мы забыли, что делает та или иная команда, то, прописав man х, терминал выведет полное описание команды x.  
Клавиша Tab поможет нам при вводе данных: после команды и первой буквы названия файла или директории, нажав Tab, терминал предложит возможные варианты или вовсе дополнит название, если на данную букву есть лишь одно название элемента.

В терминале мы можем совершать несколько команд одновременно. Для этого мы между командами пишем либо &&(и), либо ||(или). При написании &&, если одна из команд не выполняется, то не выполняются и все последующие за ней. При написании || выполнится первая успешная из команд. Пример:

root@DESKTOP-5HM2HTK:~# mkdir 1991 && (mkdir Push || mkdir Load) && cp file1 file2 && mkdir Direct

cp: cannot stat 'file1': No such file or directory

root@DESKTOP-5HM2HTK:~# ls

1991 Desktop Push file

.

Проверить успешность прошлого действия мы можем при помощи команды echo $?. Если команда была успешна, будет выведен 0, любой вывод кроме нуля означает провальное выполнение программы. Пример:

root@DESKTOP-5HM2HTK:~# mkdir Peace && touch file2 && cp file file3 && cp file2 file3 && mv file6 Desktop

mv: cannot stat 'file6': No such file or directory

root@DESKTOP-5HM2HTK:~# echo $?

1

.

В терминале можно работать с текстом. Доступ к этому дают следующие команды:

echo ‘Hello’>file – записывает в файл то, что указано в кавычках;

nano – текстовый редактор;

сat – записывает любой объём сообщения;

head – записывает/читает первые 10 строк;

tail – записывает/читает последние 10 строк;

grep – находит строчки, в которых содержится то, что ему зададут;

wc – считает слова (через пробел добавить –w), байты(через пробел добавить –с), строчки(через пробел добавить –l) в тексте.

Работают они следующим образом: если просто записать одну из команд (nano, cat, head, tail), то запись будет производиться с клавиатуры и выводиться в терминал (так называемый стандартный ввод). Если же мы после команды напишем >file, то запись будет производиться уже в файл с названием file; если после команды напишем >>file, то файл перезаписан не будет, в него будет добавлен вводимый текст. Есть ещё одна форма записи (полезна при написании скриптов): cat>file<<EOF.Так, записываться информация у нас будет ровно до строчи EOF (сама она записана не будет). Написав через пробел после команды название файла, мы прочтём, что в нём записано. Команда grep работает так: после записи команды пишем в кавычках то, что хотим найти, и далее название файла. В терминале выведется строчка с искомыми элементами, а сами они будут выделены красным. Пример:

root@DESKTOP-5HM2HTK:~# cat>file3<<EOF

> asdf

> qwerty

> EOF

root@DESKTOP-5HM2HTK:~# cat file3

asdf

qwerty

root@DESKTOP-5HM2HTK:~# grep 'rty' file3

qwerty

.

Также для команды wc можно сразу писать сообщение, в котором она посчитает нужную нам информацию, без взятия данных из файлов:

root@DESKTOP-5HM2HTK:~# wc -c <<<"Hello"

6

.

Теперь можно приступить к конвейерам. Конвейер представляет из себя последовательность действий, которые взаимосвязаны тем, что каждое из них получает входные данные из прошлого этапа. Важно понимать, что действия, которые имеют схожую функцию и в конвейере будут выполняют одно и то же, не следует засовывать в конвейер одновременно. Например, если в сообщении не больше 10 строчек, то cat и head будут выполнять одну и ту же функцию, поэтому для построения правильного конвейера нужна лишь одна из этих команд. В конвейере нужно использовать file<, когда мы хотим брать данные из файла и использовать для одной из команд. Рассмотрим пример одного конвейера:

root@DESKTOP-5HM2HTK:~# cat file3

One One

Two Two

Three Three

Four Four

Five Five

Six six

Seven Seven

Eight Eight

Nine Nine

Ten Ten

root@DESKTOP-5HM2HTK:~# head -n5 <file3 | tail -n3 | wc -w

6

.

Можно совершать не такие очевидные действия, какие мы рассматривали в прошлых примерах, используя знак $. Например, мы можем перейти в директорию, которая идёт первая (вторая, третья, …) по списку ниже текущей. Делается это следующим образом:

root@DESKTOP-5HM2HTK:~# ls

Desktop Laba file

root@DESKTOP-5HM2HTK:~# cd “$(ls “$(pwd)” | head -n1)”

root@DESKTOP-5HM2HTK:~/Desktop#

.

Есть возможность пользоваться математикой и задавать переменные:

root@DESKTOP-5HM2HTK:~# A=10

root@DESKTOP-5HM2HTK:~# cat>file3

25

root@DESKTOP-5HM2HTK:~# echo "$((A\*25-"$(cat file3)"))"

225

root@DESKTOP-5HM2HTK:~#

.

Кстати, отличия ‘…’, ”…” , …, можно легко проследить на этом примере:

root@DESKTOP-5HM2HTK:~# A="Hello World"

root@DESKTOP-5HM2HTK:~# mkdir $A

root@DESKTOP-5HM2HTK:~# ls

Desktop Hello World file

root@DESKTOP-5HM2HTK:~# mkdir '$A'

root@DESKTOP-5HM2HTK:~# ls

'$A' Desktop Hello World file

root@DESKTOP-5HM2HTK:~# mkdir "$A"

root@DESKTOP-5HM2HTK:~# ls

'$A' Desktop Hello 'Hello World' World file

.

Двигаемся дальше. При помощи команды ping мы можем отправлять пакеты данных на определённый сайт и получать обратные данные в ответ. Получение обратных данных будет означать, что доступ к файлу есть. Например:

root@DESKTOP-5HM2HTK:~# ping ya.ru

PING ya.ru (87.250.250.242) 56(84) bytes of data.

64 bytes from ya.ru (87.250.250.242): icmp\_seq=1 ttl=250 time=13.0 ms

64 bytes from ya.ru (87.250.250.242): icmp\_seq=2 ttl=250 time=6.54 ms

64 bytes from ya.ru (87.250.250.242): icmp\_seq=3 ttl=250 time=6.52 ms

“и так далее”

.

Перестать получать данные мы можем благодаря сочетанию клавиш Ctrl+C. Оно прекращает работу текущего процесса и выводится то, что успело обработаться.

У нас есть возможность запускать процессы на заднем фоне. Происходит это следующим образом: мы пишем команду, а после неё ставим знак &. Таким образом мы переводим процесс на задний фон. Проще говоря, происходит перевод процесса выполнения команды из текущего ввода на задний план, при этом сам её процесс не прекращается, а лишь останавливается. Команда jobs позволяет узнать состояние процессов на фоне. Сочетание Ctrl+Z замораживает текущий процесс команды и переводит его на задний фон. Пример:

root@DESKTOP-5HM2HTK:~# head &

[2] 25

root@DESKTOP-5HM2HTK:~# head

^Z

[2]- Stopped head

[3]+ Stopped head

root@DESKTOP-5HM2HTK:~# jobs

[1] Stopped head

[2]- Stopped head

[3]+ Stopped head

root@DESKTOP-5HM2HTK:~#

.

Команды fg и bg позволяют активировать замороженные процессы соответственно на переднем или заднем плане. Если мы просто напишем fg или bg, то активируется первый по приоритетности процесс. Слева от названия самих команд мы видим их состояние, а также “+”, “-” или “ ”. Это как раз показывается их статус: при написании fg или bg активируется сначала процесс со знаком “+”, после него процесс со знаком “-” поменяет приоритетность на “+”, а один из других получит статус “-”. Пример:

root@DESKTOP-5HM2HTK:~# cat &

[1] 28

root@DESKTOP-5HM2HTK:~# head &

[2] 29

[1]+ Stopped cat

root@DESKTOP-5HM2HTK:~# tail &

[3] 30

[2]+ Stopped head

root@DESKTOP-5HM2HTK:~# jobs

[1] Stopped cat

[2]- Stopped head

[3]+ Stopped tail

root@DESKTOP-5HM2HTK:~# fg

tail

root@DESKTOP-5HM2HTK:~# fg

head

.

С помощью команды disown мы можем убрать заданную задачу из таблицы задач командной оболочки. Рассмотрим пример:

root@DESKTOP-5HM2HTK:~# jobs

[1] Stopped head

[2]- Stopped cat

[3]+ Stopped tail

root@DESKTOP-5HM2HTK:~# disown head

-bash: warning: deleting stopped job 1 with process group 96

root@DESKTOP-5HM2HTK:~# jobs

[2]- Stopped cat

[3]+ Stopped tail

(или можно было записать: disown %1)

.

Увидеть такие задачи, а также несколько других мы можем с помощью команды ps (команда ps –e выдаст ещё больше задач). В этом списке будут указаны процессы, а также их уникальные идентификаторы в колонке PID, с которыми также можно поработать. Например, команда kill “PID” завершает процессы с указанным идентификатором, но только те, которые не заморожены. Приписка “-9” позволяет завершать ДАЖЕ замороженные процессы. Например:

root@DESKTOP-5HM2HTK:~# evince &

[1] 168

root@DESKTOP-5HM2HTK:~# evince

^Z

[2]+ Stopped evince

root@DESKTOP-5HM2HTK:~# jobs

[1]- Activated evince &

[2]+Stopped evince

root@DESKTOP-5HM2HTK:~# ps

PID TTY TIME CMD

7 tty1 00:00:00 init

8 tty1 00:00:00 bash

168 tty1 00:00:00 evince

169 tty1 00:00:00 evince

171 tty1 00:00:00 ps

root@DESKTOP-5HM2HTK:~# kill 168 169

root@DESKTOP-5HM2HTK:~# ps

PID TTY TIME CMD

7 tty1 00:00:00 init

8 tty1 00:00:00 bash

169 tty1 00:00:00 evince

173 tty1 00:00:00 ps

[1]- Terminated evince

root@DESKTOP-5HM2HTK:~# kill -9 169

root@DESKTOP-5HM2HTK:~# ps

PID TTY TIME CMD

7 tty1 00:00:00 init

8 tty1 00:00:00 bash

175 tty1 00:00:00 ps

[1]+ Killed evience

.

Двигаемся дальше. Команда ls –l выдаёт список файлов в указанной директории (или в текущей если не указать директорию), а в первой колонке приведено, какие действия доступны с тем или иным объектом для каждого ранга пользователей. Первые три буквы (не считая d для директорий и “-” для файлов) показывают доступные действия для данного пользователя (user), вторые три – для группы (group), третьи три - для всех остальных (others). Буква r (read) означает возможность чтения файла или просмотра объектов в директории, w (write) – возможность записывания/дописывания текста в файл или создание/удаление файлов/директорий в директории, x (execute) – возможность активации файла или перемещение в директорию. Изменять права на действия с объектами для каждого ранга пользователей можно при помощи команды chmod. После пробела мы указываем, для каких групп пользователей мы хотим применить изменения (“u”, “g” или “o”, а можно и для нескольких сразу, написав их слитно (без указания группы выполняет действие для всех)) и далее пишем, какое действие хотим сделать: + (разрешить) или – (запретить) и название действия (“r”, “w” или “x”, а можно, опять же, и несколько сразу, написав их слитно). Далее пишем название файла или директории. Дописав к chmod через пробел –R мы будем изменять права не только указанного объекта, но и всех объектов, которые в него входят:

root@DESKTOP-5HM2HTK:~# ls -l Desktop/

total 0

drwxrwxrwx 1 root root 4096 Sep 15 23:42 Guap

root@DESKTOP-5HM2HTK:~# chmod u-w Desktop/

root@DESKTOP-5HM2HTK:~# ls -l

total 0

dr-xrwxrwx 1 root root 4096 Sep 15 23:42 Desktop

-rw-rw-rw- 1 root root 0 Sep 10 12:17 file

root@DESKTOP-5HM2HTK:~# ls -l Desktop/

total 0

drwxrwxrwx 1 root root 4096 Sep 15 23:42 Guap

root@DESKTOP-5HM2HTK:~# chmod -R -r Desktop

root@DESKTOP-5HM2HTK:~# ls -l Desktop/

total 0

d-wx-wx-wx 1 root root 4096 Sep 15 23:42 Guap

root@DESKTOP-5HM2HTK:~#

.

А вот пример совершения действия без наличия прав:

root@DESKTOP-5HM2HTK:~# ls -l

total 0

drwxrwxrwx 1 root root 4096 Sep 16 21:58 Desktop

-rw-rw-rw- 1 root root 4 Sep 16 21:59 file

root@DESKTOP-5HM2HTK:~# ./file

-bash: ./file: Permission denied

root@DESKTOP-5HM2HTK:~#

.

Изменять права можно также с помощью цифр. Для этого нужно отдельно рассматривать каждую категорию пользователей как число в двоичной системе счисления, а права на действия – как его разряды. Например:

root@DESKTOP-5HM2HTK:~# ls -l

total 0

drwxrwxrwx 1 root root 4096 Sep 16 21:58 Desktop

-rw-rw-rw- 1 root root 4 Sep 16 21:59 file

root@DESKTOP-5HM2HTK:~# chmod 777 file

root@DESKTOP-5HM2HTK:~# ls -l

total 0

drwxrwxrwx 1 root root 4096 Sep 16 21:58 Desktop

-rwxrwxrwx 1 root root 4 Sep 16 21:59 file

root@DESKTOP-5HM2HTK:~# chmod 535 file

root@DESKTOP-5HM2HTK:~# ls -l

total 0

drwxrwxrwx 1 root root 4096 Sep 16 21:58 Desktop

-r-x-wxr-x 1 root root 4 Sep 16 21:59 file

root@DESKTOP-5HM2HTK:~#

.

Благодаря командам chown и chgrp мы можем изменять соответственно владельца и группу файла или директории, к которым он/она принадлежит.

Если перед нами стоит задача совершить конкретное действие с некоторым количеством файлов, то перечисление можно заменить очень ёмкой строкой, если их названия одинаково начинаются. Делается это просто: указывается путь к одному из файлов (все они должны лежать в одном месте) и дальше дописывается:

“?” – если символ ровно один;

“\*” – если символов любое количество.

Пример:

root@DESKTOP-5HM2HTK:~/Desktop# ls ~

Desktop Frog file file0 file1 file10 file2 file3 file4 file5 file6 file7 file8 file9

root@DESKTOP-5HM2HTK:~/Desktop# mv ~/file?\* ~/Frog

root@DESKTOP-5HM2HTK:~/Desktop# ls ~/Frog

file0 file1 file10 file2 file3 file4 file5 file6 file7 file8 file9

root@DESKTOP-5HM2HTK:~/Desktop#

.

Также можно выбирать файлы из конкретного диапазона цифр или букв:

root@DESKTOP-5HM2HTK:~# ls

Desktop file file11 file13 file8 filead fileaf filebe filecd filecf

FRog file10 file12 file14 file9 fileae filebd filebf filece

root@DESKTOP-5HM2HTK:~# mv file1[1-3] fileb[d-f] FRog

root@DESKTOP-5HM2HTK:~# ls FRog

file11 file12 file13 filebd filebe filebf

При создании/удалении файлов/директорий квадратные скобки будут восприниматься как части имени объекта, поэтому, если мы хотим создать/удалить несколько элементов, то нужно использовать фигурные скобки (промежуток в них обозначается двумя точками). Пример:

root@DESKTOP-5HM2HTK:~# touch Frog/file{oo,pp,,i}gep{3..10}

root@DESKTOP-5HM2HTK:~# ls Frog

filegep10 filegep6 fileigep10 fileigep6 fileoogep10 fileoogep6 fileppgep10 fileppgep6 filegep3 filegep7 fileigep3 fileigep7 fileoogep3 fileoogep7 fileppgep3 fileppgep7 filegep4 filegep8 fileigep4 fileigep8 fileoogep4 fileoogep8 fileppgep4 fileppgep8 filegep5 filegep9 fileigep5 fileigep9 fileoogep5 fileoogep9 fileppgep5 fileppgep9

root@DESKTOP-5HM2HTK:~#

.

На echo мы также можем пронаблюдать отличие квадратных скобок от фигурных:

root@DESKTOP-5HM2HTK:~# echo [1-5]

[1-5]

root@DESKTOP-5HM2HTK:~# echo {1..5}

1 2 3 4 5

root@DESKTOP-5HM2HTK:~#

.

Если мы не хотим писать длинное название файла, но хотим к его названию

добавить что-то в конце или создать его копию, но с новым именем, то можем это заменить такой функцией:

root@DESKTOP-5HM2HTK:~# mv ThereMustBeYourAdverdesing{,+79094567645}

root@DESKTOP-5HM2HTK:~# ls

Desktop Frog ThereMustBeYourAdverdesing+79094567645 file

root@DESKTOP-5HM2HTK:~#

.

Таким образом мы сможем также, например, перемещать директории/файлы в другие директории, если их название начинается схоже, а конец отличается.