Linux и bash-скриптинг

Узнать количество ссылок на странице сайта, количество изображений, заголовков и т.п.

Нам понадобится mojo.

```
Узнать количество ссылок на странице:
mojo get https://foo.ru 'a' | wc -l

Узнать количество изображений на странице:
mojo get https://foo.ru 'img' | wc -l

Узнать title страницы:
mojo get https://foo.ru 'title'

Узнать h1 страницы:
mojo get https://foo.ru 'h1'
```

И это только малая часть, что может mojo. В планах построить аналитический скрипт для SEO-оптимизаторов.

Сделать рандом строк в файле

```
shuf list > list.tmp && mv list.tmp list
```

Заменить запятые переносом строк

```
sed -i 's/,/\n/g' list.new # заменить запятые переносом строк
```

Узнать дату установки системы Linux

```
sudo tune2fs -l /dev/sda2 | grep create
```

Отследить изменения в каталоге средствами md5sum

```
    # формируем файл с контрольными суммами
    md5sum * | sort -u > md5.sum

    # вносим изменения в каталог
    touch test.txt

    # формируем ещё раз файл с контрольными суммами
    md5sum * | sort -u > md5-2.sum
```

```
# сравниваем 2 файла с контрольными суммами
diff md5.sum md5-2.sum

# разницу (если она есть) покажет предыдущая команда
```

Проверить целостность файлов указанных в сгенерированном списке md5.sum

```
md5sum -c md5.sum

# если какой-то файл изменён, то будет показано
# ЦЕЛ, ПОВРЕЖДЁН. СОВПАЛА или НЕ СОВПАЛА вычисленная контрольная сумма.
```

scp — копирование файлов с локального компьютера на сервер

```
scp -C ~/files/localdir/file.txt ftpuser@hostname:~/sitename/target-dir/

-r - рекурсивно
-C - использовать сжатие данных
```

Редактирование файла sudoers

Узнать права пользователя:

```
sudo -l
```

Редактирование файла sudoers — дадим возможность запускать sudo для пользователя petrovich:

Чтобы открыть /etc/sudoers, введите:

```
sudo visudo
```

Запишем в файл и сохраним:

```
petrovich ALL=(ALL:ALL) ALL
```

tmux

```
терминал tmux https://habrahabr.ru/post/126996/
```

Команда cron запускает процесс, выполняющий команды в указанные дни и время — crontab

Узнать запущен ли cron:

```
service cron status
```

Просмотр задач в crontab:

```
crontab -l
```

Настройка crontab:

```
crontab -e
```

Описание как создавать расписание: http://devacademy.ru/posts/15-otlichnykh-primierov-dlia-sozdaniia-cron-zadach-v-linux/

Можно записать задания из заранее подготовленного файла:

```
crontab cron-file.txt
```

ПРИМЕРЫ РАСПИСАНИЙ corntab: https://www.shellhacks.com/ru/crontab-format-cron-job-examples-linux/

/var/spool/cron/ — задания для планировщика cron хранятся в данной директории

crontab — http://www.opennet.ru/man.shtml?topic=crontab&category=1 at — http://www.opennet.ru/man.shtml?topic=at&category=1

Man по-русски: http://www.opennet.ru/man.shtml?topic=cron&category=1&russian=0

pass — консольный менеджер паролей

```
sudo apt-get install pass
```

http://rus-linux.net/MyLDP/sec/pass.html https://www.passwordstore.org/ https://linuxcenter.kz/node/2398/blog https://losst.ru/luchshie-menedzhery-parolej-dlya-linux

Проверка SSL-check на смешивание данных. Добиваемся отображения зелёного https

Если у https сайта на страницах есть данные подгружаемые по http, то зелёного замочка в адресной строке не будет.

Вопрос: Как обнаружить http контент?

Ответ: Сервисом https://www.jitbit.com/sslcheck/

Bash: приостановить и возобновить процесс

```
[bash]

Ctrl + Z - приостановить процесс
fg - возобновить процесс
```

Проверьте ваши учётные записи на предмет утечки

```
Сервис для проверки своих учётных записей и e-mail на наличие в базах которые уве

https://haveibeenpwned.com/
```

```
Проверяйте и принимайте меры по смене пароля, его усилению и защите.

Рекомендация сервиса найдена на сайте: https://securelist.ru/
```

Обновить систему Linux, программные пакеты и очистить ненужные зависимости

```
sudo apt-get update && sudo apt-get upgrade && sudo apt-get autoremove
```

Планирование задач в *NIX с помощью at

После запуска at предлагает вам ввести последовательность команд для выполнения. Чтобы закончить ввод команд, следует использовать комбинацию клавиш CTRL-D. Описание основных параметров командной строки at приведено ниже:

```
    at [-m] [-q очередь] [-f файл] ВРЕМЯ

    Значение этих параметров:

    -q используется для указания очереди. Очередь обозначается одной буквой;
    корректными очередями считаются очереди с идентификаторами от a до z и
    от A до Z.

    -m позволяет отправить пользователю сообщение по электронной почте после
    выполнения задачи даже в том случае, когда выполненная программа
    ничего не вывела.

    -f позволяет прочитать команды из файла, а не со стандартного ввода.

    В качестве времени at принимает строки в форматах, совместимых со
    стандартом POSIX.2.
    Принимается строка, указывающая время в формате ЧЧ:ММ, позволяющая выполнить
    команду в назначенное время в течение дня.
    Месяц, день, год: ММДДГГ или ММ/ДД/ГГ или ДД.ММ.ГГ.
    Указание даты должно следовать за указанием времени.
```

Пример:

```
    at 20:05
    [Enter]
    > touch foo.txt
    [Ctrl + D]
```

Посмотреть задания в очереди:

```
atq
```

Удалить задачу (узнав номер задачи через atq:

```
atrm 3
```

Если пользователю не разрешено выполнять команду at, то в файле /etc/at.allow указать имя пользователя, которому дать право на исполнение команды:

```
nano /etc/at.allow
```

http://rus-linux.net/MyLDP/admin/manage-planned-tasks-on-linux-with-the-command-at.html

Вывести в терминал список установленных программ Linux

```
dpkg -l | tail -n+6 | awk '{print($2)}' | less

или

dpkg --get-selections | less
```

install for me

```
    sudo apt-get install ranger mc mutt evolution geany nano keepassx imagemagick
    sinatra firefox lynx w3m pwgen liferea gimp virtualbox vlc youtube-dl shutter
    libreoffice pindgin gnumeric git htop unetbootin nemo xmlstarlet screen laby
    vim tree at kazam traceroute rake anki pass dia tmux guake git jekyll mojo

    +PDF-Shuffler + HPLIP + printer-driver + telegram + heroku + google chrome +
```

Удалить EXITF из изображения (bash)

```
    Посмотреть данные: $ exiftool image.jpeg
    Стереть данные: $ exiftool -all= image.jpeg
    Стереть из всех файлов в каталоге: $ exiftool -all= *
    Стереть из всех jpg в каталоге: $ exiftool -all= *.jpg
```

Trisquel — GNU/Linux дистрибутив с полным использованием в своем составе только свободного ПО без проприетарного ПО и использованием свободной версии ядра Linux

```
Сайт: https://trisquel.info/
Wikipedia: https://ru.wikipedia.org/wiki/Trisquel
Ещё: http://www.opennet.ru/opennews/art.shtml?num=40995
Ещё: http://zenway.ru/page/trisquel
```

SSH, SCP: ssh-keygen: избавляемся от ошибки Received disconnect from Received disconnect from Too many authentication failures for

Ранее, было описано как настроить доступ к серверу по связке ключей через ssh-keygen.

Вопрос: Почему возникает ошибка Received disconnect from Received disconnect from Too many authentication failures for при подключении к серверу? Ответ: В файл конфигурации ~/.ssh добавлено множество ключей и серверу это не нравится, и он

блокирует доступ.

Как решить: В файле ~/.ssh/config для хостов принудительно указать файлы сертификатов в строке IdentityFile

```
    Host mysite
    IdentityFile ~/.ssh/mysite_key.pub
    HostName 00.000.00.00
    User siteadmin
```

При наличии связки ключей ssh-keygen, чтобы подключиться по ssh не по связке ключей, а по паролю, надо вызывать ssh следующим образом:

```
ssh -o 'IdentitiesOnly=yes'
```

Яндекс.XML не работает с адресами IPv6. Как отключить на хостинге IPv6, чтобы работать с Яндекс.XML

Если работа приложения связана с Яндекс.XML, а на VPS работает IPv6, то придётся его отключить и включить IPv4.

Инструкция:

```
    1.Выполните команду
    echo net.ipv6.conf.all.disable_ipv6=1 > /etc/sysctl.d/disableipv6.conf

    2.В файле /etc/hosts закомментируйте все строки, относящиеся к IPv6
    # The following lines are desirable for IPv6 capable hosts
    #::1 ip6-localhost ip6-loopback
    #fe00::0 ip6-localnet
    #ff00::0 ip6-mcastprefix
    #ff02::1 ip6-allnodes
    #ff02::2 ip6-allrouters

    3. Добавьте или измените строки в /etc/ssh/ssh_config
    AddressFamily inet

    4. В /etc/avahi/avahi-daemon.conf
    use-ipv6=no

    5. В /etc/exim4/update-exim4.conf
    disable_ipv6 = true

    6. Перезагрузить и убедиться, что интерфейс не имеет адреса IPv6
    ifconfig
    - после перезагрузки не должно быть строки с IPv6 адресом.

    7. Проверить нет ли подключений и прослушиваемых портов по IPv6
    netstat --all | grep -E "tcp6|udp6"
```

Взято отсюда: http://forum.ispsystem.ru/showthread.php?24588-%D0%9A%D0%B0%D0%BA-%D0%BE%D1%82%D0%BA%D0%BB%D1%8E%D1%87%D0%B8%D1%82%D1%8C-IPv6Б

Опробовано на Debian 7.0 (Wheezy)

Добавить, блокировать, удалить пользователя в *NIX

(Debian)

С правами root. В примере пользователь krdprog

Добавить нового пользователя. Система запросит пароль (ввести 2 раза):

```
adduser krdprog
```

Блокировать пользователя:

```
passwd --lock krdprog
```

Удалить пользователя и всю его домашнюю директорию:

```
deluser --remove-home krdprog
```

Сменить пароль пользователя:

```
passwd
```

Определение ссылок на странице, их количества и типа через bash-скрипт

В справочнике по Linux нашёл описание UNIX-утилиты dog, но потом обнаружилось, что её убрали. Пока я её не нашёл, а нашёл замену. Работу средствами консольного браузера lynx.

Итак, напишем скрипт поиска внутренних ссылок на странице сайта (links.on.page.sh):

```
    #!/bin/bash
    #========================================================================
    # Скрипт поиска внутренних ссылок на странице сайта
    #========================================================================
    URL="$1" ; # адрес страницы заданный через аргумент к скрипту

    # уникальные внутренние ссылки:
    GiveMeListUrlOnPage () {
    lynx -dump -nonumbers -listonly $URL | grep $URL | sort | uniq ;
    }

    GiveMeListUrlOnPage ;

    exit 0
```

Запускаем скрипт, поиск уникальных внутренних ссылок на странице сайта:

```
    $ sh links.on.page.sh krdprog.ru

    если надо сохранить в файл:
    $ sh links.on.page.sh krdprog.ru > url.txt

    если надо показать количество ссылок, то так:
    $ sh links.on.page.sh krdprog.ru | wc -l
```

Можно определять также скрытые и внешние ссылки. Как использовать? Можно, дописать, например, определитель структуры сайта, или скрипт подготовки sitemap.xml, или в связке со скриптом достающим из ТОП-10 Яндекса адреса сайтов по

конкретному ключевому слову, создать анализатор сайтов в ТОП Яндекса по словам (покажет для каждого сайта сколько ссылок на странице), а также разработать анализатор перелинковки сайтов.

ssh-keygen — авторизация по ssh без ввода пароля (по паре ключей RSA)

```
1. Генерация пары ключей:
ssh-keygen -b 2048 -t rsa -f ~/.ssh/sitename_key -C "Key for site"

2. Разложим ключи по местам:
Публичный ключ sitename_key.pub копируем на удалённый сервер в домашнем катал
создав каталог ~/.ssh и меняя название файла на authorized_keys

# выставим права на удалённом сервере:
~$ chmod 700 .ssh/
~$ chmod 600 .ssh/authorized_keys

3. Делаем короткую запись доступа к серверу (на локальной машине):
nano ~/.ssh/config

Содержимое файла ~/.ssh/config:

Host sitename
IdentityFile ~/.ssh/sitename_key
HostName 11.111.111.11
User usernameftp

# Тут мы сделали короткую запись, чтобы не вводить ssh usernameftp@11.111.111
а набирать просто ssh sitename

Пробуем:
$ ssh sitename
```

Можно для копирования ключа использовать команду ssh-copy-id usrername@servername (попробую и при удачном опыте, допишу заметку).

screen (bash)

```
screen - запустить скрин
screen -S taskmane -t taskname - запустить скрин с именем taskname

screen -r taskname - открыть скрин с именем taskname
screen -x taskname - подключиться к скрину запущенному в другом терминале

Ctrl+a+d - свернуть скрин
ctrl+d - выйти из screen

screen -list - список активных скринов
```

Настроим screen. В файле ~/.screenrc надо добавить:

```
hardstatus on
hardstatus alwayslastline
hardstatus string "%w"
#shell /bin/bash
```

```
startup_message off
```

Упростим работу, создадим алиасы (в ~/.bashrc):

```
alias screen1='screen -S 1 -t 1'
alias screen2='screen -S 2 -t 2'
alias screen3='screen -S 3 -t 3'
alias screen4='screen -S 4 -t 4'
alias screen5='screen -S 5 -t 5'

alias screenr1='screen -r 1'
alias screenr2='screen -r 2'
alias screenr3='screen -r 3'
alias screenr4='screen -r 4'
alias screenr5='screen -r 5'

Теперь запускать скрины можно по команде screen1
Сворачивать по: ctrl+a+d
Разворачивать по: screenr1
и т.п. для каждого номера
```

Сравнение двух файлов. Сохранить в третий уникальные строки (bash)

```
grep -f ./file1 -vFx ./file2 > ./file3
```

Получим на выходе все строки из файла file2, которых нет в file1. Строки которые есть в file1 не выводятся. Уникальные строки сохраним в ./file3

Наброски скрипта с применением case (bash)

```
#!/bin/bash

# обработка фотографий в jpg
case "$1" in
     jpg)
          echo "Ок. Это jpg" ;
          ;;
     jpeg|JPEG|JPG)
          echo "Переименуем в jpg" ;
          ;;
     png|PNG)
          echo "Конвертируем png в jpg" ;
          ;;
     gif|GIF)
          echo "Конвертируем gif в jpg" ;
          ;;
     *)
          echo "Этот тип файла не поддерживается операцией" ; # все другие фо
          ;;
esac

exit 0
```

Bash фукция с параметром

```
test_me () {
echo "Hello $1!"
}

test_me "Петрович";
```

Создание каталога для log-файлов (bash)

```
#!/bin/bash

# создание каталога для log-файлов
MkdirLog () {
    if ! [ -d "./log" ] ; then
        mkdir ./log ; # каталог ./log
    fi ;

    if ! [ -d "./log/$(date +%F)" ] ; then
        mkdir ./log/$(date +%F) ; # каталог с текущей датой внутри каталога .
    fi ;

    LogDir="./log/$(date +%F)" ;
}

MkdirLog ;

exit 0
```

Проверка наличия файла из скрипта bash

```
YesOrNotFileFind () {
    if ! [ -f "$1" ] ; then
        echo "Файл $1 не найден" ;
    else
        echo "Файл $1 найден" ;
    fi
}

# передадим в аргументе функции имя файла для проверки на наличие
YesOrNotFileFind "./list" ;
YesOrNotFileFind "./my.config" ;

-e  Файл существует
-f  Файл существует и это обычный файл
-r  Файл существует и он доступен для чтения
-w  Файл существует и он доступен для записи
-x  Файл существует и он исполняемый
-L  Файл существует и это символьная ссылка
-S  Файл существует и это сокет
-d  Директория существует
```

Количество строк в переменную (bash)

Пример: для файла list

```
    ListSring="$(cat ./list | wc -l)" ;

    или

    ListSring="$(wc -l < ./list)" ;
```

Удалить N верхних строк в файле (bash)

Пример: удалить 10 первых строк из файла:

```
    sed '1,10d' test.log > test.tmp.log && mv test.tmp.log test.log

    или с переменной

    HowManyLines=10 ;

    tail -n $((`wc -l test.log | awk '{print $1}'`-$HowManyLines)) test.log > tes
```

Просмотр увеличивающегося log-файла в реальном времени (bash)

```
tail -f test.log
```

Очистка содержимого файла через /dev/null (bash)

```
cat /dev/null > test.log
```

Вывести конкретные строки из файла (bash)

Вывести с 1 по 6 строку из файла test.txt:

```
sed -n 1,6p test.txt
```

Вывести с 4 по 8 строку + строку номер 11 из файла test.txt:

```
sed -n -e 4,8p -e 11p test.txt
```

Вывести строку номер 5:

```
head -n 5 test.txt | tail -n 1
или
sed -n 5p test.txt
```

ИСКЛЮЧИТЬ ИЗ ВЫВОДА СТРОКИ (пример: исключим 1 и 2 строку из test.txt):

```
sed '1,2d' test.txt
```

Генерация случайного целого числа в диапазоне чисел (bash)

Пример: генерация 1 целого числа в диапазоне от 50 до 150

```
shuf -i 50-150 -n 1
```

Скрипт проверки битых ссылок. Используем curl

Напишу на базе curl, скрипт для SEO, который будет проверять битые ссылки на сайте.

```
curl -Lw '%{http_code}' -s -o /dev/null -I krdprog.ru
```

Выдаёт код 200 — страница существует

```
curl -Lw '%{http_code}' -s -o /dev/null -I krdprog.ru/dfhagjdsfyhja8
```

Выдаёт код 404 — страницы нет

Регулярные выражения Bash

```
\ - с обратной косой черты начинаются буквенные спецсимволы, а также он используе
если нужно использовать спецсимвол в виде какого-либо знака препинания;
^ - указывает на начало строки;
$ - указывает на конец строки;
* - указывает, что предыдущий символ может повторяться 0 или больше раз;
+ - указывает, что предыдущий символ должен повторится больше один или больше раз
? - предыдущий символ может встречаться ноль или один раз;
{n} - указывает сколько раз (n) нужно повторить предыдущий символ;
{N,n} - предыдущий символ может повторяться от N до n раз;
. - любой символ кроме перевода строки;
[az] - любой символ, указанный в скобках;
x|y - символ x или символ y;
[^az] - любой символ, кроме тех, что указаны в скобках;
[a-z] - любой символ из указанного диапазона;
[^a-z] - любой символ, которого нет в диапазоне;
\b - обозначает границу слова с пробелом;
\B - обозначает что символ должен быть внутри слова, например, ux совпадет с uxb
или tuxedo, но не совпадет с Linux;
\d - означает, что символ - цифра;
\D - нецифровой символ;
\n - символ перевода строки;
\s - один из символов пробела, пробел, табуляция и так далее;
\S - любой символ кроме пробела;
\t - символ табуляции;
\v - символ вертикальной табуляции;
\w - любой буквенный символ, включая подчеркивание;
\W - любой буквенный символ, кроме подчеркивания;
\uXXX - символ Unicdoe.
```

Подробнее: https://losst.ru/regulyarnye-vyrazheniya-linux http://www.bash-scripting.ru/abs/chunks/ch17.html http://www.k-max.name/linux/ispolzovanie-regulyarnyx-vyrazhenij-regex-v-linux/ http://www.opennet.ru/docs/RUS/bash_scripting_guide/c11895.html

Год, месяц, день, час, минуты (bash)

Запишем в файл с датой и временем:

```
echo "Hello!" > $(date +%Y-%m-%d-%H-%M).txt
```

Операции сравнения в bash-скриптах

```
Сравнение целых чисел:

-eq равно

if [ "$a" -eq "$b" ]

-ne не равно

if [ "$a" -ne "$b" ]

-gt больше

if [ "$a" -gt "$b" ]

-ge больше или равно

if [ "$a" -ge "$b" ]

-lt меньше

if [ "$a" -lt "$b" ]

-le меньше или равно

if [ "$a" -le "$b" ]

< меньше (внутри двойных круглых скобок )

(("$a" < "$b"))

<= меньше или равно (внутри двойных круглых скобок)

(("$a" <= "$b"))

знак > больше (внутри двойных круглых скобок)

(("$a" > "$b"))

знак >= больше или равно (внутри двойных круглых скобок)

(("$a" >= "$b"))
```

Архив журнала Linux Format

```
http://www.linuxformat.ru/archive
```

Конвертация файлов .csv в .xls и не только (ssconvert)

Конвертация одиночного файла:

```
    ssconvert file.csv file.xls

    или

    ssconvert -T Gnumeric_Excel:excel_biff8 file.csv
```

Узнать в какие форматы можно конвертировать:

```
ssconvert --list-exporters
```

Если надо сконвертировать .csv в .html

```
ssconvert -T Gnumeric_html:xhtml file.csv
```

Если надо сконвертировать .csv в .pdf

```
ssconvert -T Gnumeric_pdf:pdf_assistant file.csv
```

Можно конвертировать из любого доступного формата в любой (доступный для конвертации данной утилитой):

```
    ID                                | Description
    Gnumeric_lpsolve:lpsolve          | Решатель линейных программ LPSolve
    Gnumeric_sylk:sylk                | MultiPlan (SYLK)
    Gnumeric_html:roff                | TROFF (*.me)
    Gnumeric_html:latex_table_visible | LaTeX 2e (*.tex) table fragment of visibl
    Gnumeric_html:latex_table         | LaTeX 2e (*.tex) фрагмент таблицы
    Gnumeric_html:latex               | LaTeX 2e (*.tex)
    Gnumeric_html:xhtml_range         | Блок XHTML для экспорта в буфер обмена
    Gnumeric_html:xhtml               | XHTML (*.html)
    Gnumeric_html:html40frag          | Фрагмент HTML (*.html)
    Gnumeric_html:html40              | HTML 4.0 (*.html)
    Gnumeric_html:html32              | HTML 3.2 (*.html)
    Gnumeric_Excel:xlsx2              | ISO/IEC 29500:2008 & ECMA 376 2oe издание
    Gnumeric_Excel:xlsx               | ECMA 376 1oe издание (2006); [MS Excel™ 2
    Gnumeric_Excel:excel_dsf          | MS Excel™ 97/2000/XP & 5.0/95
    Gnumeric_Excel:excel_biff7        | MS Excel™ 5.0/95
    Gnumeric_Excel:excel_biff8        | MS Excel™ 97/2000/XP
    Gnumeric_dif:dif                  | Формат обмена данными (*.dif)
    Gnumeric_glpk:glpk                | Решатель линейных программ GLPK
    Gnumeric_OpenCalc:odf             | ODF 1.2 extended conformance (*.ods)
    Gnumeric_OpenCalc:openoffice      | ODF 1.2 strict conformance (*.ods)
    Gnumeric_stf:stf_csv              | Значения разделённые запятыми (CSV)
    Gnumeric_stf:stf_assistant        | Текст (настраиваемый)
    Gnumeric_XmlIO:sax:0              | Gnumeric XML несжатый (*.xml)
    Gnumeric_XmlIO:sax                | Gnumeric XML (*.gnumeric)
    Gnumeric_pdf:pdf_assistant        | Экспорт в PDF
```

Установить на linux:

```
sudo apt install gnumeric
```

БОНУС:

Конвертировать все файлы в каталоге из .csv в .xls и .csv в .html и разложить по

каталогам xls и html:

```
    mkdir xls html ;

    find * -maxdepth 0 -type f -print0 | xargs -0 -n 1 ssconvert -T Gnumeric_Exce
    xargs -0 -i mv {} xls/ ;

    find * -maxdepth 0 -type f -print0 | xargs -0 -n 1 ssconvert -T Gnumeric_html
    xargs -0 -i mv {} html/ ;
```

Действия с файлами в каталоге (bash)

Найти все файлы ТОЛЬКО в данном каталоге (без учёта вложенных каталогов), и выполнить действие.

```
    find * -maxdepth 0 -type f -print0 | xargs -0 -n 1 -i mv {} foo/
    # в качестве примера: перенесёт все файлы в каталог foo
```

Узнать сколько строк в каждом файле и записать результат в файл (bash)

Пригодится для написания seo-скриптов (работы с семантическим ядром и т. п.)

```
find * -maxdepth 0 -type f -print0 | xargs -0 wc -l >> how-many-words.txt
```

Вынести конфигурацию за пределы скрипта (bash)

Можно вынести переменные конфигурации в отдельный файл и затем подгрузить их в скрипт.

```
    Вставить в скрипт:
    . ./имя_файла

    Пример:
    . ./yandex.config
```

Консольные файловые менеджеры

```
mc
ranger
```

Повторить команду N раз (bash)

```
for i in {1..30} ; do echo "Hi!" ; done
```

Добавить символы в начале и в конце каждой строки

```
# добавить символы в начале каждой строки используя AWK
awk '{print "текст"$0}' file
# добавить символы в начале каждой строки используя SED
sed 's/^/текст/' file

# добавить символы в конце каждой строки используя AWK
awk '{print $0"текст"}' file
# добавить символы в конце каждой строки используя SED
sed 's/$/текст/' file

# добавить символы в начале и в конце каждой строки используя AWK
awk '{print "в_начале"$0"в_конце"}' file
# добавить символы в начале и в конце каждой строки используя SED
sed "s/.*/в_начале&в_конце/" file
```

sed: убрать пустые строки и строки комментариев

```
sed -e '/^$/d' -e '/^#/d' ttt.txt
```

Найти текст в файлах во всех вложенных каталогах

```
grep -r "искомый текст" ~/work
```

sort | uniq

```
sort | uniq
```

uniq убирает дубли только если они идут подряд, а не в разброс. Чтобы этого добиться, нужно использовать sort.

Получить значение: количество строк

```
wc -l
```

Краткое руководство по less (пейджер)

s

```
less file.txt # программа пейджер (просмотр длинных файлов)
```

q - выход f - пролистать на страницу вперёд b - пролистать на страницу назад g - вверх документа G - вниз документа j - вниз на строку k - вверх на строку h - вызвать справку

/искомоеслово - поиск слова n - далее по поиску shift + n - предыдущий результат

Как указать xargs куда подставлять данные

Команда xargs — по умолчанию вставляет в качестве аргумента в конец передаваемой команде.

Если при конструировании второй команды надо явно указать место, куда должны попасть выходные данные первой, достаточно воспользоваться парой фигурных скобок, {} и параметром -i для замены аргумента в нужном месте.

```
<команда 1> | xargs -i <команда 2> {} <команда 2-2>
```

Создать файл во всех вложенных каталогах (bash)

Имея вложенные каталоги dir01, dir02, ... dir500, надо создать в них файл zz.txt

```
for i in dir* ; do touch $i/zz.txt ; done
```

Генератор пароля (bash)

```
tr -dc A-Za-z0-9_ < /dev/urandom | head -c 30 | xargs
```

Или используйте pwgen

сложный пароль:

```
    pwgen -1sBnc 40 1

    получится примерно так: 4mdM3dAzNNhqhHxxvpeTVpkioWMNFTxERTvRLLHj
```

или ещё сложнее пароль:

```
    pwgen -1sBny 40 1

    получится примерно так: ~p`T&n[?V`f\HrK#-,g.[=$.|H)?'po9+[-]wy,v
```

установить pwgen:

```
sudo apt-get install pwgen
```

sed

```
8d - удалить 8-ую строку
/^$/d - удалить все пустые строки
1,/^$/d - удалить все строки до первой пустой строки, включительно
/Foo/d - вывести строки, содержащие 'Foo' (с ключом -n)
s/Foo/Bar/ - в каждой стоке заменить первое встретившееся слово Foo на слово Bar
s/Foo/Bar/g - в каждой стоке заменить все встретившиеся слова Foo на Bar
s/ *$// - удалить все пробелы в конце каждой строки
s/00*/0/g - заменить все последовательности нулей одним символом 0
/Foo/d - удалить все строки со словом Foo
s/Foo//g - удалить все найденные Foo, оставляя остальную чать строки без изменени
```

Знакомство с текстовыми утилитами UNIX

```
Использование стандартных утилит операционной системы для работы с текстом

Оставлю тут ссылку на крайне полезную статью: https://www.ibm.com/developerworks/
```

Нумерация строк через точку с запятой (для формирования .csv файла)

Если нам нужен номер строки для дальнейшего использования этих данных в csv файле, то поможет команда nl

```
nl -s';'
```

man (по-русски)

```
Мануалы команд GNU/Linux переведённые на русский язык.
Тут: http://www.opennet.ru/man_1.shtml
```

man wget (по-русски)

```
Тут: http://www.opennet.ru/man.shtml?topic=wget
```

Конвертация аудиофайла m4a в mp3

```
ffmpeg -i input.m4a -acodec libmp3lame -ab 128k output.mp3
```

Скобки {} и []

Создать 30 файлов типа file01.txt — используется конструкция {01..30}

```
touch file{01..30}.txt
```

Найти с 04 по 08 файлы — используется конструкция [4-8]

```
find file0[4-8].txt
```

grep (поиск) по нескольким словам

```
grep -e "word1" -e "word2"
```

Работа с файлами в нескольких каталогах

Скопировать файл zz.txt во все вложенные каталоги:

```
find * -type d | xargs -n 1 cp zz.txt

или можно так :

for i in $(find * -type d) ; do cp zz.txt $i ; done
```

Удалить файл zz.txt во всех вложенных каталогах:

```
find * -name zz.txt | xargs rm
# ОСТОРОЖНО! Удаляет файл и в текущем каталоге тоже.
```

Узнать время работы скрипта

```
time script.sh
```

Вывести дату bash

```
date +%F
```

Заменить foo на bar во всех файлах каталога

```
find -type f -print0 | xargs -0 sed -i 's/foo/bar/g'
```

Краснодарский программист

Краснодарский программист
info@krdprog.ru

krdprog
krdprog

Рабочие рецепты и способы решения возникающих задач + выжимки из учебных материалов.