

---

# Команда ls

Usage: ls [OPTION]... [FILE]...

Опции:

|                                |  |
|--------------------------------|--|
| -a, --all                      | показывать все файлы, включая скрытые (начинающиеся с .)                 |
| -A, --almost-all               | показывать все файлы, кроме . и ..                                       |
| --author                       | показывать имя автора каждого файла                                      |
| -b, --escape                   | показывать управляющие символы в формате \xxx                            |
| -B, --ignore-backups           | не показывать файлы, оканчивающиеся на ~                                 |
| -c                             | сортировать по времени изменения и показывать ctime                      |
| -C                             | вывод в колонках   |
| --color[=WHEN]                 | вывод с цветовой подсветкой (опции: auto, always, never)                 |
| -d, --directory                | показывать каталоги как файлы, не отображая их содержимое                |
| -D, --dired                    | вывод в формате dired  |
| -f                             | не сортировать, включить -a и отключить -l/-s                            |
| -F, --classify                 | добавлять символы-указатели к типам файлов (* для исполняемых)           |
| -g                             | как -l, но без отображения владельца                                     |
| --group-directories-first      | сначала отображать каталоги  |
| -G, --no-group                 | не показывать группу владельца   |
| -h, --human-readable           | размер файлов в читаемом формате (например, 1K, 234M)                    |
| -H, --dereference-command-line | следовать символическим ссылкам на командной строке                      |
| -i, --inode                    | показывать номер inode для каждого файла                                 |
| -I, --ignore=PATTERN           | игнорировать файлы, соответствующие PATTERN                              |
| -k, --kibibytes                | размеры в 1024-байтных блоках  |
| -l                             | подробный список файлов  |
| -L, --dereference              | отображать информацию о файле, на который ссылается символическая ссылка |
| -m                             | вывод в строку, разделяя имена файлов запятыми                           |
| -n, --numeric-uid-gid          | показывать числовые UID и GID вместо имен                                |
| -N, --literal                  | не обрабатывать управляющие символы                                      |
| -o                             | как -l, но без группы владельца  |
| -p, --indicator-style=slash    | добавлять / к именам каталогов   |
| -q, --hide-control-chars       | скрывать управляющие символы (показывать ? вместо них)                   |
| -Q, --quote-name               | заключать имена файлов в кавычки   |
| -r, --reverse                  | вывод в обратном порядке   |
| -R, --recursive                | рекурсивный вывод содержимого каталогов                                  |
| -s, --size                     | показывать размер блока для каждого файла                                |
| -S                             | сортировать по размеру (от большего к меньшему)                          |
| --sort=WORD                    | задает способ сортировки (WORD: none, size, time, version, extension)    |
| --time=WORD                    | задает тип времени (WORD: atime, access, use, ctime, status)             |
| --time-style=STYLE             | задает формат времени (STYLE: full-iso, long-iso, iso, locale)           |
| -t                             | сортировать по времени модификации                                       |
| -T, --tabsize=COLS             | задает ширину табуляции  |
| -u                             | сортировать и отображать время доступа                                   |
| -U                             | не сортировать; вывод в порядке каталогов                                |
| -v                             | сортировать по версии  |
| -w, --width=COLS               | задает ширину вывода   |
| -x                             | вывод в строках  |
| -X                             | сортировать по расширению  |

|               |                                   |
|---------------|-----------------------------------|
| -Z, --context | показывать контекст SELinux       |
| -1            | вывод в одну колонку              |
| --help        | показать эту справку и выйти      |
| --version     | показать версию программы и выйти |

---

## Komanda history

Usage: history [options] [n]

Опции:

|           |   |
|-----------|---|
| n         | отображает последние n строк истории команд                               |
| -c        | очищает текущий список истории  |
| -d offset | удаляет запись с заданным номером offset из истории                       |
| -a        | записывает команды текущей сессии в файл истории (обычно ~/.bash_history) |
| -n        | добавляет в текущую сессию команды из файла истории, добавленные другими  |
| -r        | считывает и добавляет команды из файла истории в текущую сессию           |
| -w        | записывает текущую историю команд в файл истории, перезаписывая его       |
| -p        | выводит команды из аргументов, не сохраняя их в истории                   |
| -s        | добавляет команды из аргументов в историю                                 |

Дополнительно:

|             |  |
|-------------|--|
| !n          | выполняет команду под номером n из истории           |
| !-n         | выполняет n-ю команду с конца                        |
| !!          | выполняет последнюю команду из истории               |
| !string     | выполняет последнюю команду, начинающуюся с "string" |
| !?string[?] | выполняет последнюю команду, содержащую "string"     |

---

## Komanda echo

Usage: echo [OPTION]... [STRING]...

Опции:

|    |   |
|----|---|
| -n | не добавлять символ новой строки в конце вывода                             |
| -e | интерпретировать управляющие последовательности (по умолчанию отключено)    |
| -E | явно отключить интерпретацию управляющих последовательностей (по умолчанию) |

Управляющие последовательности (при использовании -e):

|       |   |
|-------|---|
| \a    | звонок (звук)   |
| \b    | удаляет предыдущий символ (backspace)                 |
| \c    | прекращает дальнейший вывод                           |
| \e    | символ Escape   |
| \f    | разрыв страницы (form feed)                           |
| \n    | новая строка (line feed)                              |
| \r    | возврат каретки                                       |
| \t    | табуляция (horizontal tab)                            |
| \v    | вертикальная табуляция                                |
| \\    | обратный слэш   |
| \'    | одинарная кавычка                                     |
| \"    | двойная кавычка                                       |
| \0NNN | символ с восьмеричным значением NNN (от 000 до 377)   |
| \xHH  | символ с шестнадцатеричным значением HH (от 00 до FF) |

Примеры использования:

|   |  |
|---|--|
| <code>echo "Привет, мир!"</code>                    | <code># вывод текста "Привет, мир!"</code>       |
| <code>echo -n "Без новой строки"</code>             | <code># вывод без добавления новой строки</code> |
| <code>echo -e "Первая строка\nВторая строка"</code> | <code># вывод с новой строкой</code>             |
| <code>echo -e "Табуляция:\tПример"</code>           | <code># вывод с табуляцией</code>                |

---

## Komanda which

Команда `which` используется для определения местоположения исполняемого файла, который будет запущен при вводе команды в оболочке. Она ищет файлы в каталогах, указанных в переменной `PATH`.

Синтаксис:

```
which [опции] имя_команды
```

Опции:

- a Показать все экземпляры команды, найденные в `PATH`.
- s Подавить вывод; возвращает только статус выхода.

Примеры:

1. Найти путь к исполняемому файлу:

```
which ls
```

Вывод:

```
/bin/ls
```

2. Показать все версии команды в `PATH`:

```
which -a python
```

Вывод:

```
/usr/bin/python
```

```
/usr/local/bin/python
```

3. Проверить, существует ли команда (без вывода):

```
which -s gcc
```

```
echo $?
```

Вывод:

```
0 (если команда найдена) или 1 (если команда отсутствует)
```

Статус выхода:

```
0 - если команда найдена.
```

```
1 - если команда отсутствует.
```

---

## Komanda type

Команда ``type`` используется для определения типа команды в оболочке и может показать, является ли команда встроенной, алиасом, функцией или внешней программой.

Синтаксис:

```
type [опции] команда
```

Опции:

- t Показать только тип команды (например, "alias", "function", "file").
  - a Показать все возможные экземпляры команды в `PATH`, если она не встроенная.
  - p Показать только путь к исполняемому файлу команды.
  - f Применяется только к функциям: покажет имя файла функции.
- 

## Komanda man

Команда ``man`` (manual) используется для отображения справочной страницы по командам, функциям, конфигурационным файлам и другим элементам системы.

Синтаксис:

man [опции] команда

Основные опции:

- k Выполняет поиск по ключевым словам на всех доступных страницах.
- f Показывает краткую информацию о команде (аналогично whatis).
- a Отображает все доступные страницы поочерёдно.
- P Указывает пользовательский просмотрщик страниц (pager).
- M Указывает путь к каталогу с мануалами.

Примеры:

1. Просмотр справочной страницы команды:

```
man ls
```

2. Поиск по ключевым словам:

```
man -k copy
```

Вывод:

```
cp (1) - copy files and directories
```

```
memcp (3) - copy memory area
```

3. Просмотр раздела справки (например, системные вызовы):

```
man 2 open
```

Разделы:

- 1 - Исполняемые программы и команды
- 2 - Системные вызовы (syscalls)
- 3 - Библиотечные функции (stdlib)
- 4 - Специальные файлы (устройства)
- 5 - Форматы файлов
- 6 - Игры
- 7 - Разное
- 8 - Административные команды

4. Отображение всех страниц с совпадениями:

```
man -a intro
```

5. Использование пользовательского просмотрщика:

```
man -P cat ls
```

6. Узнать краткую информацию:

```
man -f ls
```

Вывод:

```
ls (1) - list directory contents
```

Навигация по `man`-страницам:

- Пробел Перейти на следующую страницу.
- B Перейти на предыдущую страницу.
- Q Выйти.

---

## Komanda whatis

Команда `whatis` используется для вывода краткого описания команды, функции или файла. Это быстрый способ узнать, что делает определённая команда.

Синтаксис:

```
whatis [опции] команда
```

Основные опции:

- r Выполняет поиск с использованием регулярного выражения.
- w Выполняет "широкий" поиск: частичное совпадение.
- l Показывает результаты в длинном формате.
- V Выводит версию команды.

Примеры:

1. Узнать краткое описание команды:

```
whatis ls
```

Вывод:

```
ls (1) - list directory contents
```

2. Использовать регулярное выражение:

```
whatis -r '^mk'
```

Вывод:

```
mkdir (1) - make directories
```

```
mkfifo (3) - make a FIFO special file (a named pipe)
```

3. Поиск с частичным совпадением:

```
whatis -w copy
```

Вывод:

```
cp (1) - copy files and directories
```

```
memcopy (3) - copy memory area
```

4. Показать информацию в длинном формате:

```
whatis -l man
```

Вывод:

```
man (1) - an interface to the system reference manuals
```

5. Если команда или тема отсутствует в базе данных:

```
whatis unknown
```

Вывод:

```
unknown: nothing appropriate.
```

Чтобы обновить базу данных, используйте команду:

```
sudo mandb
```

---

## Komanda whereis

Команда `whereis` используется для поиска расположения исполняемого файла, исходного кода и страницы справки (man) указанной команды.

Синтаксис:

```
whereis [опции] команда
```

Основные опции:

- b Искать только исполняемые файлы.
- m Искать только справочные страницы.
- s Искать только исходные файлы.
- B Задать каталоги для поиска исполняемых файлов.
- M Задать каталоги для поиска справочных страниц.
- S Задать каталоги для поиска исходных файлов.
- f Указывает конец списка опций, после которого идет название команды.

Примеры:

1. Найти все файлы, связанные с командой:

```
whereis ls
```

Вывод:

```
ls: /bin/ls /usr/share/man/man1/ls.1.gz
```

2. Найти только исполняемый файл:

```
whereis -b ls
```

Вывод:

```
ls: /bin/ls
```

3. Найти только справочную страницу:

```
whereis -m ls
Вывод:
ls: /usr/share/man/man1/ls.1.gz
```

4. Найти только исходные файлы:

```
whereis -s gcc
Вывод:
gcc:
```

5. Искать исполняемые файлы в определённом каталоге:

```
whereis -B /usr/bin -f ls
Вывод:
ls: /usr/bin/ls
```

6. Искать справочные страницы в указанном каталоге:

```
whereis -M /usr/share/man/man1 -f ls
Вывод:
ls: /usr/share/man/man1/ls.1.gz
```

Команда полезна для быстрого поиска всех связанных с программой файлов. Если результат пустой, возможно, нужные файлы отсутствуют в системе.

---

## Команда locate

Команда `locate` используется для быстрого поиска файлов в системе. Она работает на основе предварительно созданной базы данных, что делает её гораздо быстрее, чем `find`.

Синтаксис:

```
locate [опции] шаблон
```

Основные опции:

- i Игнорировать регистр символов.
- r Искать с использованием регулярных выражений.
- c Вывести только количество найденных результатов.
- l N Ограничить количество выводимых результатов до N.
- e Убедиться, что файлы существуют (может быть медленнее).
- 0 Выводить результаты, разделённые нулевым символом, а не новой строкой.

Примеры:

1. Найти все файлы, содержащие слово "example":

```
locate example
```

2. Игнорировать регистр при поиске:

```
locate -i Example
```

3. Использовать регулярные выражения:

```
locate -r '^/usr.*bin/ls$'
(Ищет файл `ls` в каталоге `/usr/bin`.)
```

4. Подсчитать количество найденных файлов:

```
locate -c example
Вывод:
15
```

5. Ограничить вывод до 5 результатов:

```
locate -l 5 example
```

6. Убедиться в существовании файлов:

```
locate -e oldfile
```

7. Поиск с нулевым символом в качестве разделителя:

```
locate -0 example | xargs -0 ls -l
```

Чтобы обновить базу данных для `locate`, используйте команду:

```
sudo updatedb
```

Замечание:

Если `locate` не находит файл, убедитесь, что база данных обновлена, запустив `updatedb`. Кроме того, файлы, к которым у пользователя нет доступа, не будут отображаться в результате поиска.

---

## Komanda info

Команда `info` используется для отображения подробной документации по командам и утилитам. В отличие от `man`, она предоставляет более детализированное описание с гиперссылками для навигации.

Синтаксис:

```
info [опции] команда
```

Основные опции:

- k Поиск по ключевым словам (аналогично `man -k`).
- subnodes Рекурсивно отображать все узлы (разделы) документа.
- vi-keys Использовать клавиши навигации в стиле редактора `vi`.

Навигация в `info`:

- Пробел Переход на следующую страницу.
- Backspace Переход на предыдущую страницу.
- n Перейти к следующему узлу (разделу).
- p Перейти к предыдущему узлу (разделу).
- u Перейти к верхнему уровню документации.
- m [название] Перейти к указанному узлу (разделу).
- q Выход из `info`.

Примеры:

1. Просмотр документации по команде:

```
info ls
```

2. Поиск по ключевым словам:

```
info -k copy
```

Вывод:

```
* Copying::                How to copy text between files.
* cp: (coreutils)cp invocation.
                           Copy files and directories.
```

3. Рекурсивный просмотр всех разделов:

```
info --subnodes bash
```

4. Использование клавиш в стиле `vi`:

```
info --vi-keys find
```

5. Перейти к определённому разделу документации:

```
info -m bash
```

Особенности:

- Документация в `info` часто содержит больше примеров и пояснений, чем в `man`.
- Если команда недоступна в `info`, может потребоваться установка пакета документации (например, `sudo apt install info` или `sudo apt install coreutils`).

doc`).

---

## Опција —help

Опция `--help` используется для вывода краткой справочной информации о команде, её использовании и доступных опциях.

Особенности:

- Показывает описание команды, список опций и их функции.
- Помогает быстро понять, как пользоваться командой.
- Работает практически с любой командой.

Синтаксис:

команда --help

Примеры использования:

1. Узнать о возможностях команды `ls`:  
ls --help
2. Получить справку по команде `grep`:  
grep --help
3. Узнать доступные опции команды `find`:  
find --help

Пример вывода для команды `ls`:

Usage: ls [OPTION]... [FILE]... List information about the FILES (the current directory by default).

Options: -a, --all do not ignore entries starting with . -l use a long listing format -r, --reverse reverse order while sorting --help display this help and exit

Особенности вывода:

- Обычно включает пример использования команды.
- Показывает основные и дополнительные параметры.
- Часто доступен и через сокращённый флаг `-h`.

Используйте `--help`, чтобы быстро разобраться с командами и их опциями!

---

## Команда pwd

Команда `pwd` (print working directory) используется для отображения полного пути текущего рабочего каталога.

Синтаксис:

pwd [опции]

Основные опции:

- L Показывает путь с учётом символических ссылок (логический путь, по умолчанию).
- P Показывает физический путь, игнорируя символические ссылки.

Примеры:

1. Узнать текущий рабочий каталог:  
pwd  
Вывод:  
/home/username/projects



2. Отобразить физический путь, игнорируя символические ссылки:

```
pwd -P
```

Вывод:

```
/mnt/data/projects
```

3. Отобразить путь с учётом символических ссылок:

```
pwd -L
```

Вывод:

```
/home/username/links/projects
```

Полезные замечания:

- Команда ``pwd`` часто используется для проверки текущего расположения в файловой системе.

- Различие между ``-L`` и ``-P`` становится заметным, если текущий каталог содержит символические ссылки.

---

## Komanda cd

Команда ``cd`` не имеет большого количества опций, поскольку она предназначена в основном для перемещения между каталогами. Однако вот доступные функциональные особенности:

Опции и возможности:

- ``cd`` без параметров: Переход в домашний каталог пользователя (обычно ``/home/username``).

- ``cd -``: Перемещение в предыдущий рабочий каталог.

- ``cd ..``: Перемещение в родительский каталог.

- ``cd ~``: Переход в домашний каталог (эквивалент ``cd``).

- ``cd ~username``: Переход в домашний каталог указанного пользователя.

- ``cd /path/to/dir``: Переход в указанный абсолютный путь.

- ``cd relative/path``: Переход в указанный относительный путь.

Важно:

- ``cd`` является встроенной командой оболочки (``builtin``) и зависит от текущей оболочки (например, Bash, Zsh).

---

## Komanda cp

Команда ``cp`` используется для копирования файлов и директорий в Unix/Linux системах.

Синтаксис:

```
cp [опции] источник назначение
```

Основные опции:

- a, --archive

Копирует файлы и директории рекурсивно, сохраняя структуру, атрибуты и временные метки.

- f, --force

Перезаписывает файлы без подтверждения.

- i, --interactive

Запрашивает подтверждение перед перезаписью.

- r, --recursive

Копирует каталоги и их содержимое рекурсивно.

- u, --update

Копирует только если исходный файл новее или отсутствует в назначении.

- v, --verbose

Показывает, какие файлы копируются.

- preserve[=ATTR\_LIST]

Сохраняет указанные атрибуты (например, временные метки, права доступа).

- parents

Сохраняет путь к файлам в назначении.

- p, --no-dereference

Не разыменовывает символические ссылки.

Примеры использования:

1. Копирование одного файла:  
`cp file.txt /destination/path/`
2. Копирование нескольких файлов в каталог:  
`cp file1.txt file2.txt /destination/path/`
3. Копирование каталога рекурсивно:  
`cp -r /source/directory /destination/`
4. Копирование с запросом подтверждения перед перезаписью:  
`cp -i file.txt /destination/path/`
5. Копирование с отображением всех действий:  
`cp -v file.txt /destination/path/`  
Вывод:  
``file.txt -> /destination/path/file.txt``
6. Сохранение атрибутов файла:  
`cp -a file.txt /destination/path/`
7. Копирование с условием обновления:  
`cp -u file.txt /destination/path/`  
(Копируется только если файл в источнике новее.)
8. Копирование с сохранением пути:  
`cp --parents /home/user/docs/file.txt /backup/`  
(Создаст ``/backup/home/user/docs/file.txt``.)

Особенности:

- Если указанный каталог назначения отсутствует, команда ``cp`` выдаст ошибку.
- Для больших данных удобнее использовать ``rsync``, так как он более функционален.

Для дополнительной информации используйте ``man cp`` или ``cp --help``.

---

## Команда mv

Команда ``mv`` используется для перемещения или переименования файлов и каталогов в Unix/Linux системах.

Синтаксис:

`mv [опции] источник назначение`

Основные опции:

- |                                       |   |
|---------------------------------------|---|
| <code>-f, --force</code>              | Выполняет перемещение без подтверждения, если файл назначения уже существует. |
| <code>-i, --interactive</code>        | Запрашивает подтверждение перед перезаписью файла назначения.                 |
| <code>-n, --no-clobber</code>         | Не перезаписывает существующие файлы.   |
| <code>-u, --update</code>             | Перемещает только если источник новее или файл назначения отсутствует.        |
| <code>-v, --verbose</code>            | Показывает действия команды (например, какой файл куда перемещается).         |
| <code>--backup[=контекст]</code>      | Создаёт резервные копии существующих файлов.                                  |
| <code>-b</code>                       | Включает создание резервных копий (аналогично <code>`--backup`</code> ).      |
| <code>--suffix=суффикс</code>         | Указывает суффикс для резервных копий (по умолчанию <code>`~`</code> ).       |
| <code>--strip-trailing-slashes</code> | Удаляет завершающие косые черты из имени источника.                           |

Примеры использования:

1. Переименование файла:  
`mv oldname.txt newname.txt`
2. Перемещение файла в другой каталог:  
`mv file.txt /destination/path/`
3. Перемещение нескольких файлов в каталог:  
`mv file1.txt file2.txt /destination/path/`
4. Перемещение каталога:  
`mv /source/directory /destination/`
5. Запрос подтверждения перед перезаписью:  
`mv -i file.txt /destination/`
6. Перемещение с отображением действий:  
`mv -v file.txt /destination/`  
Вывод:  
``file.txt -> /destination/file.txt``
7. Перемещение только новых или изменённых файлов:  
`mv -u file.txt /destination/`
8. Создание резервных копий перезаписываемых файлов:  
`mv -b file.txt /destination/`  
(Создаст резервную копию файла назначения с суффиксом ``~``.)

Особенности:

- Если файл назначения существует, команда ``mv`` по умолчанию его перезапишет без предупреждения (если не используется ``-i`` или ``-n``).
- Для перемещения файлов между файловыми системами ``mv`` копирует файл, а затем удаляет исходный.

Для дополнительной информации используйте ``man mv`` или ``mv --help``.

---

## Komanda touch

Команда ``touch`` используется для создания новых пустых файлов или для изменения временных меток (даты и времени последнего доступа и изменения) существующих файлов.

Синтаксис:

`touch [опции] файл...`

Основные опции:

- |                                       |   |
|---------------------------------------|---|
| <code>-a, --time=atime</code>         | Изменяет только время последнего доступа.       |
| <code>-m, --time=mtime</code>         | Изменяет только время последнего изменения.     |
| <code>-c, --no-create</code>          | Не создаёт файл, если он не существует.         |
| <code>-d, --date=STRING</code>        | Устанавливает указанную дату и время.           |
| <code>-r, --reference=FILE</code>     | Использует временные метки другого файла.       |
| <code>-t [[CC]YY]MMDDhhmm[.ss]</code> | Устанавливает дату и время в указанном формате. |
| <code>--no-dereference</code>         | Не разыменовывает символические ссылки.         |

Примеры использования:

1. Создание нового файла:  
`touch newfile.txt`
2. Обновление временных меток существующего файла:  
`touch existingfile.txt`
3. Создание нескольких файлов одновременно:

```
touch file1.txt file2.txt file3.txt
```

4. Изменение только времени последнего доступа:  
`touch -a file.txt`

5. Изменение времени последнего изменения:  
`touch -m file.txt`

6. Указание конкретной даты и времени:  
`touch -d "2024-12-25 15:30:00" file.txt`

7. Использование временных меток другого файла:  
`touch -r reference.txt target.txt`

8. Указание временных меток в формате:  
`touch -t 202412251530 file.txt`  
(Дата: 25 декабря 2024 года, время: 15:30)

9. Не создавать файл, если он не существует:  
`touch -c file.txt`

Особенности:

- Если файл существует, команда обновляет временные метки.
- Если файл отсутствует, он будет создан (если не указана опция `-c`).

Для подробной информации используйте `man touch` или `touch --help`.

---

## Команда rm

Команда `rm` используется для удаления файлов и каталогов в Unix/Linux системах.

Синтаксис:

```
rm [опции] файл...
```

Основные опции:

|                                  |  |
|----------------------------------|--|
| <code>-f, --force</code>         | Удаляет файлы без подтверждения, даже если они защищены от записи.                         |
| <code>-i, --interactive</code>   | Запрашивает подтверждение перед удалением каждого файла.                                   |
| <code>-I</code>                  | Запрашивает подтверждение перед удалением более трёх файлов или рекурсивным удалением.     |
| <code>-r, -R, --recursive</code> | Удаляет каталоги и их содержимое рекурсивно.   |
| <code>-d, --dir</code>           | Удаляет пустые каталоги.   |
| <code>-v, --verbose</code>       | Показывает, какие файлы или каталоги удаляются.  |
| <code>--preserve-root</code>     | Защищает корневую файловую систему ( <code>/</code> ) от удаления (включено по умолчанию). |
| <code>--no-preserve-root</code>  | Отключает защиту корневой файловой системы.  |

Примеры использования:

1. Удаление одного файла:  
`rm file.txt`

2. Удаление нескольких файлов:  
`rm file1.txt file2.txt file3.txt`

3. Удаление пустого каталога:  
`rm -d empty_directory/`

4. Удаление каталога и его содержимого рекурсивно:  
`rm -r directory/`

5. Удаление файлов без подтверждения:  
`rm -f file.txt`
6. Запрос подтверждения перед удалением каждого файла:  
`rm -i file.txt`  
Вывод:  
``rm: удалить обычный пустой файл 'file.txt'? y``
7. Показать удаляемые файлы:  
`rm -v file.txt`  
Вывод:  
``удалён 'file.txt'``
8. Удаление всех файлов в каталоге (включая скрытые файлы):  
`rm -r /path/to/directory/*`
9. Удаление без защиты корневой файловой системы (осторожно!):  
`rm -rf --no-preserve-root /`

Особенности:

- Будьте осторожны с ``rm -r`` и ``rm -rf``, так как они могут удалить большие объёмы данных без возможности восстановления.
- Для удаления файлов с подтверждением лучше использовать опцию ``-i``.
- Команда ``rm`` **\*\*не перемещает файлы в корзину\*\***; удалённые файлы нельзя восстановить стандартными средствами.

Для дополнительной информации используйте ``man rm`` или ``rm --help``.

---

## Komandas mkdir un rmdir

Команда ``mkdir`` используется для создания новых каталогов, а ``rmdir`` — для удаления пустых каталогов.

---

### mkdir

Синтаксис:

`mkdir [опции] каталог...`

Основные опции:

- |                              |   |
|------------------------------|---|
| <code>-m, --mode=MODE</code> | Устанавливает права доступа к создаваемым каталогам (например, <code>`mkdir -m 755 dir`</code> ). |
| <code>-p, --parents</code>   | Создаёт указанный каталог и его родительские каталоги, если их ещё нет.                           |
| <code>-v, --verbose</code>   | Показывает сообщение о создании каждого каталога.   |

Примеры использования ``mkdir``:

1. Создание одного каталога:  
`mkdir new_directory`
2. Создание нескольких каталогов одновременно:  
`mkdir dir1 dir2 dir3`
3. Создание каталога с указанными правами доступа:  
`mkdir -m 700 private_directory`
4. Создание вложенного каталога и его родителей:  
`mkdir -p /path/to/new_directory`
5. Показать процесс создания каталогов:

```
mkdir -v dir1 dir2
Вывод:
`создан каталог 'dir1'`
`создан каталог 'dir2'`
```

---

### rmdir

Синтаксис:  
rmdir [опции] каталог...

Основные опции:

|               |   |
|---------------|---|
| -p, --parents | Удаляет каталог и его пустые родительские каталоги. |
| -v, --verbose | Показывает сообщение о каждом удалённом каталоге.   |

Примеры использования `rmdir`:

1. Удаление пустого каталога:  
rmdir empty\_directory
2. Удаление нескольких пустых каталогов:  
rmdir dir1 dir2
3. Удаление вложенного каталога и его пустых родительских директорий:  
rmdir -p /path/to/empty\_directory
4. Показать процесс удаления:  
rmdir -v empty\_directory  
Вывод:  
`удалён каталог 'empty\_directory'`

---

### Важные замечания:

- `mkdir` создаёт только каталоги. Для создания файлов используйте команду `touch`.
- `rmdir` удаляет **только** пустые каталоги. Для удаления каталогов с содержимым используйте `rm -r`.
- При использовании опции `-p` в `rmdir` родительские каталоги удаляются только если они тоже пусты.

Для дополнительной информации используйте `man mkdir`, `man rmdir` или `mkdir --help`, `rmdir --help`.

---

## Arhivēšana

**gunzip/gzip**

**bunzip2/bzip2**

**xz/unxz**

**tar**

**zip/unzip**

### gzip и gunzip

#### gzip

Команда `gzip` сжимает файлы, заменяя оригиналы на сжатую версию с расширением `.gz`.

**\*\*Синтаксис\*\*:**  
gzip [опции] файл...

**\*\*Основные опции\*\*:**

|                  |  |
|------------------|--|
| -d, --decompress | Распаковывает файл (аналог `gunzip`).              |
| -k, --keep       | Оставляет оригинальный файл.                       |
| -r, --recursive  | Сжимает файлы в каталогах рекурсивно.              |
| -1 до -9         | Указывает уровень сжатия (1 - быстрее, 9 - лучше). |
| -v, --verbose    | Показывает информацию о процессе сжатия.           |

**\*\*Пример\*\*:**

- Сжать файл: `gzip file.txt` (результат: `file.txt.gz`)
- Сжать с сохранением оригинала: `gzip -k file.txt`

#### gunzip  
Команда `gunzip` распаковывает `.gz` файлы.

**\*\*Синтаксис\*\*:**  
gunzip [опции] файл...

**\*\*Пример\*\*:**

- Распаковать файл: `gunzip file.txt.gz` (результат: `file.txt`)
- Распаковать файл без удаления `.gz`: `gunzip -k file.txt.gz`

---

### bzip2 и bunzip2

#### bzip2  
Команда `bzip2` сжимает файлы, создавая файлы с расширением `.bz2`.

**\*\*Синтаксис\*\*:**  
bzip2 [опции] файл...

**\*\*Основные опции\*\*:**

|                  |  |
|------------------|--|
| -d, --decompress | Распаковывает файл (аналог `bunzip2`). |
| -k, --keep       | Оставляет оригинальный файл.           |
| -v, --verbose    | Показывает процесс сжатия.             |
| -z               | Сжимает файл (по умолчанию).           |
| -1 до -9         | Указывает уровень сжатия.              |

**\*\*Пример\*\*:**

- Сжать файл: `bzip2 file.txt` (результат: `file.txt.bz2`)
- Сжать с сохранением оригинала: `bzip2 -k file.txt`

#### bunzip2  
Команда `bunzip2` распаковывает `.bz2` файлы.

**\*\*Синтаксис\*\*:**  
bunzip2 [опции] файл...

**\*\*Пример\*\*:**

- Распаковать файл: `bunzip2 file.txt.bz2` (результат: `file.txt`)

---

### xz и unxz

#### xz  
Команда `xz` сжимает файлы, создавая файлы с расширением `.xz`.

**\*\*Синтаксис\*\*:**  
xz [опции] файл...

**\*\*Основные опции\*\*:**

|                  |                                     |
|------------------|-------------------------------------|
| -d, --decompress | Распаковывает файл (аналог `unxz`). |
| -k, --keep       | Оставляет оригинальный файл.        |
| -T [N]           | Указывает количество потоков.       |
| -l до -9         | Указывает уровень сжатия.           |
| -v, --verbose    | Показывает процесс сжатия.          |

**\*\*Пример\*\*:**

- Сжать файл: `xz file.txt` (результат: `file.txt.xz`)
- Сжать с сохранением оригинала: `xz -k file.txt`

#### unxz  
Команда `unxz` распаковывает `.xz` файлы.

**\*\*Синтаксис\*\*:**  
unxz [опции] файл...

**\*\*Пример\*\*:**  
- Распаковать файл: `unxz file.txt.xz` (результат: `file.txt`)

---

### tar  
Команда `tar` используется для создания архивов и их распаковки.

**\*\*Синтаксис\*\*:**  
tar [опции] файл...

**\*\*Основные опции\*\*:**

|                 |  |
|-----------------|--|
| -c, --create    | Создаёт новый архив.                     |
| -x, --extract   | Извлекает файлы из архива.               |
| -f, --file=FILE | Указывает имя архива.                    |
| -v, --verbose   | Показывает список обрабатываемых файлов. |
| -z, --gzip      | Использует сжатие gzip.                  |
| -j, --bzip2     | Использует сжатие bzip2.                 |
| -J, --xz        | Использует сжатие xz.                    |
| -t, --list      | Показывает содержимое архива.            |

**\*\*Пример\*\*:**

- Создать архив: `tar -cvf archive.tar file1 file2`
- Извлечь архив: `tar -xvf archive.tar`
- Архив с сжатием: `tar -cvzf archive.tar.gz file1 file2`
- Извлечь сжатый архив: `tar -xvzf archive.tar.gz`

---

### zip и unzip

#### zip  
Команда `zip` создаёт сжатые архивы `.zip`.

**\*\*Синтаксис\*\*:**  
zip [опции] файл.zip файл...

**\*\*Основные опции\*\*:**

|         |                                  |
|---------|----------------------------------|
| -r      | Добавляет файлы рекурсивно.      |
| -d FILE | Удаляет файл из архива.          |
| -u      | Обновляет файлы в архиве.        |
| -v      | Показывает подробную информацию. |



```
**Пример**:
- Создать архив: `zip archive.zip file1 file2`
- Создать архив рекурсивно: `zip -r archive.zip directory`

#### unzip
Команда `unzip` распаковывает `.zip` архивы.

**Синтаксис**:
unzip [опции] файл.zip

**Основные опции**:
-l          Показывает содержимое архива.
-v          Показывает подробную информацию.
-d DIR      Указывает каталог для извлечения.

**Пример**:
- Распаковать архив: `unzip archive.zip`
- Извлечь архив в папку: `unzip archive.zip -d /destination/path`
```

---

## Komanda grep

```
### grep
```

Команда `grep` используется для поиска строк, соответствующих указанному шаблону, в файлах или потоках данных.

```
**Синтаксис**:
grep [опции] шаблон [файл...]
```

```
---
```

**\*\*Основные опции\*\*:**

- \*\*Опции поиска\*\*:**
  - `-i` Игнорирует регистр (поиск нечувствителен к регистру).
  - `-v` Выводит строки, которые **\*\*не\*\*** соответствуют шаблону.
  - `-w` Ищет точные слова (не части слов).
  - `-x` Ищет строки, полностью совпадающие с шаблоном.
  - `-E` Использует расширенные регулярные выражения (аналог `egrep`).
  - `-F` Ищет фиксированные строки (аналог `fgrep`).
- \*\*Опции отображения\*\*:**
  - `-n` Показывает номер строки, в которой найдено совпадение.
  - `-c` Показывает только количество совпадений.
  - `-l` Показывает только имена файлов, содержащих совпадения.
  - `-L` Показывает только имена файлов без совпадений.
  - `-o` Показывает только совпадающие части строки.
  - `-H` Показывает имя файла перед каждой совпавшей строкой (по умолчанию для нескольких файлов).
  - `-h` Не показывает имя файла (по умолчанию для одного файла).
- \*\*Опции контекста\*\*:**
  - `-A NUM` Показывает `NUM` строк **\*\*после\*\*** найденной строки.
  - `-B NUM` Показывает `NUM` строк **\*\*до\*\*** найденной строки.
  - `-C NUM` Показывает `NUM` строк **\*\*до и после\*\*** найденной строки.
- \*\*Прочее\*\*:**
  - `-r` или `-R` Выполняет рекурсивный поиск в каталогах.
  - `--include=GLOB` Ищет только в файлах, соответствующих шаблону GLOB.
  - `--exclude=GLOB` Исключает файлы, соответствующие шаблону GLOB.
  - `--color[=WHEN]` Подсвечивает совпадения (WHERE: auto, always, never).

---

**\*\*Примеры использования\*\*:**

1. Поиск строки в файле:  
```bash  
grep "hello" file.txt

---

## Komanda cat

### `cat`

Команда `cat` (сокращение от "concatenate") используется для просмотра содержимого файлов, объединения файлов и передачи их содержимого в стандартный вывод.

---

**\*\*Синтаксис\*\*:**

`cat [опции] [файл...]

---

### **\*\*Основные опции\*\*:**

1. **\*\*Отображение содержимого файла\*\*:**

- `-n` Нумерует строки в выводе.
- `-b` Нумерует только непустые строки.
- `-s` Убирает лишние пустые строки (сводит их к одной).
- `-T` Показывает символы табуляции как `^I`.
- `-E` Показывает символ конца строки как `\$`.

2. **\*\*Объединение файлов\*\*:**

- Команда по умолчанию объединяет содержимое нескольких файлов в один поток вывода.

---

### **\*\*Примеры использования\*\*:**

- **\*\*Показ содержимого файла\*\*:**

Команда `cat filename.txt` считывает и выводит содержимое файла `filename.txt`.

- **\*\*Объединение двух файлов и вывод результата\*\*:**

Используя команду `cat file1.txt file2.txt`, можно объединить содержимое файлов `file1.txt` и `file2.txt` и вывести его в терминал.

- **\*\*Запись содержимого объединённых файлов в новый файл\*\*:**

Команда `cat file1.txt file2.txt > mergedfile.txt` считывает содержимое `file1.txt` и `file2.txt`, объединяет его и сохраняет в файл `mergedfile.txt`.

- **\*\*Нумерация строк в файле\*\*:**

Если нужно пронумеровать все строки файла, используется команда `cat -n filename.txt`.

- **\*\*Нумерация только непустых строк\*\*:**

Чтобы нумеровать только непустые строки, применяют команду `cat -b filename.txt`.

- **\*\*Удаление лишних пустых строк\*\*:**

Команда ``cat -s filename.txt`` удаляет избыточные пустые строки, оставляя только одну пустую строку.

- **\*\*Показ символов конца строк\*\***:

Для отображения символов конца строк используется команда ``cat -E filename.txt``, что позволяет увидеть символ ``$`` в конце каждой строки.

- **\*\*Показ символов табуляции\*\***:

Команда ``cat -T filename.txt`` позволяет увидеть символы табуляции, которые будут отображаться как ``^I``.

- **\*\*Показ непечатаемых символов\*\***:

Чтобы показать непечатаемые символы, кроме табуляции и конца строки, используют команду ``cat -v filename.txt``.

---

## Komanda less

# Команда ``less``

Команда ``less`` в Linux предназначена для постраничного просмотра содержимого файлов или вывода других команд. Она позволяет просматривать текстовые файлы, перемещаться по ним, а также выполнять поиск, не загружая весь файл в память, что особенно полезно для больших файлов.

## Основные возможности

- Постраничный просмотр текста.
- Перемещение вперед и назад по файлу.
- Поиск текста внутри файла.
- Просмотр данных из других команд через конвейер (pipe).

## Синтаксис

```
```bash
less [опции] [файл]
```
```

## Основные опции

- ``-N`` — отображает номера строк.
- ``-S`` — отключает перенос строк (показывает длинные строки горизонтально).
- ``-X`` — отключает очистку экрана после завершения работы.
- ``+<строка>`` — сразу выполняет поиск строки в файле.
- ``+F`` — переходит в режим "следования" (аналогично ``tail -f``).
- ``-i`` — игнорирует регистр букв при поиске.

## Навигация внутри ``less``

- **\*\*Пробел\*\*** или **\*\*стрелка вниз\*\*** — переход на следующую страницу.
- **\*\*b\*\*** или **\*\*стрелка вверх\*\*** — переход на предыдущую страницу.
- **\*\*g\*\*** — перейти в начало файла.
- **\*\*G\*\*** — перейти в конец файла.
- **\*\*/текст\*\*** — поиск текста вперед.
- **\*\*?текст\*\*** — поиск текста назад.
- **\*\*n\*\*** — повтор предыдущего поиска вперед.
- **\*\*N\*\*** — повтор предыдущего поиска назад.
- **\*\*q\*\*** — выйти из программы.

## Примеры использования

1. **\*\*Простой просмотр файла:\*\***

```
```bash
less example.txt
```
```

2. **\*\*Просмотр файла с отображением номеров строк:\*\***

```
```bash
less -N example.txt
```
```

3. **\*\*Отключение переноса строк:\*\***

```
```bash
less -S example.txt
```
```

4. **\*\*Поиск строки при открытии файла:\*\***

```
```bash
less +/поиск example.txt
```
```

5. **\*\*Просмотр вывода другой команды:\*\***

```
```bash
dmesg | less
```
```

6. **\*\*Следование за обновлением файла (журналов):\*\***

```
```bash
less +F /var/log/syslog
```
```

7. **\*\*Игнорирование регистра при поиске:\*\***

```
```bash
less -i example.txt
```
```

## Полезные сочетания

- **\*\*Ctrl+F\*\*** — прокрутка вперед на один экран.
- **\*\*Ctrl+B\*\*** — прокрутка назад на один экран.
- **\*\*Ctrl+D\*\*** — прокрутка вперед на пол-экрана.
- **\*\*Ctrl+U\*\*** — прокрутка назад на пол-экрана.

`less` используется для быстрого и удобного анализа текстовых файлов, журналов, вывода команд и других данных в терминале. Это мощный инструмент для работы с большими объемами информации.

---

## Команды head, tail

# Команда `head`

### `head`

Команда `head` используется для отображения первых нескольких строк из файла или потока данных. По умолчанию выводится первые 10 строк.

---

**\*\*Синтаксис\*\*:**

```
`head` [опции] [файл...]
```

---

### **\*\*Основные опции\*\*:**

1. **\*\*Указание числа строк\*\*:**

- **`-n` [число]** Показывает заданное количество строк (по умолчанию 10).

2. **\*\*Отображение байтов\*\*:**

- **`-с` [число]** Показывает первые заданное количество байтов.

### 3. **\*\*Использование стандартного ввода\*\*:**

- Если файл не указан, команда читает данные из стандартного ввода (например, с помощью пайпа).

---

#### ### **\*\*Примеры использования\*\*:**

##### - **\*\*Показ первых 10 строк файла\*\*:**

Команда ``head filename.txt`` выводит первые 10 строк файла ``filename.txt``.

##### - **\*\*Показ первых 20 строк файла\*\*:**

Команда ``head -n 20 filename.txt`` выводит первые 20 строк файла ``filename.txt``.

##### - **\*\*Показ первых 50 байтов файла\*\*:**

Команда ``head -c 50 filename.txt`` выводит первые 50 байтов из файла ``filename.txt``.

##### - **\*\*Использование с пайпом\*\*:**

Команда ``cat filename.txt | head -n 5`` выведет первые 5 строк файла ``filename.txt``, используя стандартный ввод через пайп.

---

# Команда ``tail``

#### ### ``tail``

Команда ``tail`` используется для отображения последних нескольких строк из файла или потока данных. По умолчанию выводятся последние 10 строк.

---

#### **\*\*Синтаксис\*\*:**

``tail [опции] [файл...]'``

---

#### ### **\*\*Основные опции\*\*:**

##### 1. **\*\*Указание числа строк\*\*:**

- ``-n [число]`` Показывает последние заданное количество строк (по умолчанию 10).

##### 2. **\*\*Отображение байтов\*\*:**

- ``-c [число]`` Показывает последние заданное количество байтов.

##### 3. **\*\*Следить за изменениями файла\*\*:**

- ``-f`` Позволяет следить за добавлением новых строк в файл в реальном времени (например, для логов).

##### 4. **\*\*Постоянное обновление вывода\*\*:**

- ``-F`` Тот же, что и ``-f``, но также будет повторно открывать файл, если он был удалён или перезаписан.

---

#### ### **\*\*Примеры использования\*\*:**

##### - **\*\*Показ последних 10 строк файла\*\*:**

Команда ``tail filename.txt`` выводит последние 10 строк файла ``filename.txt``.

- **\*\*Показ последних 20 строк файла\*\*:**  
Команда ``tail -n 20 filename.txt`` выводит последние 20 строк файла ``filename.txt``.
  - **\*\*Показ последних 50 байтов файла\*\*:**  
Команда ``tail -c 50 filename.txt`` выводит последние 50 байтов из файла ``filename.txt``.
  - **\*\*Следить за изменениями файла\*\*:**  
Команда ``tail -f filename.log`` будет выводить новые строки, добавляемые в файл ``filename.log`` в реальном времени.
  - **\*\*Показ последних строк с выводом изменений\*\*:**  
Команда ``tail -F filename.log`` будет выводить последние строки из файла ``filename.log`` и отслеживать его изменения, даже если файл будет перезаписан.
- 

## Komanda tr

### `tr`

Команда ``tr`` используется для преобразования или удаления символов в потоке данных. Она часто применяется для замены символов или удаления нежелательных символов в текстовых файлах или строках.

---

**\*\*Синтаксис\*\*:**

``tr [опции] [символы_для_замены] [символы_замены]``

---

### **\*\*Основные опции\*\*:**

1. **\*\*Удаление символов\*\*:**
  - ``-d`` — Удаляет все символы, указанные в первом наборе.
2. **\*\*Сжатие символов\*\*:**
  - ``-s`` — Сжимает повторяющиеся символы из первого набора в один символ.
3. **\*\*Использование с complement\*\*:**
  - ``-c`` — Применяет операции к complement (инвертированному) набору символов.
4. **\*\*Диапазоны символов\*\*:**
  - ``a-z`` — Пример диапазона символов: ``a-z`` для всех строчных букв.

---

### **\*\*Примеры использования\*\*:**

- **\*\*Замена символов\*\*:**  
Команда ``echo "hello world" | tr 'a-z' 'A-Z`` заменяет все строчные буквы на заглавные:  
````bash`  
`echo "hello world" | tr 'a-z' 'A-Z'`

---

## Komanda sort

# Команда ``sort``

### `sort`

Команда `\sort` используется для сортировки строк в текстовых файлах или потоке данных. По умолчанию строки сортируются в алфавитном порядке.

---

**\*\*Синтаксис\*\*:**

`\sort [опции] [файл...]`

---

### **\*\*Основные опции\*\*:**

1. **\*\*Обратный порядок\*\*:**

- `-r` — Сортирует строки в обратном порядке.

2. **\*\*Игнорирование регистра\*\*:**

- `-f` — Игнорирует различия в регистре при сортировке.

3. **\*\*Удаление дубликатов\*\*:**

- `-u` — Убирает повторяющиеся строки.

4. **\*\*Сортировка по числовым значениям\*\*:**

- `-n` — Сортирует строки как числа (по возрастанию).

5. **\*\*Сортировка по ключу\*\*:**

- `-k [номер_поля]` — Сортирует по конкретному полю в строке (по умолчанию сортирует по первому полю).

6. **\*\*Определение разделителя полей\*\*:**

- `-t [символ]` — Задаёт символ-разделитель полей (по умолчанию используется пробел).

---

### **\*\*Примеры использования\*\*:**

- **\*\*Сортировка строк в алфавитном порядке\*\*:**

Команда `\sort filename.txt` сортирует строки файла `filename.txt` в алфавитном порядке:

```
```bash
sort filename.txt
```

---

## Команда wc

# Команда `\wc`

### `\wc`

Команда `\wc` (word count) используется для подсчёта количества строк, слов и байтов в файлах или потоке данных. Это полезно для быстрого анализа текста и статистики.

---

**\*\*Синтаксис\*\*:**

`\wc [опции] [файл...]`

---

### **\*\*Основные опции\*\*:**

1. **\*\*Подсчет строк\*\***:  
- ``-l`` — Показывает количество строк в файле.
2. **\*\*Подсчет слов\*\***:  
- ``-w`` — Показывает количество слов в файле.
3. **\*\*Подсчет байтов\*\***:  
- ``-c`` — Показывает количество байтов в файле.
4. **\*\*Подсчет символов\*\***:  
- ``-m`` — Показывает количество символов в файле.
5. **\*\*Подсчет байтов в блоках по 512 байт\*\***:  
- ``-k`` — Показывает количество блоков по 512 байт.

---

**Вывод имеет четыре столбца:**

1. Количество строк
2. Количество слов
3. Количество байтов
4. Имя файла

### **\*\*Примеры использования\*\***:

```
- **Подсчет строк в файле**:  
Команда `wc -l filename.txt` показывает количество строк в файле  
`filename.txt`:  
``bash  
wc -l filename.txt
```

---

## Komanda cut

# Команда ``cut``

### ``cut``

Команда ``cut`` используется для извлечения отдельных частей строк из файла или потока данных. Она позволяет извлекать конкретные столбцы или символы из строк, что полезно при работе с текстовыми файлами, разделёнными определённым символом.

---

**\*\*Синтаксис\*\***:  
``cut [опции] [файл...]`

---

### **\*\*Основные опции\*\***:

1. **\*\*Извлечение по столбцам\*\***:  
- ``-f [номера_столбцов]`` — Извлекает указанные столбцы из строк. Столбцы нумеруются с 1.
2. **\*\*Использование разделителя\*\***:  
- ``-d [символ]`` — Указывает символ, который используется как разделитель столбцов (по умолчанию используется табуляция).
3. **\*\*Извлечение по символам\*\***:  
- ``-c [номера_символов]`` — Извлекает указанные символы по номерам.
4. **\*\*Удаление последнего столбца\*\***:



- `--complement` — Показывает все столбцы, кроме указанных.

---

### **\*\*Примеры использования\*\***:

- **\*\*Извлечение столбцов из файла\*\***:

Команда `cut -f 1,3 filename.txt` извлекает первый и третий столбцы из файла `filename.txt`, разделённого табуляцией:

`bash`

cut -f 1,3 filename.txt

apple 10

banana 20

cherry 5

---

## Komandas arch, lscpu

Šīs komandas izvadīs informāciju par procesora arhitektūru un pārējo info.

---

## Komanda free

# Команда `free`

### `free`

Команда `free` используется для отображения информации об объёме свободной и используемой оперативной памяти, а также о состоянии пространства подкачки (swap) в системе. Она помогает анализировать загрузку памяти и производительность системы.

---

**\*\*Синтаксис\*\***:

`free [опции]`

---

### **\*\*Основные опции\*\***:

1. **\*\*Формат отображения памяти\*\***:

- `-b` — Отображает объём памяти в байтах.

- `-k` — Отображает объём памяти в килобайтах (по умолчанию).

- `-m` — Отображает объём памяти в мегабайтах.

- `-g` — Отображает объём памяти в гигабайтах.

- `-h` — Показывает память в удобочитаемом формате (автоматический выбор единиц: K, M, G).

2. **\*\*Обновление в реальном времени\*\***:

- `-s [интервал]` — Обновляет информацию каждые [интервал] секунд.

3. **\*\*Не отображать заголовки\*\***:

- `-t` — Добавляет итоговую строку (total).

- `--si` — Использует кратные 1000 (десятичные) вместо 1024 (двоичных).

---

### **\*\*Примеры использования\*\***:

- **\*\*Показ информации о памяти в килобайтах\*\***:

Команда `free` выводит информацию о памяти в килобайтах (по умолчанию):  
``bash  
free

---

## Komanda lspci

Šī komanda ļauj uzzināt, kadas ierīces ir pieslēgtas caur pci.

---

## Komanda lsusb

Šī komanda ļauj uzzināt, kādas ierīces ir pieslēgtas caur usb.

---

## Komanda pstree

Šī komanda parāda procesu koku:

```
engineer@MintPC:~$ pstree
systemd--ModemManager--3*[{ModemManager}]
        |--NetworkManager--3*[{NetworkManager}]
        |--accounts-daemon--3*[{accounts-daemon}]
        |--agetty
        |--at-spi2-registr--3*[{at-spi2-registr}]
        |--avahi-daemon--avahi-daemon
        |--bwrap--bwrap--obsidian
                |--2*[cat]
                |--obsidian.sh--obsidian--obsidian--25*[{o+
                        |--obsidian--9*[{obsidian}]
                        |--obsidian--34*[{obsidian}]
                        |--zypak-sandbox
                        |--44*[{obsidian}]
        |--2*[bwrap--xdg-dbus-proxy--2*[{xdg-dbus-proxy}]]
        |--bwrap--bwrap--solanum--27*[{solanum}]
        |--chrome_crashpad--2*[{chrome_crashpad}]
        |--chrome_crashpad--{chrome_crashpad}
        |--colord--3*[{colord}]
        |--cron
        |--csd-printer--3*[{csd-printer}]
        |--cups-browsed--3*[{cups-browsed}]
        |--cupsd
        |--dbus-daemon
        |--fwupd--5*[{fwupd}]
        |--irqbalance--{irqbalance}
        |--2*[kerneloops]
        |--lightdm--Xorg--11*[{Xorg}]
                |--lightdm--cinnamon-session--agent--3*[{agent}]
```

---

## Komanda ps

# Команда `ps`

### `ps`

Команда `ps` используется для отображения информации о запущенных процессах. Она позволяет узнать ID процессов, их состояние, ресурсы, которые они используют, и другие параметры. По умолчанию `ps` показывает процессы текущего пользователя в текущем терминале.

---

**\*\*Синтаксис\*\*:**  
``ps [опции]``

---

### **\*\*Основные опции\*\*:**

1. **\*\*Отображение всех процессов\*\*:**
  - ``-e`` или ``-A`` — Показывает все процессы в системе.
  - ``-u [имя_пользователя]`` — Показывает процессы указанного пользователя.
2. **\*\*Формат вывода\*\*:**
  - ``-f`` — Полный формат вывода с дополнительной информацией.
  - ``-o [колонки]`` — Позволяет указать, какие столбцы выводить (например, PID, CMD, %CPU).
3. **\*\*Процессы в дереве\*\*:**
  - ``--forest`` — Показывает процессы в виде дерева (родительские и дочерние).
4. **\*\*Процессы в реальном времени\*\*:**
  - ``-T`` — Показывает процессы текущего терминала.
  - ``-x`` — Показывает процессы без управляющего терминала (фоновые).
5. **\*\*Фильтрация по PID или имени\*\*:**
  - ``-p [PID]`` — Показывает процесс с указанным PID.
  - ``-C [команда]`` — Показывает процессы с указанным именем команды.

---

### **\*\*Примеры использования\*\*:**

- **\*\*Просмотр процессов текущего пользователя\*\*:**  
Команда ``ps`` выводит процессы текущего пользователя:  
````bash``  
`ps`

---

## Komanda top — sistēmas monitors

# Команда ``top``

### ``top``

Команда ``top`` используется для отображения в реальном времени информации о процессах, их состоянии, и используемых ресурсах системы (CPU, память и т.д.). Она предоставляет обновляемый список запущенных процессов, сортируемый по их загрузке системы.

---

**\*\*Синтаксис\*\*:**  
``top [опции]``

---

### **\*\*Основные опции\*\*:**

1. **\*\*Интервал обновления\*\*:**
  - ``-d [секунды]`` — Задаёт интервал обновления (по умолчанию 3 секунды).
2. **\*\*Показ процессов конкретного пользователя\*\*:**
  - ``-u [имя_пользователя]`` — Показывает процессы только указанного

пользователя.

3. **\*\*Количество обновлений\*\***:
  - ``-n [число]`` — Задаёт количество обновлений перед завершением.
4. **\*\*Показ конкретного PID\*\***:
  - ``-p [PID]`` — Отображает информацию только о процессе с указанным PID.
5. **\*\*Не выводить интерактивный интерфейс\*\***:
  - ``-b`` — Работает в пакетном режиме, удобном для сохранения вывода в файл.

---

### **\*\*Основные интерактивные команды\*\*** (во время работы ``top``):

1. **\*\*Сортировка\*\***:
  - ``P`` — Сортировка по загрузке CPU (по умолчанию).
  - ``M`` — Сортировка по использованию памяти.
  - ``T`` — Сортировка по времени выполнения.
2. **\*\*Фильтрация\*\***:
  - ``u`` — Отображение процессов конкретного пользователя (запрос имени пользователя).
  - ``k`` — Завершение процесса (запрос PID).
3. **\*\*Обновление интерфейса\*\***:
  - ``Space`` — Немедленное обновление списка процессов.
  - ``h`` — Показ справки.
4. **\*\*Завершение работы\*\***:
  - ``q`` — Выход из программы.

---

### **\*\*Примеры использования\*\***:

```
engineer@MintPC: ~  
Файл  Правка  Вид  Поиск  Терминал  Справка  
top - 15:37:46 up 4:04, 1 user, load average: 0,53, 0,43, 0,38  
Tasks: 355 total, 1 running, 353 sleeping, 0 stopped, 1 zombie  
%Cpu(s): 0,9 us, 0,5 sy, 0,0 ni, 98,5 id, 0,1 wa, 0,0 hi, 0,0 si, 0,0 st  
Миб Мем : 15853,9 total, 1122,2 free, 5986,7 used, 9389,9 buff/cache  
Миб Swap: 2048,0 total, 2048,0 free, 0,0 used. 9867,2 avail Mem  


| PID   | USER     | PR | NI  | VIRT    | RES    | SHR    | S | %CPU | %MEM | TIME+   | COMMAND   |
|-------|----------|----|-----|---------|--------|--------|---|------|------|---------|-----------|
| 3714  | engineer | 20 | 0   | 2646772 | 230656 | 122144 | S | 4,0  | 1,4  | 7:24.86 | Isolate+  |
| 2421  | engineer | 20 | 0   | 5287544 | 256920 | 135900 | S | 3,3  | 1,6  | 3:38.61 | cinnamon  |
| 4272  | engineer | 20 | 0   | 1167,2g | 486536 | 166160 | S | 1,7  | 3,0  | 7:36.17 | obsidian  |
| 2842  | engineer | 20 | 0   | 11,7g   | 644048 | 304424 | S | 1,0  | 4,0  | 9:32.52 | firefox+  |
| 3796  | engineer | 20 | 0   | 1078340 | 101880 | 56444  | S | 0,7  | 0,6  | 7:23.19 | transmi+  |
| 4250  | engineer | 20 | 0   | 32,9g   | 191836 | 115660 | S | 0,7  | 1,2  | 1:53.03 | obsidian  |
| 12425 | engineer | 20 | 0   | 2700408 | 257956 | 113084 | S | 0,7  | 1,6  | 0:12.22 | Isolate+  |
| 36    | root     | 20 | 0   | 0       | 0      | 0      | S | 0,3  | 0,0  | 0:06.79 | ksofttir+ |
| 1720  | root     | 20 | 0   | 1252052 | 168100 | 109684 | S | 0,3  | 1,0  | 2:53.20 | Xorg      |
| 1990  | engineer | 9  | -11 | 122652  | 19064  | 9468   | S | 0,3  | 0,1  | 0:36.14 | pipewire  |
| 1994  | engineer | 9  | -11 | 145228  | 44028  | 10768  | S | 0,3  | 0,3  | 1:10.36 | pipewir+  |
| 2962  | engineer | 20 | 0   | 386544  | 7184   | 6520   | S | 0,3  | 0,0  | 0:00.20 | xdg-des+  |
| 3118  | engineer | 20 | 0   | 391296  | 45636  | 33072  | S | 0,3  | 0,3  | 0:49.77 | Utility+  |
| 4456  | engineer | 20 | 0   | 2362652 | 170036 | 108340 | S | 0,3  | 1,0  | 0:45.47 | solanum   |
| 11978 | root     | 20 | 0   | 0       | 0      | 0      | I | 0,3  | 0,0  | 0:04.76 | kworker+  |
| 13638 | root     | 20 | 0   | 0       | 0      | 0      | I | 0,3  | 0,0  | 0:01.46 | kworker+  |
| 14988 | engineer | 20 | 0   | 552668  | 51192  | 40808  | S | 0,3  | 0,3  | 0:00.30 | gnome-t+  |


```

## Komanda journalctl

Ši komanda izvada sistēmas žurnālu.

# Команда dmesg

Šī komanda izvada kodola žurnālu.

---

# Команда host

host example.com

Šī komanda ļauj uzzināt konkrēta domēna vārda Ip-adresi.

---

# Команда ifconfig

Šī komanda ļauj uzzināt tīkla konfigurāciju:

```
root@localhost:~# ifconfig
eth0      Link encap:Ethernet  HWaddr b6:84:ab:e9:8f:0a
          inet addr:192.168.1.2   Bcast:0.0.0.0  Mask:255.255.255.0
          inet6 addr: fe80::b484:abff:fee9:8f0a/64 Scope:Link
          UP    BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:95 errors:0 dropped:4 overruns:0 frame:0
          TX packets:9 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:25306 (25.3 KB)  TX bytes:690 (690.0 B)
lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          inet6 addr: ::1/128 Scope:Host
          UP LOOPBACK RUNNING  MTU:65536  Metric:1
          RX packets:6 errors:0 dropped:0 overruns:0 frame:0
          TX packets:6 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:460 (460.0 B)  TX bytes:460 (460.0 B)
```

# Команда ip

# Руководство по команде `ip`

Команда `ip` используется для управления сетевыми интерфейсами в системах Linux. Она предоставляет мощный набор функций для настройки, управления и мониторинга сетевых устройств и их конфигураций.

## Общий синтаксис

```
``bash
ip [OPTIONS] OBJECT COMMAND [ARGUMENTS]
``
```

- **OPTIONS**: Дополнительные параметры.
- **OBJECT**: Тип объекта (например, `link`, `addr`, `route` и др.).
- **COMMAND**: Команда для выполнения.
- **ARGUMENTS**: Дополнительные аргументы команды.

## Основные объекты

### 1. `link`

Управление сетевыми интерфейсами.

#### Команды:

- **`ip link show`**: Показать список всех интерфейсов.

```

```bash
ip link show
```
- **`ip link set dev <интерфейс> up|down`**: Включить или отключить интерфейс.
  ```bash
  ip link set dev eth0 up
  ip link set dev eth0 down
  ```
- **`ip link set dev <интерфейс> mtu <размер>`**: Установить MTU для интерфейса.
  ```bash
  ip link set dev eth0 mtu 1500
  ```

```

### ### 2. `addr` Работа с IP-адресами.

#### #### Команды:

```

- **`ip addr show`**: Показать IP-адреса всех интерфейсов.
  ```bash
  ip addr show
  ```
- **`ip addr add <адрес>/<маска> dev <интерфейс>`**: Добавить IP-адрес
  интерфейсу.
  ```bash
  ip addr add 192.168.1.10/24 dev eth0
  ```
- **`ip addr del <адрес>/<маска> dev <интерфейс>`**: Удалить IP-адрес.
  ```bash
  ip addr del 192.168.1.10/24 dev eth0
  ```

```

### ### 3. `route` Работа с маршрутами.

#### #### Команды:

```

- **`ip route show`**: Показать таблицу маршрутизации.
  ```bash
  ip route show
  ```
- **`ip route add <сеть>/<маска> via <шлюз> dev <интерфейс>`**: Добавить
  маршрут.
  ```bash
  ip route add 192.168.2.0/24 via 192.168.1.1 dev eth0
  ```
- **`ip route del <сеть>/<маска>`**: Удалить маршрут.
  ```bash
  ip route del 192.168.2.0/24
  ```

```

### ### 4. `neigh` Работа с ARP-таблицей.

#### #### Команды:

```

- **`ip neigh show`**: Показать ARP-таблицу.
  ```bash
  ip neigh show
  ```
- **`ip neigh add <адрес> lladdr <MAC-адрес> dev <интерфейс>`**: Добавить запись
  в ARP-таблицу.
  ```bash
  ip neigh add 192.168.1.2 lladdr 00:11:22:33:44:55 dev eth0
  ```

```

```

- **`ip neigh del <адрес> dev <интерфейс>`**: Удалить запись из ARP-таблицы.
  ```bash
  ip neigh del 192.168.1.2 dev eth0
  ```

```

### ## Полезные опции

```

- **`-4`**: Использовать только IPv4.
  ```bash
  ip -4 addr show
  ```
- **`-6`**: Использовать только IPv6.
  ```bash
  ip -6 addr show
  ```
- **`-c`**: Включить цветной вывод.
  ```bash
  ip -c addr show
  ```

```

### ## Примеры комбинирования

#### ### Проверка доступности интерфейсов

```

```bash
ip -c link show
```

```

#### ### Назначение статического маршрута

```

```bash
ip route add 10.0.0.0/16 via 192.168.1.1 dev eth0
```

```

#### ### Удаление всех IP-адресов с интерфейса

```

```bash
ip addr flush dev eth0
```

```

#### ### Мониторинг изменений

```

```bash
ip monitor
```

```

Эта команда позволяет отслеживать изменения в конфигурации сети в реальном времени.

### ## Полезные советы

- Используйте `ip` вместо устаревшей команды `ifconfig` для более современной работы с сетью.
- Для сохранения конфигурации сети используйте соответствующие конфигурационные файлы в вашей системе (например, `/etc/network/interfaces` или `/etc/netplan/` в зависимости от дистрибутива).

---

Для более подробной информации воспользуйтесь справкой:

```

```bash
man ip
```

```

---

## Komanda route

Šī komanda ļauj uzzināt, uz kurieni tiek sūtīti freimi (paķetes):

```
root@localhost:~# route
Kernel IP routing table
Destination    Gateway         Genmask         Flags Metric Ref    Use Iface
192.168.1.0    *               255.255.255.0   U      0      0      0 eth0
default        192.168.1.1     0.0.0.0         UG     0      0      0 eth0
```

---

## Komanda ping

ping example.com

vai arī

ping 8.8.8.8

Šī komanda ļauj nosūtīt freimu (paķeti) uz kādu adresi (hostu).  
Parasti to izmanto diagnostikai un tīkla pieslēguma testēšanai.

---

## Komanda netstat

# Руководство по команде `netstat`

Команда `netstat` используется для мониторинга сетевых подключений, таблиц маршрутизации, статистики интерфейсов и других сетевых параметров. Несмотря на то, что `netstat` постепенно заменяется командой `ss`, она все еще полезна в некоторых сценариях.

## Общий синтаксис

```
```bash
netstat [OPTIONS]
```
```

## Основные опции

### Показ подключений

- `**-a**`: Отображает все активные соединения (включая прослушиваемые порты).  
```bash  
netstat -a  
```
- `**-t**`: Показывает только TCP соединения.  
```bash  
netstat -t  
```
- `**-u**`: Показывает только UDP соединения.  
```bash  
netstat -u  
```
- `**-x**`: Показывает UNIX сокеты.  
```bash  
netstat -x  
```

### Информация о маршрутизации

- `**-r**`: Показывает таблицу маршрутизации.  
```bash



```
netstat -r
```
```

### ### Статистика сетевых интерфейсов

- \*\*`-i`\*\*: Отображает статистику сетевых интерфейсов.  
```bash  
netstat -i  
```
- \*\*`-e`\*\*: Показывает расширенную статистику интерфейсов.  
```bash  
netstat -ie  
```

### ### PID и программа

- \*\*`-p`\*\*: Отображает PID и имя программы для каждого соединения (требуется права суперпользователя).  
```bash  
sudo netstat -p  
```

### ### Состояния TCP соединений

- \*\*`-s`\*\*: Показывает статистику по протоколам.  
```bash  
netstat -s  
```
- \*\*`-n`\*\*: Показывает адреса и порты в числовом формате (без попытки разрешения имен).  
```bash  
netstat -n  
```

### ### Комбинации опций

- \*\*`-lt`\*\*: Список прослушиваемых TCP портов.  
```bash  
netstat -lt  
```
- \*\*`-lu`\*\*: Список прослушиваемых UDP портов.  
```bash  
netstat -lu  
```
- \*\*`-ant`\*\*: Список всех TCP соединений с использованием числового формата.  
```bash  
netstat -ant  
```

### ## Полезные примеры

#### ### Отображение всех активных соединений и их состояния

```
```bash  
netstat -an  
```
```

#### ### Список процессов, связанных с открытыми портами

```
```bash  
sudo netstat -plnt  
```
```

#### ### Просмотр статистики по протоколам

```
```bash
netstat -s
```
```

### Показ маршрутов сети

```
```bash
netstat -rn
```
```

## Замена команды `netstat`

Для современных систем рекомендуется использовать команду `ss`:

```
- Аналог `netstat -an`:
  ```bash
  ss -an
  ```
- Аналог `netstat -lt`:
  ```bash
  ss -lt
  ```
```

---

Для более подробной информации воспользуйтесь справкой:

```
```bash
man netstat
```
```

---

## Komanda ss

# Руководство по команде `ss`

Команда `ss` используется для отображения информации о сетевых подключениях, таблицах маршрутизации, сокетах и статистике протоколов. Это современная альтернатива устаревшей команде `netstat` с более высокой производительностью.

## Общий синтаксис

```
```bash
ss [OPTIONS]
```
```

## Основные опции

### Просмотр соединений

```
- **`-a`**: Показывает все сокеты (включая прослушиваемые).
  ```bash
  ss -a
  ```
- **`-t`**: Показывает только TCP-соединения.
  ```bash
  ss -t
  ```
- **`-u`**: Показывает только UDP-соединения.
  ```bash
  ss -u
  ```
- **`-x`**: Показывает только UNIX-сокеты.
```

```
```bash
ss -x
```
```

### ### Прослушиваемые порты

- `ss -l`: Показывает только прослушиваемые сокеты.

```
```bash
ss -l
```
```

- `ss -lt`: Список прослушиваемых TCP-портов.

```
```bash
ss -lt
```
```

- `ss -lu`: Список прослушиваемых UDP-портов.

```
```bash
ss -lu
```
```

### ### Статистика соединений

- `ss -s`: Показать сводную статистику по протоколам.

```
```bash
ss -s
```
```

### ### Фильтрация по состояниям

- `ss state <state>`: Фильтрация по состоянию TCP соединения (например, `ESTABLISHED`, `LISTEN`, `CLOSED`).

```
```bash
ss state ESTABLISHED
```
```

### ### Отображение PID и процесса

- `ss -p`: Показывает PID и имя процесса для каждого сокета (требуется права суперпользователя).

```
```bash
sudo ss -p
```
```

### ### Отображение адресов и портов в числовом формате

- `ss -n`: Отключает попытки разрешения имен хостов и сервисов.

```
```bash
ss -n
```
```

### ### Таблица маршрутизации

- `ss -r`: Показывает таблицу маршрутизации.

```
```bash
ss -r
```
```

### ### Комбинированные опции

- `ss -tan`: Показать все TCP-соединения в числовом формате.

```
```bash
ss -tan
```
```

- `ss -ltnp`: Список всех прослушиваемых TCP-сокетов с указанием процессов.

```
```bash
```

```
sudo ss -ltnp
```
```

## ## Полезные примеры

### ### Показ всех активных TCP и UDP соединений

```
```bash
ss -tua
```
```

### ### Список процессов, использующих открытые порты

```
```bash
sudo ss -ltnp
```
```

### ### Просмотр открытых соединений на определённом порту

```
```bash
ss -t src :80
```
```

### ### Показ всех установленных соединений

```
```bash
ss state ESTABLISHED
```
```

### ### Показ всех сокетов с их статистикой

```
```bash
ss -as
```
```

## ## Полезные советы

- Для более быстрого поиска конкретных соединений используйте фильтры (например, по порту или состоянию).
- Команда `ss` поддерживает мощный синтаксис фильтрации, позволяющий искать соединения по IP, порту или другим параметрам.

---

|               |  |
|---------------|--|
| Netid         | Тип сокета и транспортный протокол                               |
| State         | Подключено или неподключено, в зависимости от протокола          |
| Recv-Q        | Объем данных, поставленных в очередь для обработки, был получен  |
| Send-Q        | Объем данных, поставленных в очередь для отправки на другой хост |
| Local Address | Адрес и порт части соединения локального хоста.                  |
| Peer Address  | Адрес и порт части соединения удаленного хоста.                  |

Для подробной информации и всех доступных опций обратитесь к справке:

```
```bash
man ss
```
```

---

# Komanda dig

dig example.com

Šī komanda ļauj pārbaudīt, vai DNS serveris funkcionē.

---

# Komanda ssh

Руководство по команде `ssh`

Команда `ssh` (Secure Shell) используется для безопасного удаленного подключения к другим устройствам через сеть. Она позволяет выполнять команды, передавать файлы и управлять удаленными серверами.

## Общий синтаксис

```
```bash
ssh [OPTIONS] [USER@]HOST [COMMAND]
```
```

- \*\*`USER`\*\*: Имя пользователя на удаленном устройстве.
- \*\*`HOST`\*\*: IP-адрес или доменное имя удаленного устройства.
- \*\*`COMMAND`\*\*: Команда для выполнения на удаленном устройстве.

## Основные опции

### Подключение к удаленному серверу

- \*\*`ssh [USER@]HOST`\*\*: Простое подключение к удаленному серверу.  
```bash  
ssh user@example.com  
```

### Указание порта

- \*\*`-p <порт>`\*\*: Указывает порт подключения (по умолчанию используется порт 22).  
```bash  
ssh -p 2222 user@example.com  
```

### Выполнение команды

- \*\*`ssh [USER@]HOST COMMAND`\*\*: Выполняет команду на удаленном сервере без открытия интерактивного сеанса.  
```bash  
ssh user@example.com "ls -l"  
```

### Указание файла с ключами

- \*\*`-i <путь\_к\_файлу>`\*\*: Использует указанный приватный ключ для аутентификации.  
```bash  
ssh -i ~/.ssh/id\_rsa user@example.com  
```

### Перенаправление портов

- \*\*`-L <локальный\_порт>:<адрес\_цели>:<удалённый\_порт>`\*\*: Настраивает локальный туннель.  
```bash

```
ssh -L 8080:localhost:80 user@example.com
```
- **`-R <удалённый_порт>:<адрес_цели>:<локальный_порт>`**: Настраивает удалённый туннель.
```bash
ssh -R 8080:localhost:80 user@example.com
```
```

### ### Фоновый режим

```
- **`-f`**: Запускает SSH-сессию в фоновом режиме.
```bash
ssh -f user@example.com "sleep 60"
```
```

### ### Перенаправление X11

```
- **`-X`**: Включает перенаправление X11 (графический интерфейс).
```bash
ssh -X user@example.com
```
```

### ### Подробный вывод

```
- **`-v`**: Включает подробный вывод для отладки.
```bash
ssh -v user@example.com
```
- **`-vv`** или **`-vvv`**: Увеличивает уровень подробности вывода.
```bash
ssh -vv user@example.com
```
```

### ## Работа с конфигурационным файлом

Конфигурация SSH-клиента хранится в файле `~/.ssh/config`. Пример записи:

```
```text
Host example
  HostName example.com
  User user
  Port 2222
  IdentityFile ~/.ssh/id_rsa
```
```

Теперь можно подключаться с помощью короткой команды:

```
```bash
ssh example
```
```

### ## Примеры использования

#### ### Простое подключение к серверу

```
```bash
ssh user@example.com
```
```

#### ### Выполнение команды на сервере

```
```bash
ssh user@example.com "uptime"
```
```

### ### Настройка и использование SSH-туннеля

```
```bash
ssh -L 8080:localhost:80 user@example.com
```
```

### ### Копирование файлов через SSH

Для передачи файлов используйте команду `scp`:

```
```bash
scp файл.txt user@example.com:/путь/на/сервере
```
```

### ### Генерация SSH-ключей

Для работы с ключами выполните:

```
```bash
ssh-keygen -t rsa -b 4096 -C "your_email@example.com"
```
```

Скопируйте публичный ключ на сервер:

```
```bash
ssh-copy-id user@example.com
```
```

---

Для более подробной информации воспользуйтесь справкой:

```
```bash
man ssh
```
```

---

## Komanda su

# Руководство по команде `su`

Команда `su` (substitute user) используется для смены пользователя в текущей сессии. Чаще всего она применяется для переключения на суперпользователя (root) для выполнения административных задач.

### ## Общий синтаксис

```
```bash
su [OPTIONS] [USERNAME]
```
```

- \*\*`USERNAME`\*\*: Имя пользователя, под которым вы хотите выполнить вход (по умолчанию — `root`).

### ## Основные опции

#### ### Смена пользователя

- \*\*`su`\*\*: Переключение на суперпользователя (root). Запрашивает пароль root.

```
```bash
su
```
```

- \*\*`su USERNAME`\*\*: Переключение на указанного пользователя. Запрашивает пароль

```
пользователя.  
```bash  
su username  
```
```

### ### Запуск shell с изменением среды

```
- **`-l`** или **`--login`**: Запускает shell как при полноценном входе в  
систему. Текущая среда заменяется на среду указанного пользователя.  
```bash  
su -l username  
```  
  
или  
```bash  
su - username  
```
```

### ### Указание shell

```
- **`-s <shell>`**: Использует указанный shell вместо стандартного.  
```bash  
su -s /bin/bash username  
```
```

### ### Выполнение команды

```
- **`-c <command>`**: Выполняет указанную команду от имени пользователя и  
возвращается в исходный сеанс.  
```bash  
su -c "apt update"  
```
```

### ### Подробный вывод

```
- **`-v`**: Показать версию программы `su`.  
```bash  
su -v  
```  
  
- **`-h`**: Отобразить справку по использованию команды.  
```bash  
su -h  
```
```

### ## Примеры использования

#### ### Переключение на суперпользователя

```
```bash  
su  
```
```

#### ### Переключение на другого пользователя

```
```bash  
su username  
```
```

#### ### Выполнение команды от имени суперпользователя

```
```bash  
su -c "systemctl restart apache2"  
```
```

#### ### Использование другого shell



```
```bash
su -s /bin/zsh
```
```

### Переключение на другого пользователя с изменением среды

```
```bash
su - username
```
```

## Безопасная альтернатива: `sudo`

Рекомендуется использовать `sudo` вместо `su`, чтобы минимизировать риски и управлять доступом к административным правам.

- Выполнение команды от имени root:

```
```bash
sudo apt update
```
```
- Получение root-доступа на ограниченное время:

```
```bash
sudo -i
```
```

## Справка

Для получения дополнительной информации воспользуйтесь справкой:

```
```bash
man su
```
```

---

## Komanda id

# Руководство по команде `id`

Команда `id` используется для отображения информации о пользователе, такой как UID, GID и принадлежность к группам. Она полезна для проверки текущей идентификации пользователя и групповых прав.

## Общий синтаксис

```
```bash
id [OPTIONS] [USERNAME]
```
```

- **`USERNAME`**: Имя пользователя, для которого нужно отобразить информацию (по умолчанию используется текущий пользователь).

## Основные опции

### Отображение информации о текущем пользователе

- **Без опций**: Показывает информацию о текущем пользователе.

```
```bash
id
```
```

### UID, GID и группы

- **`-u`**: Показывает только UID пользователя.

```
```bash
```

```
id -u
```
- **`-g`**: Показывает только GID пользователя.
  ```bash
  id -g
  ```
- **`-G`**: Показывает список всех GID групп, к которым принадлежит
  пользователь.
  ```bash
  id -G
  ```
- **`-n`**: Показывает имена вместо числовых идентификаторов (может быть
  использовано с `-u`, `-g`, `-G`).
  ```bash
  id -un
  id -gn
  id -Gn
  ```
```

### ### Информация о другом пользователе

```
- **`id USERNAME`**: Показывает информацию о указанном пользователе.
  ```bash
  id username
  ```
```

### ## Примеры использования

#### ### Проверка текущего пользователя

```
```bash
id
```
```

#### ### Получение UID текущего пользователя

```
```bash
id -u
```
```

#### ### Получение имени текущего пользователя

```
```bash
id -un
```
```

#### ### Проверка групп, к которым принадлежит пользователь

```
```bash
id -G
```
```

#### ### Отображение информации о другом пользователе

```
```bash
id username
```
```

#### ### Проверка имени группы пользователя

```
```bash
id -gn
```
```

## ## Полезные советы

- Команда ``id`` часто используется в скриптах для проверки привилегий пользователя.

```
```bash
if [ $(id -u) -eq 0 ]; then
    echo "Вы запустили скрипт как root."
else
    echo "Запустите скрипт с правами root."
fi
```
```

- Сочетайте с другими командами для проверки прав доступа.

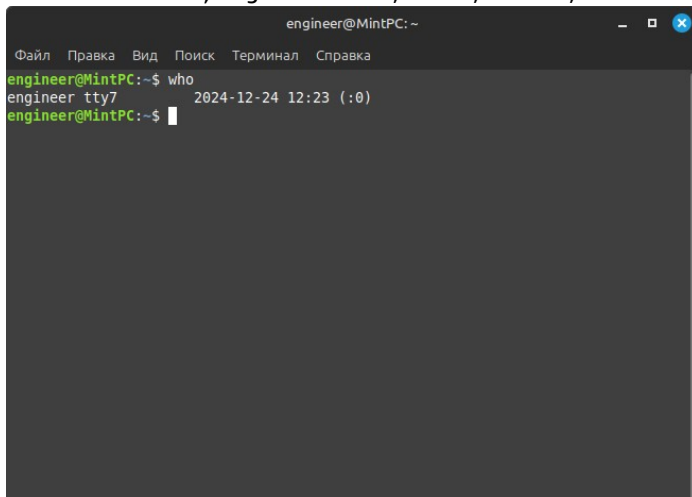
## ## Справка

Для получения дополнительной информации воспользуйтесь справкой:

```
```bash
man id
```

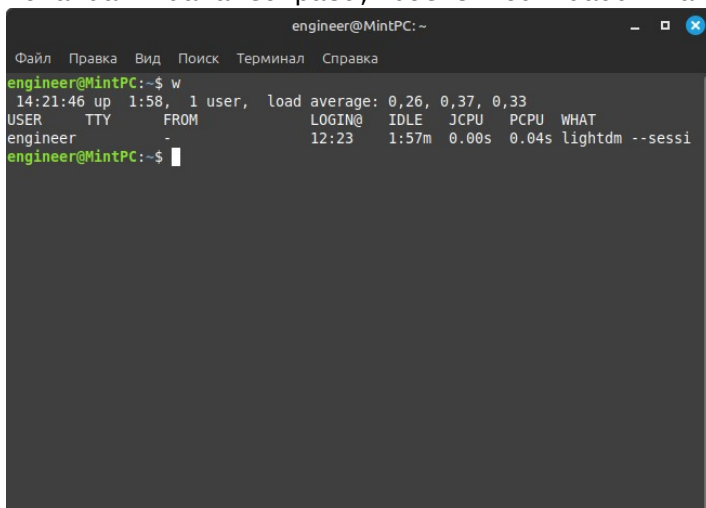
# Komandas who, w

Komanda who ļauj uzzināt, kas, kurš, kad un caur ko pieslēdzās sistēmai.



```
engineer@MintPC: ~
Файл Правка Вид Поиск Терминал Справка
engineer@MintPC:~$ who
engineer tty7        2024-12-24 12:23 (:0)
engineer@MintPC:~$
```

Komanda w dara to pašu, bet sniedz daudz vairāk info:



```
engineer@MintPC: ~
Файл Правка Вид Поиск Терминал Справка
engineer@MintPC:~$ w
14:21:46 up 1:58, 1 user, load average: 0,26, 0,37, 0,33
USER      TTY      FROM          LOGIN@   IDLE   JCPU   PCPU   WHAT
engineer  -        -             12:23    1:57m  0.00s  0.04s  lightdm --sessi
engineer@MintPC:~$
```

| Столбец | Пример | Описание                               |
|---------|--------|----------------------------------------|
| USER    | root   | Имя пользователя, вошедшего в систему. |

| Столбец | Пример      | Описание                                                                                        |
|---------|-------------|-------------------------------------------------------------------------------------------------|
| TTY     | tty2        | В каком окне терминала работает пользователь.                                                   |
| FROM    | example.com | Откуда пользователь вошел в систему.                                                            |
| LOGIN@  | 10:00       | Когда пользователь вошел в систему.                                                             |
| IDLE    | 43:44       | Как долго пользователь бездействовал с момента выполнения последней команды.                    |
| JCPU    | 0.01s       | Общее время процессора, использованное всеми процессами, запущенными с момента входа в систему. |
| PCPU    | 0.01s       | Общее время процессора для текущего процесса.                                                   |
| WHAT    | -bash       | Текущий процесс, который запускает пользователь.                                                |

## Komanda last

Ši komanda parāda visu logošanas vēsturi no faila /etc/log/wtmp

```

engineer@MintPC: ~
Файл Правка Вид Поиск Терминал Справка
engineer@MintPC:~$ last
engineer tty7 :0 Tue Dec 24 12:23 gone - no logout
reboot system boot 6.8.0-51-generic Tue Dec 24 12:23 still running
engineer tty7 :0 Tue Dec 24 12:03 - 12:14 (00:10)
reboot system boot 6.8.0-51-generic Tue Dec 24 12:02 - 12:14 (00:11)
engineer tty7 :0 Mon Dec 23 22:24 - 23:07 (00:43)
reboot system boot 6.8.0-51-generic Mon Dec 23 22:21 - 23:08 (00:46)
engineer tty7 :0 Sun Dec 22 20:02 - crash (1+02:18)
reboot system boot 6.8.0-51-generic Sun Dec 22 20:02 - 23:08 (1+03:05)
engineer tty7 :0 Sun Dec 22 17:01 - 17:50 (00:48)
reboot system boot 6.8.0-51-generic Sun Dec 22 17:01 - 17:50 (00:49)
engineer tty7 :0 Sun Dec 22 15:34 - 16:35 (01:00)
reboot system boot 6.8.0-51-generic Sun Dec 22 15:33 - 16:35 (01:01)
engineer tty7 :0 Sun Dec 22 12:39 - 14:41 (02:02)
reboot system boot 6.8.0-51-generic Sun Dec 22 12:38 - 14:41 (02:03)
engineer tty7 :0 Sat Dec 21 21:26 - 00:19 (02:53)
reboot system boot 6.8.0-51-generic Sat Dec 21 21:25 - 00:19 (02:53)
engineer tty7 :0 Sat Dec 21 18:33 - 20:36 (02:03)
reboot system boot 6.8.0-51-generic Sat Dec 21 18:32 - 20:36 (02:04)
engineer tty7 :0 Sat Dec 21 10:43 - 16:30 (05:47)
reboot system boot 6.8.0-51-generic Sat Dec 21 10:42 - 16:30 (05:48)
engineer tty7 :0 Fri Dec 20 21:24 - 22:55 (01:30)
reboot system boot 6.8.0-51-generic Fri Dec 20 21:23 - 22:55 (01:31)
engineer tty7 :0 Fri Dec 20 21:16 - 21:23 (00:07)

```

## Komanda groupadd

# Руководство по команде `groupadd`

Команда `groupadd` используется для создания новых групп пользователей в операционных системах на базе Linux. Группы позволяют управлять доступом к файлам, директориям и другим ресурсам.

## Общй синтаксис

```

` `` bash
groupadd [OPTIONS] GROUPNAME
` ``

```

- **\*\*`GROUPNAME`\*\***: Имя новой группы.

## Основные опции

### Указание GID (Group ID)

```

- **`-g <GID>`**: Задаёт числовой идентификатор группы (GID).
  ` `` bash
  groupadd -g 1001 developers
  ` ``

```

Если GID не указан, система назначит его автоматически.

### Поменять имя группы

```
- **`-n <NAME>`**: Меняет имя группы.  
  ``bash  
  groupadd -n kijasko
```

### Системная группа

```
- **`-r`**: Создает системную группу с GID из диапазона системных групп (обычно  
< 1000).  
  ``bash  
  groupadd -r backup
```

### Задание пароля для группы

```
- **`-p <PASSWORD>`**: Устанавливает зашифрованный пароль для группы.  
  ``bash  
  groupadd -p $(openssl passwd -1 "password") team  
  
> **Примечание**: Рекомендуется использовать команду `grpasswd` для управления  
паролями.
```

### Задание файла с конфигурацией

```
- **`-K <KEY=VALUE>`**: Устанавливает настройки в конфигурационном файле  
`/etc/login.defs` для этой группы.  
  ``bash  
  groupadd -K GID_MIN=2000 -K GID_MAX=3000 testgroup
```

## Примеры использования

### Создание группы с автоматическим GID

```
``bash  
groupadd developers
```

### Создание группы с заданным GID

```
``bash  
groupadd -g 1050 engineers
```

### Создание системной группы

```
``bash  
groupadd -r systemgroup
```

### Установка пароля для группы

```
``bash  
groupadd -p $(openssl passwd -1 "group_password") marketing
```

### Проверка созданной группы

После создания группы можно проверить ее наличие в файле `/etc/group`:

```
``bash  
grep developers /etc/group
```

...

### ## Сопутствующие команды

- **``groups``**: Показывает список групп, к которым принадлежит пользователь.
- **``groupdel``**: Удаляет группу.  
``bash``  
groupdel developers
- **``usermod``**: Добавляет пользователя в группу.  
``bash``  
usermod -aG developers username
- **``gpasswd``**: Управляет паролями и правами доступа для группы.

### ## Справка

Для получения дополнительной информации воспользуйтесь справкой:

```
``bash``  
man groupadd
```

---

## Komandas -nogroup, find

find / -nogroup → / (šajā piemērā meklē failus saknes katalogā)  
Šīs komandas ļauj atrast failus, kas nepieder nevienai grupai.

---

## Komanda useradd

### # Руководство по команде `useradd`

Команда `useradd` используется для создания новых пользователей в операционных системах на базе Linux. Она позволяет настроить учетные записи пользователей, включая их домашние директории, группы и начальные настройки.

### ## Общий синтаксис

```
``bash``  
useradd [OPTIONS] USERNAME  
````
```

- **``USERNAME``**: Имя нового пользователя.

### ## Основные опции

#### ### Указание домашней директории

- **``-d <DIRECTORY>``**: Устанавливает путь к домашней директории пользователя.  
``bash``  
useradd -d /home/customuser customuser

Если не указано, по умолчанию используется `/home/USERNAME`.

#### ### Автоматическое создание домашней директории

- **``-m``**: Создает домашнюю директорию для пользователя, если она отсутствует.  
``bash``  
useradd -m username

#### ### Указание начальной группы

```
- **`-g <GROUP>`**: Назначает основную группу пользователя.  
  ```bash  
  useradd -g developers username  
  ```
```

#### ### Дополнительные группы

```
- **`-G <GROUP1, GROUP2, ...>`**: Добавляет пользователя в дополнительные группы.  
  ```bash  
  useradd -G sudo, docker username  
  ```
```

#### ### Указание оболочки

```
- **`-s <SHELL>`**: Устанавливает оболочку пользователя (например, `/bin/bash`  
или `/bin/zsh`).  
  ```bash  
  useradd -s /bin/bash username  
  ```
```

#### ### Указание UID

```
- **`-u <UID>`**: Устанавливает идентификатор пользователя (UID).  
  ```bash  
  useradd -u 1001 username  
  ```
```

#### ### Без создания домашней директории

```
- **`-M`**: Не создает домашнюю директорию.  
  ```bash  
  useradd -M username  
  ```
```

#### ### Установка даты истечения учётной записи

```
- **`-e <DATE>`**: Устанавливает дату истечения учётной записи в формате `YYYY-MM-DD`.  
  ```bash  
  useradd -e 2024-12-31 username  
  ```
```

#### ### Указание комментария

```
- **`-c <COMMENT>`**: Добавляет описание или комментарий к учётной записи.  
  ```bash  
  useradd -c "John Doe, Developer" username  
  ```
```

#### ## Примеры использования

##### ### Создание пользователя с домашней директорией

```
```bash  
useradd -m -s /bin/bash john  
```
```

##### ### Создание пользователя с заданной группой и UID

```
```bash  
useradd -g developers -u 1050 alice  
```
```

### Создание пользователя с истекающей учётной записью

```
```bash
useradd -e 2025-01-01 temporaryuser
```
```

### Проверка созданного пользователя

После создания пользователя можно проверить его информацию с помощью команды:

```
```bash
id username
```
```

Или просмотреть файл `/etc/passwd`:

```
```bash
grep username /etc/passwd
```
```

## Сопутствующие команды

- `passwd`: Устанавливает пароль для пользователя.  
```bash  
passwd username  
```
- `usermod`: Изменяет настройки существующего пользователя.
- `userdel`: Удаляет пользователя.  
```bash  
userdel username  
```

## Справка

Для получения дополнительной информации воспользуйтесь справкой:

```
```bash
man useradd
```
```

---

## Komanda adduser (ērtāka un modernāka nekā useradd)

# Руководство по команде `adduser`

Команда `adduser` используется для удобного и интерактивного создания новых пользователей в системах Linux. Она является более пользовательски-ориентированной альтернативой команде `useradd`.

## Общий синтаксис

```
```bash
adduser [OPTIONS] USERNAME
```
```

- `USERNAME`: Имя нового пользователя.

## Основные возможности

1. **Интерактивный режим**: Команда автоматически запрашивает информацию, необходимую для создания пользователя.
2. **Автоматическая настройка**: Использует стандартные значения из конфигурационного файла `/etc/adduser.conf`.



3. **\*\*Поддержка дополнительных групп\*\***: Позволяет сразу добавлять пользователя в группы.

## Пример использования

### Создание нового пользователя

```
```bash
sudo adduser username
```
```

После выполнения команды будет запущен диалог:

```
```
Adding user `username' ...
Adding new group `username' (1001) ...
Adding new user `username' (1001) with group `username' ...
Creating home directory `/home/username' ...
Copying files from `/etc/skel' ...
Enter new UNIX password: ****
Retype new UNIX password: ****
Full Name []: John Doe
Room Number []: 101
Work Phone []: 555-1234
Home Phone []: 555-5678
Other []:
Is the information correct? [Y/n] Y
```
```

### Добавление пользователя в группу

Чтобы добавить существующего пользователя в группу:

```
```bash
sudo adduser username groupname
```
```

Например, чтобы дать пользователю права администратора:

```
```bash
sudo adduser username sudo
```
```

### Просмотр конфигурации по умолчанию

Конфигурационные настройки для `adduser` хранятся в файле `/etc/adduser.conf`. Пример настроек:

```
```bash
# Конфигурация для создания пользователя
DSHELL=/bin/bash
SKEL=/etc/skel
FIRST_UID=1000
LAST_UID=60000
FIRST_GID=1000
LAST_GID=60000
```
```

## Основные опции

- **\*\*`--disabled-login`\*\***: Создает пользователя без возможности входа в систему.

```
```bash
sudo adduser --disabled-login guest
```
```

- `**`--gecos "<INFO>"`**`: Указывает данные о пользователе (например, полное имя) без интерактивного диалога.

```
```bash
sudo adduser --gecos "John Doe,,555-1234" username
```
```

- `**`--disabled-password`**`: Создает пользователя без установки пароля.

```
```bash
sudo adduser --disabled-password newuser
```
```

## Сопутствующие команды

- `**`deluser`**`: Удаляет пользователя и связанные с ним данные.

```
```bash
sudo deluser username
```
```

- `**`usermod`**`: Изменяет параметры существующего пользователя.

## Справка

Для получения дополнительной информации воспользуйтесь справкой:

```
```bash
man adduser
```
```

---

## Komanda passwd

# Руководство по команде ``passwd``

Команда ``passwd`` используется для изменения пароля пользователя в системах на базе Linux. Она позволяет пользователям установить новый пароль, а администраторам — управлять паролями других пользователей.

## Общий синтаксис

```
```bash
passwd [OPTIONS] [USERNAME]
```
```

- `**`USERNAME`**`: Имя пользователя, для которого нужно изменить пароль. Если не указано, пароль изменяется для текущего пользователя.

## Основные опции

### Изменение пароля текущего пользователя

```
```bash
passwd
```
```

Пользователь будет запрошен на ввод текущего пароля, а затем нового пароля.

### Изменение пароля другого пользователя (только для root)

```
```bash
passwd username
```
```

Требуются права суперпользователя.

### Установка срока действия пароля

- `passwd -x <DAYS> username`: Задаёт максимальное количество дней, в течение которых пароль остаётся действительным.

```
```bash
```

```
passwd -x 90 username
```

- `passwd -n <DAYS> username`: Устанавливает минимальное количество дней между изменениями пароля.

```
```bash
```

```
passwd -n 7 username
```

- `passwd -w <DAYS> username`: Указывает количество дней до истечения пароля, когда пользователь начнет получать предупреждения.

```
```bash
```

```
passwd -w 14 username
```

### ### Блокировка и разблокировка учетной записи

- `passwd -l username`: Блокирует учетную запись пользователя (делает пароль недействительным).

```
```bash
```

```
passwd -l username
```

- `passwd -u username`: Разблокирует учетную запись пользователя.

```
```bash
```

```
passwd -u username
```

### ### Удаление пароля

- `passwd -d username`: Удаляет текущий пароль пользователя, что позволяет входить в систему без пароля (не рекомендуется).

```
```bash
```

```
passwd -d username
```

### ### Проверка статуса пароля

- `passwd -S username`: Показывает текущий статус пароля пользователя.

```
```bash
```

```
passwd -S username
```

### ## Примеры использования

#### ### Смена пароля для текущего пользователя

```
```bash
```

```
passwd
```

#### ### Установка пароля для другого пользователя (от имени root)

```
```bash
```

```
sudo passwd alice
```

#### ### Ограничение срока действия пароля

```
```bash
```

```
passwd -x 90 -n 7 -w 14 bob
```

```
### Блокировка учетной записи
```

```
```bash
passwd -l carol
```

```
### Разблокировка учетной записи
```

```
```bash
passwd -u carol
```

```
### Удаление пароля пользователя
```

```
```bash
passwd -d guest
```

```
### Проверка статуса пароля
```

```
```bash
passwd -S dave
```

```
## Справка
```

Для получения дополнительной информации воспользуйтесь справкой:

```
```bash
man passwd
```

---

## Komanda chage

```
# Руководство по команде `chage`
```

Команда `chage` используется для управления политиками истечения пароля пользователя в Linux. Она позволяет администратору задавать сроки действия пароля, минимальное и максимальное время между его изменениями, а также даты, после которых учетная запись становится недействительной.

```
## Общий синтаксис
```

```
```bash
chage [OPTIONS] USERNAME
```

```
- **`USERNAME`**: Имя пользователя, для которого задаются настройки.
```

```
## Основные опции
```

```
### Интерактивный режим
```

```
- **Без указания опций**: Открывает интерактивный диалог для изменения параметров.
```

```
```bash
sudo chage username
```

```
### Настройка срока действия пароля
```

- \*\*`-m DAYS`\*\*: Устанавливает минимальное количество дней между изменениями пароля.

```
```bash
sudo chage -m 7 username
```
```

- \*\*`-M DAYS`\*\*: Устанавливает максимальное количество дней, в течение которых пароль остается действительным.

```
```bash
sudo chage -M 90 username
```
```

- \*\*`-W DAYS`\*\*: Указывает количество дней до истечения пароля, когда пользователь начнет получать предупреждения.

```
```bash
sudo chage -W 14 username
```
```

### ### Учетная запись

- \*\*`-E DATE`\*\*: Устанавливает дату истечения учетной записи в формате `YYYY-MM-DD`.

```
```bash
sudo chage -E 2024-12-31 username
```
```

- \*\*`-I DAYS`\*\*: Устанавливает количество дней после истечения пароля, в течение которых пользователь может войти в систему.

```
```bash
sudo chage -I 30 username
```
```

### ### Просмотр информации о пользователе

- \*\*`-l`\*\*: Показывает текущую информацию о политике пароля пользователя.

```
```bash
sudo chage -l username
```
```

### ## Примеры использования

#### ### Просмотр настроек истечения пароля

```
```bash
sudo chage -l alice
```
```

#### ### Установка минимального и максимального срока действия пароля

```
```bash
sudo chage -m 7 -M 90 bob
```
```

#### ### Установка даты истечения учетной записи

```
```bash
sudo chage -E 2024-12-31 guest
```
```

#### ### Установка периода предупреждений перед истечением пароля

```
```bash
sudo chage -W 14 carol
```
```

...

### ## Полезная информация

- Даты и интервалы задаются в днях.
- Интерактивный режим позволяет визуальнo настроить все параметры, если не указаны опции.

### ## Справка

| Короткий вариант      | Длинный опцион                  | Описание                                                                                                                |
|-----------------------|---------------------------------|-------------------------------------------------------------------------------------------------------------------------|
| -l                    | --list                          | Перечислите информацию об устаревании учетной записи                                                                    |
| -d <i>LAST_DAY</i>    | --lastday <i>LAST_DAY</i>       | Установите дату последней смены пароля на <i>LAST_DAY</i>                                                               |
| -E <i>EXPIRE_DATE</i> | --expiredate <i>EXPIRE_DATE</i> | Установить срок действия учетной записи до истечения срока действия <i>EXPIRE_DATE</i>                                  |
| -h                    | --help                          | Показать помощь для chage команда                                                                                       |
| -I <i>INACTIVE</i>    | --inactive <i>INACTIVE</i>      | Установите учетную запись, чтобы разрешить вход для <i>INACTIVE</i> дней после истечения срока действия пароля          |
| -m <i>MIN_DAYS</i>    | --mindays <i>MIN_DAYS</i>       | Установите минимальное количество дней, прежде чем пароль можно будет изменить на <i>MIN_DAYS</i>                       |
| -M <i>MAX_DAYS</i>    | --maxdays <i>MAX_DAYS</i>       | Установите максимальное количество дней до смены пароля на <i>MAX_DAYS</i>                                              |
| -W <i>WARN_DAYS</i>   | --warndays <i>WARN_DAYS</i>     | Установите количество дней до истечения срока действия пароля, чтобы начать отображать предупреждение. <i>WARN_DAYS</i> |

Для получения дополнительной информации воспользуйтесь справкой:

```
```bash
man chage
```

---

## Komanda usermod

# Руководство по команде `usermod`

Команда `usermod` используется для изменения параметров существующих учетных записей пользователей в Linux. Она позволяет изменять имя пользователя, домашний каталог, группы, пароль, дату истечения учетной записи и другие параметры.

### ## Общий синтаксис

```
```bash
usermod [OPTIONS] USERNAME
```
```

- **\*\*`USERNAME`\*\***: Имя пользователя, параметры которого нужно изменить.

### ## Основные опции

#### ### Изменение имени пользователя

- **\*\*`-l NEW\_USERNAME`\*\***: Устанавливает новое имя пользователя.  
```bash  
sudo usermod -l newname oldname  
```

### ### Изменение домашнего каталога

- `**`-d DIRECTORY`**`: Устанавливает новый домашний каталог пользователя.  
```bash  
sudo usermod -d /new/home/directory username  
```
- `**`-m`**`: Перемещает содержимое старого домашнего каталога в новый.  
```bash  
sudo usermod -d /new/home/directory -m username  
```

### ### Изменение основной группы

- `**`-g GROUP`**`: Назначает новую основную группу пользователя.  
```bash  
sudo usermod -g developers username  
```

### ### Добавление пользователя в дополнительные группы

- `**`-G GROUP1, GROUP2, ...`**`: Устанавливает список дополнительных групп.  
```bash  
sudo usermod -G sudo, docker username  
```
- `**`-a -G GROUP1, GROUP2, ...`**`: Добавляет пользователя в дополнительные группы, сохраняя текущие.  
```bash  
sudo usermod -a -G wheel username  
```

### ### Установка даты истечения учётной записи

- `**`-e DATE`**`: Устанавливает дату истечения учётной записи в формате ``YYYY-MM-DD``.  
```bash  
sudo usermod -e 2025-01-01 username  
```

### ### Отключение учётной записи

- `**`-L`**`: Блокирует учётную запись (запрещает вход).  
```bash  
sudo usermod -L username  
```
- `**`-U`**`: Разблокирует учётную запись.  
```bash  
sudo usermod -U username  
```

### ### Изменение оболочки пользователя

- `**`-s SHELL`**`: Задаёт новую оболочку пользователя (например, ``/bin/bash`` или ``/bin/zsh``).  
```bash  
sudo usermod -s /bin/zsh username  
```

### ## Примеры использования

### ### Переименование пользователя

```
```bash
sudo usermod -l newusername oldusername
```
```

### Перемещение домашнего каталога

```
```bash
sudo usermod -d /new/home -m alice
```
```

### Добавление в группу без удаления текущих

```
```bash
sudo usermod -a -G docker bob
```
```

### Блокировка учётной записи

```
```bash
sudo usermod -L guest
```
```

### Изменение оболочки

```
```bash
sudo usermod -s /bin/bash john
```
```

## Полезная информация

- Изменения в конфигурации пользователя сохраняются в файле `/etc/passwd``.
- Изменения групп — в файле `/etc/group``.

## Справка

| Короткий вариант         | Длинный опцион                     | Описание  |
|--------------------------|------------------------------------|---|
| -c                       | <i>COMMENT</i>                     | Устанавливает значение поля GECOS или поля комментария на <i>COMMENT</i> .                                      |
| -d <i>HOME_DIR</i>       | --home <i>HOME_DIR</i>             | Наборы <i>HOME_DIR</i> в качестве нового домашнего каталога для пользователя.                                   |
| -e<br><i>EXPIRE_DATE</i> | --expiredate<br><i>EXPIRE_DATE</i> | Установите дату истечения срока действия учетной записи на <i>EXPIRE_DATE</i> .                                 |
| -f <i>INACTIVE</i>       | --inactive <i>INACTIVE</i>         | Установите учетную запись, чтобы разрешить вход для <i>INACTIVE</i> дней после истечения срока действия пароля. |
| -g <i>GROUP</i>          | --gid <i>GROUP</i>                 | Набор <i>GROUP</i> как основная группа.   |
| -G <i>GROUPS</i>         | --groups <i>GROUPS</i>             | Установите дополнительные группы в список, указанный в <i>GROUPS</i> .  |
| -a                       | --append                           | Добавить дополнительные группы пользователя к группам, указанным в -G вариант.                                  |
| -h                       | --help                             | Показать помощь для usermod команда.  |
| -l <i>NEW_LOGIN</i>      | --login <i>NEW_LOGIN</i>           | Измените имя пользователя.  |
| -L                       | --lock                             | Заблокируйте учетную запись пользователя.   |
| -s <i>SHELL</i>          | --shell <i>SHELL</i>               | Укажите оболочку входа в учетную запись.  |
| -u <i>NEW_UID</i>        | --uid <i>NEW_UID</i>               | Укажите UID пользователя, который будет <i>NEW_UID</i> .  |
| -U                       | --unlock                           | Разблокируйте учетную запись пользователя.  |



Для получения дополнительной информации воспользуйтесь справкой:

```
```bash
man usermod
```

---

## Komanda userdel

# Руководство по команде `userdel`

Команда `userdel` используется для удаления учетных записей пользователей в Linux. Она также может удалить домашние каталоги и связанные файлы.

## Общий синтаксис

```
```bash
userdel [OPTIONS] USERNAME
```
```

- **``USERNAME``**: Имя пользователя, учетная запись которого должна быть удалена.

## Основные опции

### Удаление только учётной записи

- Без указания дополнительных опций `userdel` удаляет только запись о пользователе из системных файлов (например, из `/etc/passwd`), оставляя его файлы и домашний каталог.

```
```bash
sudo userdel username
```
```

### Удаление домашнего каталога

- **``-r``**: Удаляет домашний каталог пользователя и файлы из него. Также удаляются задания cron и почта пользователя.

```
```bash
sudo userdel -r username
```
```

### Принудительное удаление

- **``-f``**: Принудительно удаляет учетную запись пользователя, даже если он активно использует систему (требуется осторожности).

```
```bash
sudo userdel -f username
```
```

## Примеры использования

### Удаление учетной записи без удаления данных

```
```bash
sudo userdel alice
```
```

### Удаление учетной записи вместе с домашним каталогом

```
```bash
```

```
sudo userdel -r bob
```
```

### Принудительное удаление активного пользователя

```
```bash
sudo userdel -f guest
```
```

## Полезная информация

- После удаления пользователя его идентификатор (UID) и файлы, которые остались на системе, больше не ассоциируются с именем пользователя.
- Файлы, принадлежащие удалённому пользователю, сохраняют его UID.
- Для удаления связанных групп можно использовать команду `groupdel`.

## Справка

Для получения дополнительной информации воспользуйтесь справкой:

```
```bash
man userdel

```

---

## Komanda newgrp

# Руководство по команде `newgrp`

Команда `newgrp` используется для переключения на новую группу в текущем сеансе оболочки. Это позволяет выполнять команды с правами другой группы без необходимости выхода из системы.

## Общий синтаксис

```
```bash
newgrp [GROUPNAME]
```
```

- **\*\*`GROUPNAME`\*\***: Имя группы, на которую вы хотите переключиться. Если имя группы не указано, команда переключится на основную группу текущего пользователя.

## Основные особенности

- Команда запускает новый сеанс оболочки с текущими правами указанной группы.
- Выход из группы происходит автоматически при завершении сеанса оболочки.
- Для переключения на группу пользователь должен быть её членом. Это можно проверить с помощью команды:

```
```bash
groups
```
```

## Примеры использования

### Переключение на другую группу

Если пользователь является членом группы `developers`, он может переключиться на неё:

```
```bash
newgrp developers
```
```

### ### Проверка текущей группы

После выполнения `newgrp` можно проверить текущую группу с помощью команды `id`:

```
```bash
id
```
```

### ### Вернуться к основной группе

Для возврата в исходную группу закройте текущую оболочку командой `exit`:

```
```bash
exit
```
```

### ### Использование без указания группы

Если не указывать имя группы, команда переключится на основную группу пользователя:

```
```bash
newgrp
```
```

### ## Примечания

- Если для группы требуется пароль, команда запросит его.
- Для добавления пользователя в группу используется команда `usermod` с флагом `-a -G`:

```
```bash
sudo usermod -a -G developers username
```
```

### ## Полезная информация

- Использование `newgrp` временно меняет права доступа, что удобно для работы с файлами, принадлежащими другой группе.
- Переключение на группу не сохраняется после завершения текущего сеанса.

### ## Справка

Для получения дополнительной информации воспользуйтесь справкой:

```
```bash
man newgrp
```
```

---

## Komanda chgrp

### # Команда chgrp

Команда `chgrp` (сокращение от "change group") используется в Unix-подобных операционных системах для изменения группы, к которой принадлежит файл или директория.

### ## Синтаксис

```
```sh
chgrp [опции] группа файл...
```

### Опции

- -c, --changes - показывает информацию только о тех файлах, которые действительно были изменены.
- -f, --silent, --quiet - подавляет сообщения об ошибках.
- -v, --verbose - выводит диагностическую информацию для каждого файла.
- -R, --recursive - рекурсивно изменяет группу для всех файлов и директорий внутри указанной директории.
- --dereference - изменяет группу символической ссылки (по умолчанию).
- -h, --no-dereference - изменяет группу самой символической ссылки, а не файла, на который она указывает.
- --reference=RFILe - использует группу файла RFILe вместо указания группы.

### Примеры использования

1. **Изменение группы для одного файла:**  
chgrp новая\_группа имя\_файла
2. **Рекурсивное изменение группы для директории и всех ее содержимых:**  
chgrp -R новая\_группа
3. **Изменение группы для символической ссылки:**  
chgrp -h новая\_группа

---

## Komanda chown

# Команда chown

Команда `chown` (сокращение от "change owner") используется в Unix-подобных операционных системах для изменения владельца и/или группы файла или директории.

## Синтаксис

```
``sh
chown [опции] владелец[:группа] файл...
```

### Опции

- -c, --changes - показывает информацию только о тех файлах, которые действительно были изменены.
- -f, --silent, --quiet - подавляет сообщения об ошибках.
- -v, --verbose - выводит диагностическую информацию для каждого файла.
- -R, --recursive - рекурсивно изменяет владельца и/или группу для всех файлов и директорий внутри указанной директории.
- --dereference - изменяет владельца и/или группу символической ссылки (по умолчанию).
- -h, --no-dereference - изменяет владельца и/или группу самой символической ссылки, а не файла, на который она указывает.
- --reference=RFILe - использует владельца и/или группу файла RFILe вместо указания владельца и/или группы.

### Изменение владельца для одного файла:

chown новый\_владелец имя\_файла

### Изменение владельца и группы для одного файла:

chown новый\_владелец:новая\_группа имя\_файла

### Рекурсивное изменение владельца для директории и всех ее содержимых:

chown -R новый\_владелец директория

### Изменение владельца для символической ссылки:

chown -h новый\_владелец имя\_ссылки

---

## Komanda chmod

# Команда chmod

Команда `chmod` (сокращение от "change mode") используется в Unix-подобных

операционных системах для изменения прав доступа к файлам и директориям.

## Синтаксис

```
```sh
chmod [опции] режим файл...
```

#### Форматы режима

- **Символьный формат:** u (пользователь), g (группа), o (другие), a (все)
  - Операторы: + (добавить), - (убрать), = (установить)
  - Права: r (чтение), w (запись), x (выполнение)

Пример: `chmod u+rw,g+rx,o+r имя_файла`

- **Оctalный формат:** Числовое представление прав доступа

- 4 - чтение (r)
- 2 - запись (w)
- 1 - выполнение (x)

- 7 - rwx
- 6 - rw-
- 5 - r-x
- 4 - r--
- 3 - -wx
- 2 - -w-
- 1 - -w-
- 0 - ---

Пример: `chmod 755 имя_файла`

#### Опции

- -c, --changes - показывает информацию только о тех файлах, которые действительно были изменены.
- -f, --silent, --quiet - подавляет сообщения об ошибках.
- -v, --verbose - выводит диагностическую информацию для каждого файла.
- -R, --recursive - рекурсивно изменяет права для всех файлов и директорий внутри указанной директории.
- --reference=RFIL - использует режим файла RFIL вместо указания режима.

#### Примеры использования

**Установка прав rwx для пользователя, gx для группы, r для остальных:**

```
chmod 755 имя_файла
```

**Добавление права выполнения для всех пользователей:**

```
chmod a+x имя_файла
```

**Рекурсивное изменение прав для директории и всех ее содержимых:**

```
chmod -R 755 директория
```

---

## Papildus pie chmod

# Что такое `setuid` в Linux

`setuid` (set user ID) — это механизм в операционных системах Unix/Linux, который позволяет запускать программу с правами пользователя, отличными от текущего пользователя, который её запустил. Это используется, например, для обеспечения прав доступа к файлам или ресурсам, которые доступны только определённому пользователю.

## Синтаксис

Чтобы установить флаг `setuid` для файла, используется команда `chmod`:

```
```bash
chmod u+s файл
```

# Что такое `setgid` в Linux

``setgid`` (set group ID) — это механизм в операционных системах Unix/Linux, который позволяет запускать программу с правами группы, отличными от текущей группы, которой принадлежит пользователь. Это используется для выполнения программ с правами определённой группы, а не группы, к которой принадлежит пользователь.

## Синтаксис

Чтобы установить флаг ``setgid`` для файла, используется команда ``chmod``:

```
```bash
chmod g+s файл
```

# Что такое ``sticky bit`` в Linux

**\*\*Sticky bit\*\*** — это специальный флаг, который используется в операционных системах Unix/Linux для контроля над удалением файлов в директориях. Когда `sticky bit` установлен на директорию, только владелец файла или суперпользователь (root) могут удалять или переименовывать файлы в этой директории.

## Синтаксис

Чтобы установить `sticky bit` для директории, используется команда ``chmod``:

```
```bash
chmod +t директория
```

---

## Komanda umask

# Команда `umask`

Команда ``umask`` (сокращение от "user file creation mode mask") используется в Unix-подобных операционных системах для задания прав доступа по умолчанию для новых файлов и директорий, создаваемых пользователем.

## Синтаксис

```
```sh
umask [опции] [маска]
```

### Форматы маски

Маска определяет, какие биты прав будут сброшены при создании нового файла или директории. Маска задается в восьмеричном формате и состоит из трех цифр:

- Первая цифра — для прав пользователя.
- Вторая цифра — для прав группы.
- Третья цифра — для прав остальных.

Примеры значений маски

- 022 — файлы создаются с правами 755 (rwxr-xr-x) для директорий и 644 (rw-r--r--) для файлов.
- 027 — файлы создаются с правами 750 (rwxr-x---) для директорий и 640 (rw-r-----) для файлов.
- 077 — файлы создаются с правами 700 (rwx-----) для директорий и 600 (rw-----) для файлов.

Опции

- -p, --preserve — сохраняет текущую маску и выводит её на стандартный вывод.

### Примеры использования

**Установить маску по умолчанию:**

```
umask 022
```

**Показать текущую маску:**

# Hard Links

## Sintakse:

```
ln [target] [link_name]
```

```
# Жесткие ссылки (Hard Links)
```

Жесткие ссылки в Unix-подобных операционных системах позволяют нескольким именам файлов указывать на одни и те же данные на диске. Это означает, что одно и то же содержимое файла может быть доступно через разные пути.

```
## Основные характеристики
```

- **Общие данные:** Все жесткие ссылки на файл указывают на одни и те же данные на диске.
- **Счетчик ссылок:** Когда создается жесткая ссылка, увеличивается счетчик ссылок на файл. Удаление одной из ссылок не удаляет данные, пока существует хотя бы одна ссылка.
- **Идентичный индексный дескриптор:** Жесткие ссылки имеют одинаковый inode (индексный дескриптор), что подтверждает их общие данные.

```
## Синтаксис команды ln
```

Для создания жестких ссылок используется команда `ln`:

```
```sh
ln исходный_файл целевой_файл
```

## Примеры использования

### Создание жесткой ссылки:

```
ln /path/to/original_file /path/to/hard_link
```

### Проверка inode для подтверждения жесткой ссылки:

```
ls -li /path/to/original_file /path/to/hard_link
```

### Удаление жесткой ссылки:

```
rm /path/to/hard_link
```

## Преимущества и недостатки

### Преимущества

- **Экономия места:** Несколько имен файлов указывают на одни и те же данные, что экономит пространство на диске.
- **Независимость:** Удаление одного из имен файла не удаляет данные, пока существует хотя бы одна жесткая ссылка.

### Недостатки

- **Ограничения на файловую систему:** Жесткие ссылки могут создаваться только в пределах одной файловой системы.
- **Сложность управления:** Множество ссылок на один и тот же файл может усложнить управление файлами.

---

# Symbolic Links

```
# Символические ссылки (Symbolic Links)
```

Символические ссылки (symlinks или soft links) в Unix-подобных операционных системах представляют собой специальные файлы, которые содержат путь к другому файлу или директории. В отличие от жестких ссылок, символические ссылки могут указывать на файлы или директории в других файловых системах и устройствах.

## ## Основные характеристики

- **\*\*Гибкость:\*\*** Символические ссылки могут указывать на файлы или директории, находящиеся в любой файловой системе или на любом устройстве.
- **\*\*Содержат путь:\*\*** Символическая ссылка хранит путь к целевому файлу или директории.
- **\*\*Легкость создания и удаления:\*\*** Символические ссылки легко создавать и удалять, не затрагивая сами данные.

## ## Синтаксис команды ln

Для создания символических ссылок используется команда `ln` с опцией `-s`:

```
```sh
ln -s источник целевой_файл
```

### Примеры использования

#### Создание символической ссылки на файл:

```
ln -s /path/to/original_file /path/to/symlink
```

#### Создание символической ссылки на директорию:

```
ln -s /path/to/original_directory /path/to/symlink
```

#### Проверка символической ссылки:

```
ls -l /path/to/symlink
```

#### Удаление символической ссылки:

```
rm /path/to/symlink
```

## Преимущества и недостатки

### Преимущества

- **Гибкость:** Символические ссылки могут указывать на файлы и директории в разных файловых системах.
- **Простота:** Легко создавать и удалять, не затрагивая исходные файлы.

### Недостатки

- **Сломанные ссылки:** Если исходный файл удален или перемещен, символическая ссылка станет недействительной (сломается).
  - **Низкая производительность:** Обращение к файлам через символические ссылки может быть немного медленнее, чем через обычные или жесткие ссылки.
-