**Занятие 1. Определение CTF, основы соревнований. Основы командной строки в Unix.**

Capture the Flag (CTF) — это упражнение в компьютерной безопасности, в котором «флаги» тайно прячутся в преднамеренно уязвимых программах или веб-сайтах.

Существует два формата CTF:

* Jeopardy. Участвующие команды должны выполнить как можно больше испытаний с разным количеством очков из заданной категории.
* Атака-защита. Соревнующиеся команды должны защищать свои уязвимые компьютерные системы, атакуя противников.

##### **Подробнее про CTF**

Описание проводится на примере Positive Hack Days CTF — международные соревнования по защите информации, которые проводятся по игровому принципу Capture the Flag. Несколько команд в течение заранее отведенного времени защищают свои сети и атакуют чужие. Основная задача участников — выявлять уязвимости в системах противников и получать доступ к секретной информации (флагам), при этом обнаруживая и устраняя подобные уязвимости в своей системе.

Квалификационный этап соревнования (PHDays CTF Quals) относится к типу task-based CTF, то есть команды должны решать задания и получать за них очки. Задания могут относиться к одной из следующих категорий:

* Forensic — компьютерно-криминалистическая экспертиза,
* Reverse (reverse engineering) — анализ бинарного кода,
* Pwn — эксплуатация уязвимостей,
* Admin — навыки администрирования,
* Network — знание сетевой инфраструктуры и протоколов,
* Crypto — криптография,
* Stegano — стеганография,
* PPC (professional programming and coding) — олимпиадное программирование,
* Web — поиск и использование веб-уязвимостей,
* Misc — разное.

##### **Сравнение Windows системы и UNIX**

**UNIX: Концепция "Toolbox"**

Поскольку UNIX разрабатывалась инженерами и для инженеров, в ее основу была положена концепция toolbox (ящик с инструментами). Что это значит? Это значит, что при создании софта и встроенных утилит для UNIX не делали универсальные программы, каждая из которых выполняла бы внутри себя все, необходимые пользователю действия, а для каждой небольшой задачи создавалась своя утилита, которая выполняла свою задачу, только одну, но делала это хорошо. Дело пользователя было при помощи набора этих утилит выполнить операции, которые ему нужно сделать.  
  
При этом из этого набора утилит можно составлять цепочки и последовательности действий, что позволяет легко автоматизировать рутинные, часто повторяющиеся операции.  
  
Для того, чтобы утилиты могли обмениваться между собой результатами своей работы, в качестве носителя информации был выбран текстовый файл. Для обмена информацией между утилитами были изобретены "pipes" (трубы). При помощи "труб" информация с выхода одной команды может быть передана на вход второй, та ее обрабатывает, выдает свою информацию на выход, которая может быть передана на вход третьей и так далее.  
  
В общем, в результате UNIX позволяет пользователю легко создавать простые программные комплексы, выполняющие повторяющиеся действия как по команде пользователя, так и в автономном режиме.  
  
Такой подход имеет как плюсы, так и недостатки. С одной стороны он дает больший контроль над системой, гибкость в настройке, но при этом повышается порог вхождения в систему, или говоря простыми словами, прежде, чем что нибудь сделать, как правило, нужно изучить основы.

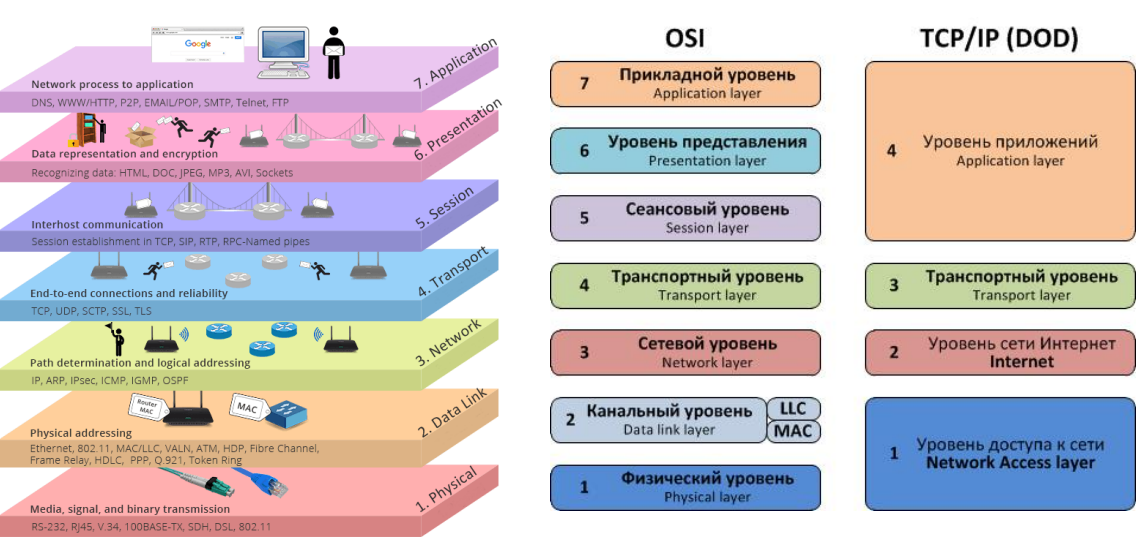
**Windows: Концепция "Тостер"**

В Windows доминирует другая концепция. Эта концепция — максимально облегчить вхождение пользователя в задачу. Программы в Windows как правило большие, на каждое действие есть пункт в меню или иконка. В системы программы связываются как правило с большим трудом.  
  
Ухудшает ситуацию о построением комплексов на базе Windows то, что большинство программ — коммерческие и используют свои, бинарные и как правило закрытые форматы данных и файлов. Такой подход превращает компьютер в устройство, которое может выполнять ограниченный изготовителем ПО набор функций, в пределе в этакий своеобразный "тостер", который выполняет только то, что задумал его изготовитель.  
  
Плюс такого подхода — легкость вхождения неподготовленного пользователя. Минус — то, что обманутый кажущейся легкостью пользователь вообще не хочет ничему учиться и не выполнять необходимых действий. На поводу идут и производители софта. Это одна из причин такого обилия документов отформатированных пробелами, пренебрежения безопасностью и как следствие вирусных эпидемий.

##### **Linux дистрибутивы**



Дистрибутив Linux - это комплект взаимосогласованного программного обеспечения, включающий ОС на базе ядра Linux, системное программное обеспечение и приложения, а также систему управления программным обеспечением • Большая часть программного обеспечения в дистрибутиве является свободным (распространяется по свободным лицензиям). Минимальным объектом дистрибутива является программный пакет. Программные пакеты могут собираться как в двоичном виде, так и в исходных кодах. Дистрибутивы Linux размещаются в специальном хранилище – репозитории (в локальном или сетевом)



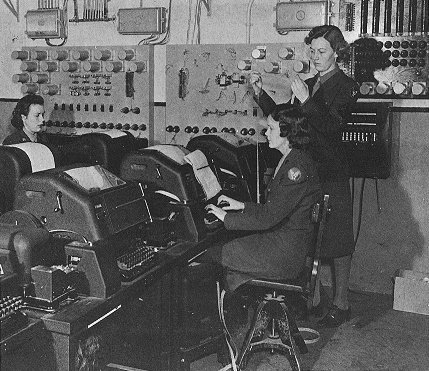
##### **Основы командной строки**

Давайте для начала разберемся, что такое Terminal, Shell, Console, чем отличается Terminal Emulator от обычного Terminal и почему он так назван. Информации об этом написано уже довольно много, поэтому ничего нового вы здесь не услышите. Почти вся информация здесь была взята из интернета, ссылки приведу в конце статьи. Кто уже знает, что все эти вещи обозначают, может смело пропускать данный раздел.

**Terminal**

Terminal (терминал) — это комбинация дисплея и клавиатуры, то есть физическое устройство. До того, как терминалы стали именно данной комбинацией, они являлись неким устройством под названием teleprinter (teletype, teletypewriter или TTY сокращенно), то есть комбинацией принтера и клавиатуры. Обычно несколько терминалов подключались к одному и тому же компьютеру. Таким образом возможно было работать нескольким пользователям за одним и тем же компьютером, причем каждому выделялась своя сессия, независимая от других. Терминал был назван так потому, что он находился на конце терминального кабеля (terminal end).

Это Teletype:



А это Terminal:



**Console**

Console (консоль) — терминал, который подключен напрямую к компьютеру. Дело в том, что большинство терминалов были соединены неявно, но хотя бы один был подключен напрямую к компьютеру. Консоль было разрешено использовать строго определенному кругу лиц, так как она позволяла настраивать компьютер.

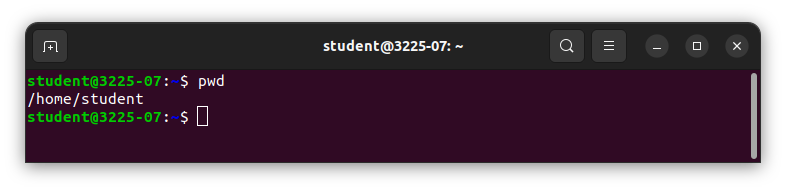
**Shell**

Если предыдущие два представляют собой физические устройства, то данное определение относится исключительно к программному обеспечению.

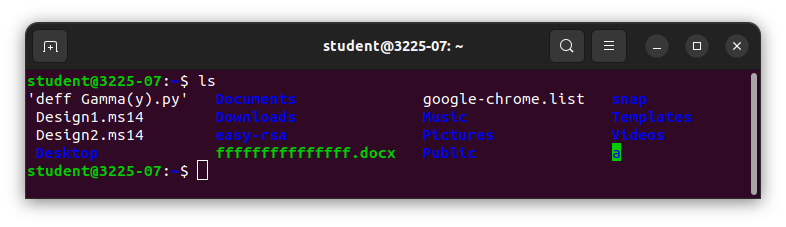
Shell — это command line interpreter. Главное предназначение — запускать другие программы. Существует большое количество различных Shell'ов. Самым распространенным является Bash (от англ. Bourne Again SHell, что как подсказывает [Википедия](https://ru.wikipedia.org/wiki/Bash), является каламбуром для «Born again» Shell, то есть «возрождённый» Shell). Другие примеры: Dash (легковесный Shell, доступен, если запустить бинарник по адресу /bin/sh), Zsh.

##### **Работа с директориями**

**pwd**— полный путь до папки, в которой находитесь сейчас:

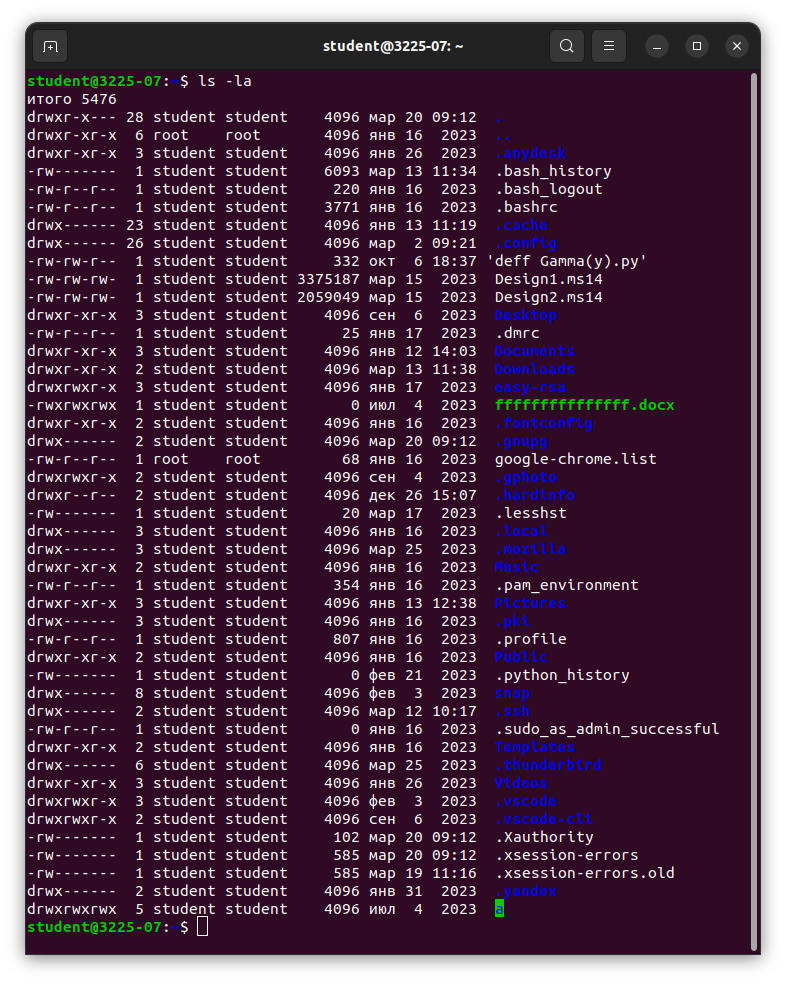


**ls**— просмотр содержимого открытой папки:

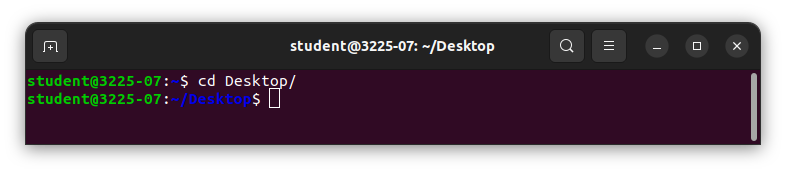


У команд есть ключи — например, можно написать команду ls с ключом -a и увидеть список всех файлов, в том числе скрытых. Как правило, у всех команд терминала есть такие дополнительные ключи. Их может быть довольно много — например, как в списке ключей для команды ls.

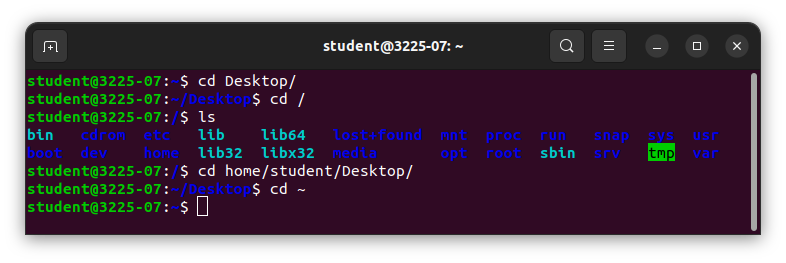
**ls -la**выводит дополнительную информацию (о правах, владельце, дате редактирования и размере).



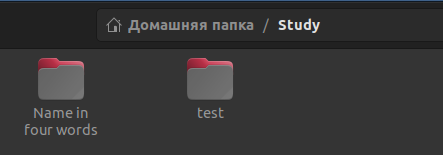
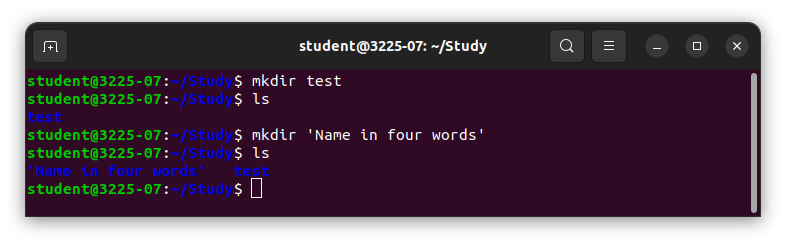
**cd**название папки прокладывает навигацию к любой папке. Это происходит примерно по той же схеме, что в проводнике или Finder.



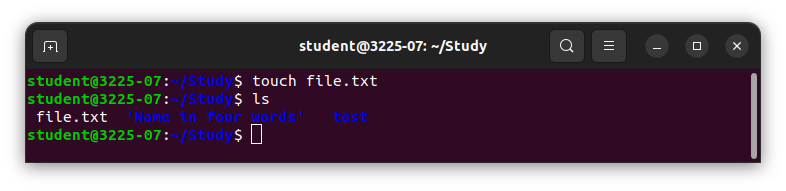
|  |  |
| --- | --- |
| Символы | На что направляет |
| **.** | Текущая директория |
| **..** | Директория, находящаяся выше |
| **/** | Корневая директория |
| **~** | Домашняя директория |



**mkdir**название директории — создаёт папку. Если у директории в названии присутствует несколько символов, название пишется в кавычках:



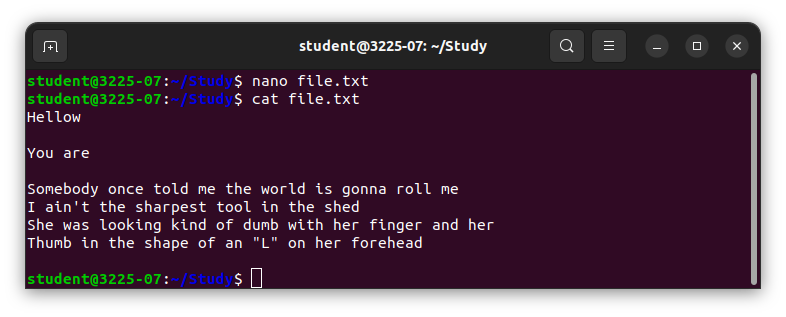
**touch название файла** — создаёт файл, а также указывает расширение:



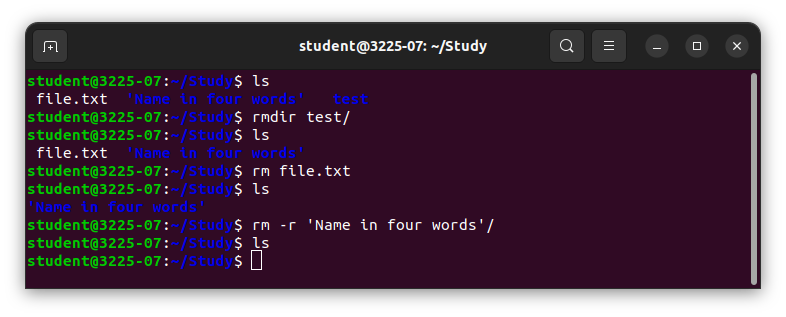
**nano название**— текстовый редактор файла:



**cat название**— построчно выводит содержимое файла:

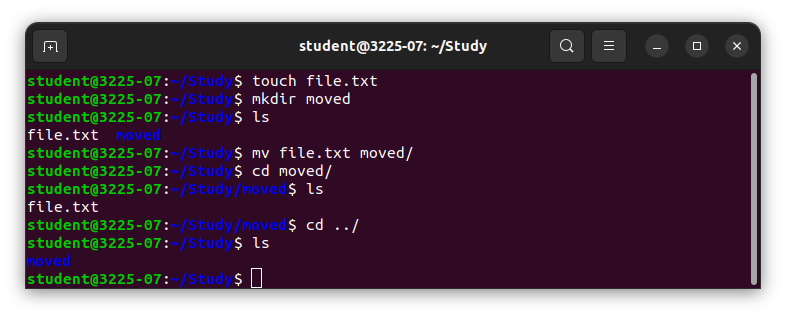


**rmdir название** — удаляет директорию. Команда **rm название** удаляет файл. Также можно удалять директорию при помощи **rm -r название** :



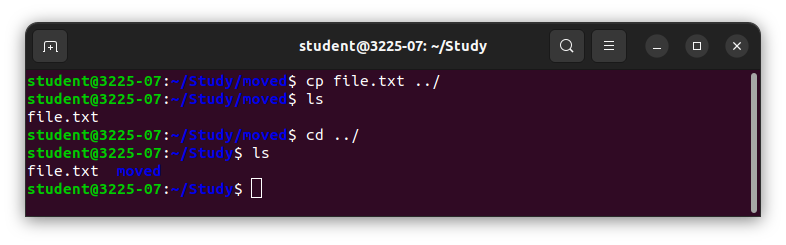
**mv** - перемещение файлов

Команда mv перемещает файл в новое место. Она также может использоваться для переименования файлов. Чтобы переместить файл в другую папку нужно указать путь к ней.



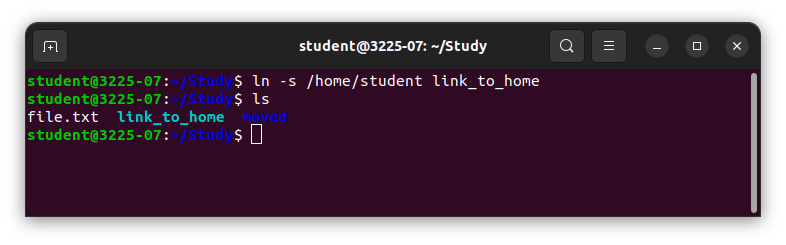
**cp** - копирование файлов

Эта cp и mv - это похожие команды linux для работы с файлами. Они работают аналогичным образом, только исходный файл остается на своем месте. Вы также можете рекурсивно скопировать всю папку с помощью команды cp -r. Эта команда скопирует всю папку вместе со всеми файлами и вложенными папками в новое место.



**ln** - создание ссылок

Утилита ln позволяет создавать жесткие и символические ссылки на файлы или папки. Для создания символической ссылки используется опция -s. Например, мы создадим ссылку на папку с загрузками:



|  |  |
| --- | --- |
| **Команда** | **Что делает** |
| **pwd** | Выводит текущую директорию |
| **ls** | Выводит список содержимого |
| **cd** | Переход в другую директорию |
| **mkdir** | Создание директории |
| **touch** | Создание файла |
| **ln** | Создание ссылки |
| **nano** | Редактор файлов |
| **cat** | Выводит содержимое файла построчно |
| **mv** | Перенос / переименовывание файла |
| **cp** | Копирование файла, директории |
| **rm** | Удаление файла / директории |
| **rmdir** | Удаление директории |
| **man** | Описание команды |

##### **Перенаправление ввода и вывода**

Ввод и вывод распределяется между тремя стандартными потоками:

stdin — стандартный ввод (клавиатура),

stdout — стандартный вывод (экран),

stderr — стандартная ошибка (вывод ошибок на экран).

< file — использовать файл как источник данных для стандартного потока ввода.

> file — направить стандартный поток вывода в файл (перезапись)

2> file — направить стандартный поток ошибок в файл (перезапись)

>>file — направить стандартный поток вывода в файл (добавление)

2>>file — направить стандартный поток ошибок в файл. (добавление)

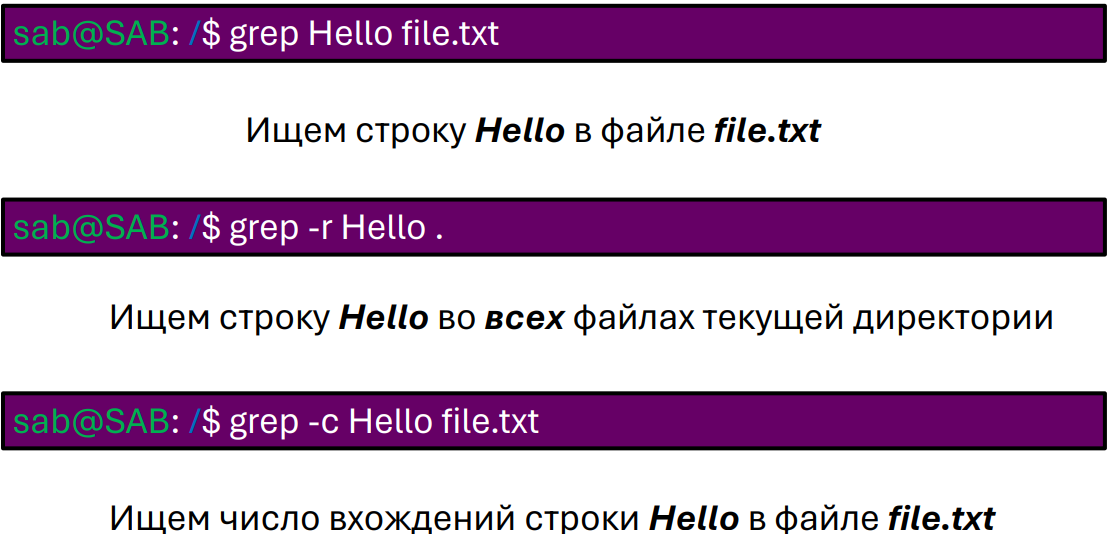
&>file или >&file — направить с.п. вывода и с.п. ошибок в файл.

##### **Команды поиска**

**Grep** (global regular expression printer) – утилита командной строки, позволяющая производить поиск строки в файле.

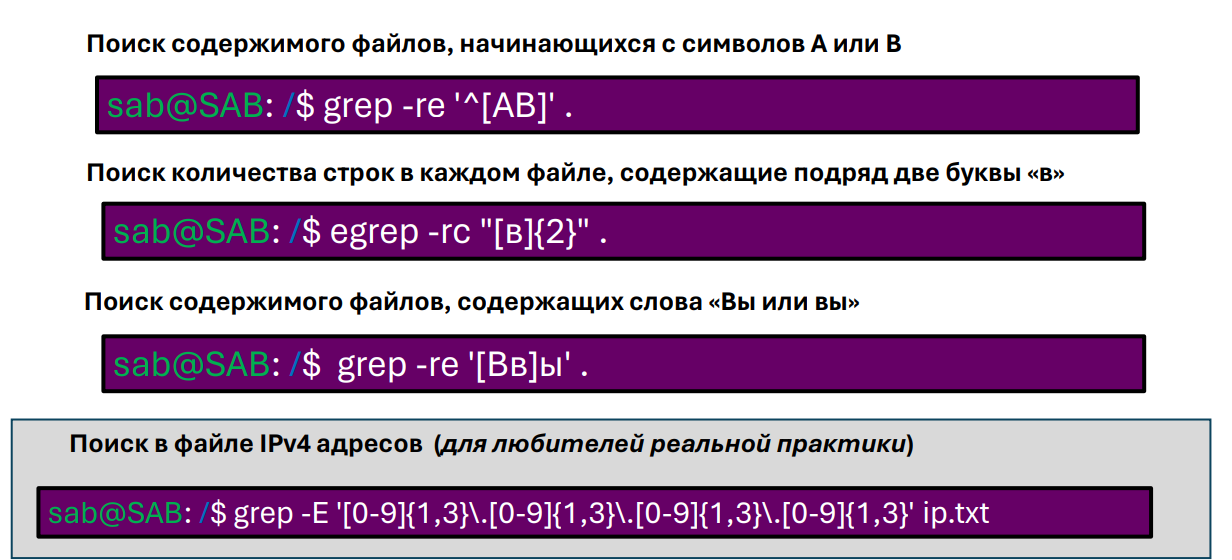


|  |  |
| --- | --- |
| **Ключ** | **Описание** |
| **-c** | Выдает только количество строк, содержащих выражение. |
| **-h** | Скрывает вывод названия файла, в котором было обнаружено вхождение. Используется при поиске по нескольким файлам. |
| **-i** | Игнорирует регистр символов при поиске. |
| **-l** | Выдает только имена файлов, содержащих сопоставившиеся строки |
| **-n** | Выдает перед каждой строкой ее номер в файле (строки нумеруются с 1). |
| **-s** | Скрывает выдачу сообщений о не существующих или недоступных для чтения файлах. |
| **-v** | Выдает все строки, за исключением содержащих выражение. |
| **-E** | Поиск с использованием регулярных выражений |
| **-o** | Вывод только обнаруженных символов |
| **-r** | Рекурсивный поиск |

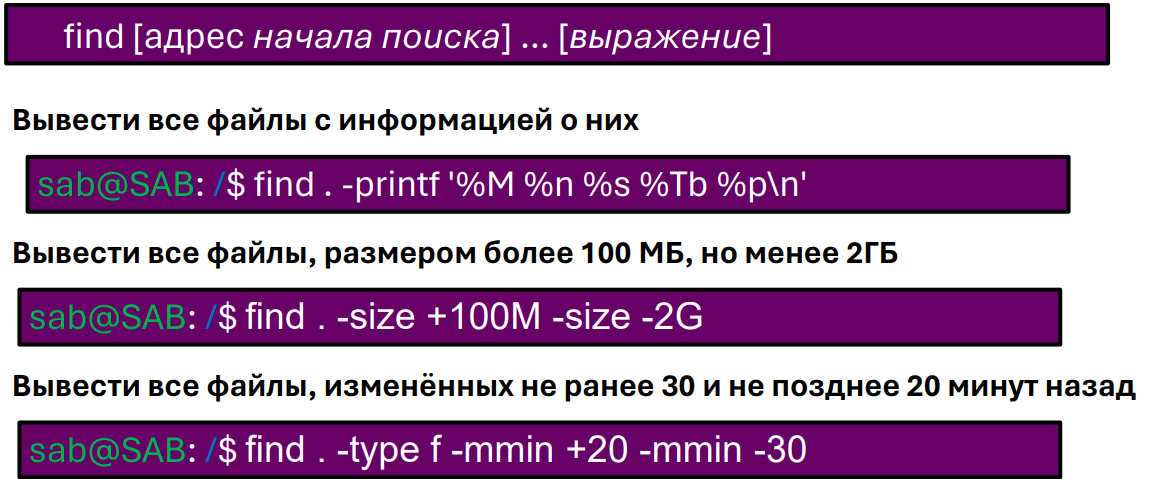


##### **Регулярные выражения**

|  |  |
| --- | --- |
| **Метасимвол** | **Описание** |
| **\** | начало буквенного спецсимвола |
| **^** | указывает на начало строки |
| **$** | указывает на конец строки |
| **\*** | указывает, что предыдущий символ может повторяться 0 или больше раз |
| **+** | указывает, что предыдущий символ должен повторится больше один или больше раз |
| **?** | предыдущий символ может встречаться ноль или один раз |
| **{n}** | указывает сколько раз (n) нужно повторить предыдущий символ |
| **{N,n}** | предыдущий символ может повторяться от N до n раз |
| **.** | любой символ кроме перевода строки |
| **[az]** | любой символ, указанный в скобках |
| **x|y** | символ x или символ y |
| **[^az]** | любой символ, кроме тех, что указаны в скобках |
| **[a-z]** | любой символ из указанного диапазона |
| **[^a-z]** | любой символ, которого нет в диапазоне |
| **[[:alpha:]]** | является алфавитным символом |
| **[[:digit:]]** | является числом |



**find** – утилита, с помощью которой возможно найти файл по его имени



**cut** – утилита, с помощью которой возможно вырезать символы из каждой переданной ей строки

**tr** (translate) – утилита командной строки, позволяющая посимвольно обрабатывать текст

**wc** (word count) – утилита командной строки, выводящая число переводов строк, слов и байт для каждого указанного файла и итоговую строку, если было задано несколько файлов.

**uniq** – утилита командной строки, предназначена для поиска одинаковых строк в массивах текста.

**awk** – скриптовый язык построчного разбора и обработки входного потока

##### **Полезные команды для повседневки**

**hexdump** — показывает шестнадцатеричное представление данных, поступающих на стандартный поток ввода.

**cat** — считывает данные со стандартного потока ввода и передает их на стандартный поток вывода.

**sudo** - запуск программы от имени других пользователей, а также от имени суперпользователя.

**bc** – калькулятор

**python3** – среда разработки Python

**sort** – сортировка переданных значений (-r – обратный порядок)

##### **Прикольные команды**

https://losst.pro/prikolnye-komandy-linux?ysclid=lu027tik2g297486326

##### **Задания**

**Задание 1.**

1. Перейдите в директорию lab1.

2. Оставаясь в директории lab1, создайте в каталоге poems/English файл, содержащий текст вашего любимого стихотворения отечественного автора. Название файла должно соответствовать названию стихотворения. Внутри файла, перед текстом произведения укажите название и автора.

3. Перенесите созданный файл из директории English в Russian.

4. Создайте в директории English каталоги, содержащие названия веков и распределите по ним расположенные в ней стихотворения.

**Задание 2.**

1. Произведите поиск всех стихотворений, названия которых содержит только кириллицу

2. Найдите все файлы с расширением jpeg

3. Найдите все файлы, которые были изменены за последние 20 минут

4. Найдите все файлы, объемом больше 500 Кб

**Задание 3.**

1. Вычислите у скольких стихотворений вместо названия стоят “\*\*\*”

2. Выведите напротив каждого файла сообщение о том, содержит ли он восклицательный знак

3. Рассчитайте, сколько раз в тексте стихотворений встречается предлог «на»

4. Вычислите самое часто встречающееся слово в монологе Гамлета