ПРАКТИЧЕСКАЯ РАБОТА №10

Тема: Архитектура Linux-based систем. Работа в Debian с bash. Базовые команды.

Цель: на примере дистрибутива Debian 12.10 рассмотреть архитектуру операционной системы на основе ядра Linux. Получить базовые навыки работы с дистрибутивом Debian через CLI и одним из редакторов на выбор: nano или vim

## Теоретические сведения

В ходе практической работы №2 вы устанавливали сразу две виртуальные машины: первая была с ОС Windows 10, а вторая – с Debian 12.10.

Теперь мы приступаем к работе с дистрибутивом Debian, основанном на ядре Linux и являющимся наиболее стабильным дистрибутивом в мире GNU/Linux систем.

Архитектура большинства дистрибутивов является устоявшейся и регламентируется рядом стандартов, включая стандарт графического окружения [www.freedesktop.org](http://www.freedesktop.org/) и Linux Standard Base, расширяющий стандарты Unix.

## Описание типового дистрибутива GNU/Linux

*(на основе материала:* [*https://habr.com/ru/articles/531872/*](https://habr.com/ru/articles/531872/) *)*

# Загрузчик

Сеанс операционной системы начинается с *загрузчика*, как театр с вешалки. Дефолтным загрузчиком сегодня является **GNU GRUB**, известный также, как **GRUB 2**. По-прежнему доступна первая ветка, называемая теперь

«**GRUB Legacy**». Другой загрузчик с давней историей — **Syslinux**.

Задача загрузчика — инициализировать ядро Linux. Для этого, в общем случае, нужно знать, где ядро лежит, и уметь прочитать это место (раздел Ext4

на диске). Ядру в помощь загрузчик обычно так же подтягивает начальный образ загрузки.

**GRUB** умеет много прочего, например: построения сложных меню, организовывать чейнлоадинг (англ. chain – цепь, loading – загрузка), т.е. цепочечную загрузку других загрузчиков (Windows Boot Manager, например). **GRUB** имеет конфигурационный синтаксис, отдалённо напоминающий shell, и расширяется с помощью модулей.

# Ядро

Могучий **Linux** («не оставляй нас, монолит!»). Ядро операционной системы, созданное, чтобы работать с POSIX-совместимыми окружениями. Обычно лежит в /boot/ и содержит в названии слово vmlinuz, где «vm» напоминает нам о поддержке виртуальной памяти, а «z» указывает, что файл сжат.

В рамках одного дистрибутива может поддерживаться несколько вариантов ядра, например:

* **Mainline** («основное»);
* **LTS** (с расширенной поддержкой);

## generic

* **rt** (со специальными патчами для поддержки исполнения в режиме реального времени);
* с различными патчами для повышения производительности или защищённости (**zen**, **hardened** etc);
* **libre** (очищенное от проприетарных блобов ядро, ожидаемо поддерживающее мало оборудования).
* совсем экзотичные варианты с не-Linux ядром по типу: Debian GNU/Hurd (с ядром **GNU Hurd**)

Debian GNU/kFreeBSD (с ядром **FreeBSD** соответственно).

Это уже, конечно, не GNU/Linux.

# Начальный образ загрузки

Начальный образ загрузки известный также как: **initrd** и **initramfs**. Представляет собой архив с образом файловой системы, развёртываемой в оперативную память в начале процесса загрузки. Несёт в себе различные драйверы и пакетные файлы (скрипты), позволяющие инициализировать оборудование и смонтировать файловые системы.

Содержимое начального образа загрузки зависит от версии ядра и потребностей пользователя (кто-то использует ZFS, а у кого-то корень зашифрован LUKS). Поэтому образ не поставляется в дистрибутивах. В дистрибутивах поставляются фреймворки для создания начальных образов по мере необходимости. Так, обычно создание свежего образа инициируется при обновлении ядра. Вот несколько популярных фреймворков:

* **initramfs-tools** — детище Debian.
* **Dracut** (произносится как драйкат ) — в RHEL и производных (CentOS, Scientific Linux etc.). Наиболее гибкий и современный инструмент из перечисленных.
* **mkinitcpio** поставляется в Archlinux, хотя мейнтейнеры [подумывают о](https://lists.archlinux.org/pipermail/arch-dev-public/2019-May/029570.html)

[**Dracut**](https://lists.archlinux.org/pipermail/arch-dev-public/2019-May/029570.html), который уже включён в репозиторий и установочные образы.

* **make-initrd** — свой путь у замечательного отечественного дистрибутива Alt Linux.

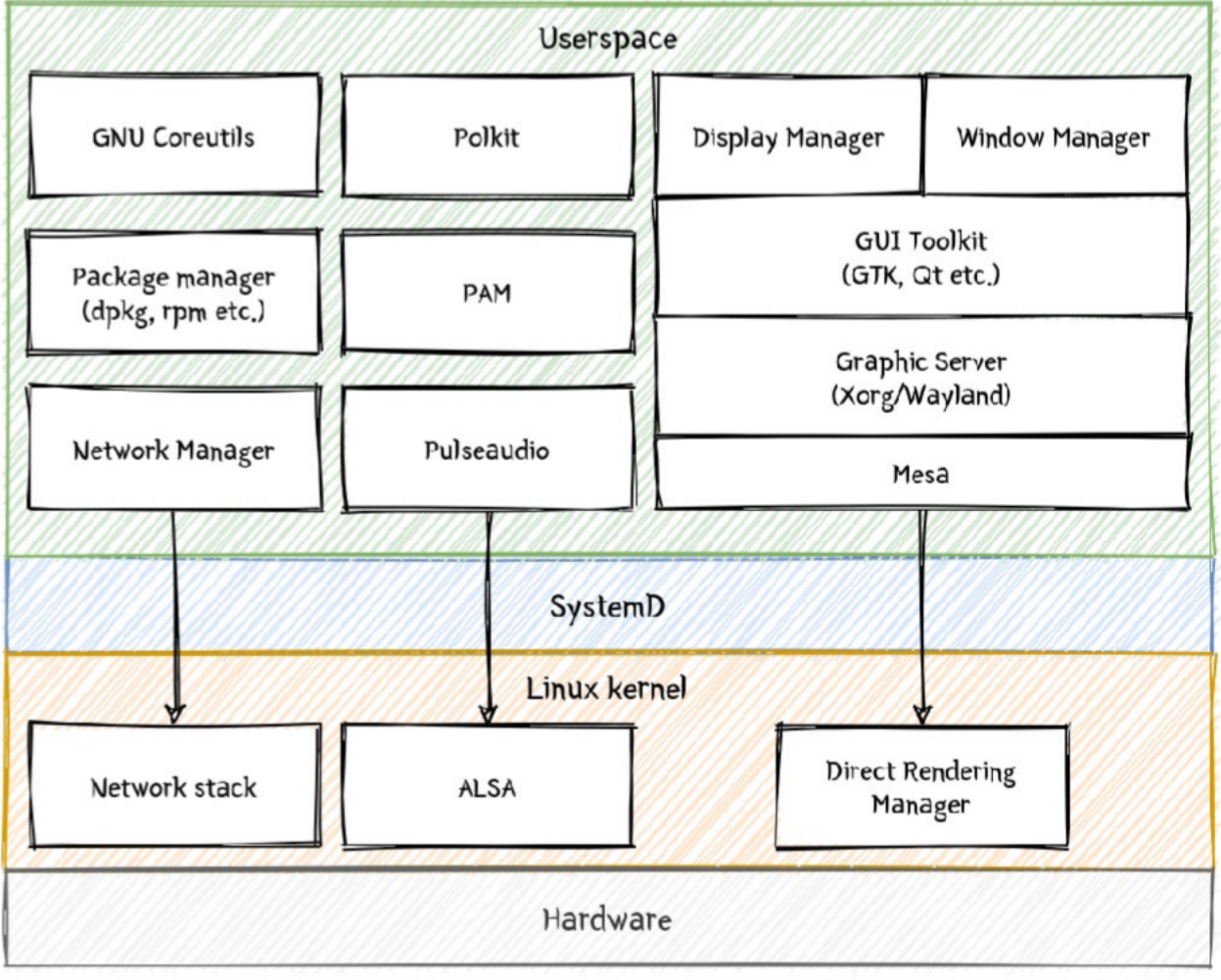
Тут же можно упомянуть **Plymouth**, размещаемый в начальном образе. Это заставка (сплэш-скрин), позволяющая заменить вывод ядра при загрузке (куча кодов с [OK] от инициализируемых программ) на произвольную анимированную картинку, например логотип дистрибутива, что принято в

«дружелюбных к пользователю» дистрибутивах типа Ubuntu и Fedora.

Вставить архитектуру систем, основанных на ядре Linux и программах фонда GNU (GNU/Linux).

Чтобы ознакомиться с понятием bash в форме видеоролика:

(Понятие bash) <https://rutube.ru/video/3ba5c39a32f3b360187b334d84d93bee> (Команды bash) <https://rutube.ru/video/ea80561d113549f14c608859bc1a9a6b>



Описать проект GNU

Графический интерфейс пользователя (GUI) возможно более адаптирован для неподготовленного пользователя, который, не зная принципов работы операционной системы, может в нем интуитивно так или

иначе освоиться. Для работы в режиме командной строки (CLI) человек должен изначально владеть знаниями, по крайней мере, **как пользоваться справочной информацией.**

Однако, у CLI можно усмотреть одно психологическое преимущество – это отсутствие отвлекающих элементов. Человек ведет диалог с машиной, при этом параллельно машина не мигает лампочками где-то внизу, не выкидывает окна с большим количеством букв и слов, не завлекает взгляд красочными закруглёнными кнопками и прочим. Когда ничто не отвлекает от основной задачи, человеку легче сосредоточиться на ее решении. И кроме того, он меньше устает. Да – меньше устает: сравните час работы в GUI и CLI, возможно, вы заметите разницу. Многие вещи мы воспринимаем неосознанно. Вероятнее всего, требование постоянно изучать меняющийся GUI перегружает эту неосознаваемую часть нас и мозг быстро устает.

Еще одно преимущество командной строки – более глубокое понимание принципов работы. Нажатие на кнопку графического интерфейса почему-то не ведет к ярко выраженной ассоциации ― дать команду компьютеру, в отличии от того, когда пользователь вводит команду в терминал.

В большинстве дистрибутивов GNU/Linux по умолчанию установлена командная оболочка Bash и эмулятор терминала в виде программы Konsole или иной. Между всеми командными оболочками, соответствующими стандарту POSIX Unix-подобных операционных систем, разница небольшая.

Bash запускается в текстовом режиме или приложении "Терминал" и работает интерактивно. Это значит, что он ожидает ввод пользователя, далее интерпретирует введенную строку как команду, которую надо выполнить. Команды бывают разными: запуск программы, операции с файлами, выполнение скрипта на языке самого Bash и др.

Unix разрабатывалась как многопользовательская операционная система, доступ к которой пользователи осуществляли через терминалы. Например, в здании был один мощный компьютер, а в рабочих помещениях

специалистов располагались терминалы – точки доступа к ресурсам общей машины, состоящие из монитора и клавиатуры.



Представьте, что подключаетесь по протоколу SSH к веб-серверу, на котором размещаете файлы сайта. Работая в ОС GNU/Linux и запустив программу "Терминал", вы даете Bash'у команду на подключение к удаленной машине под конкретным именем пользователя. Сервер отвечает вам и требует ввести пароль этого пользователя. Все потому, что пользоваться данным компьютером-сервером может далеко не один человек.

На локальном компьютере в текстовом режиме Linux эмулируется удаленный доступ к машине через терминал. Здесь Bash первым делом предлагает вам ввести логин, после чего – пароль. При вводе пароля на экране ничего не отображается. Так сделано специально, чтобы нельзя было подсмотреть даже количество вводимых символов. Если введенная пара логин/пароль соответствуют какому-либо ранее зарегистрированному в системе пользователю, то вы увидите приглашение командной строки Bash'а на ввод команды. В случае ошибки потребуется заново вводить логин/пароль.

В графическом режиме приложение "Терминал" автоматически "берет" данные запустившего его пользователя и передает Bash'у. Таким образом, вы уже оказываетесь представленными системе и первое, что видите, – приглашение командной строки.

Приглашение командной строки имеет относительно стандартный синтаксис и выглядит примерно так:

имя\_пользователя@имя\_компьютра:текущий\_каталог знак

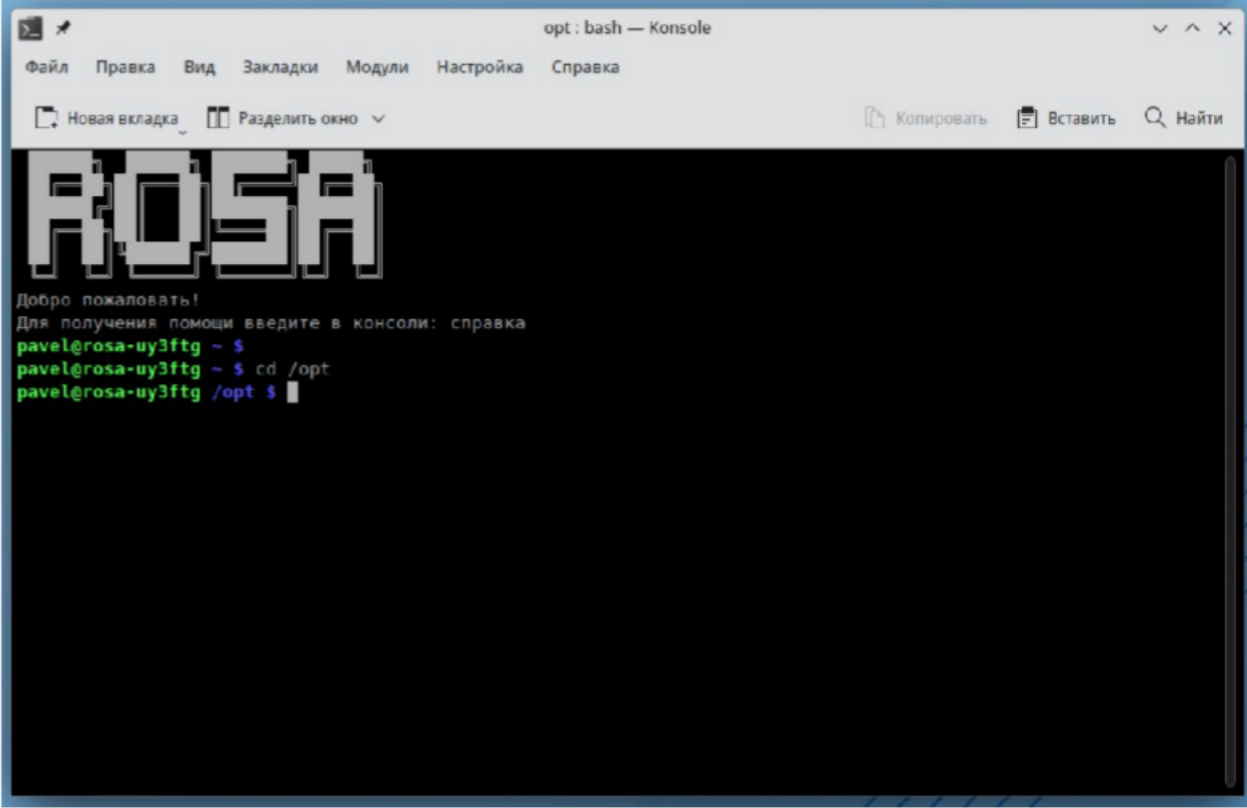


Рисунок 1

На рисунке 1 изображён приложение-эмулятор терминала Konsole, реализующего доступ к CLI интерфейсу *pavel* – имя пользователя, *rosa-uy3ftg* – имя компьютера. Символ тильды ~ в Linux-системах – это сокращенное обозначение домашнего каталога пользователя (/home/user). Именно эту директорию Bash делает текущей по умолчанию. Знак доллара $ обозначает, что мы работаем под непривилегированным пользователем. Если находиться в системе под суперпользователем (root), вместо доллара будет знак решетки #.

После приглашения автоматически ставится пробел и ожидается ввод команды. На рисунке выше сначала был просто нажат *Enter*.

Во второй строке(рис.1) была введена команда перехода в другую директорию cd. В третьей строке мы видим результат ее выполнения – текущим каталогом стал /opt, что отображено в приглашении.

Существует множество команд, которые можно отдавать в Bash. В основном это команды запуска различных программ, которые находятся в каталогах /bin*,* /usr/bin и ряде других. Проверяемые каталоги записаны в переменной $PATH самой оболочки:

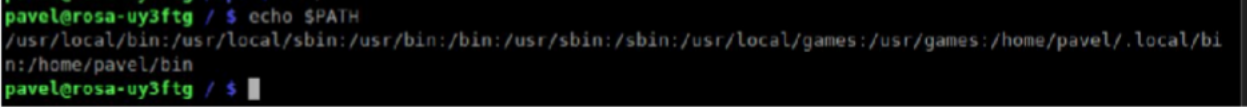


Рисунок 2

Здесь с помощью команды echo мы просим Bash выдать нам значение переменной $PATH. В выводе каталоги отделены друг от друга с помощью двоеточия.

Можно посмотреть, какие программы установлены в том или ином каталоге, после чего запустить одну из них:

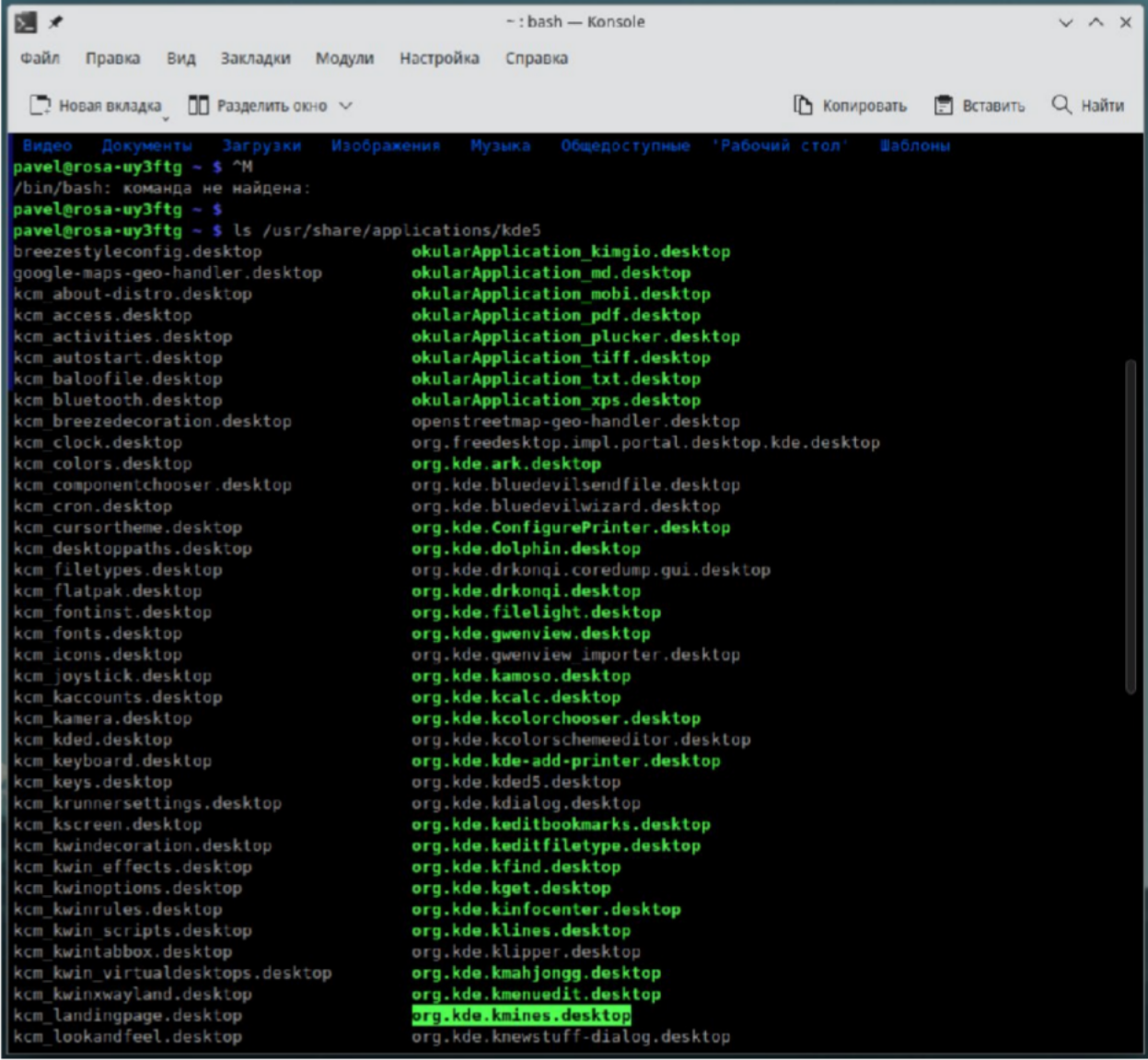


Рисунок 3

В операционной системе Linux пользователю доступны два режима работы: графический и текстовый. В текстовом режиме недоступны возможности графических интерфейсов: рисование окон произвольной формы и размера, поддержка миллионов цветов, отрисовка изображений. Все возможности текстового режима ограничены набором текстовых и псевдографических символов и несколькими десятками базовых цветов. Тем не менее в Linux в текстовом режиме можно выполнять практически любые действия в системе (кроме тех, которые требуют непосредственного просмотра изображений). Текстовый режим в Linux — это полнофункциональный способ управления системой благодаря интерфейсу

командной строки. В Linux существует огромное множество программ (включая даже игры), предназначенных для работы в текстовом режиме.

Для перехода в текстовый режим требуется выполнить комбинацию Ctrl+Alt+Fn-<3-12>. В таком случае, у вас есть возможность открыть до 10 сессий терминала в текстовом режиме. В случае, если вы будете открывать их в графическом режиме используя Ctrl+Alt+T, число не ограничено, т.к каждый раз создается экземпляр программы, которая создаёт виртуальную оболочку bash.

Для ускорения работы в терминале были придуманы следующие горячие клавиши (hotkeys):

**Ctrl+C** – остановка запущенного процесса. Посылает запущенному процессу сигнал SIGINT, что останавливает процесс.

**Ctrl+Z** – временная «остановка» процесса. Запущенному процессу посылается сигнал SIGSTP, что ставит процесс «на паузу». Для того, чтобы

«разбудить» процесс, можно воспользоваться командой *fg* или *bg*.

**Ctrl+D** – данная комбинация обозначает «EOF», что обозначает «Конец ввода данных». Например:

pavel@rosa-uy3tg ~ $ cat > /home/file.txt

ввожу текст [нажат Enter]

Пример текста: Хихик)

закончил вводить текст [Нажата комбинация Ctrl+D]

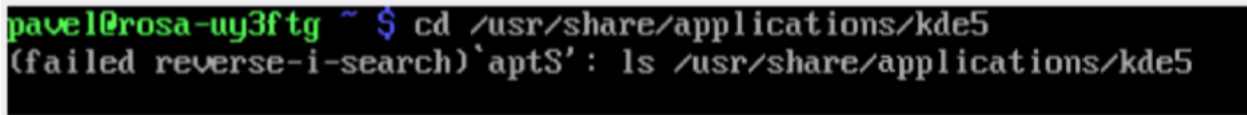
pavel@rosa-uy3tg ~ $ <тут могу снова ввести команду>

Текст полностью передавался в файл file.txt, редактируемый с помощью команды *cat.* Лишь после нажатия команда выполнилась и терминал стал готов к вводу следующих команд.

**Ctrl + V** – ввод следующего символа. Например, если вы при вводе параметра нажмете TAB, то вам bash дополнит вашу команду далее, а если использовать Ctrl+V+Tab, то он добавит TAB как символ в ваш параметр.

**Ctrl+L –** очищение экрана. Производит очистку экрана от результатов работы команд или от введенных команд.

**Ctrl+R –** позволяет искать в истории команд команды, содержащие указанный далее текст.



**Ctrl + O –** повтор серии команд, введенных ранее. То есть нужно из истории команд стрелками выбрать первую команду из серии и нажать **Ctrl + O** – это выполнит текущую команду и выведет из истории следующую. Дальше можно продолжать нажимать **Ctrl + O** с тем же эффектом.

**Ctrl + A / Ctrl+E** – эти команды позволяют вам перейти в начало и в конец текущей вводимой команды соответственно.

**Alt + b | f / Ctrl +** ß**|**à – предыдущее/следующее словосочетание

* **Alt-b** (и тот же эффект у **Ctrl-Left**) – переход в начало предыдущего слова.
* **Alt-f** (и тот же эффект у **Ctrl-Right**) – переход в конец следующего слова.

**Alt + # -** во время набора очень длинной команды вы понимаете, что вам нужно что-нибудь посмотреть или дополнительно сделать (например, посмотреть, какие файлы есть в определенной директории, прочитать мануал по команде, установить нужный пакет…). Хотелось бы посмотреть нужную информацию и продолжить набирать команду, а не начинать печатать её сначала. Alt-# (выполняется нажатием Alt + Shift + 3. Также можно использовать Esc-#) – преобразует текущую набранную команду в комментарий в истории – добавляет символ «#» в начало строки и добавляет

полученную строку в историю команд. Далее вы можете её использовать через

## Ctrl + O.

Основной командой в ОС явлется **man** — универсальный инструмент для получения справочной информации о любой команде в операционной системе. Она предоставляет доступ к руководству пользователя (man- странице) для любой установленной команды в Linux или Unix.

MAN – самая важная часть ОС для пользователя, т.к. позволяет получить справку по инструменту, с которым он планирует работать. При этом неважно, используете ли вы конкретно консольную утилиту или отдельную программу с графическим интерфейсом. При поиске информации о работе утилиты

ВСЕГДА ИЩИТЕ СНАЧАЛА В MAN-CТРАНИЦЕ УТИЛИТЫ,

а лишь затем осуществляйте поиск в иных ресурсах (интернет, современные GPT-сети для формирования контекста для поиска (а не полноценного ответа :) ) и прочее.

Команда **who** — это утилита командной строки Linux, которая используется для отображения списка всех вошедших в систему пользователей вместе с информацией об их сеансах входа в систему. Она полезна для системных администраторов, которым необходимо отслеживать, кто использует систему, а также для любых отдельных пользователей, которые хотят определить, вошел ли в систему кто-либо ещё.

Команда **mkdir** в Linux позволяет пользователям создавать новые директории (также называемые папками в некоторых ОС). Эта команда создаёт директорию(и), если они ещё не существуют. Она может создавать несколько каталогов одновременно, а также устанавливать разрешения для директорий (папок).

Команда **cd** (Change Directory) в Bash используется для изменения текущей рабочей директории. Она позволяет перемещаться по файловой

системе. Команда может принимать абсолютный или относительный путь к директории.

Команда **ls** используется в командной оболочке Linux для вывода содержимого каталогов и информации о файлах. Кроме утилиты ls, есть схожие с ней *dir* и *vdir*, отличающиеся поведением по-умолчанию.

Команда **cp** в Bash (от «copy») копирует файлы и директории из одного места в другое. Обычно ей передаются два аргумента:

* адрес с именем исходного файла
* новый адрес с именем или просто адрес каталога, куда помещается копия.

Адрес может быть как абсолютным, так относительным. Если операции с файлами выполняются в текущем каталоге, то указывать адрес смысла нет. Пишется только имя исходного файла и имя копии. Поскольку файлов с одинаковыми именами и адресами быть не может, имя копии должно отличаться от имени исходного файла.

Команда **cat** — это одна из самых часто используемых команд Linux.

Она читает данные из файла или стандартного ввода и выводит их на экран.

Команда **less** в Linux предназначена для просмотра больших текстовых файлов. Она позволяет перематывать текст не только вперёд, но и назад, осуществлять поиск в обоих направлениях, переходить сразу в конец или в начало файла. Особенность less заключается в том, что команда не считывает текст полностью, а загружает его небольшими фрагментами.

Команда **sleep** в Bash приостанавливает выполнение следующей команды на заданное время. Она используется, например, в скрипте bash при повторной попытке неудачного действия или внутри цикла.

Команда **date** в Bash используется для отображения или установки системной даты и времени. Она предоставляет различные опции для форматирования и манипуляции с датами.

Команда **ping** — инструмент командной строки Linux, используемый для проверки доступности хоста в сети. Эта команда отправляет пакеты эхо- запроса Internet Control Message Protocol (ICMP) на целевой хост и ожидает получения ответных пакетов. После этого инструмент ping выводит на экран статистику о количестве отправленных, полученных и потерянных пакетов.

Команда **ifconfig** - аналогична IPconfig в Windows, с другими ключевыми параметрами и более широким функционалом. Используется для назначения сетевого адреса, изменение настроек параметров сетевого адаптера и IP протокола. Она используется в большинстве \*nix систем на этапе загрузки операционной системы для настройки сетевых интерфейсов. После этого она обычно используется только при отладке, настройке или восстановлении системы.

Если аргументы не переданы, ifconfig выдает информацию о состоянии активных интерфейсов. Если указан один аргумент интерфейс, выдается информация только о состоянии этого интерфейса; если указан один аргумент

-a, выдается информация о состоянии всех интерфейсов, даже отключенных. Иначе команда конфигурирует указанный интерфейс.

В компьютерных сетях сетевым интерфейсом называют:

* Точку соединения между компьютером пользователя и частной (LAN) или общественной сетью (WAN);
* Сетевую карту компьютера;
* Точку соединения двух сетей между собой.

Интерфейс **loopback** имеет несколько путей применения. Он может быть использован сетевым клиентским программным обеспечением, чтобы общаться с серверным приложением, расположенным на том же компьютере. То есть если на компьютере, на котором запущен веб-сервер, указать в веб- браузере URL <http://127.0.0.1/> или <http://localhost/>, то он попадает на веб-сайт этого компьютера. Этот механизм работает без какого-либо активного подключения, поэтому он полезен для тестирования служб, не подвергая их

безопасность риску, как при удаленном сетевом доступе. Подобным образом, ping адреса loopback — это основной тест функционирования IP стека в операционной системе.

В СОВРЕМЕННЫХ Linux ДИСТРИБУТИВАХ КОМАНДА IPCONFIG

ЗАМЕНЕНА НА NetworkManager.

Редактор **nano** - текстовый редактор с простым и интуитивно понятным интерфейсом, включающим в себя основные команды для редактирования текста. Nano поддерживает подсветку синтаксиса, конвертацию файлов DOS/Mac, проверку орфографии и кодировку UTF-8. Программа Nano (с пустым буфером) занимает в оперативной памяти менее 4 Мб.

Самое важное:

**Alt+U (Esc U)** — Undo, отмена последнего действия. **Alt+E (Esc E)** — Redo, повтор отменённого действия. **F1 (Ctlr+G)** — Показать справку

**F2** — Выход с сохранением файла

**Ctrl+X** — Вернуться к редактированию из справки, и прочих диалоговых окон, выход из программы.

Копирование и вставка текста:

**Alt-A (Ctrl-6)** — установить\снять маркер начала выделения текста (либо просто стрелки с shift)

**Alt+6 (Esc 6)** — копировать выделение текст в буфер обмена.

Комбинация alt+6 у меня занята и не заработала.

**Ctrl+K (или F9)** — вырезать выделенный фрагмент в буфер обмена, при отсутствии выделения удалить строку.

**Ctrl+U (или F10)** — вставить фрагмент из буфера обмена

**Ctrl+Shift+V** — вставить из GUI буфера обмена

**Ins** — Вставка текста из другого файла, либо вывода команды.

Поиск и замена:

**Ctrl+W** — Поиск

**Alt+W** — Повторный поиск

**Ctrl+\** — Замена. Помнит последний поиск как изменяемое по умолчанию.

Навигация: **Alt+\ (Ctrl+Home)** — В начало файла **Alt+/ (Ctrl+End)** — В конец файла **Ctrl+Y (PgUp)**— На страницу вверх **Ctrl+V (PgDn)** — На страницу вниз

**Alt+S** — Переключение между плавной прокруткой страниц и фиксированной

**Ctrl+B** — Влево на символ **Ctrl+F** — Вправо на символ **Ctrl+N** — Вниз на символ

**Ctrl+P** — Вверх на символ **Ctrl+Space** — На слово вперёд, можно просто **Ctrl+курсор**. (курсор встаёт в начале слова)

**Alt+Space** (**ESC Space**) — На слово назад.

**Alt**+**-** — Прокрутка вверх, не меняя позиции курсора

**Alt**+**=** — Прокрутка вниз, не меняя позиции курсора

**Alt+7 (Esc 7)** — К предыдущему блоку текста, либо просто **Ctrl+курсор Alt+8 (Esc 8)** — К следующему блоку текста**.**

вниз).

**Alt+0 (Esc 0)** — К следующему параграфу (до первой пустой строки

**Alt+9 (Esc 9)** — К предыдущему параграфу (до первой пустой строки

вверх).

**Alt+G** — Переход на строку с указанным номером.

**Alt+<** — Переход к предыдущему открытому файлу (Если nano запущен с несколькими файлами)

**Alt+>** — Переход к следующему открытому файлу.

Редактирование текста: **Ctrl+D (Del)** — стереть символ под курсором **Ctrl+H (BSp)** — Стереть символ слева от курсора

**Ctrl+M (Enter)** — Вставка пустой строки.

**Ctrl+I (Tab)** — Вставка табуляции.

**Alt+Shift+{** — Уменьшить отступ параграфа удалением табуляции или пробелов

**Alt+Shift+}** — Увеличить отступ параграфа, вставкой табуляции. Если уже был отступ пробелами, добавит к нему табуляцию и подсветит красным.

**Ctrl+]** — Автодополнение слов, например команд в программе.

Программисты оценят!

**Alt+3 (Esc 3)** — За(рас)комментировать символом # текущую строку, выделенный параграф.

**Alt+J (F4)** — Justify, выровнять текущий абзац

Разное:

**Esc Esc 220** — Ввод символов по десятичному коду. Например 220 — ввод Ü с умляутом.

**Alt+V** — Ввод ASCII последовательности комбинации или клавиши.

**Alt+T** — Trunkate. Стирает всё от курсора и до конца файла.

**Ctrl+C** (**F11**) — Разово выводит информацию по текущей строке, колонке, номеру символа.

**Alt+D** — Подсчитать количество слов, строк, символов в файле

**Ctrl+L** — Обновить экран

Настройки nano:

**Alt+Shift+4** — Мягкий перенос длинных строк разрешить/запретить

**Alt+K** — переключает действие по **Ctrl+K** между удалением по умолчанию всей строки и удалением от курсора до конца строки.

**Alt+Q** — Переключиться с табуляций на ввод пробелов.

**Alt+L** — Включить и выключить жёсткий перенос строк.**Alt+C** — Переключает постоянный вывод инфо по строке по **Ctrl+C**

**Alt+X** — Показать\скрыть строку подсказки внизу экрана.

**Alt+Shift+3** — Показывать номера строк. **Alt+P** — Показывать пробелы символом "·" **Alt+Y** — Показывать подсветку синтаксиса.

**Alt+H (Esc H)** — Включить умную клавишу Home (переводит курсор не в 1ю колонку, а на 1й символ строки)

**Alt+I (Esc I)** — Включить автоотступы (перевод строки на новую с таким же отступом, как у предыдущей)

**Alt+B** — Переключиться на создание резервных копий и обратно

**Alt+M** — Включить «поддержку мыши». Лучше не использовать, клавиши лучше!

Редактор **vim**

Русскоязычная справка по редактору vim доступна прямо в терминале, путем набора команды

*vimtutor ru*

Для выхода из справки используйте Ctrl+C, наберите

*:qa!*

и нажмите Enter

## Требования к отчету по практике:

1. Титульный лист.
2. Ход работы: вставляются полные скриншоты рабочего стола + описание проделанной работы.
3. Вывод: краткий вывод по проделанной работе.
4. Сохраните файл man.txt

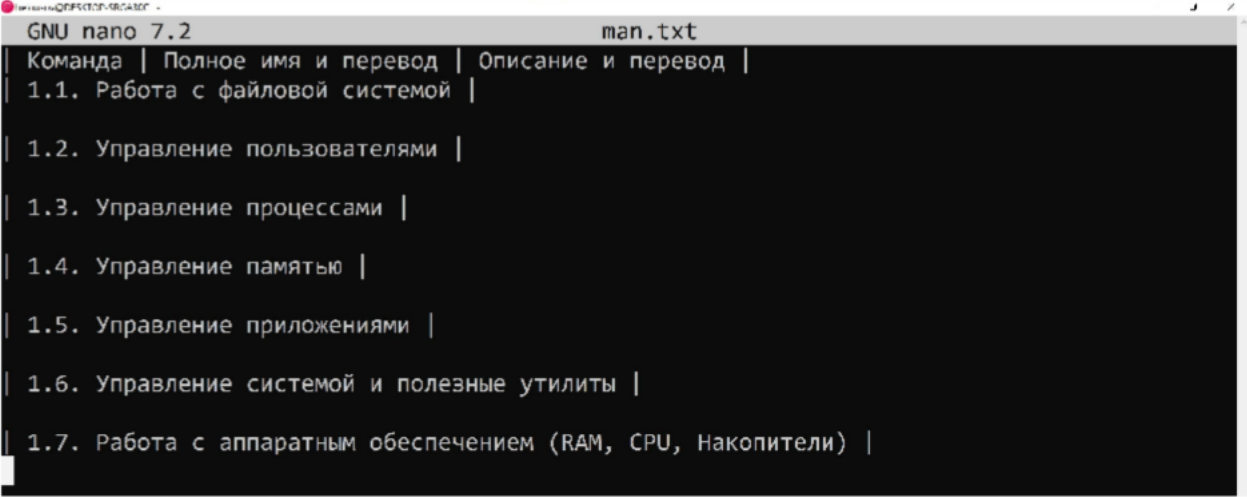
## Ход практики.

*Для выполнения практической работы используйте виртуальную машину, устанавливаемую в практической работе №2.*

## Задание 1. Используя редактор nano, в домашней директории

/home/{имя\_пользователя} **создайте файл** man.txt**.**

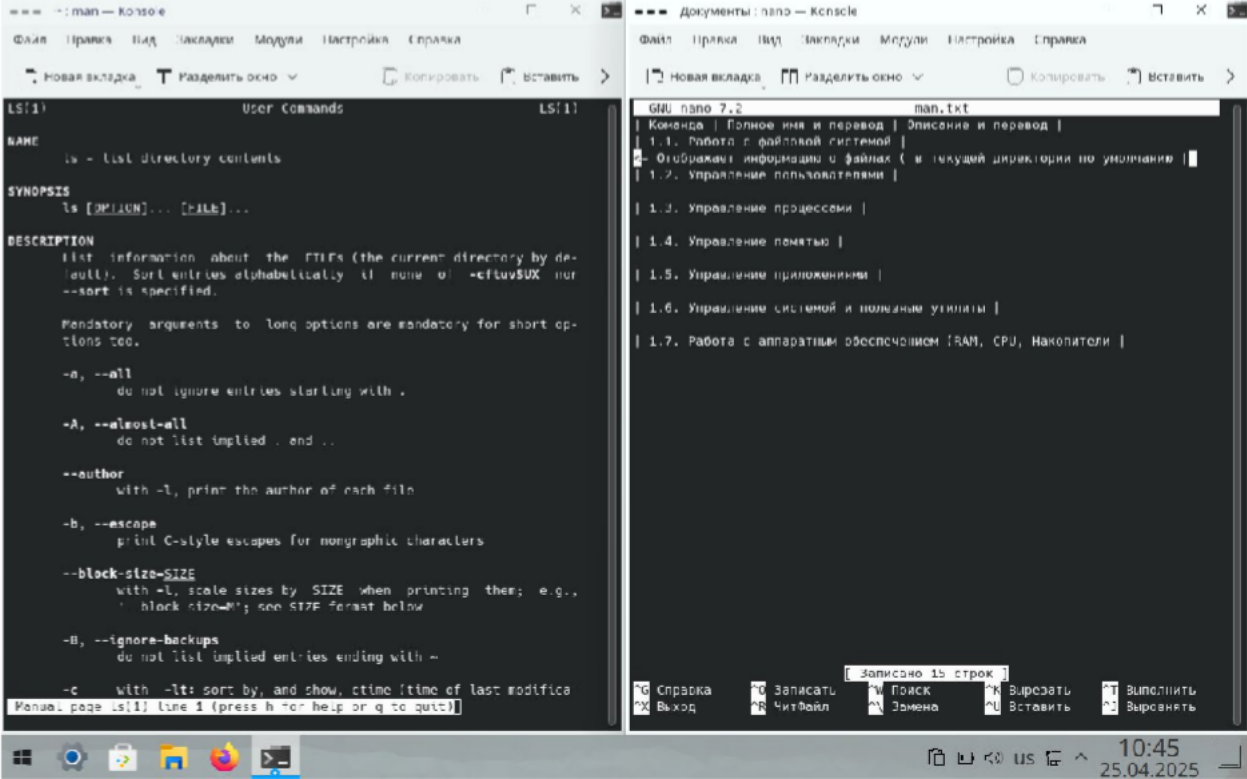
## Задание 2. Используя редактор nano, создайте следующую структуру в файле man.txt как показано на рисунке.

****

**Задание 3. Заполните данную таблицу, основываясь на таблице 1 из практической работы. Вызовите справку по следующим командам, используя** man <утилита>.

Примечание: лучший вариант работы – открыть два разных терминала, в одном писать значения команд из редактора nano в файл man.txt, а в другом – искать man-страницу и требуемые значения.

Иначе вы можете работать из одного терминала, постоянно переключаясь между редактором nano и вызывая справку man.



*В столбце «Полное имя и перевод» укажите полное смысловое именование и его перевод на русский язык. В столбце «Описание» вставьте первое предложение из справки.*

*P.S. При переводе, ради себя, постарайтесь использовать свои знания английского языка. Таким образом вы лучше запомните синтаксис и значения команд, что пригодится вам в дальнейшей работе по предмету и в общем, при работе с информационными системами.*

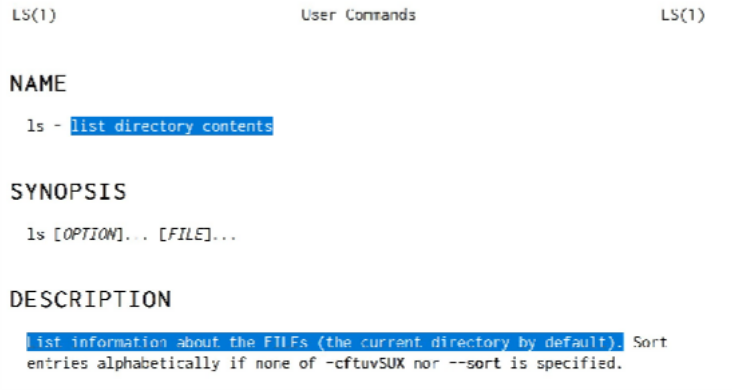
**

Рисунок 4 – выделена нужная информация

Таблица 1 – шаблон для задания 1

|  |  |  |
| --- | --- | --- |
| Команда | Полное имя и перевод (при возможности): | Описание + перевод |
| 1.1.Работа с файловой системой | | |
| *ls(пример)* | *list directory contents – список файлов в*  *директории* | *List information about the FILEs (the current directory by default) – Список с информацией о файлах (по умолчанию – из текущей директории* |
| cd |  |  |
| pwd |  |  |
| mkdir |  |  |
| rm |  |  |
| cp |  |  |
| mv |  |  |
| touch |  |  |
| cat |  |  |
| more |  |  |
| less |  |  |
| head |  |  |
| tail |  |  |
| grep |  |  |

|  |  |  |
| --- | --- | --- |
| ln |  |  |
| find |  |  |
| locate |  |  |
| chmod |  |  |
| chown |  |  |
| chgrp |  |  |
| stat |  |  |
| gpg |  |  |
| wc |  |  |
| tar |  |  |
| compress |  |  |
| zip |  |  |
| unzip |  |  |
| 1.2.Управление пользователями | | |
| adduser / addgroup |  | *тут единая справка для этих команд* |
| usermod |  |  |
| deluser / delgroup |  | *тут тоже единая справка для этих команд* |
| passwd |  |  |
| su |  |  |
| sudo / sudoedit |  | *тут ТОЖЕ) единая справка для этих команд* |

|  |  |  |
| --- | --- | --- |
| finger |  |  |
| who |  |  |
| id |  |  |
| groups |  |  |
| useradd |  |  |
| userdel |  |  |
| usermod |  |  |
| passwd |  |  |
| last |  |  |
| w |  |  |
| logout |  |  |
| 1.3.Управление процессами | | |
| top |  |  |
| ps |  |  |
| kill |  |  |
| pkill |  |  |
| htop |  |  |
| free |  |  |
| vmstat |  |  |
| killall |  |  |
| bg |  |  |

|  |  |  |
| --- | --- | --- |
| fg |  |  |
| lsof |  |  |
| renice |  |  |
| nice |  |  |
| pgrep |  |  |
| strace |  |  |
| lsof |  |  |
| sar |  |  |
| uptime |  |  |
| time |  |  |
| 1.4.Управление памятью | | |
| smem |  |  |
| sync |  |  |
| swapoff |  |  |
| swapon |  |  |
| sysctl |  |  |
| ulimit |  |  |
| pmap |  |  |
| slabtop |  |  |
| numactl |  |  |
| sysrq |  |  |

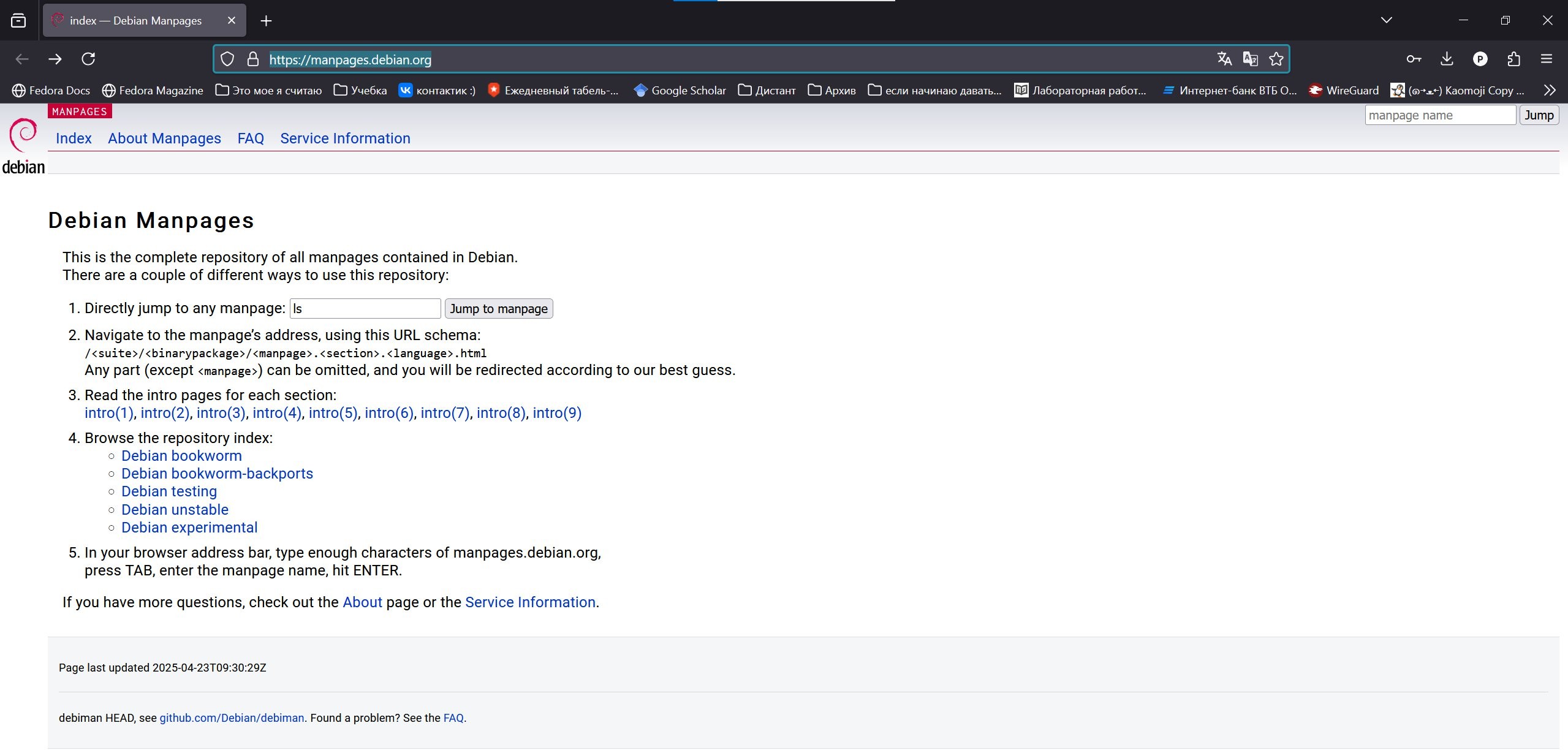
|  |  |  |
| --- | --- | --- |
| 1.5.Управление приложениями | | |
| apt-get |  |  |
| apt-get install |  |  |
| apt-get remove |  |  |
| apt-get update |  |  |
| apt-get upgrade |  |  |
| apt-cache |  |  |
| apt-cache search |  |  |
| dpkg |  |  |
| dpkg -i |  |  |
| dpkg -r |  |  |
| dpkg -l |  |  |
| snap (по возможности) |  | *SNAP – по возможности, он ставится самостоятельно* |
| snap install |  |  |
| snap remove |  |  |
| snap list |  |  |
| systemctl |  |  |
| systemctl start |  |  |
| systemctl stop |  |  |
| systemctl restart |  |  |
| systemctl enable |  |  |

|  |  |  |
| --- | --- | --- |
| systemctl disable |  |  |
| service |  |  |
| service “x” start |  |  |
| service “x” stop |  |  |
| service “x” restart |  |  |
| service “x” status |  |  |
| 1.6.Управление системой и полезные утилиты | | |
| shutdown |  |  |
| reboot |  |  |
| halt |  |  |
| poweroff |  |  |
| systemctl :] |  | НЕ УЧИТЫВАЙТЕ СМАЙЛИК, ПОИЩИТЕ, ОНО ПОВТОРЯЕТСЯ) |
| service :] |  |  |
| hostname |  |  |
| w |  |  |
| uname |  |  |
| last reboot |  |  |
| ifconfig |  |  |
| netstat |  |  |
| ip |  |  |

|  |  |  |
| --- | --- | --- |
| ping |  |  |
| traceroute |  |  |
| ssh |  |  |
| rsync |  |  |
| crontab / cron |  |  |
| at |  |  |
| nohup |  |  |
| history |  |  |
| nano |  |  |
| vim |  |  |
| telnet |  |  |
| 1.7.Работа с аппаратным обеспечением (RAM, CPU, Накопители) | | |
| dmesg |  |  |
| cat /proc/cpuinfo |  | *Достаточно описания файла* |
| cat /proc/meminfo |  | *Достаточно описания файла* |
| lshw |  |  |
| lsblk |  |  |
| lsusb |  |  |
| lspci |  |  |
| free -m |  |  |
| dmidecode |  |  |

|  |  |  |
| --- | --- | --- |
| hdparm |  |  |
| badblocks |  |  |
| df |  |  |
| fdisk |  |  |
| mount |  |  |

Вы также можете использовать веб-версию <https://manpages.debian.org/> , которая не даст ответов на все вопросы. Однако, рекомендую



Вместо ls вводите команду, но тогда будет только англоязычная страница справочник. Для переключения на русский язык (если он доступен) выберите его на man-странице. Если его нет, то *Перевод дайте на основе своих знаний английского языка. Штрафов за «неверный перевод» не предусмотрено.*

**Задание 4. Предоставьте созданный файл преподавателю.**