

Оглавление

1	Введение	3
2	Основные команды	5
2.1	man	6
2.2	naпo	8
2.3	vim	11

Глава 1

Введение

Задача данной методички – дать **практические навыки** работы в консоли Linux, на примере дистрибутива Debian. В методичке обозначаются основные команды, горячие клавиши, инструменты и варианты их использования, а так же методы быстрой и эффективной работы в консоли Linux. В то же время не уходя слишком далеко в детали, оставляя некоторые инструменты и темы открытыми, чтобы читатель провёл самостоятельное их изучение, посредством чтения оригинальной документации, поиска в Google, изучения StackOverflow и просмотра YouTube. Последнее менее желательно, т.к. концентрация полезной информации в единицу времени достаточно низкая, а качество и полнота преподносимой информации желает лучшего. Никакой сайт или обучающее видео не смогут сравниться с полнотой оригинальной документации, об этом нужно всегда помнить и акцентировать внимание при изучении **абсолютно любой** технологии, программы, языка программирования, методологии и т.п.

Умение читать оригинальную документацию на английском языке (по диагонали), при этом находя то что нужно – является критически необходимым навыком, который нужно развивать для быстрой и эффективной работы, т.к. информации на английском языке всегда было и будет больше, в силу того, что мировое сообщество его выбрало в качестве общего для общения и совместной работой между специалистами разных стран. Так же зачастую информация на английском языке является единственным источником о конфигурировании тех или иных инструментов, поэтому умение читать и понимать технические тексты на английском языке является критически важным навыком для успешной работы в IT-сфере.

P.S. Для единообразия описания, были приняты следующие обозначения и соглашения:

- `Ctrl` + `X` – нажатие сочетания клавиш Ctrl+X
- `pwd` – команда консоли (программа)
- *arg* – аргумент/параметр команды
- `/etc/passwd` – конфигурационный файл или файл с настройками
- `/proc/` – путь к директории с файлами процессов

P.S.2. В данном методическом пособии слово *команда* употребляется, как синонимом слова *программа* и по сути таковой и является, если речь не идёт о встроенных командах shell-оболочек таких как: `sh` , `bash` , `zsh` .

Глава 2

Основные команды

2.1 man

man - сокращение от manual, команда позволяет просматривать справочную информацию о программах Linux или командах shell-оболочки (**bash**, **sh**, **dash**, **zsh** и пр.), форматах конфигурационных файлов, специальных файлах устройств, описание системных вызовов или библиотечных вызовов и системных команд администратора. Пример формата вызова **man** :

```
$ man [section] page
```

- *section* тип страницы справочной информации:

1 – программы или команды shell-оболочки

2 – системные вызовы (функции ядра: **fork**, **accept**, **listen**, **select**, **mmap** и пр.)

3 – библиотечные вызовы (функции библиотек: **fopen**, **pow**, **malloc** и пр.)

4 – специальные файлы (обычно находящиеся в **/dev/** : **random**, **mem**, **tty** и пр.)

5 – форматы файлов (**/etc/passwd** , **/etc/shadow** , **~/.ssh/authorized_keys** и пр.)

6 – игры

7 – описания, соглашения и пр.

8 – команды системного администратора (доступные только для **root** и/или **sudo**-пользователя: **ss**, **adduser**, **sysctl** и пр.)

- *page* имя программы, команды, конфигурационного файла, системного вызова и т.д.

Посмотреть информацию о команде **man** :

```
$ man man
```

После входа в интерактивный режим **man** доступны следующие функции:

- q** – выход из **man** (обратить внимание, чтобы раскладка была английской)
- h** – посмотреть помощь по навигации **man**

- u** / **d** – пролистать на пол экрана вверх/вниз
- y** или **↑** – пролистать на одну строку вверх
- e** или **↓** – пролистать на одну строку вниз
- w** / **PgUp** – пролистать на один экран вверх
- z** / **PgDown** – пролистать на один экран вниз
- g** / **G** – переместиться в начало/конец документа

- /** + ввести шаблон поиска – прямой поиск по шаблону
- ?** + ввести шаблон поиска – обратный поиск по шаблону
- n** / **N** – повторить предыдущий поиск в прямом/обратном направлении

Для включения/отключения нумерации строк в режиме просмотра справочной страницы необходимо ввести соответствующие опции и нажать **Enter**:

-N – включить нумерацию строк

-n – выключить нумерацию строк

Показать все доступные разделы справочной информации по **passwd** :

```
$ man -f passwd
```

Показать раздел 1 справочной информации для команды `passwd` :

```
$ man 1 passwd
```

Показать раздел 5 справочной информации о формате файла `/etc/passwd` :

```
$ man 5 passwd
```

Поиск всех справочных страниц в названии или описании которых встречается `ls`:

```
$ man -k ls
```

Поиск по всем справочным страницам сочетания слов *password change*:

```
$ man -K "password change"
```

`Enter` – открыть страницу из результата поиска

`Ctrl`+`D` – пропустить страницу

`Ctrl`+`C` – закрыть поиск

Просмотреть страницу справки команды `man` на русском языке (если страница справки с соответствующей русской локалью `ru_RU` имеется в системе):

```
$ man -L ru_RU man
```

`/etc/manpath.config` – файл настройки `man-db`

`/usr/share/man/` – здесь расположены файлы со справочными страницами

2.2 nano

nano – текстовый редактор. Чтобы создать новый файл с именем *file* или открыть существующий в режиме редактирования, необходимо выполнить команду ¹:

```
$ nano file
```

После открытия редактора **nano** 2.1, внизу можно увидеть следующие подсказки: **^X** Exit, **^G** Help и т.д. В *nix системах сочетание нажатия клавиши **Ctrl** с нажатием другой клавиши обозначается, как символ **^** и далее название клавиши, так нажатие сочетания клавиш **Ctrl**+**X** и **Ctrl**+**C** обозначаются, как **^X** и **^C** соответственно.

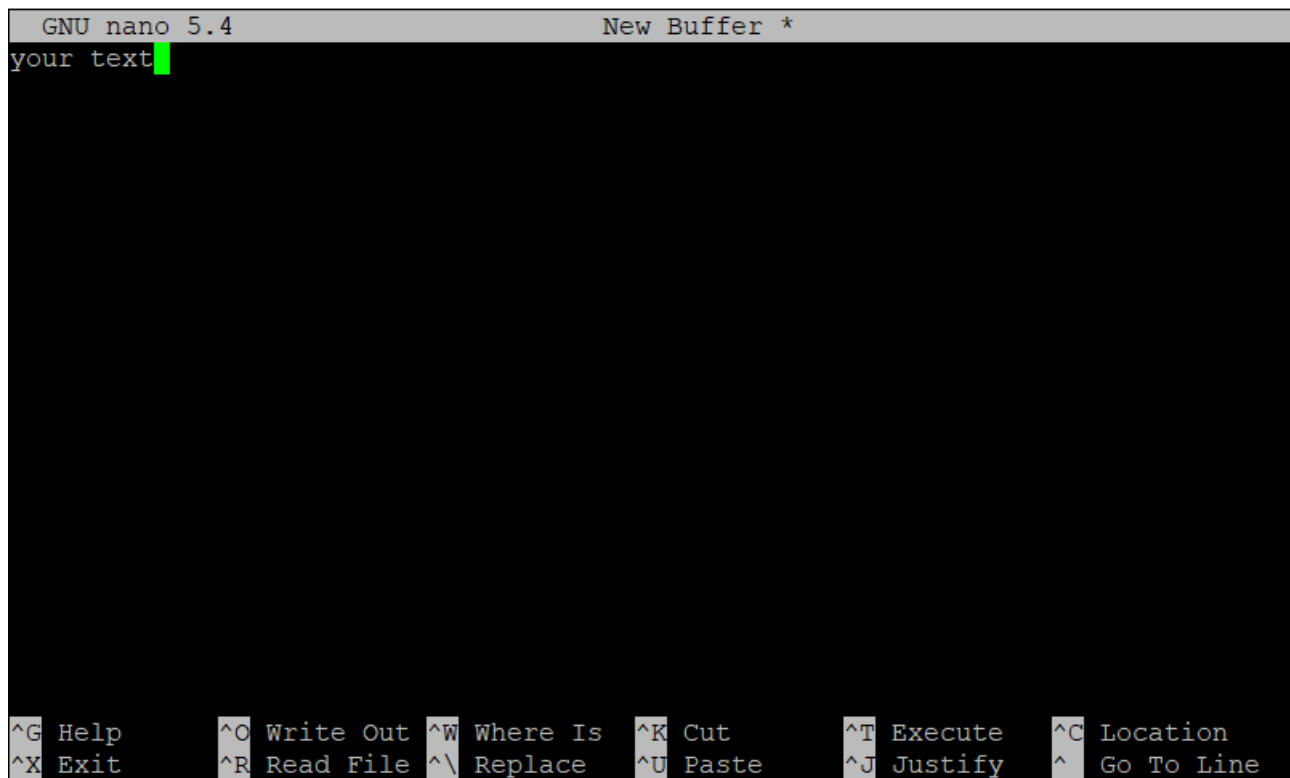


Рис. 2.1: Редактор **nano** после открытия

Так же в **nano** можно открыть более одного файла одновременно, например:

```
$ nano file1 file2 file3
```

В результате будут открыты файлы *file1*, *file2* и *file3* в так называемый *буфферах* (экраны, между которыми можно переключаться сочетаниями клавиш **Alt**+**>** и **Alt**+**<**). Так же можно набор файлов по маске, например:

```
$ nano *.log
```

- Ctrl**+**X** – закрыть текущий буффер и выйти из **nano**
- Ctrl**+**O** – сохранить изменение текущего буффера в файл
- Ctrl**+**R** – открыть файл в текущем буфере

¹если параметр *file* не указывать, то откроется редактор с пустым документом

(**Ctrl** + **R**) + (**Atl** + **F**) – открыть файл в новом буфере
 (**Ctrl** + **R**) + (**Atl** + **F**) + (**Enter**) – создать новый буфер

Редактирование

Atl + **A** / **Atl** + **6** – выделить текст с текущей позиции курсора или отменить выделение
Ctrl + **6** – скопировать выделенный текст
Ctrl + **K** – вырезать текущую строку или выделенный текст в буфер обмена
Ctrl + **U** – вставить содержимое буфера обмена в текущую позицию курсора
Atl + **M** – вырезать от позиции курсора до конца файла
Ctrl + **D** – удалить символ справа от курсора
Ctrl + **H** – удалить символ слева от курсора
Ctrl + **M** – вставить пустую строку
Atl + **R** – заменить текст или регулярное выражение
Ctrl + **W** – искать текст или регулярное выражение
Atl + **W** – повторить последний поиск

Навигация

PgUp / **PgDown** – пролистать на один экран вверх/вниз
Ctrl + **←** / **Atl** + **←** – вперёд/назад на одно слово
Atl + **E** / **End** – в конец текущей строки
Atl + **A** / **Home** – в начало текущей строки
Atl + **** / **/** – на первую/последнюю строку файла
Ctrl + **_** – перейти на указанный номер **строки** или **строки,позиции**

Прочее

Atl + **C** – отображение положения курсора включить/отключить
Ctrl + **C** – показать положение курсора (если в режиме редактирования документа)
Atl + **P** – отображение пробелов
Atl + **Y** – подсветка синтаксиса
Atl + **I** – автоотступы
Atl + **H** – умная клавиша HOME
Atl + **D** – подсчитать количество слов, строк и символов в документе
Atl + **M** – включить/отключить мышку для позиционирования курсора в редакторе
Atl + **N** – включить/отключить нумерацию строк

Открыть файл *file* и перейти к строке *line* и столбцу *col*:

```
$ nano +line,col <file>
```

Открыть файл *file* с нумерацией строк:

```
$ nano -l <file>
```

Двойное нажатие `Esc`, а затем ввод **трёхзначного числа** от **000** до **255** вставит символ с соответствующим ему hex-кодом.

Открыть файл *file* предварительно сохранив его резервную копию:

```
$ nano -B <file>
```

Чтобы указать директорию для сохранения резервных копий редактируемого файла или файлов, используется опция -C:

```
$ nano -BC <dir> <file>
```

Открыть файл *file* только для чтения используется ключ -v:

```
$ nano -v <file>
```

2.3 vim

vim – самый популярный текстовых редактор предустановленный на всех *nix-like системах, поэтому умение работать в нём является **обязательным**. Для ознакомления с базовой работой в **vim** можно воспользоваться встроенным мини-учебником **vimtutor**, который можно вызвать и командной строки. Чтобы запустить его русскую версию необходимо выполнить команду:

```
$ vimtutor ru
```

В редакторе **vim** есть два режима работы:

- **командный режим** – чтобы в него перейти необходимо нажать клавишу **Esc**
- **режим ввода текста** – чтобы в него перейти нужно нажать клавишу **i**

Создать или открыть файл с именем *file* для редактирования ²:

```
$ vi file
```

После запуска **vim** редактор по умолчанию находится в **командном режиме** и в дальнейшем вся работа в редакторе строится посредством переключения между этими двумя режимами по мере необходимости и работе в каждом из них. В **командном режиме** вводить текст невозможно, но с помощью нажатия ряда клавиш (комбинации которых будут описаны ниже), возможно редактирование документа и навигация по нему. Стоит отметить, что в **командном режиме** при вводе команд они **не отображаются** на экране, за исключением команд, которые начинаются со специальных символов «:», «/» или «?». Если в же в **командном режиме** вводится команда и она начинается с символов «:», «/» или «?», в этом случае они отображаются в самой нижней строке экрана, она никак не связана с редактируемым документом и является **командной строкой** редактора **vim**.

Если же необходимо вводить новые символы в документе или удалить посимвольно ранее набранный текст, то необходимо переключиться в **режим ввода текста** посредством нажатия клавиши **i**, для всех остальных манипуляций с текстом в **vim** используется **командный режим**, в который можно перейти посредством нажатия клавиши **Esc**.
Открыть файл *file* для просмотра:

```
$ view file
```

Открыть последнюю сохраненную версию файла *file* после аварийного выхода:

```
$ vi -r file
```

Открыть файл *file* и поместить курсор на последнюю строку *n*:

```
$ vi +n file
```

Открыть файл *file* и поместить курсор на последнюю строку:

²если параметр *file* не указывать, то откроется редактор с пустым документом, сохранить который в дальнейшем возможно посредством ввода команды **:w file** + **Enter**

```
$ vi + file
```

Открыть файл *file* и поместить курсор на первое вхождение *string* :

```
$ vi +/string file
```

Чтобы создать или открыть файл *file* с использованием шифрования, необходимо выполнить команду, а далее ввести ключ для шифрования/дешифрования в зависимости от того создаётся ли файл или открывается уже зашифрованный:

```
$ vi -x file
```

: – переход в режим **командной строки** ³, не путать с **командным режимом**. Суть режима заключается в том, что после введения двоеточия, будучи в **командном режиме**, далее появляется возможность вводить ряд команд с целью управления документом и состоянием редактора в появившейся снизу командной строке, а так же навигации по списку введённых команд в **командной строке** в **vim**, посредством стрелок на клавиатуре **↑**/**↓**. Пример таких команд перечислен ниже:

:q – выход из текстового редактора ⁴

:q! – выход без сохранения

:wq – выход с сохранением

:w – сохранить файл или внесенные изменения

:e – загрузить файл для редактирования ⁵

Редактирование

В **командном режиме** действуют следующие комбинации клавиш для редактирования текста:

yy – скопировать строку в буфер обмена (нажатие **y** + **y**)

dd – вырезать строку в буфер обмена (нажатие **p** + **p**)

p – вставить текст из буфера обмена (нажатие **p**)

u – отменить последнюю команду

U – отменить все изменения в строке

Ctrl + **r** отмена отмены последние команды

³как таковой клавиши **:** нет, для получения данного символа необходимо нажать сочетание **Shift** + **;** в английской раскладке клавиатуры, необходимо обратить на этот факт особое внимание, т.к. в дальнейшем, например, при указании клавиши **\$** подразумевается нажатие сочетания **Shift** + **4**, а при указании клавиши **_** подразумевается нажатие сочетания клавиш **Shift** + **-** и т.д.

⁴фактически это нажатие сочетания клавиш **Shift** + **:** и далее нажатие **q** и для выполнения команды нажать **Enter**. Если набираемая команда выводится не снизу экрана в **командной строке vim**, то скорее всего текущий режим это **режим ввода текста**, необходимо перейти в режим **командной строки** посредством нажатия **Esc** и повторить ввод команды

⁵после ввода команды ставится пробел и вводится имя файла, которое требуется загрузить или вводятся первые буквы названия файла и для **автодополнения** жмётся кнопка **Tab** (иногда несколько раз, если есть несколько файлов, которые начинаются с указанных первых букв), пока не дополнит до искомого имени файла. Это всё работает, если файл находится в той же директории, в противном случае указывается путь к файлу и только в конце указывается требуемый файл. В процессе указания полного пути к файлу рекомендуется пользоваться автодополнением с помощью нажатия клавиши **Tab**

. – позволяет повторить последнее действие

x – удаление одного символа под курсором ⁶

X – удаление одного символа до курсора

r – единичная замена символа под курсором ⁷

Shift + **r** – режим замены символов

~ – смена регистра символа над курсором

J – слияние следующей строки с текущей (j – от слова **join**)

nJ – слияние **n** строк (вводится с клавиатуры число **n** и далее жмётся кнопка **J**)

В **режиме ввода текста** действуют некоторые комбинации клавиш для упрощения работы:

Ctrl + **h** – удаляет символ слева от курсора, аналогично клавише **←Backspace**

Ctrl + **w** – удаляет одно слово перед курсором

Ctrl + **u** – удаляет все символы от начала строки до курсора

Ctrl + **t** – вставить табуляцию в начало текущей строки

Ctrl + **d** – удалить табуляцию из начала текущей строки

Выделение

V – выделить текст построчно

v – выделить кусок текста ⁸

o / **O** – перемещают курсор в разные концы выделенного блока для изменения размеров

Ctrl + **v** – выделить прямоугольную часть текста

Навигация

h / **l** – переместить курсор на один символ влево/вправо

j / **k** – переместить курсор на одну строку вниз/вверх

↑ / **↓** / **←** / **→** – так же возможна навигация

:number – перейти на строку с номером *number*

<number>G – перейти на конкретную строку с номером *<number>*

<number>gg – перейти на конкретную строку с номером *<number>*

z. – сделать текущую строку средней строкой экрана

z- – сделать текущую строку нижней строкой экрана

0 / **\$** – переместить курсор в начало/конец строки

gg / **G** – переместить курсор в начало/конец файла

Ctrl + **D** / **Ctrl** + **U** – на пол экрана вниз/вверх

⁶x – похожа на крестик, удалить/«перечеркнуть»

⁷r – от слова **replace**

⁸далее возможно использовать команды **d** – вырезать выделенный текст в буфер обмена, **y** – скопировать выделенный текст в буфер обмена с дальнейшей вставкой посредством команды **p**

}/**{** – абзац вниз/вверх

w – перейти на начало следующего слова

e – перейти к концу следующего слова

b – перейти на начало предыдущего слова

Также для удобной навигации, в тексте можно расставлять свои метки/«закладки» – это специальный образом отмеченные позиции, куда можно в любой момент вернуть каретку курсора, набрав соответствующую команду. Именем метки может быть любая **одна буква**. На примере метки с именем «a» посмотрим как это работает:

ma – создание метки

'a – перемещение курсора на метку «a»

Ctrl+**o**/**Ctrl**+**i** – перемещение к ранее созданным меткам назад/вперед

:marks – показать все созданные метки

Составление команд

Перед большинством команд, начинающихся с двоеточия, может быть указан диапазон строк, на которые эта команда будет действовать. Например, **:3,7d** служит для удаления строк 3-7. Диапазоны обычно используются с командой **:s** для замены в нескольких строках, например **:\$s/pattern/string/g** выполнит замены с текущей строки до конца файла.

:n,m – строки с **n** до **m**

:. – текущая строка

:\$ – последняя строка

: 'c – строка с маркером **c**

:% – все строки файла

:g/pattern/ – все строки, содержащие **pattern**

Одним из преимуществ **vim** перед рядом других редакторов, является возможность составлять комбинации из команд, например:

4dd – вырезать четыре строки

3e – перейти на три слова вперёд

7x – удалить 7 символов

8xj – заменить следующие 8 символов на символ «j»

и т.д.

Поиск и замена

Команды для поиска:

/ – прямой поиск текста

? – обратный поиск текста

n – повторить поиск вперёд

N – повторить поиск назад

Команды для поиска и замены:

:s/old/new – заменить первое вхождение **old** на **new**

:s/old/new/g – заменить все вхождения **old** на **new** во всём файле

:%s/old/new/g – замена всех вхождений **old** на **new** во всём файле

:%s/old/new/gc – замена всех вхождений **old** на **new** во всём файле с запросом подтверждения каждой замены («с» — confirmation)

:N,Ms/old/new/g – заменить вхождение **old** на **new** в диапазоне строк от **N** до **M**

Макросы

Макросы – последовательность команд, выполненных в **vim**, которая записана в именованный *регистр*, который в дальнейшем можно вызвать более лаконично, введя в командном режиме **@** и имя регистра (один символ). *Регистры* обозначаются латинскими буквами без учёта регистра («a» и «A» – один и тот же регистр), цифрами, и даже специальными символами. Пример записи макроса с именем «a»:

- **qa** – начало записи макроса **@a**
- ...
- набор команд **vim**
- ...
- **q** – окончание записи макроса

Теперь, чтобы запустить макрос с именем «a», нам нужно ввести команду **@a**, и запустится на выполнение макрос, хранящийся в регистре **a**. Мы можем традиционно для **vim** написать команду **10@a**, и макрос будет выполнен 10 раз.

Буфера

Открыть файлы *file1*, *file2* и *file3*:

```
$ vi file1 file2 file3
```

После открытия набора файлов их загрузка происходит в **буфера**, а далее происходит отображение содержимого буфера в **окнах** и **вкладках** о которых будет рассказано ниже.

:ls – список открытых буферов

:bn – перейти к следующему буферу

:bp – перейти к предыдущему буферу

:b name – переключиться на буфер **name** ⁹

:bd – удалить текущий буфер (если этот буфер единственное окно то **vim** закроется)

:bd name – удалить буфер **name**

Окна

Открыть в **vim** каждый файл из строки аргументов в отдельном окне **-o** ¹⁰:

```
$ vi -o one.txt two.txt three.txt
```

⁹(очень удобно комбинируется с табом, к примеру пишем **:b** + пробел, нажимаем несколько раз **Tab** с помощью автоподстановки меняются имена открытых буферов

¹⁰ключ **-O** приводит к аналогичным результатам, но окна будут разделены по вертикали

:new – создать окно

:sp – разделить окно по горизонтали (**Ctrl**+**w**+**s**)

:vsp – разделить окно по вертикали (**Ctrl**+**w**+**v**)

:only – развернуть текущее окно, закрыв все остальные окна (**Ctrl**+**w**+**o**). Однако, если в одном из окон есть несохранённые изменения, то вы увидите сообщение об ошибке и это окно не будет закрыто

:close – закрыть текущее окно

Ctrl+**w**+«стрелочки»(**↑**/**↓**/**←**/**→**) – перемещение между окнами

Команды для установки и изменения размера окон:

Ctrl+**w**+**_** – развернуть окно по вертикали

Ctrl+**w**+**|** – развернуть окно по горизонтали

height+**Ctrl**+**w**+**_** – установить высоту окна равную **height**

width+**Ctrl**+**w**+**|** – установить ширину окна равную **width**

Ctrl+**w**+**=** – сбросить размер всех разделенных окон и сделать их одинаковыми

count+**Ctrl**+**w**+**+**+**+** – увеличение размера окна по вертикали¹¹

count+**Ctrl**+**w**+**-** – уменьшение размера окна по вертикали

count+**Ctrl**+**w**+**>** – увеличение размера окна по горизонтали

count+**Ctrl**+**w**+**<** – уменьшение размера окна по горизонтали

Команды для перемещения положения окон:

Ctrl+**w**+**K** – поместить текущее окно в верхней части экрана¹²

Ctrl+**w**+**H** – поместить текущее окно в левой части экрана

Ctrl+**w**+**J** – поместить текущее окно в нижней части экрана

Ctrl+**w**+**L** – поместить текущее окно в правой части экрана

Если открыто несколько окон и нужно произвести манипуляции применительно к ряду окон, чтобы не вводить команды в каждом из окон, пригодятся следующие команды:

:qall – выйти из всех окон

:qall! – выйти из всех окон, не обращая внимание на несохранённые изменения

:wall – сохранить изменения во всех окнах

:wqall – сохранить изменения во всех окнах, где это требуется и затем выйти из всех окон

Вкладки

Открыть в **vim** каждый файл из строки аргументов в отдельной вкладке:

```
$ vi -p one.txt two.txt three.txt
```

:tabs – список всех открытых вкладок

:tabm 0 – открыть первую вкладку

:tabm – открыть последнюю вкладку

:tabn – перейти в следующую вкладку(**g**+**t**)

:tabp – перейти в предыдущую вкладку(**g**+**T**)

:tabnew file – открыть файл **file** в новой вкладке

:tabedit file – открыть файл **file** в новой вкладке

¹¹если перед командами параметр **count** не вводится с клавиатуры, то по умолчанию он равен единице

¹²**K** это нажатие сочетания **Shift**+**k**

:tabonly – закрыть все вкладки кроме текущей

:tabdo command – выполнить команду **command** над содержимым всех вкладок

:tabclose – закрыть текущую вкладку

:tabclose i – закрыть i-ю вкладку

- **буферы** – это некие контейнеры, которые содержат в себя загруженные в vim файлы
- **окна** – это часть экрана монитора, в котором отражается содержимое буфера
- **вкладки** – это способ организации нескольких окон на экране

Переключаясь между вкладками, вы переходите от одного набора окон - к другому набору окон. При этом содержимое любого из буферов вы можете отражать в том окне или в ином окне, как захотите. Таким образом, вы можете увидеть какой-то файл на одной вкладке, отредактировать его, потом переключиться на другую вкладку и увидеть в другом окне тот же самый файл со всеми последними изменениями.

vimdiff

Чтобы сравнить два файла между собой удобно пользоваться командой **vimdiff** :

```
$ vimdiff one.txt two.txt
```

]C – перейти на следующее место в файле, в котором есть отличия

[C – перейти на предыдущее место в файле, в котором есть отличия

do – diff obtain, получить текст из другого окна, которого не хватает в текущем

dp – diff put, записать текст из текущего окна в другое

Строки, которые повторяются в обоих файлах, сложены в одну строку, которая называется **складкой** и начинается с символов «+-». Наведя курсор на **складку**, её можно раскрывать и сворачивать используя следующие команды:

zo – open, раскрывает складку

zc – collapse, сворачивает складку

Другие команды

:set mouse=a – включить возможность позиционирования курсора в редакторе мышкой

:set nu – включить нумерацию строк (или **:set number**)

:set nonu – выключить нумерацию строк (или **:set nonumber**)

:set tabstop=4 – установить размер табуляции равный 4 символам

!command – выполнить команду UNIX не покидая редактора

:set nolist – отключить отображение скрытых символов

:set list – включить отображение скрытых символов ¹³

:= – номер текущей строки

:= – количество строк в файле

[Ctrl]+[G] имя файла, номер строки, общее число строк и положение в файле

:r file – вставить содержимое file после текущей строки

¹³спецсимволы в текущей строке: tab (î), backslash, backspace (Ĥ), newline (\$), bell (Ĝ), formfeed (Î)

:nr file – вставить содержимое **file** после строки **n**
::history – показать историю вводимых команд
::set tabpagemax=15 – установить максимальное количество вкладок равное 15

Переключение между консолью **bash** и редактором **vi**

В процессе работы с **vim** может появиться необходимость переключиться в командную оболочку **bash** и провести ряд манипуляций, для этого случая следует нажать сочетание клавиш **Ctrl**+**z**, тогда текущий экземпляр **vim** приостановится и перейдёт в список фоновых заданий. Чтобы обратно вернуться в **vim**, после необходимых манипуляций в командной оболочке **bash** необходимо посмотреть список всех фоновых заданий с помощью следующей команды:

```
$ jobs
```

И далее запустить соответствующую задачу по номеру её идентификатора *id*¹⁴ с помощью команды:

```
$ fg %1
```

При управлении фоновыми заданиями в командной оболочке **bash** доступны следующие команды:

jobs – - - список всех фоновых задач с их *id*

bg %n – - - поместить задачу с *id* равным **n** на задний план

fg %n – - - вернуть задачу с *id* равным **n** из заднего плана на передний план

kill %n – для прерывания фоновой задачи с *id* равным **n**

Ctrl+**z** – остановить текущую задачу и поместить её на задний план

Чтобы запустить сразу задачу в фоне из командной строки нужно добавить после команды символ **&**, например:

```
$ sleep 3 &
```

Но для переключения между окнами лучше всего использовать терминальный мультплексор **tmux**, о котором более подробно будет рассказано в следующей главе.

¹⁴В нашем случае с *id* равным 1