

VIM-CHEATSHEET

Тарок

АВТОРСКИЕ ЗАМЕЧАНИЯ:

Скобки не обязательны. Они написаны, чтобы не было путаницы.

РЕДАКТИРОВАНИЕ

БАЗОВЫЕ КОМАНДЫ

i[text] {insert} - вставка на позиции курсора;
I - вставка в начале строки;
a[text] {append} - вставка текста после курсора;
A - вставка текста в конце строки;
o - вставка на следующей строке;
O - вставка текста на предыдущей строке;
c[text] {change} - замена текста;
cc - замена строки;
C - то же самое, что и команда c\$;
d[text] {delete} - удаление текста;
dd - удаление строки;
D - то же самое, что и команда d\$;
x {x-out/аналогия с телетайпом} - удаление следующего символа;
X - удаление предыдущего символа;
y {yank/выдернуть} - копирование текста;
yy - копирование строки;
Y - то же самое, что и команда yy;
p {paste} - вставка на следующей строке;
P - вставка на предыдущей строке;
.
- повтор последней команды редактирования;
u {undo} - отмена последней команды;
U - отмена всех правок, которые происходили в строке, пока курсор на ней находился;
CTRL+r {redo} - отмена отмены;
s {substitute} - удаляет символ на позиции курсора и подставляет текст;
S - удаляет строку и подставляет текст;
[num] r {replace} - замена [num] следующих символов на введенный стмвол (default: [num] = 1);
R - замещает существующие символы без указания их количества;
J - объединение двух строк;
~ - смена регистра символа;
% - поиск и перемещение на ближайшую закрытую/открытую скобку (такие как (), [], { } или < >);

ПЕРЕМЕЩЕНИЕ

БАЗОВОЕ ПЕРЕМЕЩЕНИЕ

[num]h - налево на [num] символов (default: [num] = 1);
[num]j - вниз на [num] символов (default: [num] = 1);
[num]k - вверх на [num] символов (default: [num] = 1);
[num]l - направо на [num] символов (default: [num] = 1);

ПЕРЕМЕЩЕНИЕ ЭКРАНА (scrolling)

CTRL+b {back} - пролистать на экран вверх;
CTRL+u {up} - пролистать на полэкрана вверх;
CTRL+y - пролистать на строку вверх;
CTRL+e - пролистать на строку вниз;
CTRL+d - пролистать на полэкрана вниз;
CTRL+f - пролистать на экран вниз;
z+- - пролистать экран наверх относительно курсора;
z+. - выровнить экран по центру относительно курсора;
z+ENTER - пролистать экран вниз относительно курсора;

ПЕРЕМЕЩЕНИЕ КУРСОРА ПО СТРОКЕ

^ {hat} - переход на первый непустой символ строки;
0 - переход на первый символ строки;
\$ - переход на последний непустой символ строки;
b {beggining} - перейти назад в начало слова с учетом знаков препинания;
B - перейти назад в начало слова без учета знаков препинания;
w - перейти вперед на начало слова с учетом знаков препинания;
W - перейти вперед на начало слова без учета знаков препинания;
e {end} - перейти в конец слова с учетом знаков препинания;
E - перейти в конец слова без учета знаков препинания;
[num]| - переход на [num] столбец текущей строки (default: [num] = 1);

ПОИСК СИМВОЛА В ТЕКУЩЕЙ СТРОКЕ

f[sym] {find} - переход в строке вперед на символ [sym];
F[sym] - переход в строке назад на символ [sym];
t[sym] - переход в строке вперед перед символом [sym];
T[sym] - переход в строке назад после символа [sym];
; - повтор поиска в том же направлении, что и команда;
, - повтор поиска в противоположном направлении, что и команда;

ГЛОБАЛЬНОЕ ПЕРЕМЕЩЕНИЕ КУРСОРА

[num]H {[num] High} - переход в начало экрана на [num] строк ниже самой верхней строки без scrolling (default: [num] = 0);
M (Middle) - переход в середину экрана без scrolling;
[num]L {[num] Low} - переход в конец экрана на [num] строк вверх самой нижней строки без scrolling (default: [num] = 0);
[num]gg {[num] go go} - переход на [num] строку вниз относительно первой строки (default: [num] = 0);
[num]G ([num] Go To) - переход на [num] строку вверх относительно последней строки (default: [num] = last line);
- - переход на начало предыдущей строки;
+ - переход на начало следующей строки;
CTRL+g - вывод текущего положения в файле;

ОБЩИЙ ВИД КОМАНД С СОЧЕТАНИЕМ ИХ ПОВТОРЕНИЙ:

[number][command][text object]/[command][number][text object], где:
[number] - необязательный числовой аргумент,
[command] - команда редактирования (например, c, d или y),
[text object] - команда перемещения;
Например, команды
cH или dH или yH
cG или dG или yG
c13G или d13G или y13G

РАБОТА С ФАЙЛАМИ

ПОИСК ПО ФАЙЛУ

/[pattern] - поиск [pattern] вперед;
?[pattern] - поиск [pattern] назад;
n {next} - поиск в том же направлении, что и команда;
N - поиск в отличном направлении, что и команда;
/ - повтор поиска вперед;
? - повтор поиска назад;

ОТКРЫТИЕ ФАЙЛА

vi +[num] [file] - открытие файла на n-ой строке;
vi + [file] - открытие файла на последней строке;
vi +[/pattern] [file] - открыти файла на совпавшем [/pattern];
view/(vi -R) - открытие файла в read only режиме;
vi -r - список всех файлов, сохраненных системой в текущей дерриктории (*.swp формат);
vi -r [file] - восстановление файла из буфера с помощью файла подкачки;

СОХРАНЕНИЕ И ВЫХОД

ZZ - сохранение и выход;

РАБОТА С БУФФЕРАМИ

ТЕКСТОВЫЕ БУФФЕРЫ

"[(1-9)/(a-z)][P/p] - вставка из буфера, где [1-9] - последние изменения, а [a-z] - сохраненные в буфер изменения;
"[(a-Z)][num][command] - если [a-z], то буфер перезаписывается, а [A-Z] - в буфер добавляются символы;
"*[command] - работа с PRIMARY CLIPBOARD (copy-on-select) регистром;
"+[command] - работа с CLIPBOARD (CTRL+C/CTRL+V) регистром;

ОКОННЫЕ БУФФЕРЫ

:ls[!]/:files[!]/:buffers[!] - вывод списка буферов (! дает более подробную информацию);
:edit [buffer] - редактирование буфера [buffer];
:windo [cmd] - применение к видимым окнам текущей вкладки команду [cmd];
:bufdo [!][cmd] - применение ко всем окнам текущей вкладки команду [cmd];
:ball/:sball - редактировать все файлы в списке аргументов vim (sball откроет в новых окнах) (вспомни про команду args);
:unhide/:sunhide - редактировать все загруженные буферы (sunhide откроет в новых окнах);
:badd [file] - добавляет файл [file] в список буферов;
:bunload[!] - выгружает буфер из памяти (! без изменений);
:bdelete[!] - выгружает буфер и удаляет его из списка буферов (! без сохранений);
:buffer [n]/:sbuffer [num] {split buffer} - переход в буфер [n] (sbuffer откроет в новом окне);
:bnext [n]/:sbnext [n] {split buffer next} - переход в следующий по порядку n-й буфер (sbnext откроет в новом окне);
:bNext [n]/:sbNext [n] - переход в n-й предыдущий буфер (sbNext откроет новое окно);
:bfirst/:sbfirst {split {buffer first}} - переход к первому буферу (sbfirst откроет новое окно);

`:blast/:sblast {split {buffer last}}` - переход к последнему буферу (`sblast` открывает новое окно);
`:bmod [n]/:sbmod [n]` - переход к n-му измененному буферу (`sbmod` открывает новое окно);

Примечание:

Если добавить `:vertical` перед переходом с разделением в новый буффер, то окно будет разделено вертикально.

ОСНОВНЫЕ ФЛАГИ

`u` - неотображаемый буфер (увидеть можно через `!`);
`%` - буфер текущего окна;
`#` - буфер, в который можно переключиться;
`a` - активный буфер;
`h` - скрытый буфер;
`[-]` - откл. опция `modifiable` (можно только читать);
`=` - буфер нельзя сделать редактируемым;
`+` - буфер изменен;
`x` - буфер содержит ошибки чтения;

СПЕЦ. БУФФЕРЫ

`quickfix` - буфер, содержащий список ошибок при компиляции;
`help` - файлы справки `vim`;
`directory` - список файлов в каталоге;
`scratch` - текст для общих целей;

ВКЛАДКИ

`:tabnew [file]` - открывает новую вкладку;
`:tabclose` - закрывает текущую вкладку;
`:tabonly` - закрывает все остальные вкладки;
`:tabnext/[CTRL + PAGE DOWN]` - переход на следующую вкладку;
`:tabprevious/[CTRL + PAGE UP]` - переход на предыдущую вкладку;

АВТОЗАВЕРШЕНИЕ (в insert mode)

БАЗОВЫЕ КОМАНДЫ

`CTRL-X + [type autocompletion]` - стандартный шаблон автозавершения, где:
`[type autocompletion]` - определяет тип завершения,
`CTRL-N (next)` - выбор след. варианта завершения,
`CTRL-P (previous)` - выбор пред. варианта завершения;

ТИПЫ ЗАВЕРШЕНИЯ

`CTRL-L {line}` - завершение строки;
`CTRL-N` - по ключевому слову из тек. файла;
`CTRL-K` - по словарю (`.mydict` - персональный список слов);
`CTRL-T {thesaurus}` - по тезаурусу;
`CTRL-I` - по ключевому слову в текущем файле и подключаемых внешних файлах;
`CTRL-]` - по тегу в текущем и включенных файлах;
`CTRL-F {file}` - по имени файла только в тек. каталоге;
`CTRL-D {define}` - по макросу в текущем и внешних файлах (`#define`);
`CTRL-V` - по командам `vim`;
`CTRL-U` - по пользовательским функциям (`completefunc`);
`CTRL-O {omni}` - использует файл `omni`-завершения для определенных типов

файлов;
CTRL-S - для исправления орфографии;

МНОГООКОННОСТЬ

РАБОТА С ОКНАМИ

(:[n]split [+opt] [+cmd] [file])/(CTRL-W + s) - разделяет окно на [n] частей, курсор помещается в новое окно
(:[n]vsplit [+opt] [+cmd] [file])/(CTRL-W + v) - то же, что и :split, но вертикально
(:[n]new [+opt] [+cmd])/(CTRL-W + n) - то же, что и split, но буфер безымянный
:[n]vnew [+opt] [+cmd] - то же, что и :new, но вертикально
:[n]sview [+opt] [+cmd] [file] - то же, что и :split, но readonly
:[n]sfind [+opt] [+cmd] [file] - разделяет окно и открывает [file] в новом окне. file ищет в path, где [n] - количество отображаемых строк в окне,
[opt] - опции vim в сеанс работы с новым окном,
[cmd] - команда, которую надо выполнить в новом окне,
[file] - файл, который надо открыть;

КОМАНДЫ ПЕРЕХОДА

CTRL-W + j - переход на окно ниже;
CTRL-W + k - переход на окно выше;
CTRL-W + h - переход на левое окно;
CTRL-W + l - переход на правое окно;
CTRL-W + w - циклический переход на следующее снизу или справа окно;
CTRL-W + W - циклический переход на следующее сверху или слева окно;
CTRL-W + t {top} - переход в верхнее левое окно;
CTRL-W + b {bottom} - переход в нижнее правое окно;
CTRL-W + p - переход на предыдущее окно;

Примечание:

Мнемоника CTRL-W \longleftrightarrow Window

ПЕРЕМЕЩЕНИЕ ОКОН

CTRL-W + r - циклический сдвиг окна вправо и вниз;
CTRL-W + R - циклический сдвиг налево вверх;
[n]CTRL-W + x - смена двух окон в ряду или столбце;

ПЕРЕМЕЩЕНИЕ И ПЕРЕФОРМАТИРОВАНИЕ

CTRL-W + K - перемещение текущее окно на верх экрана, распаивая его во всю ширину;
CTRL-W + J - перемещение текущего окна в низ экрана, распаивая его во всю ширину;
CTRL-W + H - перемещение текущее окно в левое положение экрана, распаивая его во всю высоту;
CTRL-W + L - перемещение текущее окно в правое положение экрана, распаивая его во всю высоту;
CTRL-W + T - перемещение текущее окно в новую существующую вкладку;

ИЗМЕНЕНИЕ РАЗМЕРА ОКНА

CTRL-W + = - выравнивание всех окон (согласно winheight/winwidth);
CTRL-W + - - уменьшает высоту текущего окна на одну строку;
:resize -[n] - уменьшает размер окна по горизонтали на [n] строк;
CTRL-W + + - увеличивает высоту текущего окна на одну строку;

:resize +[n] - увеличивает размер окна по горизонтали на [n] строк;
z[n] + ENTR - устанавливает высоту текущего окна на [n] строк;
CTRL-W + < - уменьшает ширину окна на один столбец;
CTRL-W + > - увеличивает ширину окна на один столбец;
:vertical resize [n] - устанавливает ширину текущего окна на [n] столбцов;
CTRL-W + | - делает максимальным размер окна;

РАБОТА С МЕТКАМИ

БАЗОВЫЕ КОМАНДЫ

m[a-Z] {mark} - поставить метку на текущую позицию;
'[mark] - поместиться на начало строки с меткой [mark];
'[mark] - поместиться на символ с меткой [mark];
' - возвращает на начало строки, содержащей предыдущую метку;
" - возвращает на предыдущую метку;

РЕДАКТОР EX

БАЗОВЫЕ КОМАНДЫ

p {print} - печатает строки;
d {delete} - удаляет строки;
m {move} - перемещает строки;
co/t {copy} - копирует строки;
ya {yank} - копирует строки (можно использовать с именованными буферами);
pu {put} - вставляет строки (можно использовать с именованными буферами);
j {join} - объединяет строки;

АДРЕСАЦИЯ СТРОК В EX

ЯВНОЕ/ШАБЛОННОЕ УКАЗАНИЕ ДИАПАЗОНА СТРОК

: [num] - переход на [num] строку
: [(start)/(/pattern1/)], [(end)/(/pattern2/)] [command], где
[start] и [end] - числовой диапазон строк,
[pattern1] и [pattern2] - шаблоны строк,
[command] - команда ex;

СИМВОЛЫ АДРЕСАЦИИ СТРОК

. - текущая строка;
\$ - последняя строка файла;
% - все строки;

ОТНОСИТЕЛЬНЫЕ ОПЕРАЦИИ АДРЕСАЦИИ

+ [num] - прибавит к номеру строки число [num];
- [num] - убавит от номера строки число [num];

ВСПОМОГАТЕЛЬНЫЕ КОМАНДЫ АДРЕСАЦИИ

:= - вывод числа строк;
: .= - вывод текущей строки; : /pattern / - номер следующей строки, содержащей pattern;

Примечание:

Можно использовать (;) вместо (,), чтобы разделить указание диапазона от переопределения текущего положения;

ГЛОБАЛЬНЫЙ ПОИСК

:g - ищет глобально по шаблону и выводит найденные строки;
:g!/v - противоположность g;

СОЧЕТАНИЯ КОМАНД EX

- разделитель команд;

СОХРАНЕНИЕ И ВЫХОД

:w[rite] [file] - сохраняет буфер в [file] и остается в текущем редактируемом файле (default: [file] - редактируемый в данный момент файл);
:sav[eas] [file] - сохраняет текущий буфер в [file] и открывает буфер файла [file] (default: [file] - редактируемый в данный момент файл);
:q[uit] - выход из редактора;
:wq/x - запись и выход из файла;
:r[ead] [file] - вставка текста из [file] в текущий буфер;

Примечание:;

Чтобы игнорировать системные предупреждения, нужно поставить символ ! в конце команды;

РЕДАКТИРОВАНИЕ НЕСКОЛЬКИХ ФАЙЛОВ

КОМАНДЫ VI

vi [file1] [file2] - вызов редактирования двух файлов;

КОМАНДЫ EX

:n - вызов следующего файла;
:args - перечисление всех файлов, присутствующих в командной строке;
:rewind - делает текущим первый файл;
(:edit [file])/(CTRL+^^) - создать буфер и скопировать в него текст из [file];
":edit! - пересоздаст текущий буфер (НЕ РАБОТАЕТ, ИЩИ В УЧЕБНИКЕ);
% - обращение к текущему файлу (выводит его имя);
- обращение к альтернативному файлу (выводит его имя);

ГЛОБАЛЬНАЯ ЗАМЕНА

ОСНОВНЫЕ КОМАНДЫ

s (substitute) - подстановка;
g (global) - глобальность;
c (confirm) - подтверждение глобальной замены;

ШАБЛОН БАЗОВОГО ИСПОЛЬЗОВАНИЯ

:[range][command], где;
[range] - может быть как числовой диапазон, так и диапазон паттернов;
[command] - любая команда(ы) EX;
Примеры использования:
1):%s/[pattern1]/[pattern2]/g;
2):g/[pattern1]/s/[pattern2]/[pattern3]/g;
3):g/[pattern1]/s//[pattern3]/g ([pattern2] = [pattern1] = blank);

МЕТАСИМВОЛЫ, ИСПОЛЬЗУЕМЫЕ ПРИ ПОИСКЕ

. – любой одиночные символ, отличный от перевода строки;
* – повторение любого количества раз символа, стоящего перед звездочкой (включая, что символ может отсутствовать);
^ – требование нахождения в начале строки;
\$ – требование нахождения в конце строки;
_ – экранирующий символ;
[...] – любой символ, заключенный в квадратные скобки;
\\(\\) – сохранение шаблона, заключенного между скобками в (специальном месте)/(временном буфере, обращаясь через \\[num]);
< > – указание на начало (<) или конец (>) слова;
~ – замена на последнее регулярное выражение;

МЕТАСИМВОЛЫ, ИСПОЛЬЗУЕМЫЕ ПРИ ЗАМЕНЕ

\\n – замена текста, соответствующего n-му шаблону;
_ – экранирующий символ;
& – замена на текст из поиска;
~ – найденная строка заменяется на текст, определенный в последней команде подстановки;
(\\u)/(\\l) – замена следующего символа на прописной или строчной, соответственно;
(\\U)/(\\L) и (\\e)/(\\E) – такие же действия как и у \\u или \\l, только действуют, пока не встретят \\e или \\E или конец строки замены;

ТРЮКИ ПРИ ЗАМЕНАХ

1. `:s ⇔ :s//~/;`
2. `& ⇔ :&;`
3. Кроме / можно использовать любой неалфавитно-цифровой символ в качестве разделитель, кроме (, “, |);

ПРИМЕРЫ ИСПОЛЬЗОВАНИЯ РЕГУЛЯРНЫХ ВЫРАЖЕНИЙ

- `:%s/child\\([,.;:!?]\\)/children\\1/g` – \\1 заменит именно на тот символ, что был в квадратных скобках;
- `:%s/<child>/children/g` – отыщет только целое child;
- `:%s/\\(.*\\) *$/\\1/g` – удалит пробелы в конце строки;

НАСТРОЙКА РЕДАКТОРА

Файл `vimrc` – обрабатывается редактором `ex` перед переходом в `vi`;

БАЗОВЫЕ КОМАНДЫ

`:so [file] {source}` – использовать команды из файла `[file]` (МОЖНО ИСПОЛЬЗОВАТЬ В КАЧЕСТВЕ СКРИПТА `EX`);

Примечание:;

Так можно в главный `vimrc` импортировать другие настройки;

КОМАНДА SET

`:set [no][option][=num]` установка опции `vi`, где
[no] – бинарный переключатель (если есть – 1, нет – 0);
`:set all` – все установки, используемые `vi`;

:set [option]? - вывод значения параметра [option];
:set - все заданные опции, включая те, что были заданы в текущей секции;

ОПЦИИ

ОТСТУПЫ

tabstop - количество пробелов, которыми отображается символ табуляции в тексте (табуляция - управляющий символ, а не пробел);
softtabstop - количество пробелов, которыми отображается символ табуляции при добавлении;
shiftwidth - ширина отступов, добавляемых командами » и «;
smarttab - если опция включена, то приведет к добавлению отступа при нажатии tab в начале строки, равному shiftwidth;
expandtab - в режиме вставки заменяет табуляцию на пробелы;
autoindent - копирует отступы с текущей строки в следующую;
smartindent - autoindent + учитывает особенности синтаксиса со скобками;

ВЫЗОВ UNIX КОМАНД ИЗ VI

:[command] - восклицательный знак предписывает ex создать оболочку и выполнить команду [command];
:sh - создание оболочки sh (выход командой CTRL+d);

Примечание: вызов команд оболочки можно сочетать с :read, вставляя результат выполнения команды;

АББРЕВИАТУРЫ

- последовательности, автоматически расшифровывающиеся в режиме вставки;
:ab [abbr] [phrase], где;
[abbr] - аббревиатура для указанной фразы [phrase];
:unab [abbr] - отмена аббревиатуры;
:ab - отображение всех аббревиатур;

ОТОБРАЖЕНИЕ КЛАВИШ (mapping)

:map [keyboard seq] [command seq] - макрос для командного режима;
:unmap [keyboard seq] - отмена макроса с [keyboard seq] для командного режима;
:map - список [keyboard seq], для которых есть отображение;
:map! [keyboard seq] [command seq] - макрос для режима вставки;
:unmap! [keyboard seq] - отмена макроса с [keyboard seq] для режима вставки;
:map #[number] [command seq] - отображение функциональной клавиши F+[number] в [command seq];
:@[name buffer] - выполнение команды, содержащейся в буфере [name buffer] (@-функция);
CTRL+V + [ENTER/ESC/BACKSPACE/DELETE/...] - экранирование управляющих клавиш;

КОНТРОЛЬ ЗА ОТСТУПАМИ

:set autoindent - повторения отступа, что и на предыдущей;
CTRL+t - при включенном autoindent переводит курсор на след. уровень (в режиме вставки);

CTRL+d - при включенном autoindent переводит курсор на пред. уровень (в режиме вставки);

Примечание:

^ + CTRL+d - перемещение курсора на начало строки, но только для текущей;

0 + CTRL+d - перемещение курсора на начало строки с изменением уровня отступа;

[num] » - смещение на [num] строк на \tab вперед;

[num] « - смещение на [num] строк на \tab назад;

:retab! - замена начальных пробелов на табы;

ТЕГИ

!:ctags [file] - создания файла tags, содержащего данные о местоположении импортируемой функции из файла [file];

(:tag [function name])/^ - поиск в файле tags местоположение функции [function name] и перемещение курсора на ее определение;

CTRL+t - возвращение к сохраненному положению в стеке тегов;

СТЕКИ ТЕГОВ

:tag[!] [tagstring] - редактирование файла, содержащего [tagstring], как задано в файле тегов;

:[count]tag[!] - переход на запись под номером [count] в стеке тегов;

:[count]pop[!] - извлекает позицию курсора из стека, восстанавливая предыдущую позицию на [count] назад;

:tags - содержимое стека тегов;

:tselect[!] [tagstring] - вывод списка тегов, соответствующих [tagstring], используя информацию из файла тегов;

:stselect[!] [tagstring] - такая же, как и :tselect, но разделяет окно;

:[count]tnext[!] - переход на [count]-ый следующий тег;

:[count]tNext[!] - переход на [count]-ый предыдущий тег;

:[count]tlast[!] - переход на последующий [count] тег;

:[count]trewind[!] - переход на [count]-ый тег;

СВЕРТКИ (folding)

:mkview - вызов сохраненных свертки;

:loadview - сохранение свертки;

zf[move] - создать свертку с тек. строки до той, куда переместит след. команда перемещения (manual);

[count]zf - свертка, охватывающая [count] строк (manual);

zi - переключение опции foldenable (zn, zN - вкл., выкл. соотв.);

zM \iff foldlevel = 0;

zm, zr - декремент (zm) или инкремент (zr) foldlevel;

zj, zk - прыжок на след. (zj) или пред. (zk) свертку;

za - переключение состояния одной свертки;

zo (open) - открывает одну свертку;

zc (close) - закрывает одну свертку;

zd (delete) - удаляет одну свертку;

zA - переключение состояния свертки (скрытая/раскрытая), рекурсивно;

zO (open) - открывает свертку, рекурсивно;

zC (close) - закрывает свертку, рекурсивно;

zD (delete) - удаление свертки, рекурсивно;

zE - удаление всех свертки в файле;

МЕТАСИМВОЛЫ РАСШИРЕННЫХ РЕГУЛЯРНЫХ ВЫРАЖЕНИЙ (ERE)

- нельзя использовать в vim;
[str1]|[str2] - разделяет несколько возможных вариантов [str1] и [str2];
(...) - создание группы, к которой можно применять другие операторы (можно обращаться также через нумерованный буфер);
+ - соотв. одному или нескольким предшествующим регулярным выражениям;
? - ноль и одно вхождение предшествующего регулярного выражения;
{...} - интервальное выражение;

СКРИПТЫ

:source [script] - запуск скрипта [script] из vi;
ex -s [vi file] < [script] - применение скрипта [script] к [vi file] из оболочки (-s подавляет вывод ex на терминал);

МАКРОСЫ