VIM命令大全  
**光标控制命令**  
命令                   光标移动  
h                   向左移一个字符  
j                   向下移一行  
k                   向上移一行  
l                   向右移一个字符  
**G                   移到文件的最后一行**

**nG 移动到这个文件的第n行。例如20G，则会移动到这个文件的第20行（可配合:set nu）**

**n<Enter> 光标向下移动n行**

**n<space> 按下数字后再按空格键，光标会向右移动这一行的n个字符。例如20<space>，则光标会向右移动20个字符**w                   移到下一个字的开头  
W                   移到下一个字的开头，忽略标点符号  
b                   移到前一个字的开头  
B                   移到前一个字的开头，忽略标点符号  
L                   移到屏幕的最后一行  
M                   移到屏幕的中间一行  
H                   移到屏幕的第一行  
e                   移到下一个字的结尾  
E                   移到下一个字的结尾，忽略标点符号  
(                       移到句子的开头  
)                       移到句子的结尾  
{                       移到段落的开头  
}                       移到下一个段落的开头  
**0(数字), |                移到当前行的第一列**  
^                       移到当前行的第一个非空字符  
**$                       移到当前行的最后一个字符**  
+, Enter             移到下一行的第一个字符  
-                       移到前一行的第一个非空字符  
**在 vi 中添加文本**  
命令                   插入动作  
**a                   在光标后插入文本  
A                   在当前行插入文本  
i                   在光标前插入文本  
I                   在当前行前插入文本  
o                   在当前行的下边插入新行  
O                   在当前行的上边插入新行**  
s                   删除光标所在处字符，并进入插入模式  
S                   删除光标所在的行，并进入插入模式  
:r file             读入文件 file 内容，并插在当前行后  
:nr file            读入文件 file 内容，并插在第 n 行后  
Esc                 回到命令模式  
^v char             插入时忽略 char 的指定意义，这是为了插入特殊字符  
**在 vi 中删除文本**  
命令                   删除操作  
x                   删除光标处的字符  
dw                  删至下一个字的开头  
dG                  删除行，直到文件结束  
**dd                  删除整行**

**ndd 删除光标所在列的向下n列，例如，20dd则是删除20列**  
db                  删除光标前面的字  
:n,md               从第 m 行开始往前删除 n 行  
d, d$               从光标处删除到行尾  
^h, backspace       插入时，删除前面的字符  
^w                  插入时，删除前面的字  
**修改 vi 文本**  
每个命令前面的数字表示该命令重复的次数  
命令                               替换操作  
rchar                         用 char 替换当前字符  
R text escape                 用 text 替换当前字符直到按下 Esc 键  
stext escape                  用 text 代替当前字符  
S 或 cctext escape             用 text 代替整行  
cwtext escape                 将当前字改为 text  
Ctext escape                  将当前行余下的改为 text  
cG escape                     修改至文件的末尾  
ccursor\_cmd text escape       从当前位置处到光标命令位置处都改为 text  
**在 vi 中查找与替换**  
命令                               查找与替换操作  
**/text                         在文件中向前查找 text**  
?text                         在文件中向后查找 text  
n                             在同一方向重复查找  
N                             在相反方向重复查找  
ftext                         在当前行向前查找 text  
Ftext                         在当前行向后查找 text  
ttext                         在当前行向前查找 text，并将光标定位在 text 的第一个字符  
Ttext                         在当前行向后查找 text，并将光标定位在 text 的第一个字符  
:set ic                       查找时忽略大小写  
:set noic                     查找时对大小写敏感  
:ranges/pat1/pat2/g           用 newtext 替换 oldtext  
**:m,ns/oldtext/newtext         在 m 行通过 n，用 newtext 替换 oldtext**

**:1,$s/word1/word2/g 从第一行到最后一行查找word1字符串，并将该字符串替换为word2**

**:1,$s/word1/word2/gc 从第一行到最后一行查找word1字符串，并将该字符串替换为word2，且在替换前显示提示符让用户确认（conform）（常用）**&                                  重复最后的 :s 命令  
:g/text1/s/text2/text3        查找包含 text1 的行，用 text3 替换 text2  
:g/text/command               在所有包含 text 的行运行 command 所表示的命令  
:v/text/command               在所有不包含 text 的行运行 command 所表示的命令  
**在 vi 中复制文本**  
命令                      复制操作  
**yy                    将当前行的内容放入临时缓冲区**  
**nyy                   将 n 行的内容放入临时缓冲区**  
**p                     将临时缓冲区中的文本放入光标后**  
**P                     将临时缓冲区中的文本放入光标前**  
"(a-z)nyy             复制 n 行放入名字为圆括号内的可命名缓冲区，省略 n 表示当前行  
"(a-z)ndd             删除 n 行放入名字为圆括号内的可命名缓冲区，省略 n 表示当前行  
"(a-z)p               将名字为圆括号的可命名缓冲区的内容放入当前行后  
"(a-z)P               将名字为圆括号的可命名缓冲区的内容放入当前行前  
**在 vi 中撤消与重复**  
命令               撤消操作  
**u               撤消最后一次修改**  
U               撤消当前行的所有修改  
.                  重复最后一次修改  
,                  以相反的方向重复前面的 f、F、t 或 T 查找命令  
;                  重复前面的 f、F、t 或 T 查找命令  
"np             取回最后第 n 次的删除(缓冲区中存有一定次数的删除内容，一般为 9)  
n               重复前面的 / 或 ? 查找命令  
N               以相反方向重复前面的 / 或 ? 命令  
**保存文本和退出 vi**  
命令                      保存和/或退出操作  
:w                    保存文件但不退出 vi  
:w file               将修改保存在 file 中但不退出 vi  
:wq 或 ZZ 或 :x         保存文件并退出 vi  
:q!                   不保存文件，退出 vi  
:e!                   放弃所有修改，从上次保存文件开始再编辑  
**vi 中的选项**  
选项                      作用  
:set all              打印所有选项  
:set nooption         关闭 option 选项  
**:set nu               每行前打印行号**:set showmode         显示是输入模式还是替换模式  
:set autoindent          继承前一行的缩进方式，特别适用于多行注释  
:set smartindent        为C程序提供自动缩进  
:set list             显示制表符(^I)和行尾符号  
:set ts=8             为文本输入设置 tab stops  
:set window=n         设置文本窗口显示 n 行  
:set number           显示行数  
:set nonumber         取消显示行数  
**vi 的状态**  
选项                   作用  
:.=                     打印当前行的行号  
:=                     打印文件中的行数  
ctrl+g             显示文件名、当前的行号、文件的总行数和文件位置的百分比  
:l                 使用字母 "l" 来显示许多的特殊字符，如制表符和换行符  
**在文本中定位段落和放置标记**  
选项                      作用  
{                         在第一列插入 { 来定义一个段落  
[[                         回到段落的开头处  
]]                         向前移到下一个段落的开头处  
m(a-z)                用一个字母来标记当前位置，如用 mz 表示标记 z  
'(a-z)                将光标移动到指定的标记，如用 'z 表示移动到 z  
**在 vi 中连接行**  
选项                      作用  
J                     将下一行连接到当前行的末尾  
nJ                    连接后面 n 行  
**光标放置与屏幕调整**  
选项                      作用  
H                     将光标移动到屏幕的顶行  
nH                    将光标移动到屏幕顶行下的第 n 行  
M                     将光标移动到屏幕的中间  
L                     将光标移动到屏幕的底行  
nL                    将光标移动到屏幕底行上的第 n 行  
^e(ctrl+e)            将屏幕上滚一行  
ctrl+y                将屏幕下滚一行  
ctrl+u                将屏幕上滚半页  
ctrl+d                将屏幕下滚半页  
ctrl+b                将屏幕上滚一页  
ctrl+f                将屏幕下滚一页  
ctrl+l                重绘屏幕  
z-return              将当前行置为屏幕的顶行  
nz-return             将当前行下的第 n 行置为屏幕的顶行  
z.                    将当前行置为屏幕的中央  
nz.                   将当前行上的第 n 行置为屏幕的中央  
z-                    将当前行置为屏幕的底行  
nz-                   将当前行上的第 n 行置为屏幕的底行  
**vi 中的 shell 转义命令**  
选项                      作用  
:!command             执行 shell 的 command 命令，如 :!ls  
:!!                         执行前一个 shell 命令  
:r!command            读取 command 命令的输入并插入，如 :r!ls 会先执行 ls，然后读入内容  
:w!command            将当前已编辑文件作为 command 命令的标准输入并执行 command 命令，如 :w!grep all  
:cd directory         将当前工作目录更改为 directory 所表示的目录  
:sh                   将启动一个子 shell，使用 ^d(ctrl+d) 返回 vi  
:so file              在 shell 程序 file 中读入和执行命令  
**vi 中的宏与缩写**  
(避免使用控制键和符号，不要使用字符 K、V、g、q、v、\*、= 和功能键)  
选项                      作用  
:map key command\_seq 定义一个键来运行 command\_seq，如 :map e ea，无论什么时候都可以 e 移到一个字的末尾来追加文本  
:map                  在状态行显示所有已定义的宏  
:umap key             删除该键的宏  
:ab string1 string2   定义一个缩写，使得当插入 string1 时，用 string2 替换 string1。当要插入文本时，键入 string1 然后按 Esc 键，系统就插入了 string2  
:ab                   显示所有缩写  
:una string           取消 string 的缩写  
**在 vi 中缩进文本**  
选项                     作用  
ctrl+i或 tab     插入文本时，插入移动的宽度，移动宽度是事先定义好的  
:set ai               打开自动缩进  
:set sw=n             将移动宽度设置为 n 个字符  
n>                   使 n 行都向右移动一个宽度，例如 3>> 就将接下来的三行每行都向右移动一个移动宽度