VIM 教程　版本 1.5

vim 是一个具有很多命令的功能非常强大的编辑器。限于篇幅，在本教程当中就不详细介绍了。本教程的设计目标是讲述一些必要的基本命令，而掌握好这些命令，您就能够很容易将 vim 当作一个通用的万能编辑器来使用了。

完成本教程的内容大约需要 25-30 分钟，取决于您训练的时间。

每一节的命令操作将会更改本文。推荐您复制本文的一个副本，然后在副本上进行训练(如果您是通过"vimtutor"来启动教程的，那么本文就已经是副本了)。

切记一点：本教程的设计思路是在使用中进行学习的。也就是说，您需要通过执行命令来学习它们本身的正确用法。如果您只是阅读而不操作，那么您可能会很快遗忘这些命令的!

好了，现在请确定您的 Shift-Lock(大小写锁定键)还没有按下，然后按键盘上的字母键 j 足够多的次数来移动光标，直到第一节的内容能够完全充满屏幕。

第一讲 第一节：移动光标

※※ 要移动光标，请依照说明分别按下 h、j、k、l 键。 ※※

```
    ^
    k          提示： h 的键位于左边，每次按下就会向左移动。
< h   l >             l 的键位于右边，每次按下就会向右移动。
    j                 j 键看起来很象一支尖端方向朝下的箭头。
    v
```

1. 请随意在屏幕内移动光标，直至您觉得舒服为止。
2. 按下下行键(j)，直到出现光标重复下行。

提示：如果您不敢确定您所按下的字母，请按下<ESC>键回到正常(Normal)模式。然后再次从键盘输入您想要的命令。

提示：光标键应当也能正常工作的。但是使用 hjkl 键，在习惯之后您就能够快速地在屏幕内四处移动光标了。

第一讲 第二节：VIM 的进入和退出

!! 特别提示：敬请阅读完整本一节的内容，然后才能执行以下所讲解的命令。

1. 请按<ESC>键(这是为了确保您处在正常模式)。
2. 然后输入：　　:q!〈回车〉

---> 这种方式的退出编辑器绝不会保存您进入编辑器以来所做的改动。如果您想保存更改再退出，请输入：

:wq 〈回车〉

3. 如果您看到了命令行提示符，请输入能够带您回到本教程的命令，那就是：

vimtutor〈回车〉

通常情况下您也可以用这种方式：

vim tutor〈回车〉

---> 这里的 'vim' 表示进入 vim 编辑器，而 'tutor' 则是您准备要编辑的文件。

4. 如果您自信已经牢牢记住了这些步骤的话，请从步骤 1 执行到步骤 3 退出，然后再次进入编辑器。接着将光标移动到第一讲第三节来继续我们的教程讲解。

第一讲 第三节：文本编辑之删除

** 在正常(Normal)模式下，可以按下 x 键来删除光标所在位置的字符。**

1. 请将光标移动到本节中下面标记有 ---> 的那一行。
2. 为了修正输入错误，请将光标移至准备删除的字符的位置处。
3. 然后按下 x 键将错误字符删除掉。
4. 重复步骤 2 到步骤 4，直到句子修正为止。

---> The ccow jumpedd ovverr thhe mooon.

5. 好了，该行已经修正了，下一节内容是第一讲第四节。

特别提示：在您浏览本教程时，不要强行记忆。记住一点：在使用中学习。

第一讲 第四节：文本编辑之插入

** 在正常模式下，可以按下 i 键来插入文本。**

1. 请将光标移动到本节中下面标记有 ---> 的第一行。
2. 为了使得第一行内容雷同于第二行，请将光标移至文本第一个字符准备插入的位置。
3. 然后按下 i 键，接着输入必要的文本字符。
4. 所有文本都修正完毕，请按下 <ESC> 键返回正常模式。重复步骤 2 至步骤 4 以便修正句子。

---> There is text misng this .

---> There is some text missing from this line.

5. 如果您对文本插入操作已经很满意，请接着阅读下面的小结。

第一讲 小结

1. 光标在屏幕文本中的移动既可以用箭头键，也可以使用 hjkl 字母键。
   h（左移） j（下行） k（上行） l（右移）
2. 欲进入 vim 编辑器(从命令行提示符)，请输入：vim 文件名〈回车〉
3. 欲退出 vim 编辑器，请输入以下命令放弃所有修改：
   〈ESC〉 :q! 〈回车〉

或者输入以下命令保存所有修改：

   〈ESC〉 :wq 〈回车〉

4. 在正常模式下删除光标所在位置的字符，请按： x
5. 在正常模式下要在光标所在位置开始插入文本，请按：
   i 输入必要文本 〈ESC〉

特别提示：按下〈ESC〉键会带您回到正常模式或者取消一个不期望或者部分完成的命令。

好了，第一讲到此结束。下面接下来继续第二讲的内容。

第二讲 第一节：删除类命令

** 输入 dw 可以从光标处删除至一个单字/单词的末尾。**

1. 请按下 <ESC> 键确保您处于正常模式。

2. 请将光标移动到本节中下面标记有 ---> 的那一行。

3. 请将光标移至准备要删除的单词的开始。

4. 接着输入 dw 删除掉该单词。

特别提示：您所输入的 dw 会在您输入的同时出现在屏幕的最后一行。如果您输入有误，请按下 <ESC> 键取消，然后重新再来。

---> There are a some words fun that don't belong paper in this sentence.

5. 重复步骤 3 至步骤 4，直至句子修正完毕。接着继续第二讲第二节内容。

第二讲 第二节：其他删除类命令

** 输入 d$ 从当前光标删除到行末。**

1. 请按下 <ESC> 键确保您处于正常模式。

2. 请将光标移动到本节中下面标记有 ---> 的那一行。

3. 请将光标移动到该行的尾部(也就是在第一个点号 '.' 后面)。

4. 然后输入 d$ 从光标处删至当前行尾部。

---> Somebody typed the end of this line twice. end of this line twice.

5. 请继续学习第二讲第三节就知道是怎么回事了。

第二讲 第三节：关于命令和对象

删除命令 d 的格式如下：

[number] d object 或者 d [number] object

其意如下：

number - 代表执行命令的次数(可选项，缺省设置为 1)。

d - 代表删除。

object - 代表命令所要操作的对象(下面有相关介绍)。

一个简短的对象列表：

w - 从当前光标当前位置直到单字/单词末尾，包括空格。

e - 从当前光标当前位置直到单字/单词末尾，但是 *不* 包括空格。

$ - 从当前光标当前位置直到当前行末。

特别提示：

对于勇于探索者，请在正常模式下面仅按代表相应对象的键而不使用命令，则将看到光标的移动正如上面的对象列表所代表的一样。

第二讲 第四节：对象命令的特殊情况

** 输入 dd 可以删除整一个当前行。 **

鉴于整行删除的高频度，VIM 的设计者决定要简化整行删除，仅需要在同一行上击打两次 d 就可以删除掉光标所在的整行了。

1. 请将光标移动到本节中下面的短句段落中的第二行。
2. 输入 dd 删除该行。
3. 然后移动到第四行。
4. 接着输入 2dd (还记得前面讲过的 number-command-object 吗？) 删除两行。

1) Roses are red,

2) Mud is fun,

3) Violets are blue,

4) I have a car,

5) Clocks tell time,

6) Sugar is sweet

7) And so are you.

第二讲 第五节：撤消类命令

** 输入 u 来撤消最后执行的命令，输入 U 来修正整行。**

1. 请将光标移动到本节中下面标记有 ---> 的那一行,并将其置于第一个错误处。

2. 输入 x 删除第一个不想保留的字母。

3. 然后输入 u 撤消最后执行的(一次)命令。

4. 这次要使用 x 修正本行的所有错误。

5. 现在输入一个大写的 U ，恢复到该行的原始状态。

6. 接着多次输入 u 以撤消 U 以及更前的命令。

7. 然后多次输入 CTRL-R (先按下 CTRL 键不放开，接着输入 R 键) ，这样就可以执行恢复命令，也就是撤消掉撤消命令。

---> Fiix the errors oon thhis line and reeplace them witth undo.

8. 这些都是非常有用的命令。下面是第二讲的小结了。

第二讲 小结

1. 欲从当前光标删除至单字/单词末尾，请输入：dw

2. 欲从当前光标删除至当前行末尾，请输入：d$

3. 欲删除整行，请输入：dd

4. 在正常模式下一个命令的格式是：

[number] command object 或者 command [number] object

其意是： number - 代表的是命令执行的次数 command - 代表要做的事情，比如 d 代表删除 object - 代表要操作的对象，比如 w 代表单字/单词，$ 代表到行末等等。

$ (to the end of line), etc.

5. 欲撤消以前的操作，请输入：u (小写的 u)欲撤消在一行中所做的改动，请输入：U (大写的 U)欲撤消以前的撤消命令，恢复以前的操作结果，请输入：CTRL-R

第三讲 第一节：置入类命令

**** 输入 p 将最后一次删除的内容置入光标之后 ****

1. 请将光标移动到本节中下面示范段落的首行。

2. 输入 dd 将该行删除，这样会将该行保存到 vim 的缓冲区中。

3. 接着将光标移动到准备置入的位置的上方。记住：是上方哦。

4. 然后在正常模式下(<ESC>键进入)，输入 p 将该行粘贴置入。

5. 重复步骤 2 至步骤 4，将所有的行依序放置到正确的位置上。

d) Can you learn too?

b) Violets are blue,

c) Intelligence is learned,

a) Roses are red,

第三讲 第二节：替换类命令

** 输入 r 和一个字符替换光标所在位置的字符。**

1. 请将光标移动到本节中下面标记有 ---> 的第一行。
2. 请移动光标到第一个错误的适当位置。
3. 接着输入 r ，这样就能将错误替换掉了。
4. 重复步骤 2 和步骤 3，直到第一行已经修改完毕。

---> Whan this lime was tuoed in, someone presswd some wrojg keys!

---> When this line was typed in, someone pressed some wrong keys!

5. 然后我们继续学校第三讲第三节。

特别提示：切记您要在使用中学习，而不是在记忆中学习。

第三讲 第三节：更改类命令

** 要改变一个单字/单词的部分或者全部，请输入 cw **

1. 请将光标移动到本节中下面标记有 ---> 的第一行。
2. 接着把光标放在单词 lubw 的字母 u 的位置那里。
3. 然后输入 cw 就可以修正该单词了(在本例这里是输入 ine 。)
4. 最后按 <ESC> 键，然后光标定位到下一个错误第一个准备更改的字母处。
5. 重复步骤 3 和步骤 4，直到第一个句子完全雷同第二个句子。

---> This lubw has a few wptfd that mrrf changing usf the change command.

---> This line has a few words that need changing using the change command.

提示：请注意 cw 命令不仅仅是替换了一个单词，也让您进入文本插入状态了。

第三讲 第四节：使用 c 指令的其他更改类命令

** 更改类指令可以使用同删除类命令所使用的对象参数。**

1. 更改类指令的工作方式跟删除类命令是一致的。操作格式是：
[number] c object 或者 c [number] object
2. 对象参数也是一样的，比如 w 代表单字/单词，$代表行末等等。
3. 请将光标移动到本节中下面标记有 ---> 的第一行。
4. 接着将光标移动到第一个错误处。
5. 然后输入 c$ 使得该行剩下的部分更正得同第二行一样。最后按 <ESC> 键。

---> The end of this line needs some help to make it like the second.
---> The end of this line needs to be corrected using the c$ command.

第三讲 小结

1. 要重新置入已经删除的文本内容，请输入小写字母 p。该操作可以将已删除的文本内容置于光标之后。如果最后一次删除的是一个整行，那么该行将置于当前光标所在行的下一行。

2. 要替换光标所在位置的字符,请输入小写的 r 和要替换掉原位置字符的新字符即可。

3. 更改类命令允许您改变指定的对象，从当前光标所在位置直到对象的末尾。比如输入 cw 可以替换当前光标到单词的末尾的内容;输入 c$ 可以替换当前光标到行末

的内容。

4. 更改类命令的格式是：

[number] c object 或者 c [number] object

下面我们继续学习下一讲。

第四讲 第一节：定位及文件状态

** 输入 CTRL-g 显示当前编辑文件中当前光标所在行位置以及文件状态信息。输入 SHIFT-G 则直接跳转到文件中的某一指定行。**

提示：切记要先通读本节内容，之后才可以执行以下步骤!!!

1. 按下 CTRL 键不放开然后按 g 键。然后就会看到页面最底部出现一个状态信息行，显示的内容是当前编辑的文件名和文件的总行数。请记住步骤 3 的行号。

2. 按下 SHIFT-G 键可以使得当前光标直接跳转到文件最后一行。

3. 输入您曾停留的行号，然后按下 SHIFT-G。这样就可以返回到您第一次按下 CTRL-g 时所在的行好了。注意：输入行号时，行号是不会在屏幕上显示出来的。

4. 如果愿意，您可以继续执行步骤 1 至步骤三。

第四讲 第二节：搜索类命令

** 输入 / 以及尾随的字符串可以用以在当前文件中查找该字符串。**

1. 在正常模式下输入 / 字符。您此时会注意到该字符和光标都会出现在屏幕底

部，这跟 : 命令是一样的。

2. 接着输入 errroor <回车>。那个 errroor 就是您要查找的字符串。

3. 要查找同上一次的字符串，只需要按 n 键。要向相反方向查找同上一次的字符串，请输入 Shift-N 即可。

4. 如果您想逆向查找字符串，请使用 ? 代替 / 进行。

---> When the search reaches the end of the file it will continue at the start.

"errroor" is not the way to spell error; errroor is an error.

提示：如果查找已经到达文件末尾，查找会自动从文件头部继续查找。

第四讲 第三节：配对括号的查找

** 按 % 可以查找配对的括号)、]、}。**

1. 把光标放在本节下面标记有 --> 那一行中的任何一个 (、[或 { 处。
2. 接着按 % 字符。
3. 此时光标的位置应当是在配对的括号处。
4. 再次按 % 就可以跳回配对的第一个括号处。

---> This (is a test line with ('s, ['s] and {'s } in it.))

提示：在程序调试时，这个功能用来查找不配对的括号是很有用的。

第四讲 第四节：修正错误的方法之一

** 输入 :s/old/new/g 可以替换 old 为 new。**

1. 请将光标移动到本节中下面标记有 ---> 的那一行。
2. 输入 :s/thee/the<回车> 。请注意该命令只改变光标所在行的第一个匹配串。
3. 输入 :s/thee/the/g 则是替换全行的匹配串。

---> the best time to see thee flowers is in thee spring.

4. 要替换两行之间出现的每个匹配串，请输入 :#,#s/old/new/g (#,#代表的是两行的行号)。输入 :%s/old/new/g 则是替换整个文件中的每个匹配串。

第四讲 小结

1. Ctrl-g 用于显示当前光标所在位置和文件状态信息。Shift-G 用于将光标跳转至文件最后一行。先敲入一个行号然后按 Shift-G 则是将光标移动至该行号代表的行。
2. 输入 / 然后紧随一个字符串是则是在当前所编辑的文档中向后查找该字符串。输入问号 ? 然后紧随一个字符串是则是在当前所编辑的文档中向前查找该字符串。完成一次查找之后按 n 键则是重复上一次的命令，可在同一方向上查找下一个字符串所在；或者按 Shift-N 向相反方向查找下该字符串所在。
3. 如果光标当前位置是括号(、)、[、]、{、}，按 % 可以将光标移动到配对的括号上。
4. 在一行内替换头一个字符串 old 为新的字符串 new，请输入 :s/old/new 在一行内替换所有的字符串 old 为新的字符串 new，请输入 :s/old/new/g 在两行内替换所有的字符串 old 为新的字符串 new，请输入 :#,#s/old/new/g 在文件内替换所有的字符串 old 为新的字符串 new，请输入 :%s/old/new/g 进行全文替换时询问用户确认每个替换需添加 c 选项，请输入 :%s/old/new/gc

第五讲 第一节：在 VIM 内执行外部命令的方法

** 输入 :! 然后紧随著输入一个外部命令可以执行该外部命令。**

1. 按下我们所熟悉的 : 命令设置光标到屏幕底部。这样就可以让您输入命令了。

2. 接着输入感叹号 ! 这个字符，这样就允许您执行外部的 shell 命令了。

3. 我们以 ls 命令为例。输入 !ls〈回车〉。该命令就会列举出您当前目录的内容，就如同您在命令行提示符下输入 ls 命令的结果一样。如果 !ls 没起作用，您可以试试 :!dir 看看。

---> 提示： 所有的外部命令都可以以这种方式执行。

---> 提示： 所有的 : 命令都必须以〈回车〉告终。

第五讲 第二节：关于保存文件的更多信息

** 要将对文件的改动保存到文件中，请输入 :w FILENAME 。**

1. 输入 :!dir 或者 :!ls 获知当前目录的内容。您应当已知道最后还得敲<回车>吧。

2. 选择一个尚未存在文件名，比如 TEST 。

3. 接着输入 :w TEST （此处 TEST 是您所选择的文件名。）

4. 该命令会以 TEST 为文件名保存整个文件 (VIM 教程)。为了确保正确保存，请再次输入 :!dir 查看您的目录列表内容。

---> 请注意：如果您退出 VIM 然后在以文件名 TEST 为参数进入，那么该文件内容应该同您保存时的文件内容是完全一样的。

5. 现在您可以通过输入 :!rm TEST 来删除 TEST 文件了。

第五讲 第三节：一个具有选择性的保存命令

** 要保存文件的部分内容，请输入 :#,# w FILENAME **

1. 再来执行一次 :!dir 或者 :!ls 获知当前目录的内容,然后选择一个合适的不重名的文件名，比如 TEST 。

2. 接着将光标移动至本页的最顶端，然后按 CTRL-g 找到该行的行号。别忘了行号哦。

3. 接着把光标移动至本页的最底端，再按一次 CTRL-g 。也别忘了这个行好哦。

4. 为了只保存文章的某个部分，请输入 :#,# w TEST 。这里的 #,# 就是上面要求您记住的行号(顶端行号,底端行号)，而 TEST 就是选定的文件名。

5. 最后，用 :!dir 确认文件是否正确保存。但是这次先别删除掉。

第五讲 第四节：提取和合并文件

** 要向当前文件中插入另外的文件的内容，请输入 :r FILENAME **

1. 请键入 :!dir 确认您前面创建的 TEST 文件还在。

2. 然后将光标移动至当前页面的顶端。

特别提示： 执行步骤 3 之后您将看到第五讲第三节，请届时再往下移动回到这里来。

3. 接着通过 :r TEST 将前面创建的名为 TEST 的文件提取进来。

特别提示：您所提取进来的文件将从光标所在位置处开始置入。

4. 为了确认文件已经提取成功，移动光标回到原来的位置就可以注意有两份第五讲第三节，一份是原本，另外一份是来自文件的副本。

第五讲 小结

1. :!command 用于执行一个外部命令 command。

请看一些实际例子：

:!dir - 用于显示当前目录的内容。

:!rm FILENAME - 用于删除名为 FILENAME 的文件。

2. :w FILENAME 可将当前 VIM 中正在编辑的文件保存到名为 FILENAME 的文件中。

3. :#,#w FILENAME 可将当前编辑文件第 # 行至第 # 行的内容保存到文件 FILENAME 中。

4. :r FILENAME 可提取磁盘文件 FILENAME 并将其插入到当前文件的光标位置后面。

第六讲 第一节：打开类命令

** 输入 o 将在光标的下方打开新的一行并进入插入模式。**

1. 请将光标移动到本节中下面标记有 ---> 的那一行。
2. 接着输入小写的 o 在光标 *下方* 打开新的一行并进入插入模式。
3. 然后复制标记有 ---> 的行并按 <ESC> 键退出插入模式而进入正常模式。

---> After typing o the cursor is placed on the open line in Insert mode.

4. 为了在光标 *上方* 打开新的一行，只需要输入大写的 O 而不是小写的 o 就可以了。请在下行测试一下吧。当光标处在在该行上时，按 Shift-O 可以在该行上方新开一行。

Open up a line above this by typing Shift-O while the cursor is on this line.

第六讲 第二节：光标后插入类命令

** 输入 a 将可在光标之后插入文本。 **

1. 请在正常模式下通过输入 $ 将光标移动到本节中下面标记有 ---> 的第一行的末尾。

2. 接着输入小写的 a 则可在光标之后插入文本了。大写的 A 则可以直接在行末插入文本。

提示：输入大写 A 的操作方法可以在行末插入文本，避免了输入 i，光标定位到最后一个字符，输入的文本，<ESC> 回复正常模式，箭头右键移动光标以及 x 删除当前光标所在位置字符等等诸多繁杂的操作。

3. 操作之后第一行就可以补充完整了。请注意光标后插入文本与插入模式是基本完全一致的，只是文本插入的位置定位稍有不同罢了。

---> This line will allow you to practice

---> This line will allow you to practice appending text to the end of a line.

第六讲 第三节：另外一个置换类命令的版本

** 输入大写的 R 可连续替换多个字符。**

1. 请将光标移动到本节中下面标记有 ---> 的第一行。

2. 移动光标到第一行中不同于标有 ---> 的第二行的第一个单词的开始，即单词 last 处。

3. 然后输入大写的 R 开始把第一行中的不同于第二行的剩余字符逐一输入,就可以全部替换掉原有的字符而使得第一行完全雷同第二行了。

---> To make the first line the same as the last on this page use the keys.

---> To make the first line the same as the second, type R and the new text.

4. 请注意：如果您按 <ESC> 退出置换模式回复正常模式,尚未替换的文本将仍然保持原状。

第六讲 第四节：设置类命令的选项

** 设置可使查找或者替换可忽略大小写的选项 **

1. 要查找单词 ignore 可在正常模式下输入 /ignore 。要重复查找该词，可以重复按 n 键。

2. 然后设置 ic 选项(ic 就是英文忽略大小写 Ignore Case 的首字母缩写词)，即输入：

:set ic

3. 现在可以通过键入 n 键再次查找单词 ignore。重复查找可以重复键入 n 键。

4. 然后设置 hlsearch 和 incsearch 这两个选项，输入以下内容：:set hls is

5. 现在可以再次输入查找命令，看看会有什么效果：/ignore

第六讲 小结

1. 输入小写的 o 可以在光标下方打开新的一行并将光标置于新开的行首,进入插入模式。输入大写的 O 可以在光标上方打开新的一行并将光标置于新开的行首，进入插入模式。

2. 输入小写的 a 可以在光标所在位置之后插入文本。输入大写的 A 可以在光标所在行的行末之后插入文本。

3. 输入大写的 R 将进入替换模式，直至按 <ESC> 键退出替换模式而进入正常模式。

4. 输入 :set xxx 可以设置 xxx 选项。

第七讲：在线帮助命令

** 使用在线帮助系统 **

Vim 拥有一个细致全面的在线帮助系统。要启动该帮助系统，请选择如下三种方法之一：

- 按下 <HELP> 键 (如果键盘上有的话)
- 按下 <F1> 键 (如果键盘上有的话)
- 输入 :help <回车>

输入 :q <回车> 可以关闭帮助窗口。

提供一个正确的参数给":help"命令，您可以找到关于该主题的帮助。请试验以下参数(可别忘了按回车键哦。:)：

:help w <回车>

:help c_<T <回车>

:help insert-index <回车>

:help user-manual <回车>

第八讲：创建一个启动脚本

** 启用 vim 的功能 **

Vim 的功能特性要比 vi 多得多，但大部分功能都没有缺省激活。为了启动更多的功能，您得创建一个 vimrc 文件。

1. 开始编辑 vimrc 文件，这取决于您所使用的操作系统：

:edit ~/.vimrc　　这是 Unix 系统所使用的命令

:edit $VIM/_vimrc　　这是 Windows 系统所使用的命令

2. 接着导入 vimrc 范例文件：

:read $VIMRUNTIME/vimrc_example.vim

3. 保存文件，命令为：

:write

在下次您启动 vim 的时候，编辑器就会有了语法高亮的功能。您可以继续把您喜欢的其它功能设置添加到这个 vimrc 文件中。

vim 教程到此结束。本教程只是为了简明地介绍一下 vim 编辑器，但已足以让您很容易学会使用本编辑器了。毋庸质疑，vim 还有很多很多的命令，本教程所介绍的还差得远著呢。所以您要精通的话，还望继续努力哦。下一步您可以阅读 vim 手册，使用的命令是：

:help user-manual

为了更进一步的参考和学习，以下这本书值得推荐：

Vim - Vi Improved - 作者：Steve Oualline
出版社：New Riders

这是第一本完全讲解 vim 的书籍。对于初学者特别有用。其中还包含有大量实例和图示。欲知详情，请访问 http://iccf-holland.org/click5.html

以下这本书比较老了而且内容主要是 vi 而不是 vim，但是也值得推荐：

Learning the Vi Editor - 作者：Linda Lamb
出版社：O'Reilly & Associates Inc.

这是一本不错的书，通过它您几乎能够了解到全部 vi 能够做到的事情。此书的第六个版本也包含了一些关于 vim 的信息。

本教程是由来自 Calorado School of Minese 的 Michael C. Pierce、Robert K. Ware 所编写的，其中来自 Colorado State University 的 Charles Smith 提供了很多创意。编者通信地址是：

bware@mines.colorado.edu

本教程已由 Bram Moolenaar 专为 vim 进行修订。