第十章、vim 程序编辑器

一、Linux 中绝大部分的配置文件都是以 ADCII 的纯文本形态存在，因此用 vim 编辑器就可以修改设定了。而且几乎所有 Linux 系统上面的指令都默认使用 vi 作为数据编辑的接口
此外，vim 还有程序编辑能力，可以主动的以字体颜色辨别语法的正确性，方便程序设计。
vi：文字处理器
vim：程序开发工具

二、vi

vi 有三种模式：

1、一般模式：

以 vi 打开一个文件局默认进入一般模式，可以使用上下左右按键来移动光标，可以删除字符或删除整行，可以复制或粘贴

2、编辑模式：

一般模式中可以删除、复制、粘贴等，但无法编辑文件内容（无法键盘输入编辑）。在按下“i、o、a、r”这四个键的任意一个后会进入编辑模式（左下角有 INSERT 或 REPLACE 字样）。按 Esc 键则退出编辑模式

3、指令列命令模式：

在一般模式下，按”：　/　?”这三个键中的任意一个，就可以将光标移到最底行，然后可以输入指令进行搜寻、读取、存盘等动作
编辑模和指令列模式之间不可互相切换

三、vi+filename 进入一般模式

上面部分显示文件内容，最底下一行是状态显示列（文件名，行数，字符数等）或命令下达列
按 i、o、a 任意一个按键进入编辑模式，此时除了 Esc 按键外其他都可以作为输入按键。
编辑完毕后按 Esc 重新回到一般模式，然后怎么保存并退出呢？
先打出：符号，然后输入 wq 即可离开

```
[root@localhost ~]# ll te*
-rw-r--r--. 1 root root 28 3月  23 19:14 test.txt
```

当前用户 root 是有权限编辑文档的

一般模式下的操作（截取一些常用的）：

移动光标：

[Ctrl] + [f]	屏幕『向下』移动一页，相当于 [Page Down]按键 (常用)
[Ctrl] + [b]	屏幕『向上』移动一页，相当于 [Page Up] 按键 (常用)
G	移动到这个档案的最后一行(常用)
gg	移动到这个档案的第一行，相当于 1G 啊！(常用)
n<Enter>	n 为数字。光标向下移动 n 行(常用)

搜寻/替换

/word	向光标之下寻找一个名称为 word 的字符串。例如要在档案内搜寻 vbird 这个字符串，就输入 /vbird 即可！(常用)

n	这个 n 是英文按键。代表『重复前一个搜寻的动作』。举例来说， 如果刚刚我们执行 /vbird 去向下搜寻 vbird 这个字符串，则按下 n 后，会向下继续搜寻下一个名称为 vbird 的字符串。如果是执行 ?vbird 的话，那么按下 n 则会向上继续搜寻名称为 vbird 的字符串！
:n1,n2s/word1/word2/g	n1 与 n2 为数字。在第 n1 与 n2 行之间寻找 word1 这个字符串，并将该字符串取代为 word2 ！举例来说，在 100 到 200 行之间搜寻 vbird 并取代为 VBIRD 则：『:100,200s/vbird/VBIRD/g』。(常用)
:1,$s/word1/word2/g	从第一行到最后一行寻找 word1 字符串，并将该字符串取代为 word2 ！(常用)
:1,$s/word1/word2/gc	从第一行到最后一行寻找 word1 字符串，并将该字符串取代为 word2 ！且在取代前显示提示字符给用户确认 (confirm) 是否需要取代！(常用)

复制、粘贴、删除：

x, X	在一行字当中，x 为向后删除一个字符 (相当于 [del] 按键)， X 为向前删除一个字符(相当于 [backspace] 亦即是退格键) (常用)
dd	删除游标所在的那一整列(常用)
ndd	n 为数字。删除光标所在的向下 n 列，例如 20dd 则是删除 20 列 (常用)

yy	复制游标所在的那一行(常用)
nyy	n 为数字。复制光标所在的向下 n 列，例如 20yy 则是复制 20 列(常用)

p, P	p 为将已复制的数据在光标下一行贴上，P 则为贴在游标上一行！ 举例来说，我目前光标在第 20 行，且已经复制了 10 行数据。则按下 p 后， 那 10 行数据会贴在原本的 20 行之后，亦即由 21 行开始贴。但如果是按下 P 呢？ 那么原本的第 20 行会被推到变成 30 行。 (常用)

撤销和重做和重复上一个动作：

u	复原前一个动作。(常用)
[Ctrl]+r	重做上一个动作。(常用)
.	不要怀疑！这就是小数点！意思是重复前一个动作的意思。 如果你想要重复删除、重复贴上等等动作，按下小数点『.』就好了！ (常用)

大小写的 i,a,o,r 总共 8 个字符各有不同的作用，都是编辑文本，一般使用小写 i 就 ok 了：在当前光标所在处插入

四、指令列的存储、离开等指令

:w	将编辑的数据写入硬盘档案中(常用)

:q	离开 vi (常用)
:q!	若曾修改过档案，又不想储存，使用！为强制离开不储存档案。

我觉得下面这个指令很棒，完全符合我们的使用习惯

ZZ	这是大写的 Z 喔！若档案没有更动，则不储存离开，若档案已经被更动过，则储存后离开！

下面这个也很棒，因为 Linux 好像没有跨文件之间数据的复制粘贴，可以直接选几行生成新文件

:n1,n2 w [filename]	将 n1 到 n2 的内容储存成 filename 这个档案。

设置 VIM 环境

:set nu	显示行号，设定之后，会在每一行的前缀显示该行的行号
:set nonu	与 set nu 相反，为取消行号！

鸟哥的梳理经常用到/etc 目录下的 man.config 这个文件，但系统里是没有的，网上一查发现，是改名字了，叫这个

```
[root@localhost vitest]# ls /etc/man*
/etc/man_db.conf
```

当我们使用 vim 编辑时，它会在被编辑的文件所在的目录下，建立一个名为 filename.swp 的文件，我们对这个文件的操作都会记录进去。如果系统宕机，文件没有保存，这时候就可以用 filename.swp 来恢复了

1、用 vim 打开文件：vim man.config

2、ctrl+z，回到命令行，vim 会在后台运行，此时用 ls -al 会发现出现了一个.swp 文件

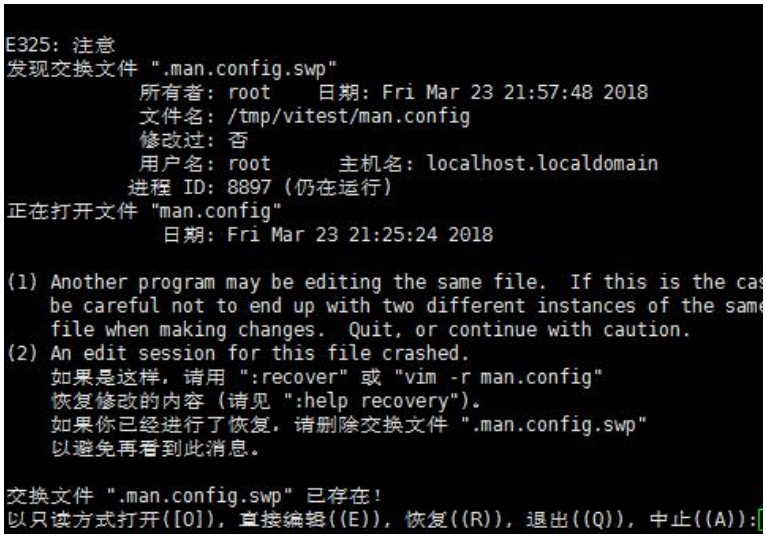
3、强行停止 vim 程序

```
[root@localhost vitest]# ls -a
.  ..  man.config  .man.config.swp  man.test.
[root@localhost vitest]# kill -9 %1

[1]-  已停止               vim man.config
[root@localhost vitest]# ls -a
.  ..  man.config  .man.config.swp  m
```

可见 swp 文件仍存在

4、此时尝试再用 vim 打开文件，会这样报错



这是因为 vim 发现存在当前目录下有这个文件的 swp 文件（如果上一次编辑是正常关闭的话 swp 文件会消失），所以判断这个文件可能有问题。

我尝试把 swp 文件移动到上一级目录去，然后 vim 就没有报错提示，说明只检测本目录下的

VIM 提示的主要问题和解决方法：

1、可能有其他人或程序同时在编辑这个文件（Linux 是多人多任务环境）

此时如果大家同时修改后存储，则文件内容会很混乱。如果只是想看内容而不编辑的话，可以按 o 选择只读模式

2、在前一个 vim 环境中，没有来得及保存 vim 就被中断了

（1）如果想恢复上一次的编辑动作，可以按 r 恢复。但即使是恢复后 vim 也不会自动删除 swp 文件，还要自己手动删除

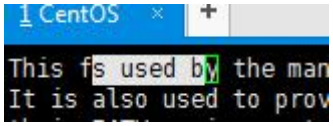
（2）如果认为这个 swp 是没有意义的，可以按 d 删除掉

鸟哥的 Linux 里 vi 已经被 vim 替代了，而我的系统里 vi、vim 还是分开的，一个没有颜色，一个有颜色

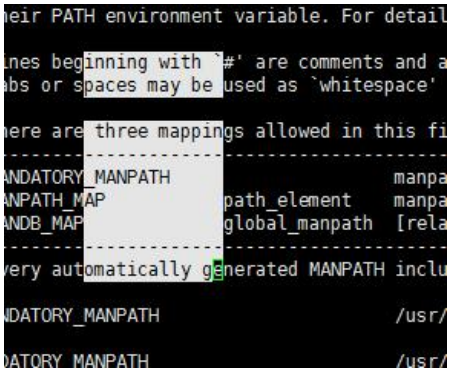
区块选择 Visual Block

	区块选择的按键意义
v	字符选择，会将光标经过的地方反白选择！
V	行选择，会将光标经过的行反白选择！
[Ctrl]+v	区块选择，可以用长方形的方式选择资料
y	将反白的地方复制起来
d	将反白的地方删除掉

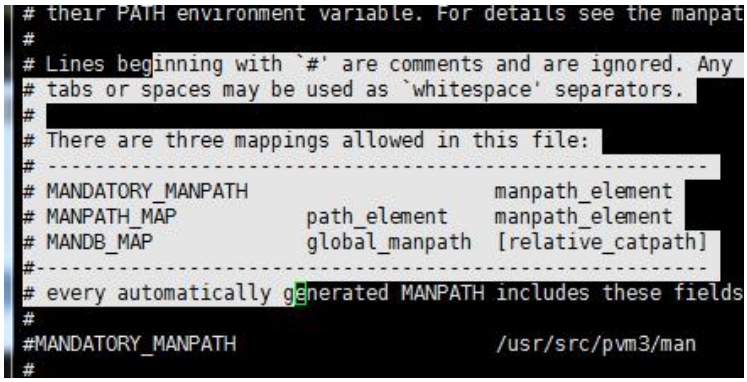
按下一次 v 后，再按向右按键，就可以进行区块选择了：



ctrl+v，然后按向下按键能这样选择



如果只是 v，则按向下按键是这样的：



此时按 y 进行复制，然后进入编辑模式，选好粘贴的位置（将光标移动到那里），然后回到一般模式，按 p，就能将刚才选择的区块复制到目标区域

五、多文件编辑

如果想要将 A 文档内的十条语句『移动』到 B 文档去，通常要开两个 vim 窗口来复制， 然而每个 vim 都是独立的，因此并没有办法在 A 文档下达『 nyy 』再跑到 B 文档去『 p 』

1、vim A B

2、在文档 A 通过 4yy 指令复制 4 行

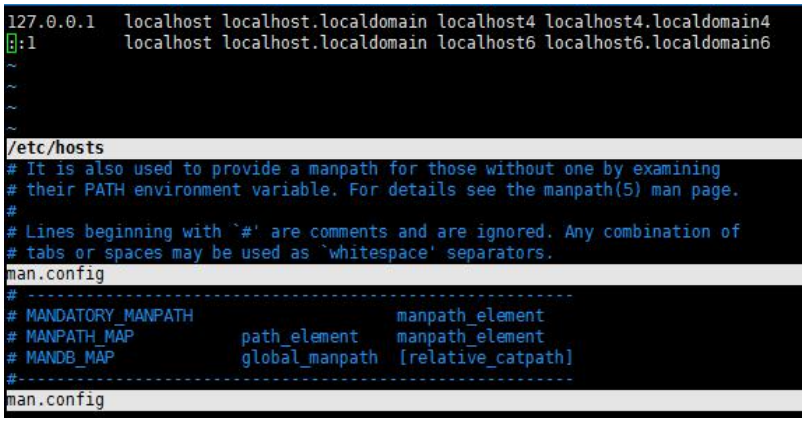
3、输入:n 去到第二个文档 B

4、按 G 去到最后一行，按 p 粘贴

六、多窗口功能

如果想对照一个大文档的前后，或者对照两个文档的内容，可以通过在指令模式下输入 :sp filename 即可。如果只输入:sp，则会分成上下两个窗口，都是同一个文档的内容。

再输入个:sp /etc/hosts，会变成三窗口



以利用『[ctrl]+w+↑』及『[ctrl]+w+↓』 在两个窗口之间移动

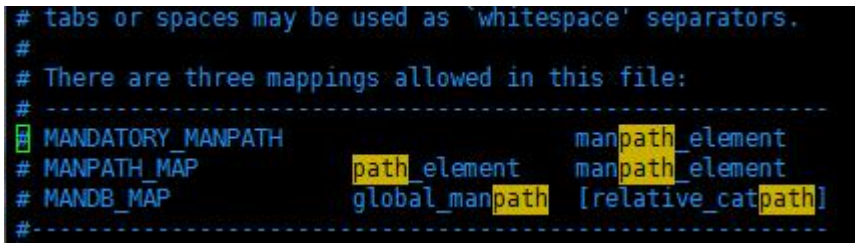
如此则文档之间的复制粘贴、查阅等都很方便了

操作手法是，先按一次 ctrl+w，然后再按向下按键就能去到下一各窗口了

还有 ctrl+w+q，就是关闭当前窗口

七、vim 环境设定与记录

我们在 vim 里面搜寻过 path 这个字符串，反白显示（黄），退出后再打开，发现 path 还是被标记着，而且光标位置也在上次离开时的位置。



这是因为 vim 会主动将我们做过的行为记录下来。记录动作的文档是 ~/.viminfo

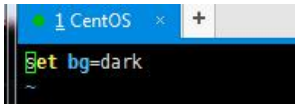
这个文档在根目录底下。用户在 vim 这个程序里所做过的动作都能在这个文档里查到。

vim 的环境设定参数

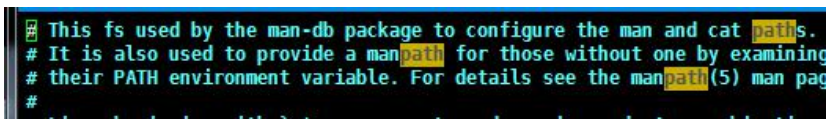
这个太多指令了，而且也没什么用，不过这个还是挺有意思的

:set bg=dark :set bg=light	可用以显示不同的颜色色调，预设是『 light 』。如果你常常发现批注的字体深蓝色实在很不容易看， 那么这里可以设定为 dark 喔！试看看，会有不同的样式呢！

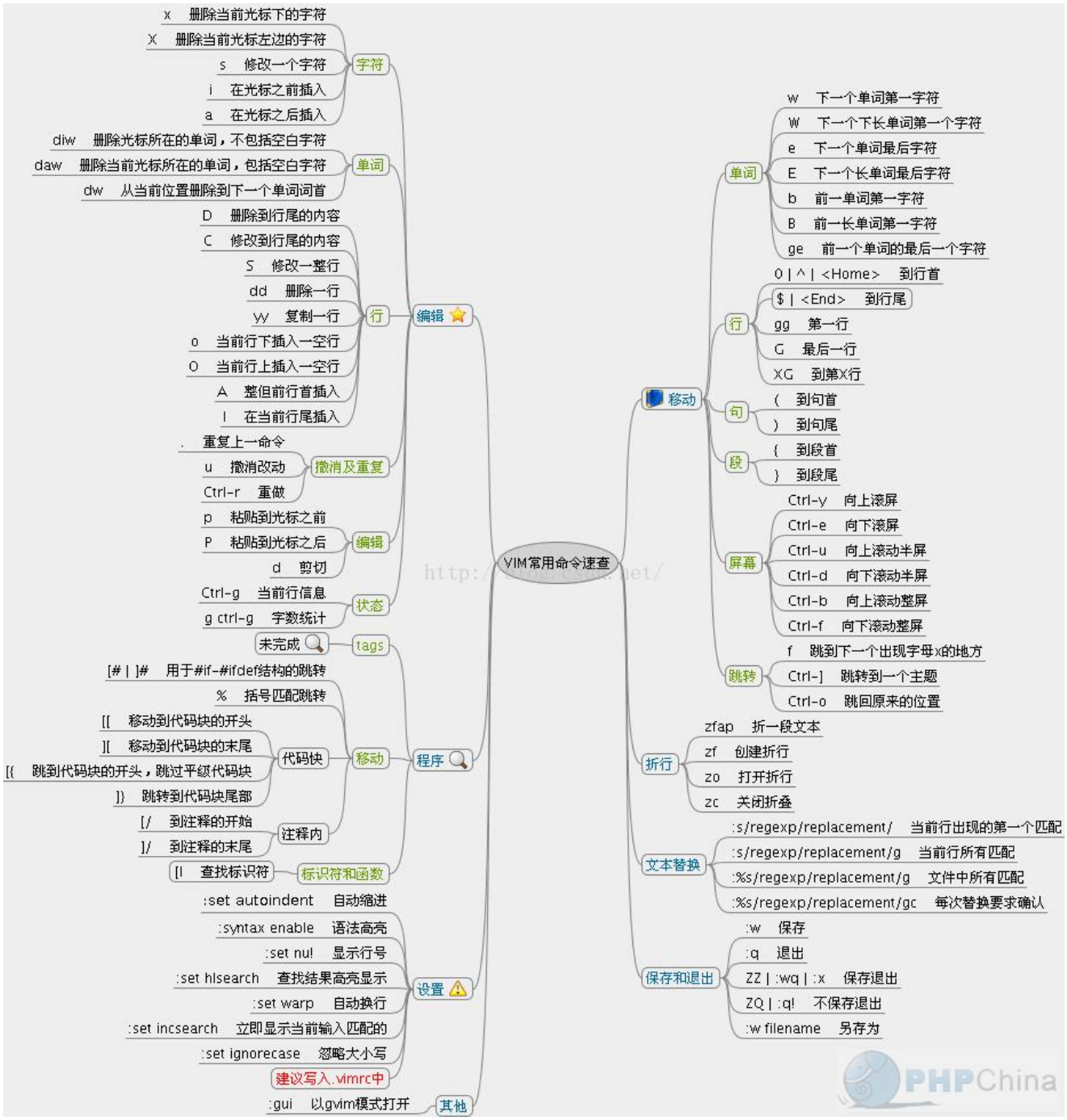
但如何保存这些设定呢（如果只是在 vim 里输入上述语句，则第二次打开 vim 时又需要再输入一遍设定），可以在建立~/.vimrc 文件，并写入设定值



此时再打开 vim,发现颜色已经变了



VIM 指令总结



VIM 使用注意事项

1、vim 里面无法显示中文，都是乱码，这是编码的问题

https://blog.csdn.net/zhaohaifan/article/details/24022811

Linux中使用vim乱码

原创　2014年04月18日 15:40:53　标签：乱码 / LINUX / vim

找到 vimrc 文件，我的是在/etc/vimrc 在最后添加

```
[plain]
1. set fileencodings=utf-8,gb2312,gbk,gb18030
2.
3. set termencoding=utf-8
4.
5. set fileformats=unix
6.
7. set encoding=prc
```

保存退出就OK了

鸟哥的方法没用，最后还是靠上面那个方法解决了。

然后就发现问题了，为什么每一行最后有个“^M”？

原因是 windows 系统里使用的是 dos 系统(磁盘 Disk 管理 Operating 系统 System)，而 Linux 不使用 DOS。

在 DOS 中断行符为 ^M$ ，而 Linux 断行符为 $ ，所以在 Linux 中就多了一个 ^M 符号出来。

不需要一个个的删除每一行的 CR，有指令进行格式转换

dos2unix [-kn] file newfile

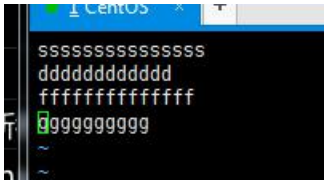
unix2dos [-kn] file newfile

-k:保留文档原来的 mtime（ modify time ），转换后的新文档把原文档替换

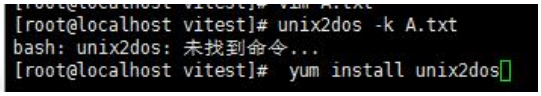
-n:保留原文档，转换后的内容输出到新文档

1、Unix 转 dos

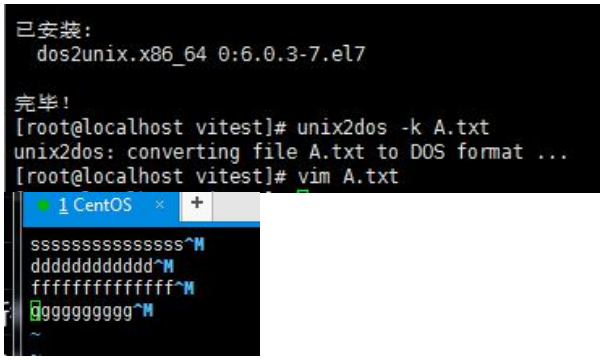
原本的文档 A



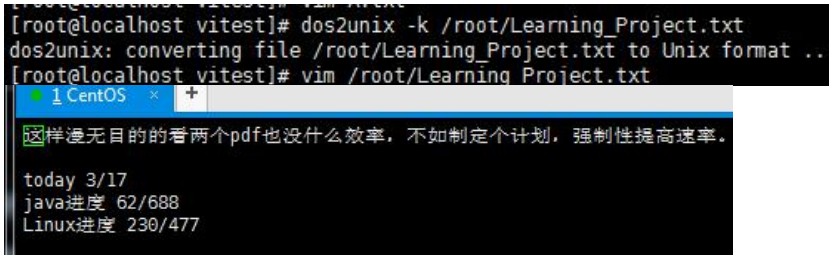
开始转换：



还没安装这个程序，于是安装



2、dos 转 Unix



语系、编码转换

将 /tmp/vitest/vi.big5 转成 utf8 编码

iconv -f big5 -t utf8 vi.big5 -o vi.utf8

Q&A

1、我用 vi 开启某个文档后，要在第 34 行向右移动 15 个字符，应该在一般模式中下达什么指令？

(1)先按下 34G 到第 34 行；

(2)再按下 [15 + 向右键]

2、如何去到该文档的页首或页尾？

去页首按下 1G ；去页尾按下 G 即可

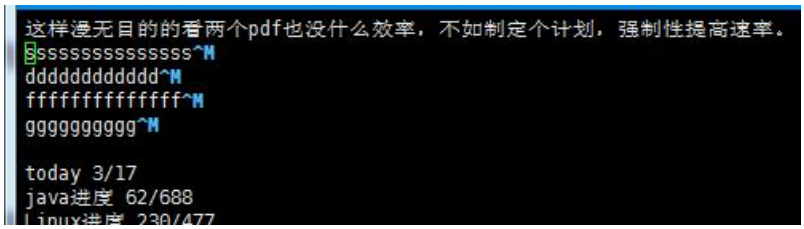
3、如何将目前正在编辑的文档另存新文件名为 newfilename？

:w newfilename

4、在一般模式下，如何读取一个文档 filename 进来目前这个文档

:r filename

这个挺厉害的，直接就把 A.txt 这个文档的内容全部复制到光标处了



5、取消之前的全部操作

:e!