



---

# 第 13 章

## 文書編輯器

---

每  
個  
參  
考

Linux

## 第 13 章 文書編輯器

Linux 是以輸入文字為主的作業系統，因此，使用 Script 來執行多個指令，或者寫 C 語言的程式、寫 e-mail，這些都要使用文書編輯器來編輯文字。文書編輯器可以讓我們簡單的觀看文字內容，也可以讓我們有效率的編輯檔案。文書編輯器可以讓我們以全螢幕的方式編輯。我們可以從螢幕上看到我們檔案的部份，我們也可以移動我們的游標到任何的文書編輯器。我們可以將文字暫存在儲存器上(通常為記憶體上)，這稱為文書緩衝區。當我們編輯檔案時，我們編輯的是檔案的備份，而這就是文書編輯器的緩衝區，而且我們可以管理好幾個備份的緩衝區，當我們作完工作後，才將結果除存在硬碟上。文書編輯器的操作是以鍵盤按鈕的操作為主，例如我們按下單一鍵 <enter>或是兩個鍵一起按 <Ctrl-D>。

我們在這裏將介紹 pico 文書編輯器、vi 文書編輯器、emacs 文書編輯器。

### 13-1 vi 文書編輯器

vi 文書編輯器有 Linux 文書編輯器所有的特徵，它可以編輯處理我們的文字。vi 雖然比 pico 文書編輯器較不易學習，但它強大的功能和高容納度，與作業系統的相容性，卻是最好，而且是最多人使用的 Linux 文書編輯器。

我們可以先使用 vi 編輯器。我們在 shell 指令提示器上輸入 vi first.sh。

```
[root@flash 1-1]# vi first.sh
```

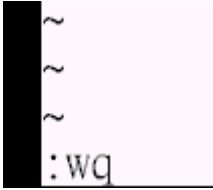
我們進入 vi 文書編輯器後，我們可以按下 a 鍵或 l 鍵，來進入輸入模式。我們可以輸入 ls -la，我們再按下 <enter> 鍵。再輸入 who 再按下 <enter> 鍵。最後輸入 pwd，輸入完後我們按下 <Esc> 鍵到命令模式。

```
ls -la
who
pwd
~
~
~
```

我們在命令模式，也就是最底層，輸入 :wq。這樣就可以儲存檔案，並且離開



vi 文書編輯器。



我們可以輸入/bin/bash first.sh 這樣就可以執行我們剛才輸入的指令了。

```
[root@flash 1-1]# /bin/bash first.sh
total 24
drwxr-xr-x  2 root   root    4096  8月 14 00:43 .
drwxrwxrwx 22 chaiyen chaiyen 4096  8月 13 23:53 ..
-rwxrwxrwx  1 root   root     15  8月 14 00:43 first.sh
-rw-r--r--  1 root   root     40  8月 14 00:04 lib.h
-rw-r--r--  1 root   root    369  8月 13 23:17 power.c
-rw-----  1 root   root    439  8月 13 23:25 power.c.save
chaiyen pts/1    Aug 13 23:59 (61-218-29-5.HINET-IP.hinet.net)
root pts/0      Aug 13 23:49
/home/chaiyen/1-1
```

### 13-1-1 如何開始 vi 文書編輯器、儲存檔案、離開 vi 文書編輯器

vi 指令讓我們編輯新的檔案或已存在的檔案。

語法：

指令：vi 參數 檔案

參數：

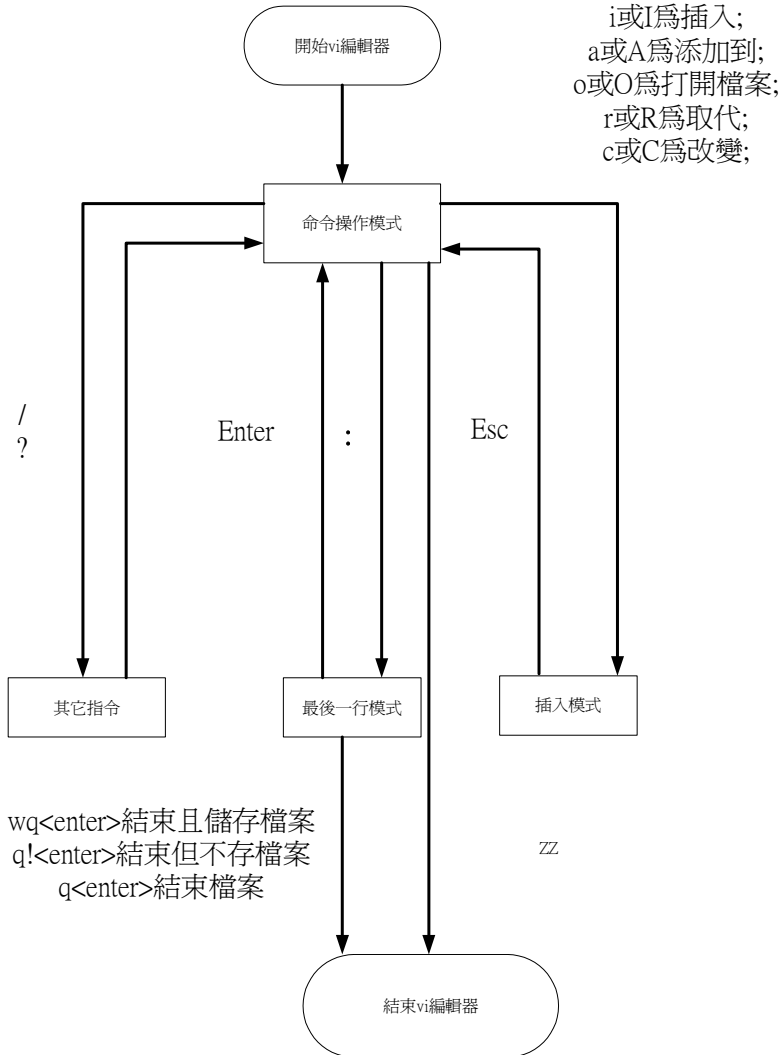
+n：在第 n 行開始編輯檔案。

+/exp：在字串 exp 開始的那一行開始編輯。

當我們進入 vi 文書編輯器後，會有兩種操作模式，一個為命令操作模式，另外一個為插入操作模式。命令操作模式是由一些指令參數所組成的命令。插入操作模式允許我們輸入文字。



我們一開始會進入到命令操作模式，我們按下<A>鍵就可以插入文字到游標那一行的最後。當我們按下<Esc>時，就可以從插入模式到命令操作模式。當我們在命令模式輸入：冒號，在後面再輸入指令 wq，就可以將資料輸入檔案，並且離開 vi。我們也開以在命令模式輸入：冒號，在後面再輸入命令 w 檔名，這樣就可以將資料寫入指定的檔案。



vi 文書編輯器預設的是以命令模式開始。當我們要從命令模式轉變成插入模式時有三個鍵，分別是 a 鍵，i 鍵和 o 鍵。

a：從目前游標下一個位置開始插入。

i：從目前游標所在位置開始插入。

o：從目前游標下一行位置開始插入。

我們使用 vi 文書編輯器編輯 power1.c 檔案。

```
[root@flash chaiyen]# vi power1.c
```

當我們進入 vi 時，如果按下 i 就會進入插入模式。

```
double compute(double x,double y);
main ()
{
    float x, y;
    printf ("請輸入x的y次方，以便求x的y次方的值\n");
    printf ("請輸入x:");
    scanf ("%f", &x);
    printf ("請輸入y:");
    scanf ("%f", &y);
    printf ("x的y次方: %f\n",compute(x,y));
}
~
~
~
~
~
~
~
~
~
~
~
-- INSERT --
```



我們按下 i 進入插入模式，從目前游標所在位置開始插入。

```
double
main ()
{
  ..
}
```

當我們按下 a 進入插入模式，從目前游標下一個位置開始插入

```
double compute
main ()
{
  ..
}
```

當我們按下 o 進入插入模式，從目前游標下一行位置開始插入。

```
double compute(double x,double y);
main ()
{
  float x, y;
  printf ("請輸入x的y次方，以便求x的y次方的值\n");
  printf ("請輸入x:");
  scanf ("%f", &x);
  printf ("請輸入y:");
  scanf ("%f", &y);
  printf ("x的y次方: %f\n",compute(x,y));
}
```

下面是 vi 編輯器命令的語法的範例

語法	說明
5dw	開始在目前游標所在的地方，刪除 5 個字元。
7dd	在目前所在位置刪除七行。
7o	在目前所在位置後空七行
7O	在目前所在位置前空七行
1G	將游標移到第一行



10yy	拷背 yanks 下十行到緩衝記憶體
d	刪除字元或行數
u	回覆到上一個動作
P	貼上所複製的文字在目前所在的位置之後
p	貼上所複製的文字在目前所在的位置之前
: r 檔名	從指令檔名讀取資料，再將資料插入目前所在位置
: q!	離開 vi 編輯器而沒有儲存任緩充區的何資料
: wq	儲存緩衝區的資料然後離開
: w 檔名	儲存緩衝區的資料到指定的檔名
: w! 檔名	覆寫目前的資料到指定的檔名中
ZZ	離開 vi 編輯器，並且儲存檔案
: e 檔名	建立新檔案
: n 檔名	載入新的檔案
: set nu	顯示每一行的編號
: set nonu	不顯示每一行的編號
/ 字串	尋找所輸入的字串。找到字串時，再輸入 n 就會再找下一個。
? 字串	尋找所輸入的字串。找到字串時，再輸入 n 就會再找下一個。
^b	將游標向前捲動一頁
^f	將游標向後捲動一頁

在進入檔案後

命令模式：

用 esc 鍵切換至此模式，開始編輯時，vi 在此模式。

輸入模式：

vi 會將鍵入的字當作本文而顯示在螢幕上。

最後一行模式：在命令模式下鍵入：或 / 可使 vi 進入此模式

: set nu 可以顯示行數。

: / 字 可以搜尋所要搜尋的字。



- : q 離開 vi 編輯器
- : q! 離開 vi 編輯器而不儲存
- : qw 離開 vi 編輯器且儲存

我們使用 vi test.php 來編輯 test.php 的檔案。

```
Telnet - flash.aasir.com
連線(C) 編輯(E) 終端機(T) 說明(H)
[root@flash chaiyen]# vi test.php
```

我們使用 set nu 來顯示資料的行號。

```
Telnet - flash.aasir.com
連線(C) 編輯(E) 終端機(T) 說明(H)
 1 PHP-Nuke: Web Portal System
 2 =====
 3
 4
 5 Welcome to PHP-Nuke!
 6
 7 First of all, thanks a lot to download this software, I hope that you
 8 can enjoy it.
 9
10 PHP-Nuke is free software, released under GNU GPL Licence version 2.0
11 (see COPYING file for details)
12
13 PHP-Nuke is the result of many years administrating a news site called
14 Linux Preview (http://linuxpreview.org). First, around August 1998, I
15 wrote my own code in Perl called NUKE and used it for about 1 year,
16 then my site goes big, so I needed a more powerfull system and decided
17 to use Slash, the same used in Slashdot site. It's good but you realy
18 need to know Perl to modify it, need too many modules, need to load
19 a damn daemon that sucks all your CPU power. My Pentium III just appears
20 to be a 386 each minute the daemon make its work. Well, then I discovere
d
21 Thatware, a good project to have a news site under PHP. I learned PHP
22 in less than a week and initiate to modify it. There are too many mods
:set nu
```

我們使用 : / Linux 來搜尋 Linux 這個字串。

```
:/Linux
```

我們使用 :qw 來儲存所編輯的資料。





我們使用 :q! 不除存資料而離開 vi 編輯器。

字串替代命令	說明
: s/scanf/good	只有將這一行的 scanf 字串替換成 good 字串，一次。
: s/scanf/good/g	在游標所在位置的那一行字串 scanf 都替換成 good
: 1,10s/scanf/good/g	從第一行到第十行的 scanf 字串都替換成 good 字串
: 1,\$s/scanf/good/g	將整個檔案的 scanf 字串都替換成 good 字串

替代字串語法為 : [替代範圍]s/舊字串/新字串。

### 13-1-2 複製與剪下與貼上文字

我們使用 vi 編輯 power1.c。

當我們進入 vi 時，我們使用 yy 命令來複製游標所在的那一行，並且使用 5p 來將所複製的那一行貼上去五遍。

```
double compute(double x,double y);
main ()
{
    float x, y;
    printf ("請輸入x的y次方，以便求x的y次方的值\n");
    printf ("請輸入x:");
    scanf ("%f", &x);
    printf ("請輸入y:");
    scanf ("%f", &y);
    printf ("x的y次方: %f\n",compute(x,y));
}
```

我們使用 5p 來將所複製的那一行貼上去五遍。

```
double compute(double x,double y);
main ()
{
    float x, y;
    printf ("請輸入x的y次方，以便求x的y次方的值\n");
    printf ("請輸入x:");
    printf ("請輸入x:");
    printf ("請輸入x:");
    printf ("請輸入x:");
    printf ("請輸入x:");
    printf ("請輸入x:");
    scanf ("%f", &x);
    printf ("請輸入y:");
    scanf ("%f", &y);
    printf ("x的y次方: %f\n",compute(x,y));
}
```

我們使用 3dd 將游標所在的下面三行給刪除。









: 5,7s/xz/g , 將第五行到第七行的 x 字元替換成 z 字元。

```

1 double compute(double x,double y);
2 main ()
3 {
4     float x, y;
5     printf ("請輸入x的y次方，以便求x的y次方的值\n");
6     printf ("請輸入x:");
7     printf ("請輸入x:");
8 void good(char *);
9 void lucky(char *);
10
11     printf ("請輸入x:");
12     掃描 ("%f", &x);
13     printf ("請輸入y:");
14     掃描 ("%f", &y);
15     printf ("x的y次方: %f\n",compute(x,y));
16 }
~
~
~
~
~
~
~
:5,7s/x/z/g

```

這是替換的情況。

```

5     printf ("請輸入z的y次方，以便求z的y次方的值\n");
6     printf ("請輸入z:");
7     printf ("請輸入z:");

```

我們使用<Esc>回到命令模式中，我們按下 u 就可以回到還沒替換的步驟前。u 就是 undo 的意義。u 就是回到前一步驟。這是還未替換 x 成 z 前的情況。

```

1 double compute(double x,double y);
2 main ()
3 {
4     float x, y;
5     printf ("請輸入x的y次方，以便求x的y次方的值\n");
6     printf ("請輸入x:");
7     printf ("請輸入x:");
8 void good(char *);
9 void lucky(char *);
10
11     printf ("請輸入x:");
12     掃描 ("%f", &x);
13     printf ("請輸入y:");
14     掃描 ("%f", &y);
15     printf ("x的y次方: %f\n",compute(x,y));
16 }

```

我們使用 w power.c 來將資料儲存到 power.c 上面。



```

1 double compute(double x,double y);
2 main ()
3 {
4     float x, y;
5     printf ("請輸入x的y次方，以便求x的y次方的值\n");
6     printf ("請輸入x:");
7     printf ("請輸入x:");
8     void good(char *);
9     void lucky(char *);
10
11    printf ("請輸入x:");
12    掃描 ("%f", &x);
13    printf ("請輸入y:");
14    掃描 ("%f", &y);
15    printf ("x的y次方: %f\n",compute(x,y));
16 }
~
~
~
~
:w power.c

```

我們按下 ZZ 就可以離開 vi 編輯器了。

我們在 vi 文書編輯器上的命令模式下也可以執行 shell。

語法：

:!shell 指令

我們在命令模式輸入 :!ls 就可以顯示目錄的內容。

```

:!ls
[No write since last change]
\      1-7      good.c      love.txt  power.c,v  test.TXT.bak
001.tif compute.c good.txt    lucky.c   power.c    total
002.tif cvsroot   good.txt.save macrol    powerv     total2
003.tif fol.txt   heapsort   makefile  -print    total.c
1-1    fo2.txt   heapsort1  oo.txt    rect      total.c,v
1-3    fool.txt heapsort.c power     rectangle.c uu.tt
1-4    foo.paths lib.a     powerl    rectangle.c~ win.c
1-5    foo.txt   lib.h     powerl.c  test.c    test.c
1-6    foo.txt,v love_pico.txt power.c   test.TXT

```

我們在命令模式輸入 :!pwd 就可以顯示目前的目錄。

```

:!pwd
[No write since last change]
/home/chaiyen

```

## 13-2 pico 文書編輯器





```
[root@flash chaiyen]# pico -h
```

這就是 pico 的參數與說明。

Possible Starting Arguments for Pico editor:

Argument	Meaning
-e	Complete - allow file name completion
-k	Cut - let ^K cut from cursor position to end of line
-a	ShowDot - show dot files in file browser
-j	Goto - allow 'Goto' command in file browser
-g	Show - show cursor in file browser
-m	Mouse - turn on mouse support
-x	NoKeyhelp - suppress keyhelp
defaults	
-d	Rebind - let delete key delete current character
-f	Keys - force use of function keys
-b	Replace - allow search and replace
-h	Help - give this list of options
-r[#cols]	Fill - set fill column to #cols columns, default=72
80	
-s <speller>	Speller - specify alternative speller
-t	Shutdown - enable special shutdown mode
-o <dir>	Operation - specify the operating directory
-z	Suspend - allow use of ^Z suspension
-w	NoWrap - turn off word wrap
+ [line#]	Line - start on line# line, default=1

當我們在編輯檔案時，只要按下<Ctrl-O>這樣就可以把檔案儲存起來。<Ctrl-O>是 Ctrl 控制鍵和 O 鍵一起按下的意義，也可以表示成^O 的符號。而如果我們要離開 pico 則我們只要輸入<Ctrl-X>，這樣就可以離開 pico 編輯器了。

我們按下^O 時，只要輸入要儲存的檔名，就可以把資料存入檔案中。這是在鍵盤按鈕指令區。

```
File Name to write : good.txt
^G Get Help      ^T To Files
^C Cancel       TAB Complete
```





## 13-2-1 一般鍵盤指令和游標移動

除了將資料寫入檔案的指令<Ctrl-O>^O(控制鍵 Ctrl 加 O 鍵)和離開 pico 編輯器的指令<Ctrl-X>^X(控制鍵加 X 鍵)，pico 還有下表的指令。

按鍵指令	說明
<^M>	開始標示要剪下的文字區
<^C>	報告目前游標所在的位置。 以行#和字元#表示。
<^G>	Pico 的說明
<^J>	格式排列所選取的文字。
<^K>	剪下所選取的文字。
<^O>	儲存目前的資料到檔案中。
<^R>	從指定檔案中讀取文字，再將它貼到目前檔案中。
<^T>	拼字檢查。
<^U>	將複製的文字貼到目前所在的行位置。
<^V>	在 help 說明的地方往下拉一頁。
<^W>	搜尋所指定的文字。
<^X>	儲存檔案的內容然後離開 pico 編輯器。
<^Y>	在 help 說明中往上一頁。

我們可以用上下左右鍵來控制游標的上下左右移動。

<^M>開始標示要剪下的文字區。我們可以使用這個按鈕來開始標示我們所要的區域。

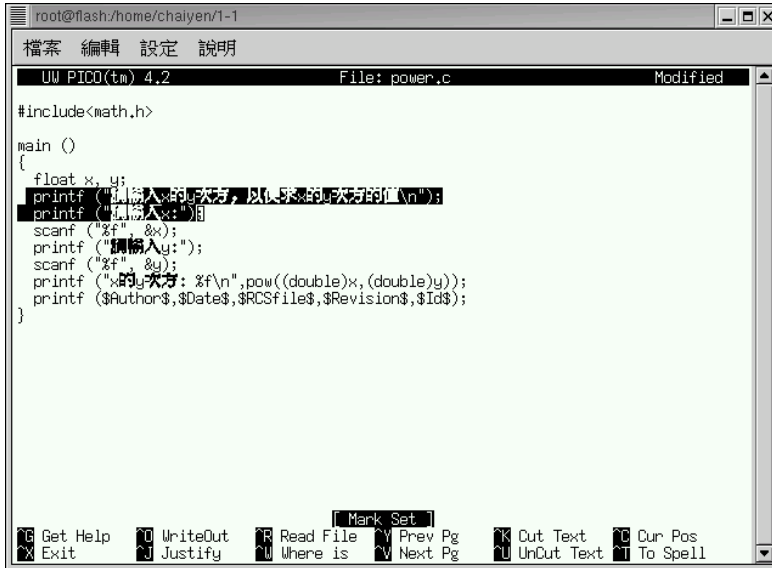
```

root@flash:/home/chaiyen/1-1
檔案 編輯 設定 說明
[root@flash 1-1]# pico power.c

```

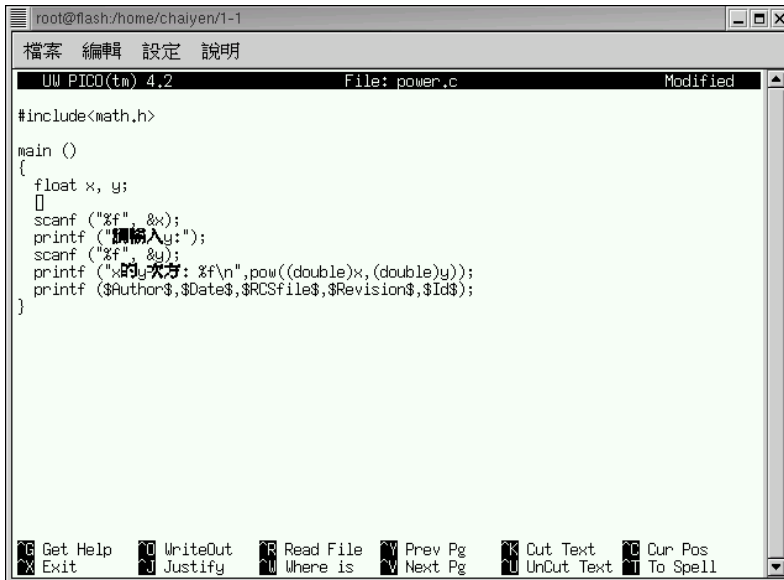


我們使用 $\langle \wedge \rangle$ 鍵 $\langle \text{Ctrl-}\wedge \rangle$ 就可以將游標反白，再用移動游標，就可以將指定的地方給標示 mark 起來。



```
root@flash:/home/chaiven/1-1
檔案 編輯 設定 說明
UW PICO(tm) 4.2 File: power.c Modified
#include<math.h>
main ()
{
float x, y;
printf ("請輸入x的次方,以及x的次方的值\n");
printf ("請輸入x:");
scanf ("%f" &x);
printf ("請輸入y:");
scanf ("%f" &y);
printf ("x的y次方: %f\n",pow((double)x,(double)y));
printf (" $Author$, $Date$, $RCSfile$, $Revision$, $Id$");
}
Mark Set
Get Help WriteOut Read File Prev Pg Cut Text Cur Pos
Exit Justify Where is Next Pg UnCut Text To Spell
```

我們使用 $\langle \wedge \rangle$ 鍵 $\langle \text{Ctrl-K} \rangle$ ，將所標示反白的地方剪下來。



```
root@flash:/home/chaiven/1-1
檔案 編輯 設定 說明
UW PICO(tm) 4.2 File: power.c Modified
#include<math.h>
main ()
{
float x, y;
scanf ("%f" &x);
printf ("請輸入x:");
scanf ("%f" &y);
printf ("x的y次方: %f\n",pow((double)x,(double)y));
printf (" $Author$, $Date$, $RCSfile$, $Revision$, $Id$");
}
Get Help WriteOut Read File Prev Pg Cut Text Cur Pos
Exit Justify Where is Next Pg UnCut Text To Spell
```



這是我們使用<^U>將所剪下的文字貼上去。

```

root@flash:/home/chaiken/1-1 - Shell - Konsole
工作階段 編輯 檢視 設定 說明
UW PICO(tm) 4.2 File: power.c Modified
#include<math.h>
main ()
{
float x, y;
printf ("請輸入x的y次方，以便求x的y次方的值\n");
printf ("請輸入x:");
printf ("請輸入x的y次方，以便求x的y次方的值\n");
printf ("請輸入x:");
scanf ("%f", &x);
printf ("請輸入y:");
scanf ("%f", &y);
printf ("x的y次方: %f\n", pow((double)x, (double)y));
printf (" $Author$, $Date$, $RCSfile$, $Revision$, $Id$");
}
    
```

我們按下<^J>就可以格式排列所選取的文字。將我們的文字排列格式化。

```

UW PICO(tm) 4.2 File: power.c Modified
#include<math.h>
main ()
{
    float x, y;
    printf ("請輸入x的y次方，以便求x的y次方的值\n");
    printf ("請輸入x:");
    scanf ("%f", &x);
    printf ("請輸入y:");
    scanf ("%f", &y);
    printf ("x的y次方: %f\n", pow((double)x, (double)y));
    printf (" $Author$, $Date$, $RCSfile$, $Revision$, $Id$");
}
    
```

我們現在可以使用<^R>鍵將其它檔案的內容輸入到我們游標所在的位置。我們現在要將 lib.h 的檔案輸入到我們游標所在的地方。

```
[root@flash 1-1]# ls  
lib.h power.c power.c.save
```

我們按下<^R>鍵，就可以將 lib.h 的檔案輸入到我們游標所在的位置。

```
UW PICO(tm) 4.2 File: power.c  
#include<math.h>  
  
main ()  
{  
    float x, y;  
    printf ("請輸入x的y次方，以便求x的y次方的值\n");  
    printf ("請輸入x:");  
    scanf ("%f", &x);  
    printf ("請輸入y:");  
    scanf ("%f", &y);  
    printf ("x的y次方: %f\n", pow((double)x,(double)y));  
    printf (" $Author$, $Date$, $RCSfile$, $Revision$, $Id$");  
}
```

```
File to insert from home directory:  
^G Get Help ^T To Files  
^C Cancel
```

這是我們要輸入的檔案 lib.h，我們在指令上輸入我們的檔案。

```
File to insert from home directory: lib.h  
^G Get Help ^T To Files  
^C Cancel
```



這是我們將 lib.h 的檔案輸入到螢幕的情況。

```

UW PICO(tm) 4.2                               File: power.c                               Modifie
#include<math.h>

main ()
{
  float x, y;
  printf ("請輸入x的y次方，以便求x的y次方的值\n");
  printf ("請輸入x:");
  scanf ("%f", &x);
  printf ("請輸入y:");
  scanf ("%f", &y);
  printf ("x的y次方: %f\n",pow((double)x,(double)y));
  printf ($Author$, $Date$, $RCSfile$, $Revision$, $Id$);
  void good(char *);
  void lucky(char *);
}

Inserted 3 lines
^G Get Help  ^O WriteOut  ^R Read File  ^Y Prev Pg  ^K Cut Text  ^C Cur Pos
^X Exit      ^J Justify   ^W Where is  ^V Next Pg  ^U UnCut Text ^T To Spell

```

我們按下<^G>控制鍵加 G 鍵，就可以打開 pico 的說明檔。我們按下<^X>鍵就可以離開 pico 的說明檔。

```

UW PICO(tm) 4.2                               File: power.c                               Modified
Pico Help Text

Pico is designed to be a simple, easy-to-use text editor with a
layout very similar to the pine mailer. The status line at the
top of the display shows pico's version, the current file being
edited and whether or not there are outstanding modifications
that have not been saved. The third line from the bottom is used
to report informational messages and for additional command input.
The bottom two lines list the available editing commands.

Each character typed is automatically inserted into the buffer
at the current cursor position. Editing commands and cursor
movement (besides arrow keys) are given to pico by typing
special control-key sequences. A caret, '^', is used to denote
the control key, sometimes marked "CTRL", so the CTRL-d key
combination is written as ^O.

The following functions are available in pico (where applicable,
corresponding function key commands are in parentheses).

Inserted 3 lines
^X Exit Help                               ^V Next Pg

```



我們按下<^V>鍵就可以到說明的下一頁。

```

UW PICO(tm) 4.2                               File: power.c                               Modified
The following functions are available in pico (where applicable,
corresponding function key commands are in parentheses).

^G (F1)   Display this help text.

^F         move Forward a character.
^B         move Backward a character.
^P         move to the Previous line.
^N         move to the Next line.
^A         move to the beginning of the current line.
^E         move to the End of the current line.
^V (F8)   move forward a page of text.
^Y (F7)   move backward a page of text.

^W (F6)   Search for (where is) text, neglecting case.
^L         Refresh the display.

^D         Delete the character at the cursor position.
^A         Mark cursor position as beginning of selected text.
          Inserted 3 lines
          ^Y Prev Pg
          ^V Next Pg
^X Exit Help

```

我們按一下<^Y>鍵就可以到說明的前一頁。

```

UW PICO(tm) 4.2                               File: power.c                               Modified
^D         Delete the character at the cursor position.
^A         Mark cursor position as beginning of selected text.
          Note: Setting mark when already set unselects text.
^K (F9)   Cut selected text (displayed in inverse characters).
          Note: The selected text's boundary on the cursor side
          ends at the left edge of the cursor. So, with
          selected text to the left of the cursor, the
          character under the cursor is not selected.
^U (F10)  Uncut (paste) last cut text inserting it at the
          current cursor position.
^I         Insert a tab at the current cursor position.

^J (F4)   Format (justify) the current paragraph.
          Note: paragraphs delimited by blank lines or indentation.
^T (F12)  To invoke the spelling checker
^C (F11)  Report current cursor position

^R (F5)   Insert an external file at the current cursor position.
^O (F3)   Output the current buffer to a file, saving it.
          Inserted 3 lines
          ^Y Prev Pg
          ^V Next Pg
^X Exit Help

```



我們按下<^W>鍵就可以到搜尋字串的指令。

```
UW PICO(tm) 4.2                               File: power.c
#include<math.h>
main ()
{
    float x, y;
    printf ("請輸入x的y次方，以便求x的y次方的值\n");
    printf ("請輸入x:");
    scanf ("%f", &x);
    printf ("請輸入y:");
    scanf ("%f", &y);
    printf ("x的y次方: %f\n", pow((double)x, (double)y));
    printf ($Author$, $Date$, $RCSfile$, $Revision$, $Id$);
}

Search :
^G Get Help  ^Y FirstLine  ^T LineNumber ^O End of Par
^C Cancel    ^V LastLine   ^W Start of P
```

我們在 Search 上輸入我們要搜尋的字串 pow。

```
Search : pow
^G Get Help  ^Y FirstLine  ^T LineNumber ^O End of Par
^C Cancel    ^V LastLine   ^W Start of P
```

游標就會到我們要搜尋的地方 pow。

```
printf ("x的y次方: %f\n", pow((double)x, (double)y));
printf ($Author$, $Date$, $RCSfile$, $Revision$, $Id$);
```



我們按下<^T>鍵就可以到拼字檢查的指令。

```
UW PICO(tm) 4.2                               File: power.c
#include<math.h>
main ()
{
    float x, y;
    printf ("請輸入x的y次方，以便求x的y次方的值\n");
    printf ("請輸入x:");
    scanf ("%f", &x);
    printf ("請輸入y:");
    scanf ("%f", &y);
    prin ("x的y次方: %f\n", pow((double)x,(double)y));
    printf ($Author$, $Date$, $RCSfile$, $Revision$, $Id$);
}

Edit a replacement: prin
^G Get Help
^C Cancel
```

我們找到一個 prin 拼字錯誤的字，因該是 printf。

```
Edit a replacement: prin
^G Get Help
^C Cancel
```





我們按下<^C>鍵就可以報告目前游標所在的位置。以行#和字元#表示。

```

UW PICO(tm) 4.2      File: power.c      Modified
#include<math.h>

main ()
{
  float x, y;
  printf ("請輸入x的y次方，以便求x的y次方的值\n");
  printf ("請輸入x:");
  scanf ("%f", &x);
  printf ("請輸入y:");
  scanf ("%f", &y);
  printf ("x的y次方: %f\n", pow((double)x,(double)y));
  printf (" $Author$, $Date$, $RCSfile$, $Revision$, $Id$");
}

```

[ line 11 of 14 (78%), character 267 of 369 (72%) ]

^G Get Help	^O WriteOut	^R Read File	^Y Prev Pg	^K Cut Text	^C Cur Pos
^X Exit	^J Justify	^W Where is	^V Next Pg	^U UnCut Text	^T To Spell

這顯示我們的游標目前在第十一行處，並且在字元第 267。

[ line 11 of 14 (78%), character 267 of 369 (72%) ]

^G Get Help	^O WriteOut	^R Read File	^Y Prev Pg	^K Cut Text	^C Cur Pos
^X Exit	^J Justify	^W Where is	^V Next Pg	^U UnCut Text	^T To Spell

### 13-3 emacs 文書編輯器

在 Linux 上面，emacs 文書編輯器是功能最多，而且最好用的文書編輯器。

Emacs 文書編輯器是全螢幕的文書編輯器，我們使用<Ctrl>加上其它鍵來控制，或者使用<Esc>。典型的螢幕是由大型的區域進入區所組成，我們可以在此區域輸入文字，而螢幕的底下則是命令模式。在命令模式區，可以顯示資訊和提示。下面是最重要的 emacs 命令。



指令	說明
<Ctrl-X>+<Ctrl-C>	離開 emacs
<Ctrl-G>	取消目前的指令和操作
<Ctrl-X>+<Ctrl-W>	儲存到目前都還未儲存的緩衝區
<Ctrl-X>+<Ctrl-S>	儲存緩充區
<Ctrl-X>+<U>	回複到前一步驟
<Ctrl-H>	說明文件
<Ctrl-X  >	從檔案插入到目前游標所在位置
<Ctrl-X 1>	刪除說明視窗

我們使用 emacs alien 就可以使用 emacs 文書編輯器來編輯 alien。

```
[root@flash chaiyen]# emacs alien
```

這是我們進入 emacs 文書編輯器的畫面。

```
File Edit Options Buffers Tools Help
#dos aliases
alias dir = "ls -la"
alias type = "more"
alias del = "rm"
alias type = "more"

-BB-:**-F1 alien (Fundamental)--L5--All--
```



我們按下<Ctrl-X>+<Ctrl-W>就可以將緩衝區的資料寫入並且儲存。

```
-BB-:**-F1 alien
C-x-
```

這是我們將資料寫入目錄中。

```
-BB-:--F1 alien
Write file: /home/chaiken/ _
```

我們按下<Ctrl-x>+<Ctrl-C>就可以離開 emacs 文書編輯器了。

```
-BB-:**-F1 alien (Fundamental)--L2--All-----
Save file /home/chaiken/alien? (y, n, !, ., q, C-r or C-h)
```

我們使用 emacs 檔案就可以使用 emacs 文書編輯器來編輯指定的檔案。

語法：

指令：emacs [參數] [檔案]

參數：

+n：從第 n 行開始編輯檔案。

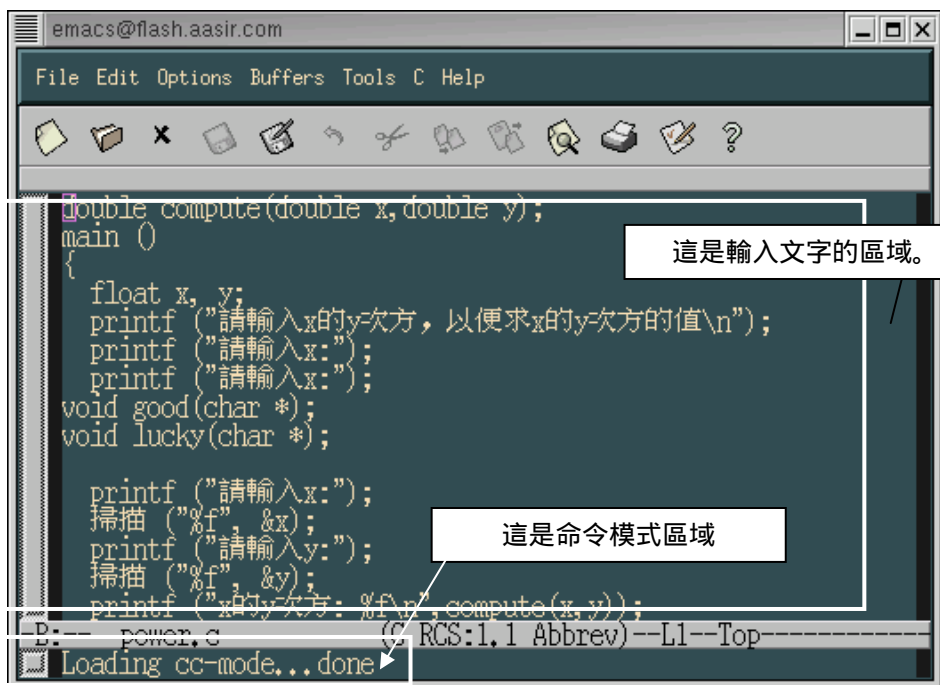
-nw：沒有打開視窗而直接執行，適合在使用者圖形化介面。

我們在 x 視窗中的終端機打開 emacs。我們輸入 emacs power.c 就可以使用 emacs 來編輯 power.c 檔案。

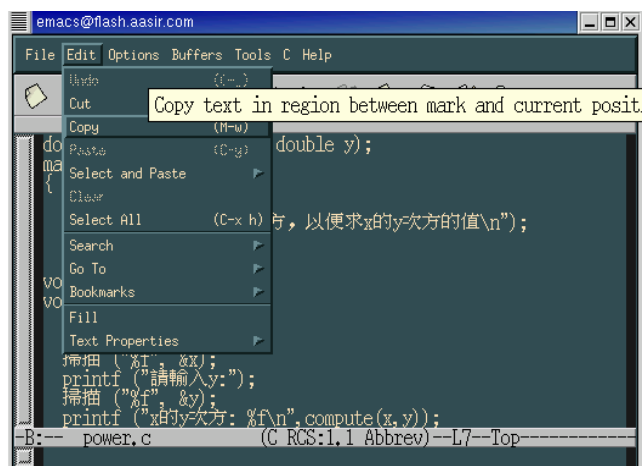
```
root@flash:/home/chaiken - Shell - Konsole
工作階段 編輯 檢視 設定 說明
[root@flash chaiken]# emacs power.c
```



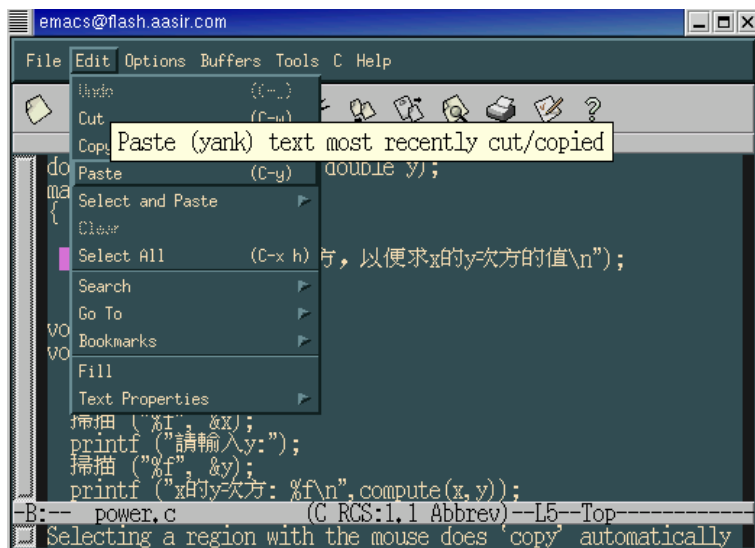
這是我們進入 emacs 的畫面。



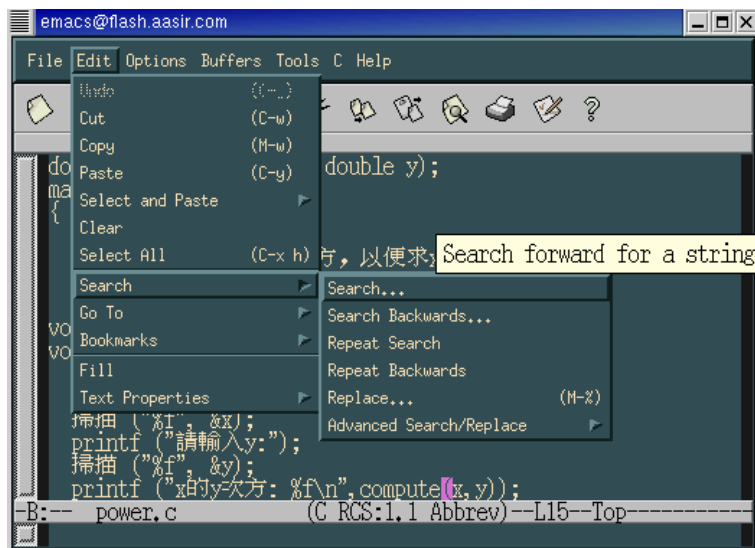
我們如果要複製文字，我們可以先選取反白的區域，我們再選取 Edit 編輯，再選取 Copy 就可以將反白的文字複製到緩衝區。



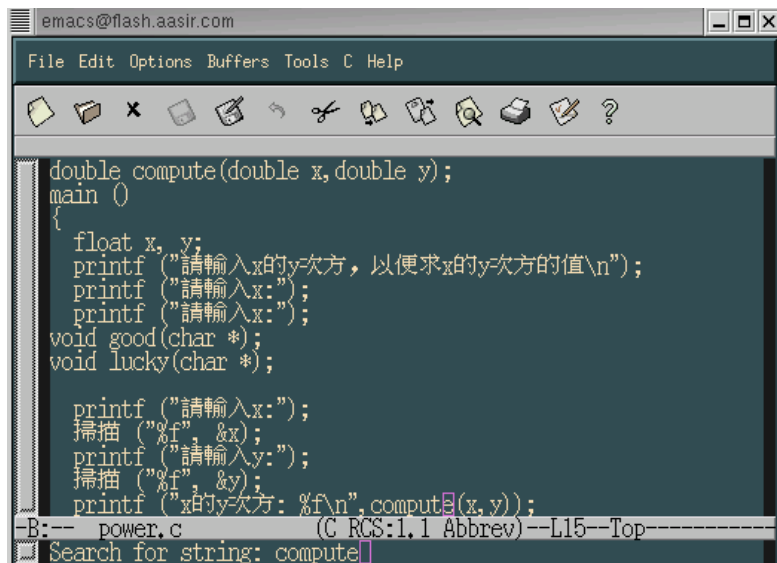
我們也可以再使用 Edit 編輯，再選取 Paste 貼上，將緩衝區的文字貼到游標所在的地方。



我們如果要搜尋在檔案中的字串，我們可以選取 Edit 再選取 Search，這樣就可以搜尋我們所要的字串。



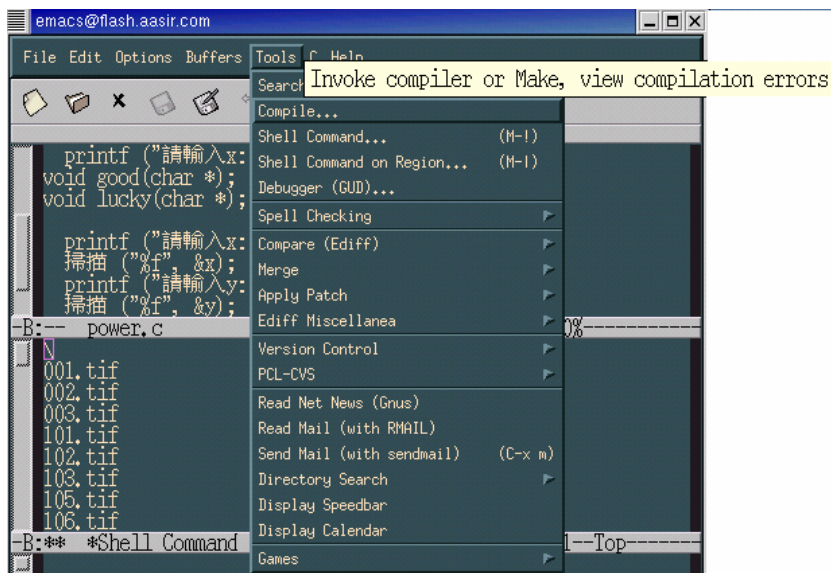
我們在底下的命令區輸入所要搜尋的字串 compute。



```
emacs@flash.aasir.com
File Edit Options Buffers Tools C Help
double compute(double x,double y);
main ()
{
  float x, y;
  printf ("請輸入x的y次方, 以便求x的y次方的值\n");
  printf ("請輸入x:");
  printf ("請輸入x:");
  void good(char *);
  void lucky(char *);

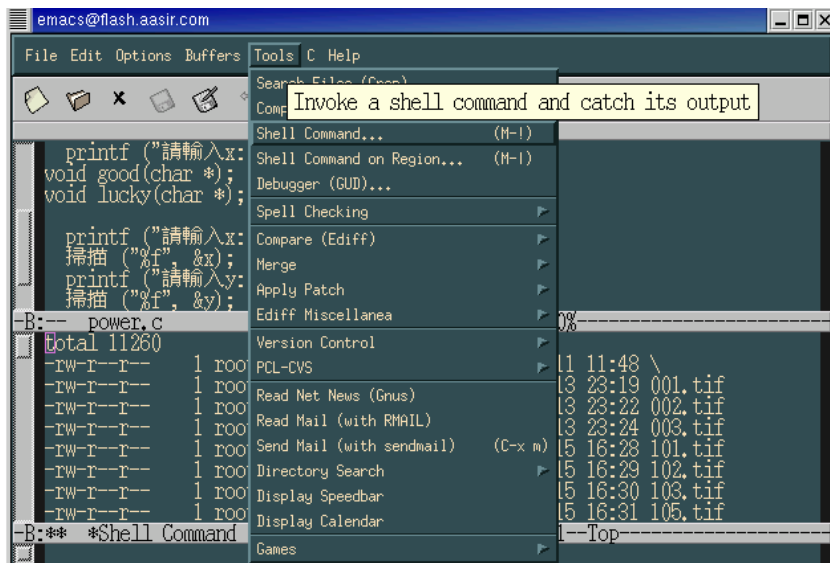
  printf ("請輸入x:");
  掃描 ("%f", &x);
  printf ("請輸入y:");
  掃描 ("%f", &y);
  printf ("x的y次方: %f\n", compute(x,y));
}
-B:-- power.c (C RCS:1.1 Abbrev)--L15--Top
Search for string: compute
```

我們可以使用 Tools 的工具、來編譯我們所寫的 C 語言程式。選取 Tools 再選取 Compile 編譯。

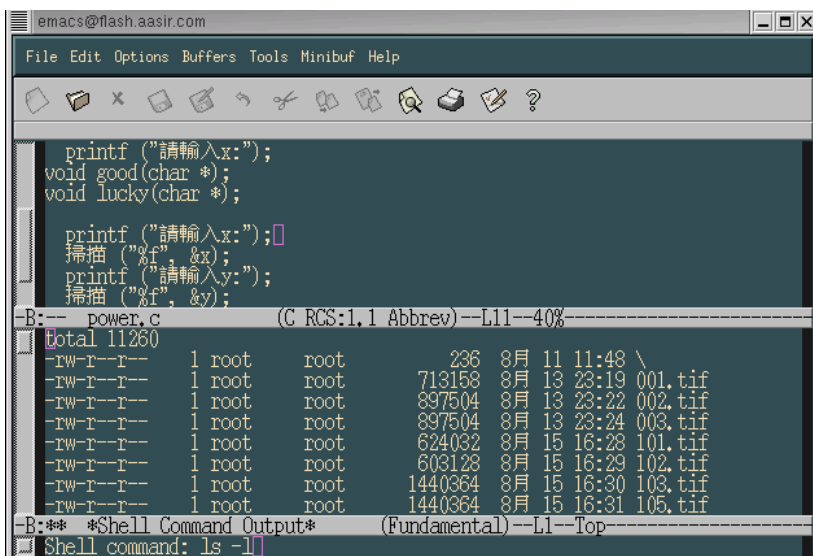


```
emacs@flash.aasir.com
File Edit Options Buffers Tools C Help
Search Invoke compiler or Make, view compilation errors
Compile...
Shell Command... (M-I)
Shell Command on Region... (M-I)
Debugger (GUD)...
Spell Checking
Compare (Ediff)
Merge
Apply Patch
Ediff Miscellaneous
Version Control
PCL-CVS
Read Net News (Gnus)
Read Mail (with RMAIL)
Send Mail (with sendmail) (C-x m)
Directory Search
Display Speedbar
Display Calendar
Games
```

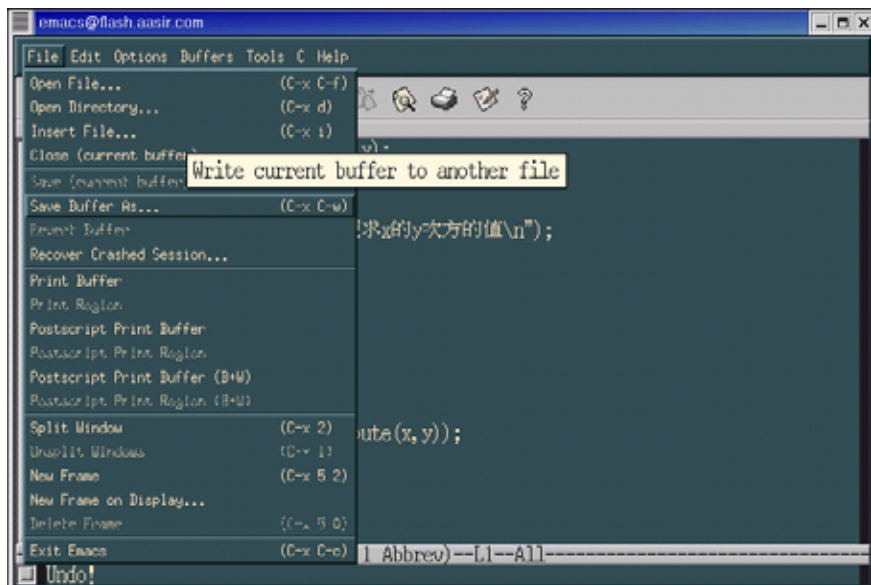
如果我們要在 emacs 上執行 shell 的指令，我們可以先選取 Tools 工具，再選取 Shell Command。



我們然後就可以在底下的 Shell Command 命令模式下輸入指令。我們輸入 `ls -l` 就可以顯示目錄的內容。



我們在選單上選取 File 檔案，我們再選取 Save Fuffer As 就可以將檔案儲存。



我們在視窗底部的命令模式將檔案儲存。

