

# Contents

<b>1</b>	<b>small tips</b>	<b>1</b>
1.1	default directory . . . . .	1
1.2	number list . . . . .	2
1.3	calculate . . . . .	2
1.4	lambda function . . . . .	2
1.5	hex conversion . . . . .	2
1.6	mail in command line . . . . .	2
1.7	cron table . . . . .	3
1.8	merge output . . . . .	3
1.9	distro check . . . . .	3
1.10	different authority for different user . . . . .	3
1.11	use footnote in table or other environments . . . . .	3
1.12	use locate to find directories . . . . .	3
1.13	list directory by size . . . . .	4
1.14	get some informations in sqlite . . . . .	4
1.14.1	get column names from certain table . . . . .	4
1.14.2	get tables names from current dbfile . . . . .	4
1.15	batch rename file with sequential numbers . . . . .	4
1.16	some tips of regular expression . . . . .	5
1.16.1	find pattern in many files . . . . .	5
1.17	change the calling priority of commands . . . . .	5
1.18	some things for git . . . . .	5
1.19	reverse string . . . . .	6
1.20	generate package list installed in archlinux . . . . .	6
1.21	count the matched pattern in vim . . . . .	6
1.22	convert markdown to pdf . . . . .	6
1.23	get http status code by curl . . . . .	7
1.24	compare the contents in two directories . . . . .	7
1.25	change contents inside "", [], () in vim . . . . .	7
1.26	shell string calculating . . . . .	7
1.27	some regex common used in shell . . . . .	8

## 1 small tips

### 1.1 default directory

the directory of xdg-default were written in `/.config/xdg-dirs.dirs`. For example, if you want to set the default directory gnome search for wallpaper, you can change the value of “XDG\_PICTURE\_DIR” in that file to the directory path you want to.

## 1.2 number list

to put a list of continuous of number in vim, use `:put = range(n,m)`, and will generated a list from n to m in document. if want random list, should use shell command `! echo $RANDOM`<sup>1</sup>, and m is the limit of random.

to generate a continuous number list by shell, use `seq $BEGIN $END`, so for loop can use: `for i in $(seq $BEGIN $END);do ...` or `for i in {$BEGIN .. $END}; do ...`

## 1.3 calculate

to calculate the sum of the first line in file, use:

`awk 's+=$1 END print s' file_include_data`

if the data is not in the first line, change \$1.

and some times awk will get into bug if the sum exceeding  $2^{31}$ , so use `printf` is a good choice: change the 'print s' to '`printf "%0f", s`'.

## 1.4 lambda function

python lambda function example: `test = lambda x, n: [x[i:i+n] for i in range(0, len(x), n)]` this can apart the string every n chars<sup>2</sup>, and lambda function usually used for the functions needed in a short time, it can also use with `filter()` or `map()` function, the syntax of lambda function: "lambda arguments: expression"

## 1.5 hex conversion

`awk '{print "ibase=10;obase=2;" $1 }' $file |bc |xargs printf "%08d\n"`

this can convert every line in \$file from decimal to binary of 8 bits.

"`xxd -p`" can print only the value, no line number and characters.

python convert hex to bin: `bin(int3(str, 16))[2:]`

## 1.6 mail in command line

config before use "mail" command to send e-mail: edit config file<sup>4</sup>, add these lines:

```
set from="your_email_address" smtp="smtp_address_of_your_mail_server"
set smtp-auth-user="your_email_address" smtp-auth-password="your_password"
set smtp-auth=login set smtp-use-starttls(this enable SSL)
```

---

<sup>1</sup>:`! echo $((RANDOM%m))` seems can not be used in vim

<sup>2</sup>this can use `re.findall('..?',str)` to replace

<sup>3</sup>`int()` function have two arguments, the first is string of number, and the last is the base of this number, for example, `int('AB9', 16)` can get  $2745((2745)_{10} = (AB9)_{16})$ .

<sup>4</sup>archlinux use `/etc/mail.rc`, ubuntu use `/etc/s-nail.rc`

## 1.7 cron table

set cron work:

use crontab -e, and add work in the file, the format is: *min hour day month year work*.

## 1.8 merge output

if you want to use the output of multi command as input by pipe, then you need to parenthesis these two commands, such as:

```
(echo "test" ; cat hello.txt) |mail -s "test" username@mailaddress
```

## 1.9 distro check

if you want to check your distro without screenfetch or neofetch installed, you can use: "cat /etc/\*-release", then it will output the distro information of your distro.

## 1.10 different authority for different user

if you want to make different users have different authority to a certain file, you can use *setfacl*: for example, if you want to set all authority to *file* file with group *group*, you can use:

```
setfacl -m g:group:rwx -R5 file
```

and then use:"setfacl -m g:<sup>6</sup>test:r -R file"to set only read authority to group test.

## 1.11 use footnote in table or other environments

use \footnotemark in the position you need to note, and then use \footnotetext{*your\_\_footnote*} out the environment.

## 1.12 use locate to find directories

locate didn't have that option, but we can use the feature of locate to get that:

```
locate -b '/dir_name'
```

or you can use regex:

```
locate -r dir_name$
```

---

<sup>5</sup>-R is to use recursive

<sup>6</sup>for user, use u:

### 1.13 list directory by size

```
du -sh -B BLOCKSIZE */ |sort -nr
```

This command will list your subdirectory in current directory by size and from big to small, *BLOCKSIZE* such as M, it will output the size by certain format, \*/ is path, you can also use another path replace it, but you should add “/\*” in the tail.

```
du -sh */ |sort -h
```

This will directly sort by size, not need to appoint block size.

### 1.14 get some informations in sqlite

#### 1.14.1 get column names from certain table

if you are in sqlite command, you can execute ‘.schema *table\_name*’ to get it;  
or if you are in python or you just want to use sql command to get it, you can try:

```
SELECT sql FROM sqlite_master WHERE tbl_name = 'table_name' AND type = 'table'
```

#### 1.14.2 get tables names from current dbfile

if you are in sqlite command, you can execute ‘.tables’ to get it;  
or if you are in python or you just want to use sql command to get it, you can try:

```
SELECT name FROM sqlite_master WHERE type = 'table'
```

### 1.15 batch rename file with sequential numbers

Firstly, set a variable:

```
a=1
```

and then execute the while loop (or for loop):

```
ls *.jpg | while read line;do mv $line` printf "%03d.jpg" "$a"` ;let a=a+1;done
```

## 1.16 some tips of regular expression

To change a certain line a a certain text block, you can use:

```
sed -i "/pattern1/,/pattern2/s/origin_pattern/dest_pattern/g" filename
```

### 1.16.1 find pattern in many files

```
grep -rnw '/path/to/destination' -e 'pattern'
```

## 1.17 change the calling priority of commands

To call commands in linux, the system will find the command in the paths which *PATH* variable stores, and the order of path in *PATH* variable depend the priority of calling commands. So, you can adjust the order of paths in *PATH* variable to make you call certain commands.

For example, if you have a different version of gcc from gcc in your system in your current directoy, you can use

```
export PATH=.:$PATH$
```

and then when you type */usr/bin/envgcc*, or execute the program include it, you use the one in your current directory <sup>7</sup>

## 1.18 some things for git

### 1. tracking

- `git add -A` stages All
- `git add .` stages new and modified, without deleted
- `git add -A` stages modified and deleted.
- `git clean` delete all untracked files, you can use `'-n'` to only list them.

- ### 2. do delete, modify or some other operations to your older commits:
- use `git rebase -i <commit_name>`, and you will enter the editor, and there will be commands in comments, such as drop, pick, squash and so on.

---

<sup>7</sup>It's temporarily, it malfunctions after you logout the shell, expect you put this in your *./bashrc*.

3. about github: **if you want to amend or cancel a commit**<sup>8</sup>, use `git push -f github_repository branch` to force push to your github repository.

### 1.19 reverse string

- if just reverse the order of lines in file, use `tac filename`
- if you want to reverse the order of every character in file, use `tac -r -s '' filename`

### 1.20 generate package list installed in archlinux

```
pacman -Qqe > filename
```

And if you want to list the low-level packages as dependencies, you can add '-t' option.<sup>9</sup>

If you don't want to list packages in aur, you can add '-n' option.

Then, when you want to reinstall these packages in a new archlinux, you can run

```
pacman -S - < filename
```

### 1.21 count the matched pattern in vim

Type this in commandline mode in vim:

```
%s/<pattern>/gn
```

### 1.22 convert markdown to pdf

```
pandoc source -f markdown -o destination
```

if you want to convert document which include unicode characters, you can appoint the engine:

```
pandoc source -f markdown -o destination --pdf-engine=xelatex -V CJKmainfont=fontname
```

---

<sup>8</sup>use `git reset --soft HEAD~1`

<sup>9</sup>Reference in Archwiki

### 1.23 get http status code by curl

```
curl -o -I -L -s -w "%{http_code}\n" host_url
```

### 1.24 compare the contents in two directories

```
diff --brief -r directory1 directory2
```

shorter for:

```
diff -qr directory1 directory2
```

If you also want to see differences for files that may not exist in either directory:

```
diff --brief -Nr dir1/ dir2/
```

### 1.25 change contents inside "", [], () in vim

For ", you can use this in command mode:

```
ci"
```

and for (), [], you can use `ci(`, `ci[`

for y, it also works, if use a, it will copy the symbol also.

`cit` can be used to change the contents in html or xml tags.

and for ", you can use `\textquotedbl` to print it in L<sup>A</sup>T<sub>E</sub>X.

### 1.26 shell string calculating

if you have a variable which stores a string, like `TEST="this is a test"`

Then, if you want to get the length of it, you can simply type

```
echo ${#TEST}
```

on you shell, and then you get the length of variable TEST.

```
echo ${TEST:n}
```

This can output the  $n_{th}$  to the tail of string \$TEST.

```
echo ${TEST:n:m}
```

This can output  $m$  characters begin from  $n_{th}$  of string \$TEST, if  $m < 0$ , you will get  $n_{th}$  to count down  $m_{th}$  characters.

As for string add, you can just put the two variables as the arguments of echo >\_<.

You can use this in the batch processing, for example, traversal all jpg files, and remove the extension, and then convert them to png by imagemagick.

```
for i in *.jpg ; do convert $i ${i:0:-4}.png ; done
```

## 1.27 some regex common used in shell

I think the most common regex in shell is '\*', and it match any string.

This may be 'or': '{str1,str2}', match str1 or str2, you can also use more, septate by ','.