Ten Steps to Linux Survival

Bash for Windows People

Jim Lehmer

2015

Figure 1: Merv sez, “Don’t panic.”

Step -1 Overview

Ten Steps

- Step 0 - Some History

Ten Steps

- Step 0 - Some History
- Step 1 - Come Out of Your Shell

Ten Steps

- Step 0 - Some History
- Step 1 - Come Out of Your Shell
- Step 2 - File Under “Directories”

Ten Steps

- Step 0 - Some History
- Step 1 - Come Out of Your Shell
- Step 2 - File Under “Directories”
- Step 3 - Finding Meaning

Ten Steps

- Step 0 - Some History
- Step 1 - Come Out of Your Shell
- Step 2 - File Under “Directories”
- Step 3 - Finding Meaning
- Step 4 - Grokking `grep`

Ten Steps

- Step 0 - Some History
- Step 1 - Come Out of Your Shell
- Step 2 - File Under “Directories”
- Step 3 - Finding Meaning
- Step 4 - Grokking `grep`
- Step 5 - “Just a Series of Pipes”

Ten Steps

- Step 0 - Some History
- Step 1 - Come Out of Your Shell
- Step 2 - File Under “Directories”
- Step 3 - Finding Meaning
- Step 4 - Grokking `grep`
- Step 5 - “Just a Series of Pipes”
- Step 6 - `vi`

Ten Steps

- Step 0 - Some History
- Step 1 - Come Out of Your Shell
- Step 2 - File Under “Directories”
- Step 3 - Finding Meaning
- Step 4 - Grokking `grep`
- Step 5 - “Just a Series of Pipes”
- Step 6 - `vi`
- Step 7 - The Whole Wide World

Ten Steps

- Step 0 - Some History
- Step 1 - Come Out of Your Shell
- Step 2 - File Under “Directories”
- Step 3 - Finding Meaning
- Step 4 - Grokking `grep`
- Step 5 - “Just a Series of Pipes”
- Step 6 - `vi`
- Step 7 - The Whole Wide World
- Step 8 - The Man Behind the Curtain

Ten Steps

- Step 0 - Some History
- Step 1 - Come Out of Your Shell
- Step 2 - File Under “Directories”
- Step 3 - Finding Meaning
- Step 4 - Grokking `grep`
- Step 5 - “Just a Series of Pipes”
- Step 6 - `vi`
- Step 7 - The Whole Wide World
- Step 8 - The Man Behind the Curtain
- Step 9 - How Do You Know What You Don’t Know, `man`?

Ten Steps

- Step 0 - Some History
- Step 1 - Come Out of Your Shell
- Step 2 - File Under “Directories”
- Step 3 - Finding Meaning
- Step 4 - Grokking `grep`
- Step 5 - “Just a Series of Pipes”
- Step 6 - `vi`
- Step 7 - The Whole Wide World
- Step 8 - The Man Behind the Curtain
- Step 9 - How Do You Know What You Don’t Know, `man`?
- Step 10 - And So On

Step 0 Some History

Remember ONE Thing!

There is **no such thing** as “UNIX”

Remember ONE Thing!

There is **no such thing** as “UNIX”
…and that matters!

Step 1

Come Out of Your Shell

What is a shell?

Windows has a shell.
Two, in fact:

- `CMD.EXE`

What is a shell?

Windows has a shell.
Two, in fact:

- `CMD.EXE`
- `PowerShell.EXE`

What is a shell?

Windows has a shell.
Two, in fact:

- `CMD.EXE`
- `PowerShell.EXE`

What is a shell?

Windows has a shell.
Two, in fact:

- `CMD.EXE`
- `PowerShell.EXE`

Technically, Windows Explorer is a “shell” for the GUI environment.

"UNIX" has **lots** of shells

- **sh** - Bourne shell

"UNIX" has **lots** of shells

- **sh** - Bourne shell
  - **ash** - Almquist shell

"UNIX" has **lots** of shells

- **sh** - Bourne shell
  - **ash** - Almquist shell
    - **dash** - Debian Almquist shell

"UNIX" has **lots** of shells

- **sh** - Bourne shell
  - **ash** - Almquist shell
    - **dash** - Debian Almquist shell
  - **bash** - "Bourne-again" shell

"UNIX" has **lots** of shells

- **sh** - Bourne shell
  - **ash** - Almquist shell
    - **dash** - Debian Almquist shell
  - **bash** - "Bourne-again" shell
  - **ksh** - Korn shell

"UNIX" has **lots** of shells

- **sh** - Bourne shell
  - **ash** - Almquist shell
    - **dash** - Debian Almquist shell
  - **bash** - "Bourne-again" shell
  - **ksh** - Korn shell
  - **zsh** - Z shell

"UNIX" has **lots** of shells

- **sh** - Bourne shell
  - **ash** - Almquist shell
    - **dash** - Debian Almquist shell
  - **bash** - "Bourne-again" shell
  - **ksh** - Korn shell
  - **zsh** - Z shell
- **csh** - C shell

"UNIX" has **lots** of shells

- **sh** - Bourne shell
  - **ash** - Almquist shell
    - **dash** - Debian Almquist shell
  - **bash** - "Bourne-again" shell
  - **ksh** - Korn shell
  - **zsh** - Z shell
- **csh** - C shell
- ***and many more!***

Linux default shell

Typically `bash`

Comparing CMD.EXE and bash

set in bash

```
~ $ set
BASH=/bin/bash
BASHOPTS=checkwinsize:cmdhist:complete_fullquote:...
BASH_ALIASES=()
BASH_ARGC=()
BASH_ARGV=()
BASH_CMDS=()
...and so on...
```

set in CMD.EXE

```
C:\Users\myuser>set
ALLUSERSPROFILE=C:\ProgramData
APPDATA=C:\Users\myuser\AppData\Roaming
CommonProgramFiles=C:\Program Files\Common Files
CommonProgramFiles(x86)=C:\Program Files (x86)\Common Files
CommonProgramW6432=C:\Program Files\Common Files
COMPUTERNAME=JLEHMER650
...and so on...
```

echo

bash:

```
~ $ echo $HOME
/home/myuser
```

CMD.EXE:

```
C:\> echo %homepath%
\Users\myuser
```

Similar, but different

- `$variable` (bash) vs. `%variable%` (CMD.EXE)

Similar, but different

- `$variable` (`bash`) vs. `%variable%` (`CMD.EXE`)
- `bash` is case-sensitive, `CMD.EXE` is not

Product of your environment

Setting variables

```
~ $ FOO=myval /home/myuser/myscript
~ $ CURRDATE=`date`
~ $ echo $CURRDATE
Wed Oct 28 11:43:38 CDT 2015
```

A path! A path!

```
$ echo $PATH
/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin:...
```

Getting lazy

- Tab expansion

Getting lazy

- Tab expansion
- Command history

Step 2

File under “Directories”

ls - list files

```
~ $ ls
Audiobooks  Downloads  KindleGen  Podcasts  Templates
Desktop     Dropbox    Music      Public    Videos
Documents   FreeRDP    Pictures   Temp      VSCode-linux-x64
```

"Dotfiles"

```
~ $ ls -a
.                .dmrc          .gnome2_private  Pictures     .themes
..               Documents      .hplip           .pki         .thumbnails
.adobe           Downloads      .hugin           Podcasts     .thunderbird
.atom            .dropbox       .ICEauthority    .profile     Videos
Audiobooks       Dropbox        .icons           .ptbt1       .vscode
.bash_history    .dropbox-dist  KindleGen        Public       VSCode-linux-x64
.bash_logout     .face          .lastpass        .sbd         .wine
.cache           FreeRDP        .lesshst         .ssh         .Xauthority
.cinnamon        .gconf         .linuxmint       .swp         .xinputrc
.cmake           .gimp-2.8      .local           Temp         .xsession-errors
.config          .gitconfig     .macromedia      Templates
.dbus            .gksu.lock     .mozilla         .texmf-var
Desktop          .gnome2        Music            .TeXworks
```

List details

```
~ $ ls -l
total 92
drwxr-xr-x  2 myuser mygroup      4096 Sep  7 04:16 Desktop
drwxr-xr-x  2 myuser mygroup      4096 Oct 13 10:02 Documents
drwxr-xr-x  2 myuser mygroup      4096 Oct 14 09:45 Downloads
drwx------  8 myuser mygroup      4096 Oct 16 19:58 Dropbox
drwxr-xr-x 19 myuser mygroup      4096 Oct 12 09:48 FreeRDP
-rwxr-x---  1 myuser sambashare    883 Oct 12 11:34 installrdp
drwxr-xr-x  5 myuser mygroup      4096 Oct 16 10:47 LightTable
drwxr-xr-x  2 myuser mygroup      4096 Sep  7 04:16 Music
drwxr-xr-x  3 myuser mygroup     36864 Oct 12 17:29 Pictures
drwxr-xr-x  2 myuser mygroup      4096 Sep  7 04:16 Public
-rwxr-xr-x  1 myuser mygroup       816 Oct 15 18:00 rdp
...and so on...
```

Combining parameters

```
$ ls -al
total 344
drwxr-xr-x 40 myuser mygroup      4096 Oct 17 07:14 .
drwxr-xr-x  3 root   root         4096 Sep  7 04:09 ..
drwx------  3 myuser mygroup      4096 Sep  7 09:33 .adobe
drwxr-xr-x  5 myuser mygroup      4096 Oct 12 15:48 .atom
-rw-------  1 myuser mygroup      6428 Oct 17 06:11 .bash_history
-rw-r--r--  1 myuser mygroup       220 Sep  7 04:09 .bash_logout
drwx------ 18 myuser mygroup      4096 Oct 13 07:31 .cache
drwxr-xr-x  5 myuser mygroup      4096 Oct 16 19:57 .cinnamon
drwxr-xr-x  3 myuser mygroup      4096 Oct 12 09:45 .cmake
drwxr-xr-x 26 myuser mygroup      4096 Oct 15 10:23 .config
drwx------  3 myuser mygroup      4096 Sep  7 04:16 .dbus
...and so on...
```

Parameter types

- **Short**

Parameter types

- **Short**
  - `rm -rf *`

Parameter types

- **Short**
  - `rm -rf *`
  - Easier to type

Parameter types

- **Short**
  - `rm -rf *`
  - Easier to type
- **Long**

Parameter types

- **Short**
  - `rm -rf *`
  - Easier to type
- **Long**
  - `rm --recursive --force *`

Parameter types

- **Short**
  - `rm -rf *`
  - Easier to type
- **Long**
  - `rm --recursive --force *`
  - Easier to understand

cat - concatenate files

```
~ $ cat installrdp
#!/bin/bash
sudo apt-get -y install git
cd ~
git clone git://github.com/FreeRDP/FreeRDP.git
cd FreeRDP
sudo apt-get -y install build-essential git-core cmake libssl-dev \
  libx11-dev libxext-dev libxinerama-dev libxcursor-dev libxdamage-dev \
  libxv-dev libxkbfile-dev libasound2-dev libcups2-dev   libxml2 \
  libxml2-dev libxrandr-dev libgstreamer0.10-dev \
  libgstreamer-plugins-base0.10-dev libxi-dev \
  libgstreamer-plugins-base1.0-dev libavutil-dev libavcodec-dev \
  libcunit1-dev libdirectfb-dev xmlto doxygen libxtst-dev
cmake -DCMAKE_BUILD_TYPE=Debug -DWITH_SSE2=ON .
make
...and so on...
```

tail - show end of files

```
~ # tail dmesg
[    2.774931] loop: module loaded
[    3.349880] eth0: intr type 3, mode 0, 3 vectors allocated
[    3.351331] eth0: NIC Link is Up 10000 Mbps
[    3.422647] RPC: Registered named UNIX socket transport module.
[    3.422649] RPC: Registered udp transport module.
[    3.422650] RPC: Registered tcp transport module.
[    3.422651] RPC: Registered tcp NFSv4.1 backchannel transport module.
[    3.432437] FS-Cache: Loaded
[    3.443980] FS-Cache: Netfs 'nfs' registered for caching
[    3.449794] Installing knfsd (copyright (C) 1996 okir@monad.swb.de).
```

"Follow" a file

```
~ # tail -f dmesg
[    2.774931] loop: module loaded
[    3.349880] eth0: intr type 3, mode 0, 3 vectors allocated
[    3.351331] eth0: NIC Link is Up 10000 Mbps
[    3.422647] RPC: Registered named UNIX socket transport module.
[    3.422649] RPC: Registered udp transport module.
[    3.422650] RPC: Registered tcp transport module.
[    3.422651] RPC: Registered tcp NFSv4.1 backchannel transport module.
[    3.432437] FS-Cache: Loaded
[    3.443980] FS-Cache: Netfs 'nfs' registered for caching
[    3.449794] Installing knfsd (copyright (C) 1996 okir@monad.swb.de).
...new lines will appear here over time...
```

sort - sorting files

```
~ $ sort -k 3 -n * | tail -n 3
Combine motor   1500
Tractor tires   2000
Combine tires   2500
```

cp - copy files

```
~ $ cp diary.txt diary.bak
~ $ cp -r thisdir thatdir
~ $ cp --recursive thisdir thatdir
```

mv - move files

```
~ $ mv thismonth.log lastmonth.log
```

- mv is simple rename

mv - move files

```
~ $ mv thismonth.log lastmonth.log
```

- mv is simple rename
- rename offers more options

rm - remove files

```
~ $ rm desktop.ini
```

Danger, Will Robinson!

```
~ $ cd MyDissertation
~ $ ls
Citations.bak  Citations.doc  Dissertation.bak  Dissertation.doc  Notes.doc
~ $ rm * .bak
rm: cannot remove ‘.bak’: No such file or directory
~ $ ls
```

And all was null and void...

```
~ $ rm -rf *
```

touch - update file time

```
~ $ touch NewEmptyDissertation.doc
~ $ ls -l
total 0
-rw-rwxr--+ 1 myuser mygroup 0 Oct 19 14:12 NewEmptyDissertation.doc
```

Reset file time

```
~ $ touch -t 201412242300 NewEmptyDissertation.doc
~ $ ls -l
total 0
-rw-rwxr--+ 1 myuser mygroup 0 Dec 24  2014 NewEmptyDissertation.doc
```

mkdir - make directory

```
~ $ mkdir Bar
~ $ ls
Bar
```

cd - change directory

```
~ $ cd /etc
~ $ pwd
/etc
```

Absolute paths

- Always includes the root, /

Absolute paths

- Always includes the root, /
- `cd /etc`

Relative paths

- Starts from current directory, .

Relative paths

- Starts from current directory, .
- Parent directory is ..

Relative paths

- Starts from current directory, `.`
- Parent directory is `..`
- `cd child`

Relative paths

- Starts from current directory, `.`
- Parent directory is `..`
- `cd child`
- `cd ../sibling`

. and ..

- . - current directory

. and ..

- . - current directory

. and ..

- . - current directory
  - we will see why this is useful later

. and ..

- . - current directory
  - we will see why this is useful later
- .. - parent directory

. and ..

- . - current directory
  - we will see why this is useful later
- .. - parent directory

. and ..

- . - current directory
  - we will see why this is useful later
- .. - parent directory
  - useful to navigate “up and out”

May I?

- **3x3 “grid”** - who by what?

May I?

- **3x3 “grid”** - who by what?
- **UGO** - who?

May I?

- **3x3 “grid”** - who by what?
- **UGO** - who?
- **RWX** - what?

Who?

- **U** - primary *user* or “owner”

Who?

- **U** - primary *user* or “owner”
- **G** - primary *group*

Who?

- **U** - primary *user* or “owner”
- **G** - primary *group*
- **O** - other (everyone else)

What?

- **R** - *read* permission

What?

- **R** - *read* permission
- **W** - *write* permission

What?

- **R** - *read* permission
- **W** - *write* permission
- **X** - *execute* permission

What?

- **R** - *read* permission
- **W** - *write* permission
- **X** - *execute* permission

What?

- **R** - *read* permission
- **W** - *write* permission
- **X** - *execute* permission
  - “list directory” permission

ls -l, again

```
~ $ ls -l
total 92
drwxr-xr-x  2 myuser mygroup      4096 Sep  7 04:16 Desktop
drwxr-xr-x  2 myuser mygroup      4096 Oct 13 10:02 Documents
drwxr-xr-x  2 myuser mygroup      4096 Oct 14 09:45 Downloads
drwx------  8 myuser mygroup      4096 Oct 16 19:58 Dropbox
drwxr-xr-x 19 myuser mygroup      4096 Oct 12 09:48 FreeRDP
-rwxr-x---  1 myuser sambashare    883 Oct 12 11:34 installrdp
drwxr-xr-x  5 myuser mygroup      4096 Oct 16 10:47 LightTable
drwxr-xr-x  2 myuser mygroup      4096 Sep  7 04:16 Music
drwxr-xr-x  3 myuser mygroup     36864 Oct 12 17:29 Pictures
drwxr-xr-x  2 myuser mygroup      4096 Sep  7 04:16 Public
-rwxr-xr-x  1 myuser mygroup       816 Oct 15 18:00 rdp
...and so on...
```

```
-rwxr-xr-x
```

- - - file

`-rwxr-xr-x`

- `-` - file
- `rwx` - `myuser` can read, write and execute

`-rwxr-xr-x`

- `-` - file
- `rwx` - `myuser` can read, write and execute
- `r-x` - `mygroup` and anyone else can read and execute

`drwx------`

- `d` - directory

`drwx------`

- `d` - directory
- `rwx` - `myuser` can read, write and list contents

drwx------

- `d` - directory
- `rwx` - `myuser` can read, write and list contents
- `---` - no one else can do anything

Changing owners

- `chown myuser foo` - change owner of `foo` to user `myuser`

Changing owners

- `chown myuser foo` - change owner of `foo` to user `myuser`
- `chgrp mygroup bar` - change group for `bar` to `mygroup`

Changing access

- `chmod u+rw foo` - give primary owner read/write to `foo`

Changing access

- `chmod u+rw foo` - give primary owner read/write to `foo`
- `chmod o-x bar` - remove execute permission for “others” from `bar`

Will this be on the test?

If `rwx` were octal:

- `r` == 2^2 (4)

Will this be on the test?

If `rwx` were octal:

- `r` == 2^2 (4)
- `w` == 2^1 (2)

Will this be on the test?

If `rwx` were octal:

- `r` == 2^2 (4)
- `w` == 2^1 (2)
- `x` == 2^0 (1)

Will this be on the test?

If `rwx` were octal:

- `r` == 2^2 (4)
- `w` == 2^1 (2)
- `x` == 2^0 (1)
- `-` == 0

Olde Skool `chmod`

Then `chmod 754 foo`:

- `7` = `rwx` for user

Quicker than `chmod u=rwx,g=rx,o=r foo`

Olde Skool `chmod`

Then `chmod 754 foo`:

- `7` = `rwx` for user
- `5` = `r-x` for group

Quicker than `chmod u=rwx,g=rx,o=r foo`

Olde Skool `chmod`

Then `chmod 754 foo`:

- `7` = `rwx` for user
- `5` = `r-x` for group
- `4` = `r--` for other

Quicker than `chmod u=rwx,g=rx,o=r foo`

Why won't it run?

```
~ # echo "echo Hello world" > foo
~ # ls -l
total 4
-rw-r--r-- 1 root root 17 Oct 20 10:07 foo
~ # ./foo
-bash: ./foo: Permission denied
~ # chmod u+x foo
~ # ls -l
total 4
-rwxr--r-- 1 root root 17 Oct 20 10:07 foo
~ # ./foo
Hello world
```

Compressing files

```
~ $ zip -r foo foo
updating: foo/ (stored 0%)
  adding: foo/c (stored 0%)
  adding: foo/b (stored 0%)
  adding: foo/d/ (stored 0%)
  adding: foo/d/e (stored 0%)
  adding: foo/a (stored 0%)
~ $ ls -l foo.zip
-rw-r--r-- 1 myuser mygroup 854 Oct 24 15:56 foo.zip
~ $ unzip foo
Archive:  foo.zip
 extracting: foo/c
 extracting: foo/b
 extracting: foo/d/e
 extracting: foo/a
```

tarballs

```
~ $ tar cvzf foo.tgz foo
foo/
foo/c
foo/b
foo/d/
foo/d/e
foo/a
~ $ ls -l foo.tgz
-rw-r--r-- 1 myuser mygroup 191 Oct 24 16:19 foo.tgz
~ $ tar xvf foo.tgz
foo/
foo/c
foo/b
foo/d/
foo/d/e
foo/a
```

Soft links

```
~ $ ln -s d Dee
```

- Equivalent to a shortcut

Soft links

```
~ $ ln -s d Dee
```

- Equivalent to a shortcut
- Target can be directory or file

Soft links

```
~ $ ln -s d Dee
```

- Equivalent to a shortcut
- Target can be directory or file
- Target can be any file system

Soft links

```
~ $ ln -s d Dee
```

- Equivalent to a shortcut
- Target can be directory or file
- Target can be any file system
- Deleting link doesn't affect target

Soft links

```
~ $ ln -s d Dee
```

- Equivalent to a shortcut
- Target can be directory or file
- Target can be any file system
- Deleting link doesn't affect target
- Deleting target breaks link, doesn't remove it

Hard links

```
~ $ ln d Dee
```

- ‘Equivalent to NTFS junction point

Hard links

```
~ $ ln d Dee
```

- ‘Equivalent to NTFS junction point
- Target can be only files

Hard links

```
~ $ ln d Dee
```

- ‘Equivalent to NTFS junction point
- Target can be only files
- Target must be on same file system

Hard links

```
~ $ ln d Dee
```

- 'Equivalent to NTFS junction point
- Target can be only files
- Target must be on same file system
- File not deleted until ***ALL*** hard links deleted

File systems

```
~ $ df
Filesystem                1K-blocks     Used Available Use% Mounted on
/dev/mapper/mint--vg-root 118647068 28847464  83749608  26% /
none                              4        0         4   0% /sys/fs/cgroup
udev                        1965068        4   1965064   1% /dev
tmpfs                        396216     1568    394648   1% /run
none                           5120        0      5120   0% /run/lock
none                        1981068      840   1980228   1% /run/shm
none                         102400       24    102376   1% /run/user
/dev/sda1                    240972    50153    178378  22% /boot
```

What's the diff?

```
~ $ diff orig.conf new.conf
1c1
< FOO=1
---
> FOO=2
7d6
< BAR=Xyzzy
```

Step 3

Finding Meaning

find in 3 steps

1. Starting at location X

find in 3 steps

1. Starting at location X
2. Recursively find all entries that match

`find` in 3 steps

1. Starting at location X
2. Recursively find all entries that match
3. Do something for each match

Example

```
~ $ find //myserver/myshare/logs/000[4-9] -name \*.dat -newer logchecker.csv \
    -exec /home/myuser/Sandbox/FileCheckers/logchecker \{\} \;
```

1. Starting at `//myserver/myshare/logs/000[4-9]`

Example

```
~ $ find //myserver/myshare/logs/000[4-9] -name \*.dat -newer logchecker.csv \
    -exec /home/myuser/Sandbox/FileCheckers/logchecker \{\} \;
```

1. Starting at `//myserver/myshare/logs/000[4-9]`
2. Find all files that end in `.dat`

Example

```
~ $ find //myserver/myshare/logs/000[4-9] -name \*.dat -newer logchecker.csv \
    -exec /home/myuser/Sandbox/FileCheckers/logchecker \{\} \;
```

1. Starting at `//myserver/myshare/logs/000[4-9]`
2. Find all files that end in `.dat`
3. That are also newer than `logchecker.csv`

Example

```
~ $ find //myserver/myshare/logs/000[4-9] -name \*.dat -newer logchecker.csv \
    -exec /home/myuser/Sandbox/FileCheckers/logchecker \{\} \;
```

1. Starting at `//myserver/myshare/logs/000[4-9]`
2. Find all files that end in `.dat`
3. That are also newer than `logchecker.csv`
4. Execute `logchecker`, passing in path to file

What's with the backslashes?

```
~ $ find //myserver/myshare/logs/000[4-9] -name \*.dat -newer logchecker.csv \
    -exec /home/myuser/Sandbox/FileCheckers/logchecker \{\} \;
```

Backslashes prevent "shell expansion"

Useful `find` tests

- **`-executable`** - the file is executable or the directory is searchable
- **`-group <gname>`** - file belongs to group *gname*
- **`-iname <pattern>`** - case-insensitive name search
- **`-name <pattern>`** - case-sensitive name search
- **`-newer <file>`** - newer than *file*
- **`-size <n>`** - file uses *n* units of space
  - various measures like 512-byte blocks (`b`) through gigabytes (`G`).
- **`-type <c>`** - file is of type *c*
  - two most common - `d` (directory) or `f` (file).
- **`-user <uname>`** - file is owned by *uname*.

Useful `find` actions

- **`-delete`** - deletes any files matched so far
  - Actions are also tests (predicates)
  - Don't put this first!
- **`-exec` and `-execdir`** - executes a command or script
- **`-print`** - prints the full path of the found file or directory
- **`-printf`** - prints a formatted string, useful for reports

find -printf

```
~ $ find . -type f -printf "%p\n%u\n%TY-%Tm-%TdT%TT\n\n"
./a
myuser
2015-10-21T11:02:51.7014527000
```

Step 4

Grokking grep

grep

Finds files based on their ***content***

```
~ $ touch a b c
~ $ echo This sequence of characters is called a \"string\". > d
~ $ cat d
This sequence of characters is called a "string".
~ $ ls
a  b  c  d
~ $ grep is *
d:This sequence of characters is called a "string".
```

Famous quote

"Some people, when confronted with a problem, think 'I know, I'll use regular expressions.' Now they have two problems." - *Jamie Zawinski*

Regular expressions

A “regex” is a pattern for matching strings

- `dir *.txt` - match zero or more characters

Regular expressions

A “regex” is a pattern for matching strings

- `dir *.txt` - match zero or more characters
- `find //myserver/myshare/logs/000[4-9] -print`

Regular expressions

A “regex” is a pattern for matching strings

- `dir *.txt` - match zero or more characters
- `find //myserver/myshare/logs/000[4-9] -print`
- `grep is *` - find “is” in all files in current directory

Helpful regexes

- **`one|other`** - find `one` pattern or the `other`.
- **`^`** - pattern for the beginning of a line.
- **`$`** - pattern for the end of a line.
- **`?`** - match exactly one character.
- **`*`** - match zero or more characters.
- **`+`** - match one or more characters.
- **`[A-Z]`** - match any character in a range (such as in this case any uppercase Latin alphabetic character).
- **`[n|y]`** - match one character or another (such as `n` or `y` here).

Why 2 problems?

What does the following do?

```
(?bhttp://[-A-Za-z0-9+&@#/%?=~_()|!:,.;]*[-A-Za-z0-9+&@f
```

- Checks a Web URL for validity

Why 2 problems?

What does the following do?

```
(?bhttp://[-A-Za-z0-9+&@#/%?=~_()|!:,.;]*[-A-Za-z0-9+&@f
```

- Checks a Web URL for validity
- Are you going to remember that?

Why 2 problems?

What does the following do?

```
(?bhttp://[-A-Za-z0-9+&@#/%?=~_()|!:,.;]*[-A-Za-z0-9+&@f
```

- Checks a Web URL for validity
- Are you going to remember that?
- Are you going to be able to figure it out?

Step 5

A Series of Pipes

stdin, stdout, stderr

- **stdin** - input, file descriptor 0
- **stdout** - output, file descriptor 1
- **stderr** - “error” output, file descriptor 2
- All three use the console in interactive mode by default

Output redirection

```
~ $ echo Hello, world > hw
~ $ ls -l
total 1
-rw-rwxr--+ 1 myuser mygroup 13 Oct 22 10:40 hw
~ $ cat hw
Hello, world
```

Input redirection

```
~ $ cat < hw
Hello, world
```

Equivalent to:

```
~ $ cat hw
Hello, world
```

Errors

```
~ $ find . -exec cat \{\} \;
cat: .: Is a directory
This is a
This is b
This is c
cat: ./d: Is a directory
This is e
```

“Is a directory” is an error message

Error redirection

```
~ $ find . -exec cat \{\} \; 2>/tmp/finderrors.log
This is a
This is b
This is c
This is e
~ $ cat /tmp/finderrors.log
cat: .: Is a directory
cat: ./d: Is a directory
```

This is where those “file descriptors” come in

Logging ALL output to file

```
~ $ find . -exec cat \{\} \; >/tmp/find.log 2>&1
~ $ cat /tmp/find.log
cat: .: Is a directory
This is a
This is b
This is c
cat: ./d: Is a directory
This is e
```

The `2>&1` trick works in `CMD.EXE`, too!

Rewrite vs. append

```
~ $ find . -exec cat \{\} \; >/tmp/find.log
```

vs.

```
~ $ find . -exec cat \{\} \; >>/tmp/find.log
```

Everyone line up

```
~ $ cat *.txt | tr '\\' '/' | while read line ; do ./mycmd "$line" ; done
```

1. `cat` echos all `.txt` files to `stdout`, piped to...
2. `tr` translates any backslash characters before sending it into...
3. A `while` loop that reads each line into a variable called `$line` and then calls...
4. Some custom script or program called `./mycmd` passing in the value of each `$line`.

Two places at once

```
~ $ find . -name error.log | tee > errorlogs.txt
```

- Log output to `error.log`

Two places at once

```
~ $ find . -name error.log | tee > errorlogs.txt
```

- Log output to `error.log`
- Monitor its progress on the console at the same time

Step 6

vi

Why vi?

- Use nano if available

Why vi?

- Use nano if available
- But vi is (almost) ***always*** there

Why vi?

- Use nano if available
- But vi is (almost) ***always*** there
- Good to know the basics “just in case”

vi strangeness

vi is a “modal” editor

- In “command” mode to start

vi strangeness

vi is a “modal” editor

- In “command” mode to start
- Need to go into “insert mode” to insert new text

vi strangeness

vi is a “modal” editor

- In “command” mode to start
- Need to go into “insert mode” to insert new text
- Confusing to almost everyone at first

vi commands

- `d` - “delete”

vi commands

- `d` - “delete”
- `b` - “jump ‘back’ one ‘word’ ”

vi commands

- `d` - “delete”
- `b` - “jump ‘back’ one ‘word’ ”
- `i` - enter “insert” mode

vi commands

- `d` - “delete”
- `b` - “jump ‘back’ one ‘word’ ”
- `i` - enter “insert” mode
- `ESC` - exit “insert” mode

vi commands

- `d` - “delete”
- `b` - “jump ‘back’ one ‘word’ ”
- `i` - enter “insert” mode
- `ESC` - exit “insert” mode
- `dw` - “delete ‘word’ ”

vi commands

- `d` - “delete”
- `b` - “jump ‘back’ one ‘word’ ”
- `i` - enter “insert” mode
- `ESC` - exit “insert” mode
- `dw` - “delete ‘word’ ”
- `3dw` - “delete 3 ‘words’ ”

Get me out of here!

- `:q!` - exit without saving

Get me out of here!

- `:q!` - exit without saving
- `u` - “undo” command

Get me out of here!

- `:q!` - exit without saving
- `u` - “undo” command
- `3u` - “undo” last three changes

Get me out of here!

- `:q!` - exit without saving
- `u` - “undo” command
- `3u` - “undo” last three changes
- `view` - “read-only” version of `vi`

Navigating

- Arrow and page keys ***tend*** to work right

Navigating

- Arrow and page keys ***tend*** to work right
  - Except in insert mode!

Navigating

- Arrow and page keys ***tend*** to work right
  - Except in insert mode!
- `0` - jump to beginning of line

Navigating

- Arrow and page keys ***tend*** to work right
  - Except in insert mode!
- `0` - jump to beginning of line
- `$` - jump to end of line

Navigating

- Arrow and page keys ***tend*** to work right
  - Except in insert mode!
- `0` - jump to beginning of line
- `$` - jump to end of line
- `w` - jump forward a “word”

Navigating

- Arrow and page keys ***tend*** to work right
  - Except in insert mode!
- `0` - jump to beginning of line
- `$` - jump to end of line
- `w` - jump forward a “word”
- `b` - jump backward a “word”

Navigating

- Arrow and page keys ***tend*** to work right
  - Except in insert mode!
- `0` - jump to beginning of line
- `$` - jump to end of line
- `w` - jump forward a “word”
- `b` - jump backward a “word”
- `:0` - jump to beginning of file

Navigating

- Arrow and page keys ***tend*** to work right
  - Except in insert mode!
- `0` - jump to beginning of line
- `$` - jump to end of line
- `w` - jump forward a “word”
- `b` - jump backward a “word”
- `:0` - jump to beginning of file
- `G` - jump to end of file

I've been searching

- `/foo` - find "foo" from cursor forward

I've been searching

- `/foo` - find "foo" from cursor forward
- `?foo` - find "foo" from cursor backward

I've been searching

- `/foo` - find "foo" from cursor forward
- `?foo` - find "foo" from cursor backward
- `n` - find next instance of last search

I've been searching

- `/foo` - find "foo" from cursor forward
- `?foo` - find "foo" from cursor backward
- `n` - find next instance of last search
- `p` - find previous instance of last search

Insertion

All of the following enter “insert mode”:

- `i` - at cursor

Insertion

All of the following enter “insert mode”:

- `i` - at cursor
- `I` - at beginning of line

Insertion

All of the following enter “insert mode”:

- `i` - at cursor
- `I` - at beginning of line
- `A` - “append” at end of line

Insertion

All of the following enter “insert mode”:

- `i` - at cursor
- `I` - at beginning of line
- `A` - “append” at end of line
- `o` - insert line below (lowercase) current line

Insertion

All of the following enter “insert mode”:

- `i` - at cursor
- `I` - at beginning of line
- `A` - “append” at end of line
- `o` - insert line below (lowercase) current line
- `O` - insert line above (uppercase) current line

Insertion

All of the following enter “insert mode”:

- `i` - at cursor
- `I` - at beginning of line
- `A` - “append” at end of line
- `o` - insert line below (lowercase) current line
- `O` - insert line above (uppercase) current line
- `ESC` - exit insert mode

Ctrl-X

- `d` - “delete” is same as “cut”

Ctrl-X

- `d` - “delete” is same as “cut”
- `dd` - delete/cut current line

Ctrl-X

- `d` - “delete” is same as “cut”
- `dd` - delete/cut current line
- `3dw` - delete/cut three “words”

Ctrl-C

- y - “yank” is the same as “copy”

Ctrl-C

- `y` - “yank” is the same as “copy”
- `yy` - yank/copy current line

Ctrl-C

- `y` - “yank” is the same as “copy”
- `yy` - yank/copy current line
- `3yw` - yank/copy three “words”

Ctrl-V

- `p` - paste contents of buffer at cursor

Ctrl-V

- `p` - paste contents of buffer at cursor
- `P` - paste contents of buffer above (uppercase) current line

Ctrl-V

- `p` - paste contents of buffer at cursor
- `P` - paste contents of buffer above (uppercase) current line
- `u` - remember “undo” when you need it!

“X” marks the spot

You can constrain the lines you want to affect by a command by “marking” a “range”:

1. Mark line with `m` command followed by a character

“X” marks the spot

You can constrain the lines you want to affect by a command by “marking” a “range”:

1. Mark line with `m` command followed by a character
2. Mark another line with `m` command, but ***with a different label character***

"X" marks the spot

You can constrain the lines you want to affect by a command by "marking" a "range":

1. Mark line with `m` command followed by a character
2. Mark another line with `m` command, but ***with a different label character***
3. Use the `'` character to reference a label

“X” marks the spot

You can constrain the lines you want to affect by a command by “marking” a “range”:

1. Mark line with `m` command followed by a character
2. Mark another line with `m` command, but ***with a different label character***
3. Use the `'` character to reference a label
4. `:'m,'ns/This/That/`

Invoking external commands

- `:1,$!sort`

Invoking external commands

- `:1,$!sort`
- `:'m,'n!sort`

Step 7

The Whole Wide World

Network commands

- `ping yahoo.com` - works like `ping -t` in `CMD.EXE`

Network commands

- `ping yahoo.com` - works like `ping -t` in `CMD.EXE`
- `traceroute yahoo.com`

Network commands

- `ping yahoo.com` - works like `ping -t` in `CMD.EXE`
- `traceroute yahoo.com`
- `dig yahoo.com`

Network commands

- `ping yahoo.com` - works like `ping -t` in `CMD.EXE`
- `traceroute yahoo.com`
- `dig yahoo.com`
- `whois yahoo.com`

sudo make me a sandwich

- Many commands require “super-user” privileges

sudo make me a sandwich

- Many commands require “super-user” privileges
- One way to get it is to log-in as “root”

sudo make me a sandwich

- Many commands require “super-user” privileges
- One way to get it is to log-in as “root”
  - Not recommended in general

sudo make me a sandwich

- Many commands require “super-user” privileges
- One way to get it is to log-in as “root”
  - Not recommended in general
- sudo - allows a pre-authorized user to run privileged commands

sudo make me a sandwich

- Many commands require “super-user” privileges
- One way to get it is to log-in as “root”
  - Not recommended in general
- `sudo` - allows a pre-authorized user to run privileged commands
- `sudo apt-get update`

Surfin' the command prompt

- `lynx` - command-line browser

Surfin' the command prompt

- `lynx` - command-line browser
- `wget` - get files over HTTP, FTP, etc.

Surfin' the command prompt

- `lynx` - command-line browser
- `wget` - get files over HTTP, FTP, etc.
- `curl` - alternative to `wget`

Sending mail

```
~ $ email --blank-mail --subject "Possibly corrupted files found..." \
  --smtp-server smtp --attach badfiles.csv --from-name NoReply \
  --from-addr noreply@mycorp.com alert@mycorp.com
```

Logging in elsewhere

- `ssh` - secure shell

Logging in elsewhere

- `ssh` - secure shell
  - `ssh myuser@remoteserver`

Logging in elsewhere

- `ssh` - secure shell
  - `ssh myuser@remoteserver`
- `scp` - secure copy (over `ssh`)

Logging in elsewhere

- `ssh` - secure shell
  - `ssh myuser@remoteserver`
- `scp` - secure copy (over `ssh`)
  - `scp -r myfiles/* myuser@remoteserver:/home/myuser/myfiles/.`

Network configuration

- `ifconfig` - display current network settings

Network configuration

- `ifconfig` - display current network settings
- `cat /etc/resolv.conf` - display current DNS settings

Network configuration

- `ifconfig` - display current network settings
- `cat /etc/resolv.conf` - display current DNS settings
- `cat /etc/hosts` - display local network aliases

Step 8

The Man Behind the Curtain

View running processes

- `ps` - shows running processes

View running processes

- `ps` - shows running processes
  - `ps -A` - shows ***all*** running processes

View running processes

- `ps` - shows running processes
  - `ps -A` - shows ***all*** running processes
  - `ps -A | grep bash` - show all running bash processes

View running processes

- `ps` - shows running processes
  - `ps -A` - shows ***all*** running processes
  - `ps -A | grep bash` - show all running bash processes
- `top` - show “top” processes by CPU, memory and other criteria

/proc file system

```
~ # cat /proc/cpuinfo
processor       : 0
vendor_id       : GenuineIntel
cpu family      : 6
model           : 37
model name      : Intel(R) Xeon(R) CPU           X5690  @ 3.47GHz
stepping        : 1
microcode       : 0x15
cpu MHz         : 3458.000
cache size      : 12288 KB
fpu             : yes
fpu_exception   : yes
cpuid level     : 11
...and so on...
```

- Many live system metrics, presented as “files”

Sawing logs

```
~ # ls /var/log
alternatives.log       auth.log.2.gz    debug       dmesg.4.gz     kern....
alternatives.log.1     auth.log.3.gz    debug.1     dpkg.log       kern....
alternatives.log.2.gz  auth.log.4.gz    debug.2.gz  dpkg.log.1     kern....
alternatives.log.3.gz  btmp             debug.3.gz  dpkg.log.2.gz  kern....
apache2                btmp.1           debug.4.gz  dpkg.log.3.gz  kern....
apt                    daemon.log       dmesg       dpkg.log.4.gz  lastlog
aptitude               daemon.log.1     dmesg.0     exim4          lpr.log
aptitude.1.gz          daemon.log.2.gz  dmesg.1.gz  faillog        mail.err
auth.log               daemon.log.3.gz  dmesg.2.gz  fsck           mail....
auth.log.1             daemon.log.4.gz  dmesg.3.gz  installer      mail....
```

It's all temporary

- `/tmp` - standard location for temp files

It’s all temporary

- `/tmp` - standard location for temp files
- Cleared at reboot

Step 9

How Do You Know What You Don't Know, man?

man pages

- man - “manual” command

man pages

- man - “manual” command
  - man ls - help on ls

man pages

- `man` - “manual” command
  - `man ls` - help on `ls`
- `vi` style navigation and searching work

man pages

- man - “manual” command
  - man ls - help on ls
- vi style navigation and searching work
- Divided into “sections”

man pages

- man - “manual” command
  - man ls - help on ls
- vi style navigation and searching work
- Divided into “sections”
  - Section 1 - user commands (default)

man pages

- man - “manual” command
  - man ls - help on ls
- vi style navigation and searching work
- Divided into “sections”
  - Section 1 - user commands (default)
  - Section 5 - system files

man pages

- man - “manual” command
  - man ls - help on ls
- vi style navigation and searching work
- Divided into “sections”
  - Section 1 - user commands (default)
  - Section 5 - system files
  - Section 8 - system commands

man pages

- `man` - “manual” command
  - `man ls` - help on `ls`
- `vi` style navigation and searching work
- Divided into “sections”
  - Section 1 - user commands (default)
  - Section 5 - system files
  - Section 8 - system commands
  - `man passwd` - help on `passwd` command

man pages

- `man` - “manual” command
  - `man ls` - help on `ls`
- `vi` style navigation and searching work
- Divided into “sections”
  - Section 1 - user commands (default)
  - Section 5 - system files
  - Section 8 - system commands
  - `man passwd` - help on `passwd` command
  - `man 5 passwd` - info on `/etc/passwd` file

man pages

- `man` - "manual" command
  - `man ls` - help on `ls`
- `vi` style navigation and searching work
- Divided into "sections"
  - Section 1 - user commands (default)
  - Section 5 - system files
  - Section 8 - system commands
  - `man passwd` - help on `passwd` command
  - `man 5 passwd` - info on `/etc/passwd` file
- `info` - like `man` for some GNU programs

man pages

- `man` - "manual" command
  - `man ls` - help on `ls`
- `vi` style navigation and searching work
- Divided into "sections"
  - Section 1 - user commands (default)
  - Section 5 - system files
  - Section 8 - system commands
  - `man passwd` - help on `passwd` command
  - `man 5 passwd` - info on `/etc/passwd` file
- `info` - like `man` for some GNU programs
- `apropos` - search man page titles for a string

Books and stuff

Following are some of the better resources on the web:

- Linux Documentation Project

Books and stuff

Following are some of the better resources on the web:

- Linux Documentation Project
- Arch Linux wiki

Books and stuff

Following are some of the better resources on the web:

- Linux Documentation Project
- Arch Linux wiki
- Debian Administrator's Handbook and reference

Books and stuff

Following are some of the better resources on the web:

- Linux Documentation Project
- Arch Linux wiki
- Debian Administrator's Handbook and reference
- linux.die.net - online `man` pages

Books and stuff

Following are some of the better resources on the web:

- Linux Documentation Project
- Arch Linux wiki
- Debian Administrator's Handbook and reference
- linux.die.net - online `man` pages
- Stackoverflow

Books and stuff

Following are some of the better resources on the web:

- Linux Documentation Project
- Arch Linux wiki
- Debian Administrator's Handbook and reference
- linux.die.net - online `man` pages
- Stackoverflow
- Docs for individual packages at maintainer's site (Samba, etc.)

Step 10

And So On...

/etc

```
~ # ls -l /etc
total 844
drwxr-xr-x 3 root root    4096 Feb 25  2015 acpi
-rw-r--r-- 1 root root    2981 Apr 23  2014 adduser.conf
-rw-r--r-- 1 root root      45 Jul  9 08:46 adjtime
-rw-r--r-- 2 root root     621 May 22  2014 aliases
-rw-r--r-- 1 root root   12288 May 22  2014 aliases.db
drwxr-xr-x 2 root root   20480 Feb 25  2015 alternatives
-rw-r--r-- 1 root root    4185 Dec 28  2011 analog.cfg
drwxr-xr-x 7 root root    4096 Feb 25  2015 apache2
drwxr-xr-x 6 root root    4096 Feb 25  2015 apt
-rw-r----- 1 root daemon   144 Jun  9  2012 at.deny
-rw-r--r-- 1 root root    1895 Dec 29  2012 bash.bashrc
...and so on...
```

Most system configuration information is here

Some helpful /etc files

- **fstab** - file systems currently mounted
- **group** - security groups
- **hosts** - network aliases
- **init.d** - startup and shutdown scripts for “services.”
- **mtab** - list of current “mounts.”
- **passwd** - “shadow” file containing all the user accounts
- **resolv.conf** - DNS settings.
- **samba** - file sharing settings for CIFS-type shares

May I be of service?

- “Services” (or “daemons”) are long-running processes

May I be of service?

- “Services” (or “daemons”) are long-running processes
- Typically controlled via `/etc/init.d` scripts

May I be of service?

- “Services” (or “daemons”) are long-running processes
- Typically controlled via `/etc/init.d` scripts
  - `/etc/init.d/samba stop`

May I be of service?

- “Services” (or “daemons”) are long-running processes
- Typically controlled via `/etc/init.d` scripts
  - `/etc/init.d/samba stop`
  - `/etc/init.d/samba start`

May I be of service?

- “Services” (or “daemons”) are long-running processes
- Typically controlled via `/etc/init.d` scripts
  - `/etc/init.d/samba stop`
  - `/etc/init.d/samba start`
  - `/etc/init.d/samba restart` - the above two commands combined

Package management

- Most Linux distros have a package manager

Package management

- Most Linux distros have a package manager
  - `dpkg` and `apt-get` on Debian flavors

Package management

- Most Linux distros have a package manager
  - `dpkg` and `apt-get` on Debian flavors
  - `rpm` on Fedora flavors

Package management

- Most Linux distros have a package manager
  - `dpkg` and `apt-get` on Debian flavors
  - `rpm` on Fedora flavors
- Package managers are like “Add/Remove Programs” - can install, update or delete applications

Package management

- Most Linux distros have a package manager
  - `dpkg` and `apt-get` on Debian flavors
  - `rpm` on Fedora flavors
- Package managers are like “Add/Remove Programs” - can install, update or delete applications
- Package managers are like “Windows Update” - can update and upgrade the OS

Package management on Debian

(...and Ubuntu, Mint and others)

- `apt-get update` - pull down latest package definitions

Package management on Debian

(...and Ubuntu, Mint and others)

- `apt-get update` - pull down latest package definitions
- `apt-get upgrade` - upgrade all packages

Package management on Debian

(...and Ubuntu, Mint and others)

- `apt-get update` - pull down latest package definitions
- `apt-get upgrade` - upgrade all packages
- `apt-get install curl` - install package “curl”

Package management on Debian

(...and Ubuntu, Mint and others)

- `apt-get update` - pull down latest package definitions
- `apt-get upgrade` - upgrade all packages
- `apt-get install curl` - install package “curl”
- `apt-get` is an admin command and usually requires `sudo`

Package management on Debian

(...and Ubuntu, Mint and others)

- `apt-get update` - pull down latest package definitions
- `apt-get upgrade` - upgrade all packages
- `apt-get install curl` - install package “curl”
- `apt-get` is an admin command and usually requires `sudo`
- `dpkg -i somesoftware.deb` - install a package file downloaded from the web

Which `which` is which?

- `which curl` - show which `curl` will execute

Which `which` is which?

- `which curl` - show which `curl` will execute
- `locate curl` - show all files on system with “curl” in the path

Which `which` is which?

- `which curl` - show which `curl` will execute
- `locate curl` - show all files on system with “curl” in the path
- `./curl` - regardless of `$PATH`, execute `curl` that is in current directory

Over and over and over

- `cron` - service that runs “cron jobs” (scheduled task)

Over and over and over

- `cron` - service that runs “cron jobs” (scheduled task)
- `crontab` - show cron jobs for current user

Over and over and over

- `cron` - service that runs “cron jobs” (scheduled task)
- `crontab` - show cron jobs for current user
- `crontab -e` - edit cron jobs for current user

Over and over and over

- `cron` - service that runs “cron jobs” (scheduled task)
- `crontab` - show cron jobs for current user
- `crontab -e` - edit cron jobs for current user
  - `sudo crontab -e -u otheruser` - edit cron jobs for another user

Start me up

- `reboot` - reboot the system (typically requires `sudo`)

Start me up

- `reboot` - reboot the system (typically requires `sudo`)
- `shutdown -h now` - shut down system now

Turn on your signals

- `kill` - send a signal to a process

Turn on your signals

- `kill` - send a signal to a process
  - Most “signals” allow process to cleanup

Turn on your signals

- `kill` - send a signal to a process
  - Most “signals” allow process to cleanup
  - `kill -9` - does ***NOT*** allow process to cleanup, may corrupt data

Exit smiling

- `echo $?` - show “return code” or exit code for last command or program

Exit smiling

- `echo $?` - show “return code” or exit code for last command or program
- `a && b` - execute `a` and if it is successful execute `b`

Exit smiling

- `echo $?` - show “return code” or exit code for last command or program
- `a && b` - execute `a` and if it is successful execute `b`
- `a || b` - execute `a` and then execute `b` regardless of `a`