```
# vim: set filetype=text:

* Pipe stdin/out to/from vim - vipe (moreutils)

* Allow some_user to run X programs:
        xhost +si:localuser:some_user

* Show all files in a directory with fbvsp:
        for f in img/*; do [ -f "$f" ] && { clear; fbvsp "$f"; \
        printf "%s\nPress ENTER..." "$f"; read; }; done; clear

* Next cron date+time (needs package cronie for cronnext):
        date -Ins -d "@$(cronnext | cut -d' ' -f2)"

* Just list UUID (needs root, otherwise might output nothing):
        lsblk -no UUID /dev/sda1

* csv2tsv (opinionated - delim=',',quote='"', GNU sed):
        sed -e's/\("\?\),/\1\t/g' -e's/"\?\([^"]*\)"\?/\1/g' \
                < file.csv > file.tsv

* tsv2csv (opinionated - delim=',',quote='"', GNU sed):
        sed -e's/\t/","/g' -e's/^/"/' -e's/$/"/' < file.tsv > file.csv

* is the process running? human-readable:
        ps -ef | grep [p]rogram_name    # [] to skip including grep itself
  machine-readable:
        pidof program_name >/dev/null 2>&1 && do_something

* create a PDF from a text file (needs cups, cups-filters):
        /usr/lib/cups/filter/texttopdf 1 1 1 1 1 file.txt > file.pdf
  using the user's font (CUPS_DATADIR has charsets subdirectory):
        CUPS_DATADIR=~/.config/cups CHARSET=utf-8 \
                /usr/lib/cups/filter/texttopdf 1 1 1 1 1 file.txt > file.pdf
  optimize the above PDF and make the text searchable:
        pdftocairo file.pdf -pdf file-opt.pdf

* list X input devices:
        xinput list

* Necessary changes in config.mk to build 9base statically:
        CFLAGS += -static -fcommon
        LDFLAGS += -static -zmuldefs
  Nice to have: uncomment "@strip" line in "all:"

* Restore tmux client after suspend-client (C-b C-z):
        kill -CONT $pidof_tmux_client

* Set xterm (or compatible) title:
        printf '\e]N;new_title\b'
  where N is: 0 = set icon + title, 1 = set icon, 2 = set title

* mpv as image viewer:
        mpv --keep-open=always /image/directory
  < = Previous image, > = Next image

* List terminal capabilities of terminal type $some_term (for tput):
        infocmp $some_term

* `basename $file` in POSIX shell (note: shell parameter substitution is slow):
        basename=${file%/}; echo ${basename##*/}
  `basename -s .ext $file`:
        basename=${file%/}; basename=${basename%.ext}; echo ${basename##*/}

* `dirname $file` in POSIX shell:
        case $file in */*) echo ${file%/*};; *) echo .; esac
```

```
* Get the name of the first X.Org monitor:
        xrandr --listmonitors | sed '/^[^ ]/d;s/.*  \(\S\+\)$/\1/;1q'

* Set window of an X.Org application "on top":
  - With $part_of_title inside title (case-insensitive):
        wmctrl -r $part_of_title -b add,above
  - With $window_id (find out with `wmctrl -l`):
        wmctrl -i -r $window_id -b add,above

* Scroll to the end of file with less (useful for logs):
        less +G /some/file
  Scroll to the end of file and wait for data (like `tail -f`):
        less +F /some/file
  Jump to line 123 immediately when starting vim:
        vim /some/file +123

* List in a tree:
  - Files and directories:
        tree
  - Mounts:
        findmnt
  - Processes:
        pstree
  - Packages in Arch-based distributions:
        pactree

* Restrict output to lines with "foo", replacing "foo" with "bar", using only
  POSIX sed:
  - Print all occurences:
        sed '/foo/s/foo/bar/;t;d'
    alternative:
        sed -n '/foo/s/foo/bar/p'
  - Print only the first such line:
        sed '/foo/s/foo/bar/;te;d;:e q'
    alternative:
        sed -n '/foo/{s/foo/bar/p;q}'

* Click a window to get its command with arguments (ps from procps-ng):
        ps --no-headers -o '%a' $(xprop | sed -n \
        '/_NET_WM_PID/{s/_NET_WM_PID\S\+\s\+=\s\+\([0-9]\+\)$/\1/p;q}')

* Print information about installed packages in an Arch-based distribution:
        pacreport
  Check for files unowned by packages:
        pacreport --unowned-files

* If `$HOME/.profile` is not sourced by Bash, check for the existence of
  `$HOME/.bash_profile`, as it gets sourced first/instead of `$HOME/.profile` if
  present. Possible solutions:
  - Move/rename `$HOME/.bash_profile`, or
  - Source `$HOME/.profile` from `$HOME/.bash_profile`

* Directory holding DKMS modules, logs and links to sources: /var/lib/dkms

* If `tcc -static` on musl complains about missing `abort`, use
        export LDFLAGS='-L/usr/lib/tcc -ltcc1'
  Sometimes LIBS needs to be set instead, as those parameters need to come at
  the end of the command (for example, mksh requires LIBS instead of LDFLAGS).
```

```
* Mount a QEMU qcow2 disk:
        modprobe nbd max_part=8
        qemu-nbd -c /dev/nbd0 /path/to/image.qcow2
        mount /dev/nbd0p1 /mnt
  Unmount it:
        umount /mnt
        qemu-nbd -d /dev/nbd0
        rmmod nbd

* Make tcc work for static linking on a glibc-based system (use separately
  installed musl-libc, for example precompiled from https://musl.cc): $s - dir
  where musl is installed (ex. /cross/x86_64-linux-musl)
        ./configure --prefix=/usr --sysroot=$s --config-musl \
                --sysincludepaths=$s/include \
                --crtprefix=$s/lib \
                --libpaths=$s/lib
        # sysincludepaths - colon-separated list of directories with files like
        #                   stdio.h, stdlib.h etc
        # crtprefix       - directory with crt[1in].o
        # libpaths        - colon-separated list of directories with libc.a,
        #                   ld-*.so.1 -> libc.so etc
  inside the tcc source directory, then make and make install. Check with:
        tcc -vv
  Related:
        https://lists.nongnu.org/archive/html/tinycc-devel/2010-04/msg00082.html
  (tcc still gives similar errors about "undefined symbols" when -static is
  used on glibc-based systems).

* For GCC, multiple directories in startfile_prefix_spec search path are
  separated with a space.

* When booting from a USB flash medium, if it isn't recognized and thus booting
  stops with a kernel panic, try adding
        rootdelay=5
  to the kernel command line. For example:
        root=/dev/sda2 rootdelay=5

* If neomutt setup using Luke Smith's mutt-wizard refuses to open attachments
  using programs which xdg-open is set to open, edit
        $PREFIX/lib/mutt-wizard/openfile
  Probably your system doesn't support `setsid -f`.

* If the output from date(1) differs for root and ordinary users, besides the
  obvious (environment variable `$TZ`), check the permissions on
  `/etc/localtime` and `/etc/zoneinfo/*` files and directories. They need to be
  "R"eadable for "O"thers (`o+r`).

* In LibreOffice Math, `func` directive formats text as upright (regular,
  non-italic...), useful for units eg.
        30 func min, 2.5 func kg

* Interactively edit environment variables in `$EDITOR` (needs vipe from
  moreutils):
        eval $(printenv | vipe | while read -r line; do
                ident=$(echo "$line" | cut -d= -f1);
                rest=$(echo "$line" | cut -d= -f2);
                printf "export %s=\"%s\"\n" "$ident" "$rest"; done)
```

```
* When configuring tcc, be **very** careful of what is assigned to `tccdir`:
        $ ./configure --tccdir=$tccdir
  This directory will be `rm -frv`ed on uninstall! (Found out the hard way -
        $ ./configure --tccdir=/lib
  with later
        # make uninstall
  nuked my /lib.)

* Current ELinks requires GNU Bash to compile. It declares /bin/sh in shebang,
  but uses Bash-specific features such as <<<.
  To build it, instead of
        ./configure --prefix= # etc.
  it is necessary to instead execute
        bash ./configure --prefix= # etc.

* doas in Alpine Linux uses configuration file(s) in /etc/doas.d/ over
  /etc/doas.conf, resulting in (most likely) unexpected "operation not
  permitted", which doesn't depend on the contents of /etc/doas.conf at all!
  Solution: `mv /etc/doas.conf /etc/doas.d/`

* If sled is used as EDITOR, Busybox's `crontab -e` won't work unless
        #define BACKUP_FAIL_OK
  is uncommented in config.h.

* Try to make Syncthing less privacy-intrusive:
        # Disable automatic check for upgrades
        syncthing cli config options auto-upgrade-intervalh set 0
        # Disable sending crash reports
        syncthing cli config options crenabled set false
        # Set "Global announce" (pinging centralized server) to false
        syncthing cli config options global-ann-enabled set false
        # Disable upgrading to pre-releases
        syncthing cli config options upgrade-to-pre-releases set false
        # Set "Usage Report" (telemetry) sending to "not accepted"
        syncthing cli config options uraccepted set -- -1
        # Set "Usage Report Unique ID" to empty
        syncthing cli config options urunique-id set ''

* ASan only works for Clang/LLVM under Alpine Linux. Libraries for it are
  located in the package compiler-rt.

* If `df -h` on Alpine Linux complains about "Permission denied", make sure that
  the service hwdrivers is *not* added to a runlevel. Otherwise, the following
  error is printed on stdout:
        df: /sys/kernel/debug/tracing: Permission denied

* xenodm is recommended on OpenBSD, but for those who would like to not have the
  overhead of running a display manager with a dedicated service, there is still
  a way to use startx (autostart or not) without a DM. To do so, it is needed to
  set setuid bit on Xorg binary:
        # chmod +s /usr/X11R6/bin/Xorg
  of course, it is needed to disable and stop xenodm if it is running:
        # rcctl disable xenodm && rcctl stop xenodm
  Caution: This does come with the cost of having X setuid. Still, it is
  possible, contrary to what is stated in the most replies on the matter.

* If elinks hangs on startup, try stopping gpm. If elinks was compiled with gpm
  support, sometimes it can get "confused" like that.

* In suckless st, if the text in bold is shown in yellow instead of white, check
  the fonts in config.h. You can test this by launching st from within st:
  mismatched fonts will produce warnings on stderr.
```

```
* Quick and dirty running git-daemon(1) from ~/src to export somerepo:
        cd ~/src
        git daemon --reuseaddr --verbose --base-path=. --export-all \
                ./somerepo.git
  then access with got(1) from another machine as:
        got clone git://X.X.X.X/somerepo.git
```