Vanier College

Faculty of Careers and Technical Programs

Department of Computer Science Technology

Advanced UNIX

Lab #2

Title: Linux Installation and Usage

Student Names: Marissa Gonçalves
Amy Yip
Yulia Bakaleinik

Submitted to Florin Pilat

September 4th, 2020

Review Questions *(p.73-76)*:

1. D) BASH
2. D) `info`
3. D) `halt`
4. C) `apropos` *keyword*
5. C) firewall settings
6. False
7. D) `clear`
8. C) Ctrl + Alt + F1
9. True
10. B) shell
11. False
12. E) Precede it with a \.
13. A) `man -k flush`
14. B) `who`
15. C) #
16. A) $
17. D) sdb3
18. A) / and C) swap
19. False
20. C) Spiceworks

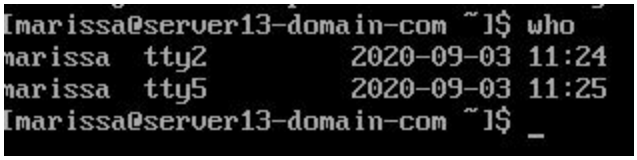
Project 2-2 *(p.77-78)*:

1. The main reason that we received this prompt message indicating my last login attempt was since we logged in earlier during the day.

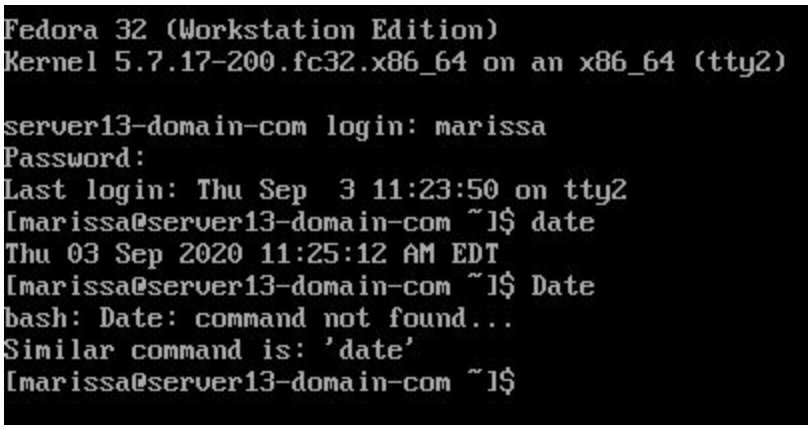
2. We have received an error because when the command `Date` is entered, it is not recognized since the command-line is case-sensitive. The BASH shell has given me this error message.

3. The main reason that we received a prompt message indicating my last login attempt was because we logged in to the other command-line terminal (tty2) earlier.

4. Another user is still logged in on the other command-line terminal (tty2).



5. Yes, the previous outputs from the command-line terminal (tty2) are still visible.



6.

```
[marissa@server13-domain-com ~]$ clear_
```

```
[marissa@server13-domain-com ~]$ _
```

```
[marissa@server13-domain-com ~]$ reset
```

```
[marissa@server13-domain-com ~]$ _
```

```
[marissa@server13-domain-com ~]$ who
marissa  tty2         2020-09-03 11:24
marissa  tty5         2020-09-03 11:25
[marissa@server13-domain-com ~]$ w
 11:40:04 up  1:16,  2 users,  load average: 0.06, 0.04, 0.00
USER     TTY        LOGIN@   IDLE   JCPU   PCPU WHAT
marissa  tty2      11:24    1.00s  0.17s  0.04s w
marissa  tty5      11:25   10:49   0.11s  0.11s -bash
[marissa@server13-domain-com ~]$ whoami
marissa
[marissa@server13-domain-com ~]$ id
uid=1000(marissa) gid=1000(marissa) groups=1000(marissa),10(wheel) context=unconfined_u:unconfined_r
:unconfined_t:s0-s0:c0.c1023
[marissa@server13-domain-com ~]$ date
Thu 03 Sep 2020 11:40:33 AM EDT
[marissa@server13-domain-com ~]$ cal
   September 2020
Su Mo Tu We Th Fr Sa
       1  2  3  4  5
 6  7  8  9 10 11 12
13 14 15 16 17 18 19
20 21 22 23 24 25 26
27 28 29 30

[marissa@server13-domain-com ~]$ uname
Linux
```

After executing the `exit` command, we were redirected back to the login prompt.

```
[marissa@server13-domain-com ~]$ exit_
```

```
Fedora 32 (Workstation Edition)
Kernel 5.7.17-200.fc32.x86_64 on an x86_64 (tty2)

server13-domain-com login: _
```

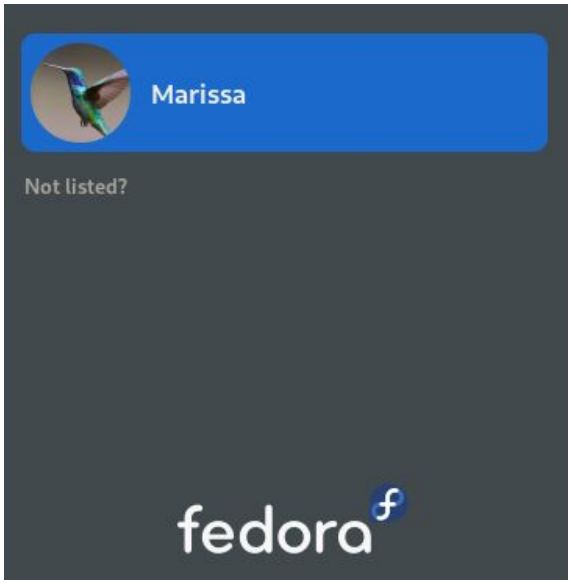
7.

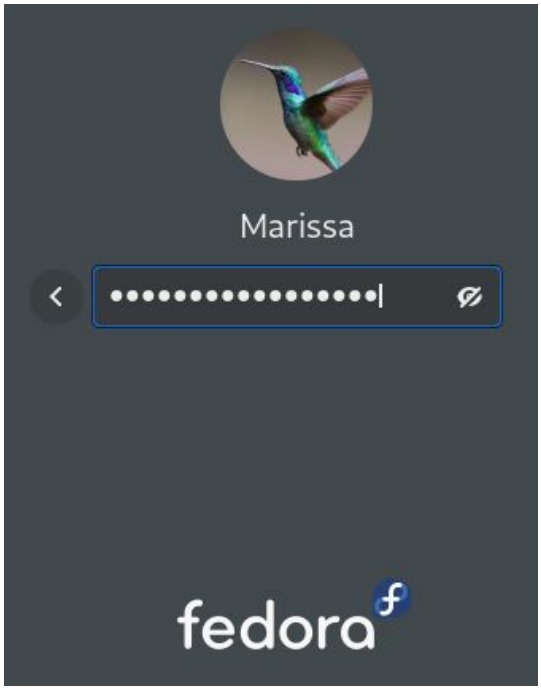
```
Fedora 32 (Workstation Edition)
Kernel 5.7.17-200.fc32.x86_64 on an x86_64 (tty5)

server13-domain-com login: marissa
Password:
Last login: Thu Sep  3 11:24:24 on tty2
[marissa@server13-domain-com ~]$ who
marissa  tty2         2020-09-03 11:24
marissa  tty5         2020-09-03 11:25
[marissa@server13-domain-com ~]$ exit
```

```
Fedora 32 (Workstation Edition)
Kernel 5.7.17-200.fc32.x86_64 on an x86_64 (tty5)

server13-domain-com login:
```

Project 2-3 [Excluding Exercises 6-11] *(p.78-79)*:

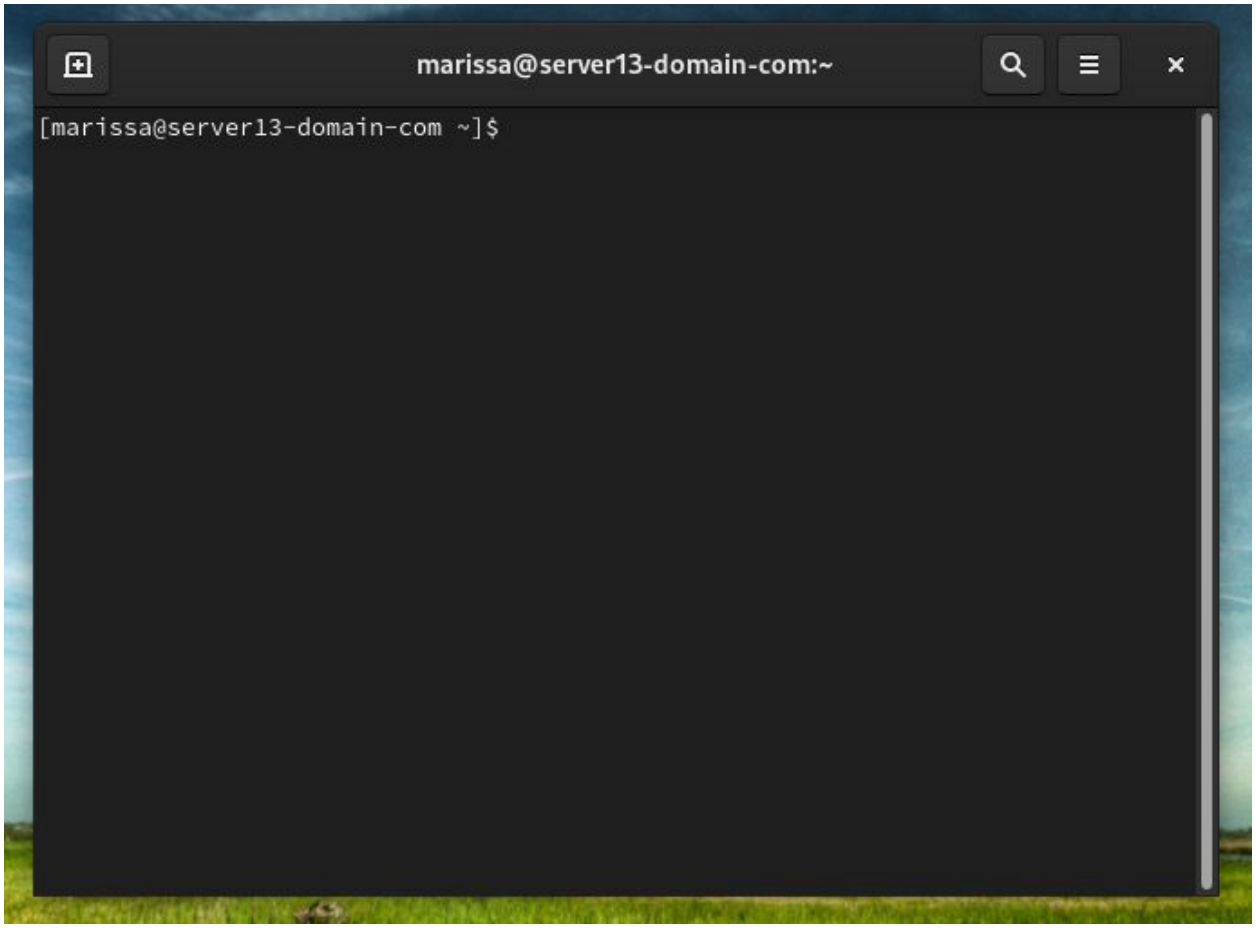
1.





2. Since we already managed to set up the desktop preferences previously after installing Fedora, we are redirected to the main desktop screen.

3. Once we run the terminal, we receive a prompt indicating the current user's name and the name of the Fedora set by that user.



4.

```
[marissa@server13-domain-com ~]$ who
marissa  tty3         2020-09-03 12:16 (tty3)
```

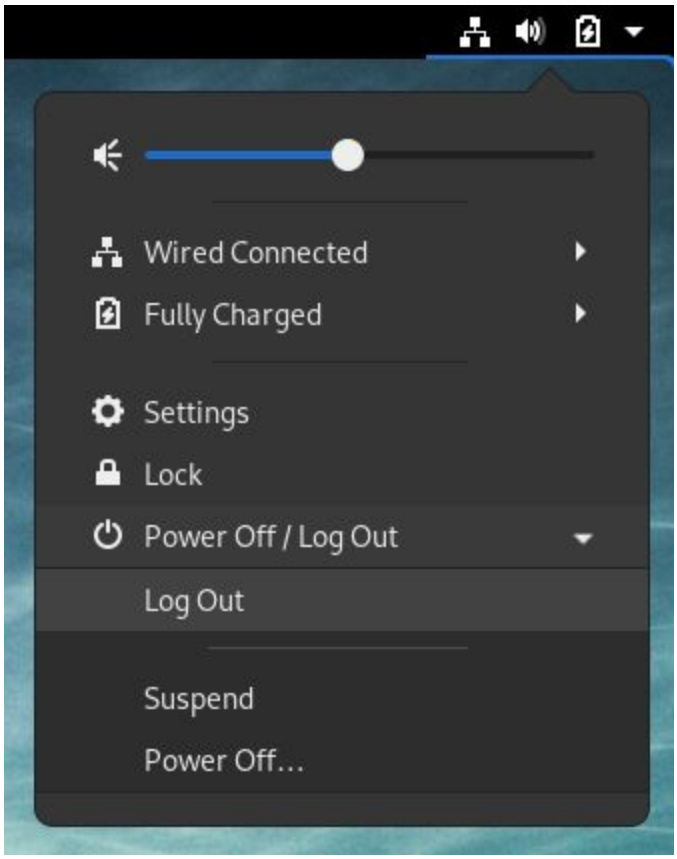
5.

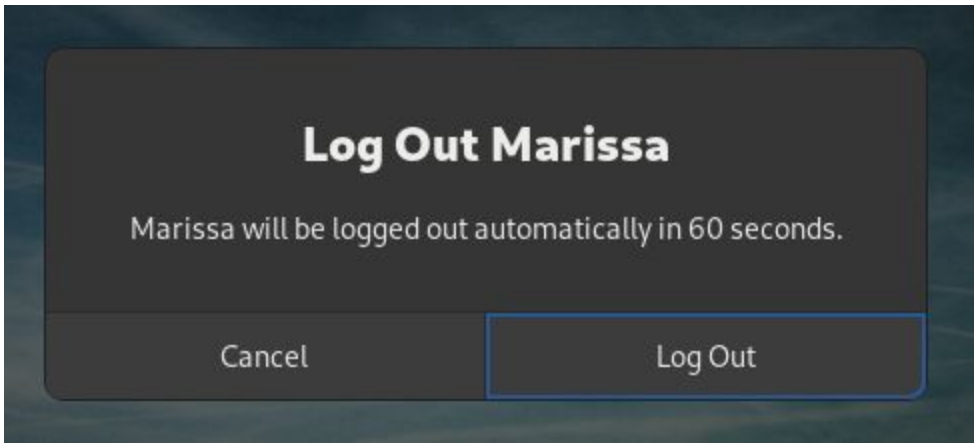
```
[marissa@server13-domain-com ~]$ sudo su
[sudo] password for marissa:
[root@server13-domain-com marissa]#
```

12. Yes, the BASH shell is used for the command-line terminals as well.

```
[root@server13-domain-com marissa]# echo $SHELL
/bin/bash
```

13.


Wired Connected
Fully Charged
Settings
Lock
Power Off / Log Out
Log Out
Suspend
Power Off...

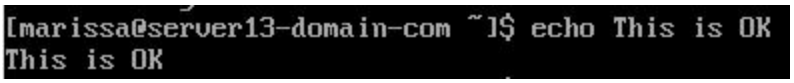

Log Out Marissa
Marissa will be logged out automatically in 60 seconds.
Cancel
Log Out

Project 2-4 *(p.79-80)*:

1.

2.

3.

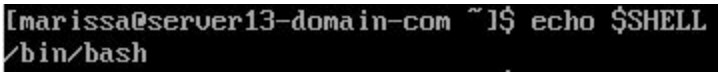


4. The apostrophe (’) in this command needs to be protected.

5. The message within the double quotation marks (“ ”) is displayed on the terminal screen.

6. The original message without the backslash (\) is displayed on the terminal screen.

7. The name of the current shell is displayed on the terminal screen.



Once I execute this command, nothing is displayed on the terminal screen.

```
[marissa@server13-domain-com ~]$ echo $TEST

[marissa@server13-domain-com ~]$
```

8. The message `You have .50` is displayed, since the shell interprets the command differently. So, the dollar sign ($) needs to be protected. You can protect this character by inserting single quotation marks (‘ ’) to enclose the message or using a backslash (\) before the character to display the entire message successfully on the terminal screen.

```
[marissa@server13-domain-com ~]$ echo You have $4.50
You have .50
```

9. The original message is displayed on the terminal screen. The single quotation marks (‘ ’) have successfully protected the metacharacter from shell interpretation.

```
[marissa@server13-domain-com ~]$ echo 'You have $4.50'
You have $4.50
```

10. The message `You have .50` is displayed because the shell interprets the command differently. The double quotation marks (“ ”) did not protect the metacharacter from shell interpretation.

```
[marissa@server13-domain-com ~]$ echo "You have $4.50"
You have .50
```

11. The original message is displayed on the terminal screen. The backslash (\) before the dollar sign ($) has successfully protected the metacharacter from shell interpretation.

```
[marissa@server13-domain-com ~]$ echo You have \$4.50
You have $4.50
```

12. Essentially, any command between the backquotes (` `) has the permission to be interpreted by the shell.

```
[marissa@server13-domain-com ~]$ echo My name is `whoami`
My name is marissa
```

13.

```
[marissa@server13-domain-com ~]$ exit
```

```
Fedora 32 (Workstation Edition)
Kernel 5.7.17-200.fc32.x86_64 on an x86_64 (tty2)

server13-domain-com login: _
```

Project 2-5 ***(p.80)*****:**

1.

```
Fedora 32 (Workstation Edition)
Kernel 5.7.17-200.fc32.x86_64 on an x86_64 (tty2)

server13-domain-com login: marissa
Password:
Last login: Thu Sep  3 13:28:26 on tty2
[marissa@server13-domain-com ~]$
```

2. Once we have executed this command, there are three manual pages for `crontab`. Each `crontab` manual page has a different function which involves file formats and commands that any user can execute.

```
[marissa@server13-domain-com ~]$ man -k cron
anacrontab (5)       - configuration file for Anacron
anacron (8)          - runs commands periodically
cron (8)             - daemon to execute scheduled commands
crond (8)            - daemon to execute scheduled commands
cronnext (1)         - time of next job cron will execute
crontab (1)          - maintains crontab files for individual users
crontab (1p)         - schedule periodic background work
crontab (5)          - files used to schedule the execution of programs
crontabs (4)         - configuration and scripts for running periodical jobs
```

3.

```
[marissa@server13-domain-com ~]$ man crontab_
```

```
CRONTAB(1)                          User Commands                          CRONTAB(1)

NAME
       crontab - maintains crontab files for individual users

SYNOPSIS
       crontab [-u user] <file | ->
       crontab [-u user] <-l | -r | -e> [-i] [-s]
       crontab -n [ hostname ]
       crontab -c
       crontab -V

DESCRIPTION
       Crontab  is  the program used to install a crontab table file, remove or list the existing
       tables used to serve the cron(8) daemon.  Each user can have their own crontab, and though
       these  are files in /var/spool/, they are not intended to be edited directly.  For SELinux
       in MLS mode, you can define more crontabs for  each  range.   For  more  information,  see
       selinux(8).

       In  this  version  of  Cron it is possible to use a network-mounted shared /var/spool/cron
       across a cluster of hosts and specify that only one of the hosts should  run  the  crontab
       jobs  in  the  particular directory at any one time.  You may also use crontab from any of
       these hosts to edit the same shared set of crontab files, and to set and query which  host
       should run the crontab jobs.

       Scheduling  cron  jobs with crontab can be allowed or disallowed for different users.  For
       this purpose, use the cron.allow and cron.deny files.  If the cron.allow  file  exists,  a
       user  must  be listed in it to be allowed to use crontab.  If the cron.allow file does not
       exist but the cron.deny file does exist, then a user must not be listed in  the  cron.deny
       file  in  order to use crontab.  If neither of these files exist, then only the super user
       is allowed to use crontab.

       Another way to restrict the scheduling of cron jobs beyond crontab is to use PAM authenti-
       cation  in  /etc/security/access.conf  to set up users, which are allowed or disallowed to
       use crontab or modify system cron jobs in the /etc/cron.d/ directory.

 Manual page crontab(1) line 1 (press h for help or q to quit)
```

4.

```
[marissa@server13-domain-com ~]$ man 5 crontab_
```

```
CRONTAB(5)                        File Formats                        CRONTAB(5)

NAME
       crontab - files used to schedule the execution of programs

DESCRIPTION
       A  crontab  file  contains instructions for the cron(8) daemon in the following simplified
       manner: "run this command at this time on this date".  Each  user  can  define  their  own
       crontab.   Commands defined in any given crontab are executed under the user who owns that
       particular crontab.  Uucp and News usually have their own crontabs, eliminating  the  need
       for explicitly running su(1) as part of a cron command.

       Blank  lines,  leading  spaces,  and  tabs are ignored.  Lines whose first non-white space
       character is a pound-sign (#) are comments, and are not processed.  Note that comments are
       not  allowed  on  the  same line as cron commands, since they are considered a part of the
       command.  Similarly, comments are not allowed on the same  line  as  environment  variable
       settings.

       An  active line in a crontab is either an environment setting or a cron command.  An envi-
       ronment setting is of the form:

           name = value

       where the white spaces around the equal-sign (=) are optional,  and  any  subsequent  non-
       leading  white  spaces in value is a part of the value assigned to name.  The value string
       may be placed in quotes (single or double, but matching) to preserve leading  or  trailing
       white spaces.

       Several  environment  variables  are set up automatically by the cron(8) daemon.  SHELL is
       set to /bin/sh, and LOGNAME and HOME are set from the /etc/passwd line  of  the  crontab's
       owner.  HOME and SHELL can be overridden by settings in the crontab; LOGNAME can not.

       (Note:  the LOGNAME variable is sometimes called USER on BSD systems and is also automati-
       cally set).

       In addition to LOGNAME, HOME, and SHELL, cron(8) looks at the MAILTO variable  if  a  mail
Manual page crontab(5) line 1 (press h for help or q to quit)_
```

5.

```
[marissa@server13-domain-com ~]$ info_
```

```
File: dir,      Node: Top,      This is the top of the INFO tree.

This is the Info main menu (aka directory node).
A few useful Info commands:

  'q' quits;
  'H' lists all Info commands;
  'h' starts the Info tutorial;
  'mTexinfo RET' visits the Texinfo manual, etc.

* Menu:

Archiving
* Cpio: (cpio).                 Copy-in-copy-out archiver to tape or disk.
* Tar: (tar).                   Making tape (or disk) archives.

Basics
* Bash: (bash).                 The GNU Bourne-Again SHell.
* Common options: (coreutils)Common options.
* Coreutils: (coreutils).       Core GNU (file, text, shell) utilities.
* Date input formats: (coreutils)Date input formats.
* File permissions: (coreutils)File permissions.
                                Access modes.
* Finding files: (find).        Operating on files matching certain criteria.
* Time: (time).                 GNU time utility.

C++ Libraries
* Source-highlight-lib: (source-highlight-lib).
                                Highlights contents

Compression
* Gzip: (gzip).                 General (de)compression of files (lzw).

DOS
* Mtools: (mtools).             Mtools: utilities to access DOS disks in Unix.
-----Info: (dir)Top, 2418 lines --Top-----------------------------------------
Welcome to Info version 6.7.  Type H for help, h for tutorial.
```

6.

```
[marissa@server13-domain-com ~]$ info date_
```

```
Next: arch invocation,  Up: System context

21.1  date : Print or set system date and time
==============================================

Synopses:

     date [OPTION]... [+FORMAT]
     date [-u|--utc|--universal] [ MMDDhhmm[[CC]YY][.ss] ]

   Invoking  date  with no FORMAT argument is equivalent to invoking it
with a default format that depends on the  LC_TIME  locale category.  In
the default C locale, this format is  '+%a %b %e %H:%M:%S %Z %Y' , so
the output looks like  Thu Mar  3 13:47:51 PST 2005 .

   Normally,  date  uses the time zone rules indicated by the  TZ
environment variable, or the system default rules if  TZ  is not set.
*Note Specifying the Time Zone with  TZ : (libc)TZ Variable.

   If given an argument that starts with a  + ,  date  prints the
current date and time (or the date and time specified by the  --date
option, see below) in the format defined by that argument, which is
similar to that of the  strftime  function.  Except for conversion
specifiers, which start with  % , characters in the format string are
printed unchanged.  The conversion specifiers are described below.

   An exit status of zero indicates success, and a nonzero value
indicates failure.

* Menu:

* Time conversion specifiers::     %[HIklMNpPrRsSTXzZ]
* Date conversion specifiers::     %[aAbBcCdDeFgGhjmuUVwWxyY]
* Literal conversion specifiers::  %[%nt]
* Padding and other flags::        Pad with zeros, spaces, etc.
-----Info: (coreutils)date invocation, 40 lines --Top-----------------------------------------------
Welcome to Info version 6.7.  Type H for help, h for tutorial.
```

7.

```
[marissa@server13-domain-com ~]$ help_
```

```
. filename [arguments]                          jobs [-lnprs] [jobspec ...] or jobs -x command>
:                                               kill [-s sigspec | -n signum | -sigspec] pid |>
[ arg... ]                                      let arg [arg ...]
[[ expression ]]                                local [option] name[=value] ...
alias [-p] [name[=value] ... ]                  logout [n]
bg [job_spec ...]                               mapfile [-d delim] [-n count] [-O origin] [-s >
bind [-lpsvPSVX] [-m keymap] [-f filename] [-q > popd [-n] [+N | -N]
break [n]                                       printf [-v var] format [arguments]
builtin [shell-builtin [arg ...]]               pushd [-n] [+N | -N | dir]
caller [expr]                                   pwd [-LP]
case WORD in [PATTERN [| PATTERN]...) COMMANDS > read [-ers] [-a array] [-d delim] [-i text] [->
cd [-L|[-P [-e]] [-@]] [dir]                    readarray [-d delim] [-n count] [-O origin] [->
command [-pVv] command [arg ...]                readonly [-aAf] [name[=value] ...] or readonly>
compgen [-abcdefgjksuv] [-o option] [-A action]> return [n]
complete [-abcdefgjksuv] [-pr] [-DEI] [-o optio> select NAME [in WORDS ... ;] do COMMANDS; done
compopt [-o|+o option] [-DEI] [name ...]        set [-abefhkmnptuvxBCHP] [-o option-name] [--]>
continue [n]                                    shift [n]
coproc [NAME] command [redirections]            shopt [-pqsu] [-o] [optname ...]
declare [-aAfFgilnrtux] [-p] [name[=value] ...> source filename [arguments]
dirs [-clpv] [+N] [-N]                          suspend [-f]
disown [-h] [-ar] [jobspec ... | pid ...]       test [expr]
echo [-neE] [arg ...]                           time [-p] pipeline
enable [-a] [-dnps] [-f filename] [name ...]    times
eval [arg ...]                                  trap [-lp] [[arg] signal_spec ...]
exec [-cl] [-a name] [command [arguments ...]] > true
exit [n]                                        type [-afptP] name [name ...]
export [-fn] [name[=value] ...] or export -p    typeset [-aAfFgilnrtux] [-p] name[=value] ...
false                                           ulimit [-SHabcdefiklmnpqrstuvxPT] [limit]
fc [-e ename] [-lnr] [first] [last] or fc -s [p> umask [-p] [-S] [mode]
fg [job_spec]                                   unalias [-a] name [name ...]
for NAME [in WORDS ... ] ; do COMMANDS; done    unset [-f] [-v] [-n] [name ...]
for (( exp1; exp2; exp3 )); do COMMANDS; done   until COMMANDS; do COMMANDS; done
function name { COMMANDS ; } or name () { COMMA> variables - Names and meanings of some shell v>
getopts optstring name [arg]                    wait [-fn] [id ...]
hash [-lr] [-p pathname] [-dt] [name ...]       while COMMANDS; do COMMANDS; done
help [-dms] [pattern ...]                       { COMMANDS ; }
```

8.

```
[marissa@server13-domain-com ~]$ help exit
exit: exit [n]
    Exit the shell.

    Exits the shell with a status of N.  If N is omitted, the exit status
    is that of the last command executed.
```

9.

```
[marissa@server13-domain-com ~]$ exit_
```

```
Fedora 32 (Workstation Edition)
Kernel 5.7.17-200.fc32.x86_64 on an x86_64 (tty2)

server13-domain-com login: _
```