

# CSGE602055 Operating Systems

## CSF2600505 Sistem Operasi

### Week 03: File System & FUSE

Rahmat M. Samik-Ibrahim (ed.)

University of Indonesia

<https://os.vlsm.org/>

Always check for the latest revision!

REV272 01-Mar-2021

# Operating Systems 211<sup>3)</sup> — PJJ from HOME

## ZOOM: A [Mon (or Wed) 10:00] — B [Mon (or Wed) 15:40] — C [Tue (or Thu) 08:00]

Week	Schedule & Deadline <sup>1)</sup>	Topic	OSC10 <sup>2)</sup>
Week 00	22 Feb - 28 Feb 2021	Overview 1, Virtualization & Scripting	Ch. 1, 2, 18.
Week 01	01 Mar - 07 Mar 2021	Overview 2, Virtualization & Scripting	Ch. 1, 2, 18.
Week 02	08 Mar - 16 Mar 2021	Security, Protection, Privacy, & C-language.	Ch. 16, 17.
Week 03	17 Mar - 23 Mar 2021	File System & FUSE	Ch. 13, 14, 15.
Week 04	24 Mar - 30 Mar 2021	Addressing, Shared Lib, & Pointer	Ch. 9.
Week 05	31 Mar - 06 Apr 2021	Virtual Memory	Ch. 10.
Week 06	07 Apr - 25 Apr 2021	Concurrency: Processes & Threads	Ch. 3, 4.
Week 07	26 Apr - 02 May 2021	Synchronization & Deadlock	Ch. 6, 7, 8.
Week 08	03 May - 09 May 2021	Scheduling + W06/W07	Ch. 5.
Week 09	17 May - 23 May 2021	Storage, Firmware, Bootloader, & Systemd	Ch. 11.
Week 10	24 May - 06 Jun 2021	I/O & Programming	Ch. 12.

<sup>1)</sup> The **DEADLINE** of Week 00 is 28 Feb 2021, whereas the **DEADLINE** of Week 01 is 07 Mar 2021, and so on...

<sup>2)</sup> Silberschatz et. al.: **Operating System Concepts**, 10<sup>th</sup> Edition, 2018.

<sup>3)</sup> This information will be on **EVERY** page two (2) of this course material.

# STARTING POINT — <https://os.vlsm.org/>

- ❑ **Text Book** — Any recent/decent OS book. Eg. (**OSC10**) Silberschatz et. al.: **Operating System Concepts**, 10<sup>th</sup> Edition, 2018. See also <https://www.os-book.com/OS10/>.
- ❑ **Resources**
  - ❑ **SCELE OS211** — <https://scele.cs.ui.ac.id/course/view.php?id=3134>.  
The enrollment key is **XXX**.
  - ❑ **Download Slides and Demos from GitHub.com**  
<https://github.com/UI-FASILKOM-OS/SistemOperasi/>:  
os00.pdf (W00), os01.pdf (W01), os02.pdf (W02), os03.pdf (W03),  
os04.pdf (W04), os05.pdf (W05), os06.pdf (W06), os07.pdf (W07),  
os08.pdf (W08), os09.pdf (W09), os10.pdf (W10).
  - ❑ **Problems** — <https://rms46.vlsm.org/2/>:  
195.pdf (W00), 196.pdf (W01), 197.pdf (W02), 198.pdf (W03),  
199.pdf (W04), 200.pdf (W05), 201.pdf (W06), 202.pdf (W07),  
203.pdf (W08), 204.pdf (W09), 205.pdf (W10).
  - ❑ **LFS** — <http://www.linuxfromscratch.org/lfs/view/stable/>
  - ❑ **OSP4DISS** — <https://osp4diss.vlsm.org/>
  - ❑ **DOIT** — <https://doit.vlsm.org/001.html>

# Agenda

- 1 Start
- 2 Schedule
- 3 Agenda
- 4 Week 03
- 5 File System Interface
- 6 File System Organization
- 7 FHS: Filesystem Hierarchy Standard
- 8 Devices
- 9 File System Implementation
- 10 File System Internals
- 11 FUSE
- 12 Week 03: Check List
- 13 The End

# Week 03 File System & FUSE: Topics<sup>1</sup>

- Files: data, metadata, operations, organization, buffering, sequential, nonsequential
- Directories: contents and structure
- File systems: partitioning, mount/unmount, virtual file systems
- Standard implementation techniques
- Memory-mapped files
- Special-purpose file systems
- Naming, searching, access, backups
- Journaling and log-structured file systems

---

<sup>1</sup>Source: ACM IEEE CS Curricula 2013

# Week 03 File System & FUSE: Learning Outcomes<sup>1</sup>

- Describe the choices to be made in designing file systems.  
[Familiarity]
- Compare and contrast different approaches to file organization, recognizing the strengths and weaknesses of each. [Usage]
- Summarize how hardware developments have led to changes in the priorities for the design and the management of file systems.  
[Familiarity]
- Summarize the use of journaling and how log-structured file systems enhance fault tolerance. [Familiarity]

---

<sup>1</sup>Source: ACM IEEE CS Curricula 2013

# File System Interface

- File Concept

- File Attributes: Name, Id, Type, Location, Size, Protection, Time Stamp: create, last modified, last accessed.
- File Operation
  - Create/Delete/Truncate
  - Open/Close
  - Read/Write
- File Types: Executable, Object, Source Code, Library, Markup, Markdown, Archive, Compressed.
- File Structure: No Structure (just a string).
- Access Methods: Sequential vs Direct Access

- Directory and Disk Structure

- Three-Structured Directories
- Directory Operation: create/delete, search/list, rename, traverse
- Path Name: Absolute vs. Relative
- FS Mounting vs. Volume Based System

- File Sharing

- Protection: Access Control (eg. -rwx-x-x)

# File System Organization

- Disk Partition
  - One Disk — Many Partitions
  - Many Disks — One Partitions
  - Many Disks — Many Partitions
  - One Partition — One File System (Volume)
- Mounting vs. Volumes

```
demo@badak:~$ df
```

Filesystem	1K-blocks	Used	Available	Use%	Mounted on
/dev/sda2	9515660	1435776	7573468	16%	/
/dev/sdb1	32895760	12156672	19045036	39%	/usr
/dev/sdc1	412322216	79695252	311639116	21%	/home
udev	10240	0	10240	0%	/dev
tmpfs	16508828	0	16508828	0%	/dev/shm
tmpfs	6603532	8880	6594652	1%	/run
tmpfs	5120	0	5120	0%	/run/lock
tmpfs	16508828	0	16508828	0%	/sys/fs/cgroup
tmpfs	3301768	0	3301768	0%	/run/user/1002

```
demo@badak:~$
```



# FHS: Filesystem Hierarchy Standard

- Source (URL) [http://refspecs.linuxfoundation.org/FHS\\_3.0/fhs-3.0.pdf](http://refspecs.linuxfoundation.org/FHS_3.0/fhs-3.0.pdf)
- A file placement guidelines/requirements for GNU/Linux-like OS.

FILES	shareable (multiple hosts)	unshareable (single hosts)
static (read only, except for update)	/usr, /opt	/etc, /boot
variable (r/w)	/var/mail, /var/spool/news	/var/run, /var/lock

## ● The Root File System (Required)

Directory	Description
/bin	Essential command binaries
/boot	Static files of the boot loader
/dev	Device files
/etc	Host-specific system configuration
/lib	Essential shared libraries and kernel modules
/media	Mount point for removable media
/mnt	Mount point for mounting a filesystem temporarily
/opt	Add-on application software packages
/run	Data relevant to running processes
/sbin	Essential system binaries
/srv	Data for services provided by this system
/tmp	Temporary files
/usr	Secondary hierarchy
/var	Variable data

- Specific Options

Directory	Description
/home	User home directories (optional)
/lib<qual>	Alternate format essential shared libraries(optional)
/root	Home directory for the root user (optional)

- The /usr Hierarchy

Directory	Description
/usr/bin	Most user commands (required)
/usr/lib	Libraries (required)
/usr/local	Local hierarchy (empty after main installation) (required) /usr/local/{bin etc games include lib man sbin share src} (required)
/usr/sbin	Non-vital system binaries (required)
/usr/share	Architecture-independent data (required) /usr/share/{man misc} (required) /usr/share/{color dict doc games info locale} (optional) /usr/share/{nls ppd sgml terminfo tmac xml zoneinfo} (optional)
/usr/games	Games and educational binaries (optional)
/usr/include	Header files included by C programs (optional)
/usr/libexec	Binaries run by other programs (optional)
/usr/lib<qual>	Alternate Format Libraries (optional)
/usr/src	Source code (optional)

## • The /var Hierarchy

Directory	Description
/var/cache	Application cache data (required)
/var/lib	Variable state information (required) /var/lib/misc (required)
/var/local	Variable data for /usr/local (required)
/var/lock	Lock fileslogLog files and directories (required)
/var/opt	Variable data for /opt (required)
/var/run	Data relevant to running processes (required)
/var/spool	Application spool data (required)
/var/tmp	Temporary files preserved between system reboots (required)
/var/backups	(reserved names, do not use)
/var/cron	(reserved names, do not use)
/var/msgs	(reserved names, do not use)
/var/preserve	(reserved names, do not use)
/var/account	Process accounting logs (optional)
/var/crash	System crash dumps (optional)
/var/games	Variable game data (optional)
/var/mail	User mailbox files (optional)
/var/yp	Network Information Service (NIS) database files(optional)

- (Mostly) Linux

Directory	Description
/proc	Kernel and process information virtual filesystem
/sys	Kernel and system information virtual filesystem
/usr/include	Header files included by C programs
/usr/src	Source code
/var/spool/cron	cron and at jobs

- the `/dev/` directory
  - `/etc/fstab`: configuration of filesystems
  - `/etc/mtab` → `/proc/mounts`: mounted filesystems
  - `/proc/swaps`: swap filesystems
  - `df`: checking disk space and filesystems
  - Device Major and Minor Numbers
  - UUID - Universally Unique Identifier (128 bits)
  - GUID - Globally Unique Identifiers: `ls -al /dev/disk/by-uuid`
  - practically is NOT guaranteed unique
  - FUSE: Filesystem in Userspace
  - BBFS: Big Brother File System
- More Storage Structure
  - `tmpfs`
  - `objfs`
  - `ctfs`
  - `lofs`
  - `procfs`
  - `ufs`
  - `zfs`

# A Typical Ubuntu 18.04 Work Station

```
rms46@rmsbase:~$ df
```

Filesystem	1K-blocks	Used	Available	Use%	Mounted on
/dev/sda1	511996	31772	480224	7%	/otr/ntfs1
/dev/sda2	250823676	167941776	82881900	67%	/otr/ntfs2
/dev/sda5	31588496	9181304	20779532	31%	/
/dev/sda6	123866100	39281464	78269476	34%	/home
/dev/sda7	490099792	270878316	194302800	59%	/extra
/dev/sda8	778472088	538257360	200647612	73%	/arsip
/dev/sda9	197809844	66848396	120890188	36%	/u1904
/dev/sda10	51851620	7784424	41410236	16%	/u1810
udev	8159412	0	8159412	0%	/dev
tmpfs	8189664	142196	8047468	2%	/dev/shm
tmpfs	1637936	2108	1635828	1%	/run
tmpfs	5120	4	5116	1%	/run/lock
tmpfs	1637932	16	1637916	1%	/run/user/121
tmpfs	1637932	44	1637888	1%	/run/user/1000
tmpfs	1637932	0	1637932	0%	/run/user/0
tmpfs	8189664	0	8189664	0%	/sys/fs/cgroup
/dev/sdc1	259103	8	259096	1%	/media/rms46/FAT32
/dev/sdc2	60360796	4694276	52600360	9%	/media/rms46/FLASHDISK
/dev/sdd1	7799912	331988	7467924	5%	/media/rms46/OS
/dev/loop0	93312	93312	0	100%	/snap/core/6259
/dev/loop1	14976	14976	0	100%	/snap/gnome-logs/45
/dev/loop2	35712	35712	0	100%	/snap/gtk-common-themes/1122
/dev/loop3	13312	13312	0	100%	/snap/gnome-characters/103
/dev/loop4	93184	93184	0	100%	/snap/core/6350
/dev/loop5	13312	13312	0	100%	/snap/gnome-characters/139
/dev/loop6	35456	35456	0	100%	/snap/gtk-common-themes/818
/dev/loop7	35584	35584	0	100%	/snap/gtk-common-themes/319
/dev/loop8	144128	144128	0	100%	/snap/gnome-3-26-1604/74
/dev/loop9	93184	93184	0	100%	/snap/core/6405
/dev/loop10	14848	14848	0	100%	/snap/gnome-logs/37

# File Systems Implementation

- File System Layers / Structure
  - Application Programs
  - Logical File Systems
  - File-Organization Module
  - Basic File Systems
  - I/O Control
  - Hardware Device
- File System Implementation
- File Control Block
- FS In Memory Structure
- VFS: Virtual File Systems
  - How to support multiple File Systems
  - I.e. How to support multiple `open()/close()` `read()/write()` operations

# Implementation and Allocation Method

- Directory Implementation
  - Linear List
  - Hash Table
- Allocation Method
  - Contiguous
  - Linked
  - Indexed
  - Combined Scheme
- Free Space Management
- Performance & Efficiency
- Unified Buffer Cache
- Recovery
- Log Structured File System



- File Systems
- File-System Mounting
- Partitions and Mounting
- File Sharing
- Virtual File Systems
- Remote File Systems
- Consistency Semantics
- NFS

```
demo@badak:~/mydemo/W03-demos$ ls -al
total 20
drwxr-xr-x  4 demo demo 4096 Feb 27 19:32 .
drwx----- 14 demo demo 4096 Feb 27 19:32 ..
-rw-r--r--  1 demo demo  672 Feb 27 19:32 1-READ-THIS-FIRST.txt
drwxr-xr-x  2 demo demo 4096 Feb 27 19:32 Files
drwxr-xr-x  2 demo demo 4096 Feb 27 19:32 FUSE
demo@badak:~/mydemo/W03-demos$ cat 1-READ-THIS-FIRST.txt
```

[...etc...]

Folder Name:

Week03/

To copy the folder to your home directory:

```
cp -r /extra/Demos/W03-demos/ W03-demos/
```

=====

File Listing:

\* 1-READ-THIS-FIRST.txt (this file)

\* Files

[...etc...]

# FUSE (2)

```
demo@badak:~/mydemo/W03-demos$ cd FUSE/
demo@badak:~/mydemo/W03-demos/FUSE$ ls -al
total 164
drwxr-xr-x 2 demo demo 4096 Feb 27 19:32 .
drwxr-xr-x 4 demo demo 4096 Feb 27 19:32 ..
-rw-r--r-- 1 demo demo 2321 Feb 27 19:32 1-READ-ME.txt
-rw-r--r-- 1 demo demo 151814 Feb 27 19:32 fuse-tutorial.tgz
demo@badak:~/mydemo/W03-demos/FUSE$ cat 1-READ-ME.txt
[...etc...]
```

FUSE DEMO STEP by STEP

=====

ATTN: This does not work for WSL! See also

<http://www.secfs.net/winfsp/blog/files/winfsp-2017.html>

<https://wpdev.uservoice.com/forums/266908-command-prompt-console-windows-subsystem-for-l/suggestions/13522>

1. UBUNTU's deb packages (privilege):  

```
sudo apt-get install autoconf automake build-essential \
fuse libfuse-dev lynx pkg-config sshfs
```
2. Get a NEW tarball with  

```
wget http://www.cs.nmsu.edu/~pfeiffer/fuse-tutorial.tgz
```

  
OR use the current fuse-tutorial.tgz
3. List and open the tarball with  

```
tar tfz fuse-tutorial.tgz
tar xfz fuse-tutorial.tgz
```
4. Enter the directory (yours may be a different version)  

```
cd fuse-tutorial-2018-02-04/
ls -al
```

# FUSE (3)

5. Read the manual with  
lynx index.html

---

## Writing a FUSE Filesystem: a Tutorial

Joseph J. Pfeiffer, Jr., Ph.D. (pfeiffer@cs.nmsu.edu)  
Emeritus Professor  
Department of Computer Science, New Mexico State University

Version of 2018-02-04

One of the real contributions of Unix has been the view that "everything is a file". A tremendous number of radically different sorts of objects, from data storage to file format conversions to internal operating system data structures, have been mapped to the file abstraction.

One of the more recent directions this view has taken has been Filesystems in User Space, or FUSE (no, the acronym really doesn't work. Oh well). The idea here is that if you can envision your interaction with an object in terms of a directory structure and filesystem operations, you can write a FUSE file system to provide that interaction. You just write code that implements file operations like `open()`, `read()`, and `write()`; when your filesystem is mounted, programs are able to access the data using the standard file operation system calls, which call your code.

FUSE filesystems have been written to do everything from providing remote access to files on a different host without using NFS or CIFS (see SSHFS at [2]<https://github.com/libfuse/sshfs>) to implementing a filesystem to talk to devices using the Media Transfer protocol (see

[.....])

# FUSE (4)

```
6. Run
   ./configure
   make
```

```
7 cd example
```

TO TRY:

```
$ ls -al rootdir
$ ls -al mountdir
$ df
$ ../src/bbfs rootdir/ mountdir/
$ df
$ ls -al rootdir
$ ls -al mountdir
```

TO PLAY:

```
$ cd mountdir
$ touch blah-blah-blah.txt
$ ls -al
$ cd ..
$ ls -al rootdir
```

TO FINISH:

```
$ fusermount -u mountdir
```

EXTRA:

```
# /etc/fstab: configuration of filesystems
# /etc/mtab --> /proc/mounts: mounted filesystems
# /proc/swaps: swap filesystems
# df: checking disk space and filesystems
# GUID (Globally Unique Identifiers) ls -al /dev/disk/by-uuid
RMS
```

# FUSE (5)

```
>>>> $ ./configure
checking for a BSD-compatible install... /usr/bin/install -c
checking whether build environment is sane... yes
checking for a thread-safe mkdir -p... /bin/mkdir -p
checking for gawk... gawk
checking whether make sets $(MAKE)... yes
checking whether make supports nested variables... yes
checking for gcc... gcc
checking whether the C compiler works... yes
checking for C compiler default output file name... a.out
checking for suffix of executables...
checking whether we are cross compiling... no
checking for suffix of object files... o
checking whether we are using the GNU C compiler... yes
checking whether gcc accepts -g... yes
checking for gcc option to accept ISO C89... none needed
checking whether gcc understands -c and -o together... yes
checking for style of include used by make... GNU
checking dependency style of gcc... gcc3
checking how to run the C preprocessor... gcc -E
checking for grep that handles long lines and -e... /bin/grep
checking for egrep... /bin/grep -E
checking for ANSI C header files... yes
[...]
checking for fdatasync... yes
checking that generated files are newer than configure... done
configure: creating ./config.status
config.status: creating Makefile
config.status: creating html/Makefile
config.status: creating src/Makefile
config.status: creating src/config.h
config.status: executing depfiles commands
```

# FUSE (6)

```
>>>> $ make
Making all in example
make[1]: Entering directory '/home/demo/mydemo/W09-demos/fuse-tutorial-2018-02-04/example'
mkdir -p mountdir
mkdir -p rootdir
echo "bogus file" > rootdir/bogus.txt
make[1]: Leaving directory '/home/demo/mydemo/W09-demos/fuse-tutorial-2018-02-04/example'
Making all in html
make[1]: Entering directory '/home/demo/mydemo/W09-demos/fuse-tutorial-2018-02-04/html'
make[1]: Nothing to be done for 'all'.
make[1]: Leaving directory '/home/demo/mydemo/W09-demos/fuse-tutorial-2018-02-04/html'
Making all in src
make[1]: Entering directory '/home/demo/mydemo/W09-demos/fuse-tutorial-2018-02-04/src'
make all-am
make[2]: Entering directory '/home/demo/mydemo/W09-demos/fuse-tutorial-2018-02-04/src'
gcc -DHAVE_CONFIG_H -I. -D_FILE_OFFSET_BITS=64 -I/usr/include/fuse -g -O2 -MT bbfs.o -MD -MP -MF
.deps/bbfs.Tpo -c -o bbfs.o bbfs.c
mv -f .deps/bbfs.Tpo .deps/bbfs.Po
gcc -DHAVE_CONFIG_H -I. -D_FILE_OFFSET_BITS=64 -I/usr/include/fuse -g -O2 -MT log.o -MD -MP -MF
.deps/log.Tpo -c -o log.o log.c
mv -f .deps/log.Tpo .deps/log.Po
gcc -D_FILE_OFFSET_BITS=64 -I/usr/include/fuse -g -O2 -o bbfs bbfs.o log.o -lfuse -pthread
make[2]: Leaving directory '/home/demo/mydemo/W09-demos/fuse-tutorial-2018-02-04/src'
make[1]: Leaving directory '/home/demo/mydemo/W09-demos/fuse-tutorial-2018-02-04/src'
make[1]: Entering directory '/home/demo/mydemo/W09-demos/fuse-tutorial-2018-02-04'
make[1]: Nothing to be done for 'all-am'.
make[1]: Leaving directory '/home/demo/mydemo/W09-demos/fuse-tutorial-2018-02-04'
>>>> $
```

# FUSE (7)

```
>>>> $ cd example/
>>>> $ ls -al rootdir/
total 12
drwxr-xr-x 2 demo demo 4096 Apr 25 18:23 .
drwxr-xr-x 4 demo demo 4096 Apr 25 18:23 ..
-rw-r--r-- 1 demo demo 11 Apr 25 18:23 bogus.txt
>>>> $ ls -al mountdir/
total 8
drwxr-xr-x 2 demo demo 4096 Apr 25 18:23 .
drwxr-xr-x 4 demo demo 4096 Apr 25 18:23 ..
>>>> $ df
Filesystem      1K-blocks      Used Available Use% Mounted on
udev              10240          0       10240   0% /dev
tmpfs            1639412    103116    1536296   7% /run
/dev/vda2        9515660    1677648    7331596  19% /
/dev/vdc1        32895760 12093508    19108200  39% /usr
tmpfs            4098528          0    4098528   0% /dev/shm
tmpfs            5120          0       5120   0% /run/lock
tmpfs            4098528          0    4098528   0% /sys/fs/cgroup
/dev/vdb1        515929528 38454128 451244668   8% /home
tmpfs            819708          0     819708   0% /run/user/1002
>>>> $ ../src/bbfs rootdir/ mountdir/
Fuse library version 2.9
about to call fuse_main
>>>> $ df
Filesystem      1K-blocks      Used Available Use% Mounted on
udev              10240          0       10240   0% /dev
[...]
tmpfs            819708          0     819708   0% /run/user/1002
bbfs            515929528 38454136 451244660   8% /home/demo/mydemo/W09-demos/
                                     fuse-tutorial-2018-02-04/example/mountdir
>>>> $
```



# FUSE (8)

```
>>>> $ ls -al rootdir/
total 12
drwxr-xr-x 2 demo demo 4096 Apr 25 18:23 .
drwxr-xr-x 4 demo demo 4096 Apr 25 18:26 ..
-rw-r--r-- 1 demo demo 11 Apr 25 18:23 bogus.txt
>>>> $ ls -al mountdir/
total 12
drwxr-xr-x 2 demo demo 4096 Apr 25 18:23 .
drwxr-xr-x 4 demo demo 4096 Apr 25 18:26 ..
-rw-r--r-- 1 demo demo 11 Apr 25 18:23 bogus.txt
>>>> $ cd mountdir/
>>>> $ touch blah-blah-blah.txt
>>>> $ ls -al
total 12
drwxr-xr-x 2 demo demo 4096 Apr 25 18:30 .
drwxr-xr-x 4 demo demo 4096 Apr 25 18:26 ..
-rw-r--r-- 1 demo demo 0 Apr 25 18:30 blah-blah-blah.txt
-rw-r--r-- 1 demo demo 11 Apr 25 18:23 bogus.txt
>>>> $ cd ..
>>>> $ ls -al rootdir/
total 12
drwxr-xr-x 2 demo demo 4096 Apr 25 18:30 .
drwxr-xr-x 4 demo demo 4096 Apr 25 18:26 ..
-rw-r--r-- 1 demo demo 0 Apr 25 18:30 blah-blah-blah.txt
-rw-r--r-- 1 demo demo 11 Apr 25 18:23 bogus.txt
>>>> $ fusermount -u mountdir
>>>> $ ls -al mountdir/
total 8
drwxr-xr-x 2 demo demo 4096 Apr 25 18:23 .
drwxr-xr-x 4 demo demo 4096 Apr 25 18:26 ..
>>>> $
```

# Week 03: Check List (Deadline: Monday, 12-Oct-2020).

- ☐ Week 03 Token: **OS202**
- ☐ **Starting Points:** <https://os.vlsm.org/>, CBKadal, and OSP4DISS.
- ☐ Week 03: Assignment (Details in <https://osp4diss.vlsm.org/W03.html>).
  - ① Read: (OSC10 chapter 13 + chapter 14 + chapter 15)
  - ② Try Demos Week 03 (and Week 02).
  - ③ Visit your Virtual Guest, and update file ".bash\_aliases" (See OSP4DISS).
  - ④ Visit <https://os.vlsm.org/GitHubPages/>. Review **Last Week TOP 10 List** and pick at least 3 out of your 10 closest neighbors. See <https://cbkadal.github.io/os211/TXT/myrank.txt>.
  - ⑤ Create your **TOP 10 List** of Week 03 (e.g. <https://cbkadal.github.io/os211/W03/>). Do not use lecture material. Please be more creative!
  - ⑥ Run "chktoken OS202" and write the result into myW03token.txt.
  - ⑦ Download TLPI Code and write the "disk usage" (du) into myW03disk.txt.
  - ⑧ Update your log (e.g. <https://cbkadal.github.io/os211/TXT/mylog.txt>).
  - ⑨ Update bash script (e.g. <https://cbkadal.github.io/os211/TXT/myscript.sh>).
  - ⑩ Make SHA256SUM and sign it (detached, armor) as SHA256SUM.asc.

# The End

- ☐ This is the end of the presentation.
- ☒ This is the end of the presentation.
  - This is the end of the presentation.