PRAKTIKUM I

INSTALLASI SISTEM OPERASI LINUX

Tujuan Instruksional Umum:

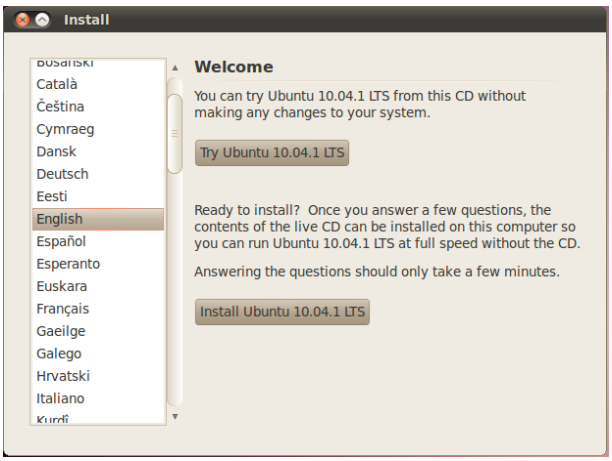
- Setelah mengikuti praktikum ini, mahasiswa diharapkan mampu menggunakan sebuah sistem operasi berbasis *Open Source* (Linux).

Tujuan Instruksional Khusus:

- Mahasiswa mampu menginstall salah satu distro sistem operasi Linux, dalam hal ini dipilih Ubuntu.
- Mahasiswa familiar dengan *Desktop Environment* sistem operasi berbasis Linux.

I.1 Installasi Ubuntu 10.04

1. Untuk memulai proses installasi Ubuntu 10.04, pertama kali masukkan CD Ubuntu 10.04 lalu boot ulang komputer menggunakan CD room drive sebagai *first boot sequence*. Kemudian akan tampil seperti berikut:

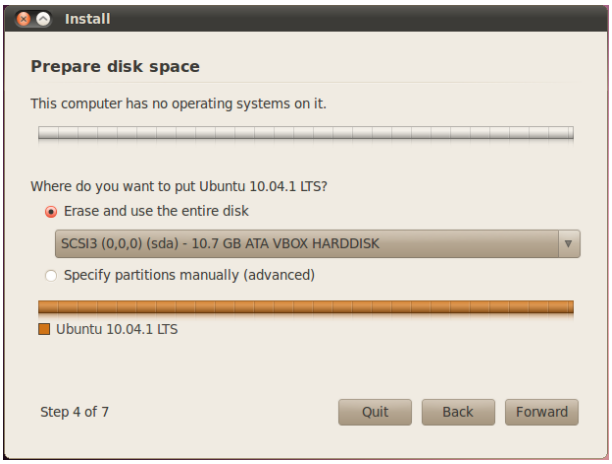


2. Pada gambar di atas, dapat dipilih bahasa yang akan memandu proses installasi pada bagian kiri. Kemudian dapat dipilih salah satu dari dua tombol yang ada di sebelah kanan, yaitu:
   - Tombol “Try Ubuntu 10.04 LTS” digunakan jika ingin mencoba Ubuntu tanpa mengubah sistem yang sudah ada.
   - Tombol “Install Ubuntu 10.04 LTS” digunakan jika ingin menginstall Ubuntu.

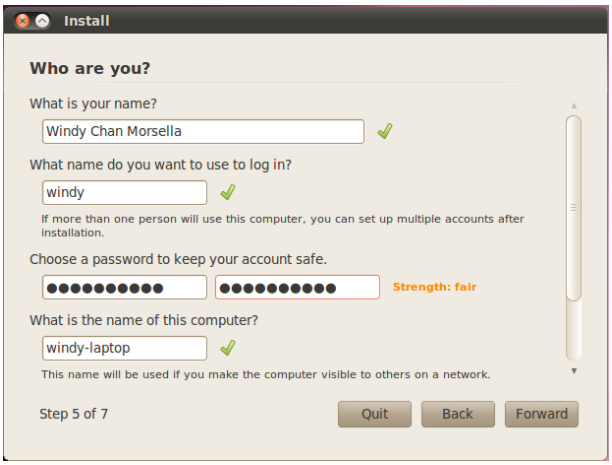
3. Tekan pada tombol “Install Ubuntu 10.04 LTS”. Kemudian akan muncul tampilan:

4. Pada bagian **Where are you?**, peta dunia muncul. Pilih dengan menggunakan mouse bagian kota tempat domisili Anda, misalkan dipilih kota Jakarta. Lalu secara otomatis *Region* dan *Time Zone* akan berubah mengikuti kota yang dipilih pada peta. Lalu pilih tombol **Forward**, sehingga tampil:

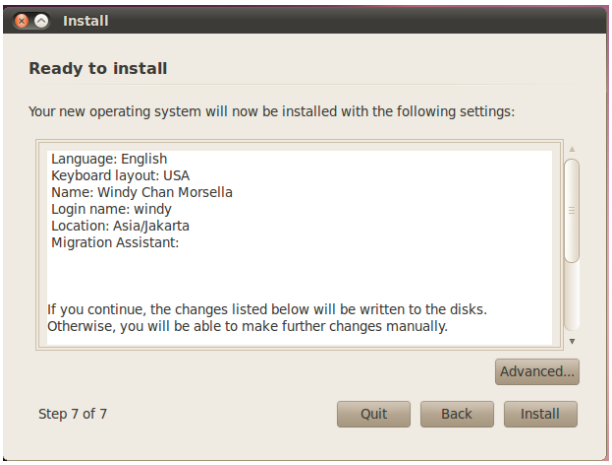
5. Pilih **Keyboard layout** sesuai dengan keyboard yang digunakan. Default-nya akan terpilih sendiri. Pada contoh di atas terpilih USA. Kemudian pilih tombol **Forward**, sehingga muncul tampilan:



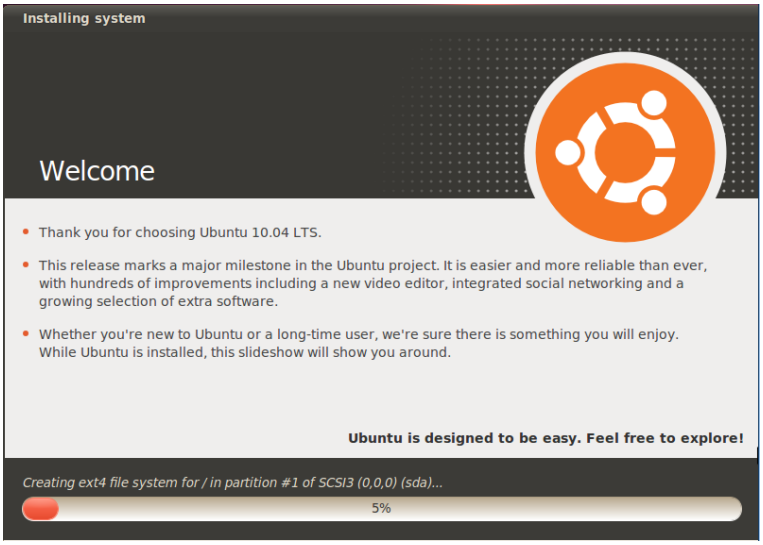
6. Karena pada sistem belum ada sistem operasi yang lain, maka hanya terdapat satu pilihan pada bagian **Prepare disk space** ini, yaitu “Erase and use the entire disk” yang berarti akan menggunakan seluruh ruang hardisk untuk installasi Ubuntu. Lalu pilih tombol **Forward**, sehingga muncul tampilan:



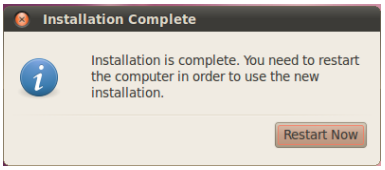
7. Pada bagian Who are you?, masukkan nama lengkap Anda pada isian pertama, lalu masukkan nama yang akan digunakan untuk *log in* pada isian kedua, serta masukkan password sebanyak dua kali di isian selanjutnya. Bagian nama komputer akan terisi secara otomatis, tetapi tetap dapat diubah jika Anda menginginkannya. Tekan tombol **Forward**.



8. Tampilan **Ready to install** akan menunjukkan setting dari berbagai optional yang telah dipilih pada tahap-tahap sebelumnya. Tombol **Back** dapat dipilih jika ingin mengulangi tahap sebelumnya. Tekan tombol **Install** untuk memulai proses installasi.

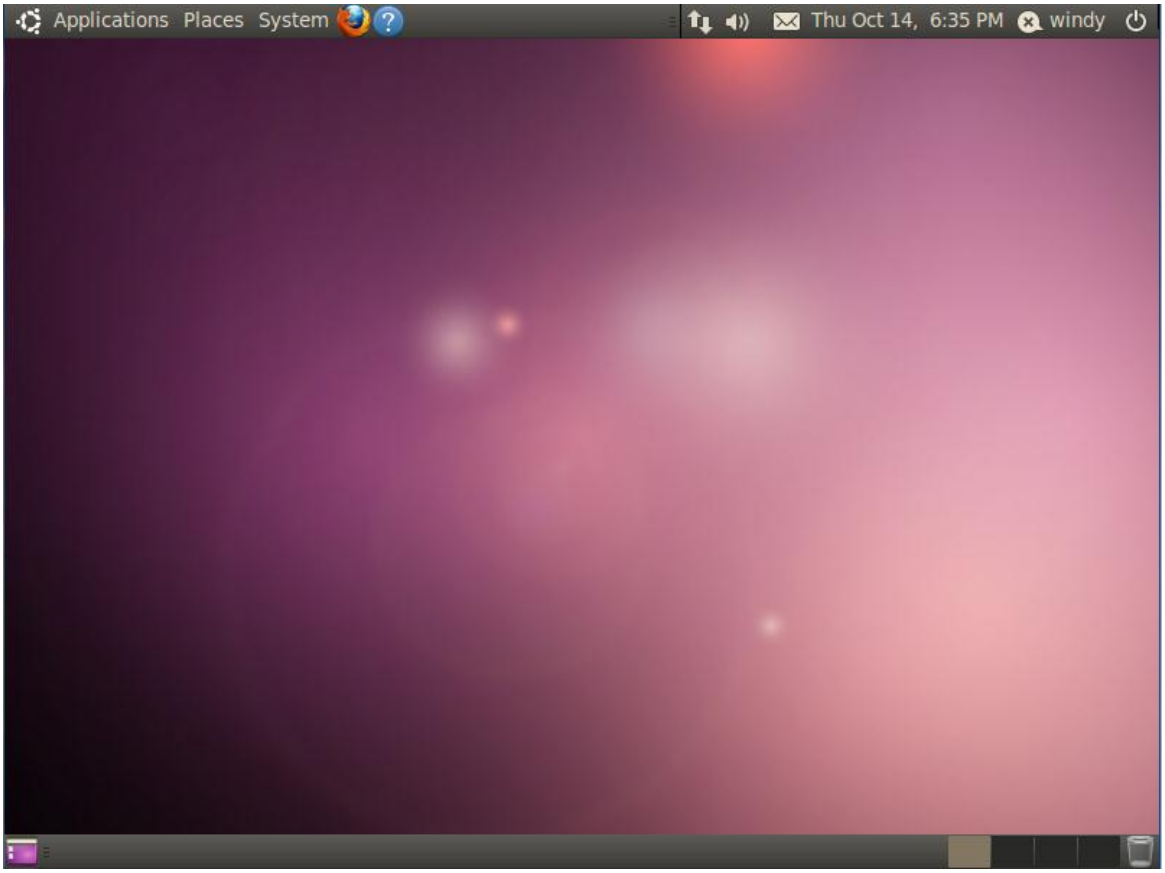


9. Tampilan **Installing system** di atas akan memakan waktu beberapa menit, tergantung kecepatan prosesor dan kapasitas memory komputer. Progress bar akan menunjukkan prosestasi yang telah ditempuh proses installasi. Jika sudah selesai, akan muncul tampilan:



10. Lalu sistem akan meminta Anda untuk me-*restart* komputer. Tekan tombol **Restart Now**.
11. Lalu muncul permintaan log in. Pilih nama pengguna dan masukkan password yang telah dimasukkan pada saat installasi.

12. Sistem Ubuntu 10.04 siap digunakan. Tombol menu ada di sebelah kanan atas, yaitu menu **Applications**, **Places**, dan **System**.



13. Enjoy the system!

I.2 Tugas

- Perhatikan prosedur installasi Ubuntu yang ditunjukkan di depan kelas, konsultasikan jika ada pertanyaan, dan catat semua hal yang dianggap perlu.
- Bagi yang punya komputer sendiri, praktekkan semua hal yang dipelajari di rumah. Pastikan sistem operasi Ubuntu terinstall dengan baik.

-ooOoo-

PRAKTIKUM II

MENGENAL PERINTAH SHELL

Tujuan Instruksional Umum:

- Setelah mengikuti praktikum ini, mahasiswa diharapkan mampu menggunakan *shell command* (perintah *shell*) pada Linux.

Tujuan Instruksional Khusus:

- Mahasiswa mampu mengaplikasikan beberapa *syntax shell command* pada terminal.
- Mahasiswa mampu mengaplikasikan pengoperasian file menggunakan *shell command* yang telah dipelajari.

II.1 Pendahuluan

Linux shell merupakan tool berbasis teks untuk berinteraksi dengan komputer. *Linux shell* sering juga disebut dengan *xterm, konsole, terminal, shell command,* ataupun *shell* saja. *Shell* ini merupakan antarmuka penghubung user dengan sistem. *Shell* juga sering disebut dengan *interpreter* yang mengoperasikan sebuah loop sederhana yakni menerima perintah, menginterpretasikan perintah, menjalankan perintah, dan menunggu perintah masukan berikutnya.

Satu hal yang menarik dari sistem operasi GNU / linux adalah penggunaan *shell command* tetap dipertahankan karena beberapa alasan berikut:

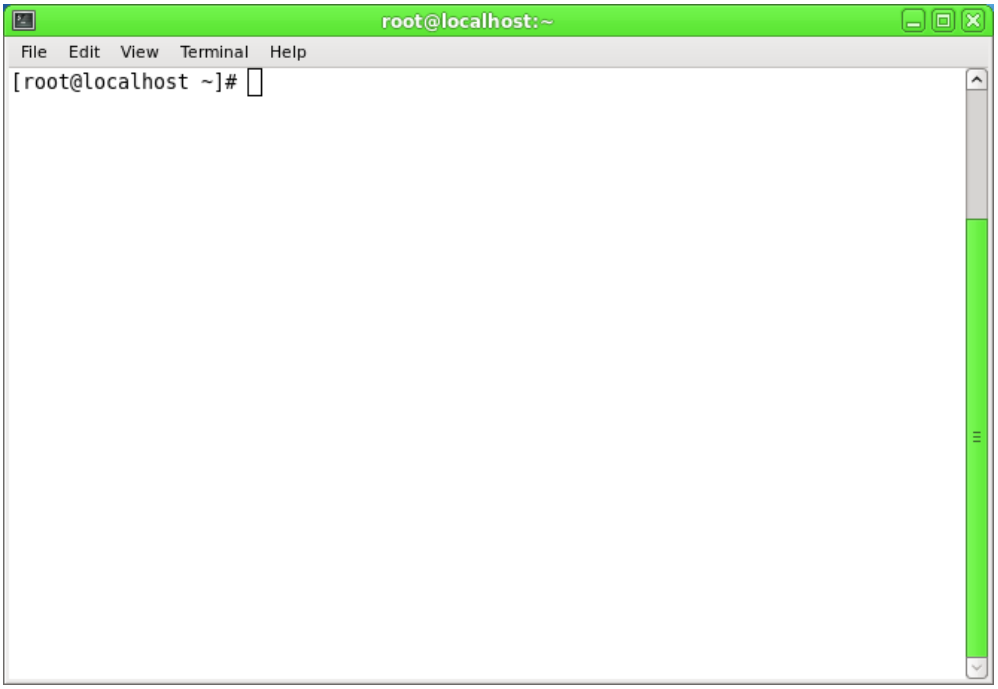
1. Kebanyakan konfigurasi sistem dapat dilakukan dari *shell command*.
2. *Linux shell* memberikan fleksibilitas terutama saat anda bekerja dengan banyak file.
3. Untuk tindakan penyelamatan terhadap sebuah data atau recovery terhadap sistem dan pekerjaan perawatan sistem pada *single mode* dapat anda lakukan lewat *shell command*.

Pada sistem operasi lain, penggunaan *shell command* sudah mulai dikurangi atau bahkan dihilangkan sama sekali. Pada beberapa sistem operasi lain tersebut, para penggunanya semakin dimanja dengan penggunaan mode grafis (*Graphical desktop*) dan tidak memperdulikan lagi penggunaan *shell command* tersebut.

(Debian GNU/Linux 2nd Edition, ©2004-2007, *http://www.debianindonesia.org*).

II.2 Menjalankan aplikasi terminal

Untuk menjalankan aplikasi terminal. pilih pada menu Application → Accessories → Terminal. Kemudian muncul:



Tanda kotak berkedip pada tampilan di atas sedang menunggu perintah *shell* untuk dimasukkan. Tekan tombol enter untuk mengeksekusi setiap perintah yang dimasukkan.

Beberapa perintah *shell* yang umum digunakan di Linux di antaranya sebagai berikut:

Command	Linux Shell Command	Usage
Copy files	`cp`	`cp <filename> <new location>`
Move files	`mv`	`mv <filename> <new location>`
Rename files	`mv`	`mv <old filename> <new filename>`
Delete files	`rm`	`rm <filename>`
Create directories	`mkdir`	`mkdir <directory name>`
Delete directories	`rm`	`rm –rf <directory name>`
Change directory	`cd`	`cd <directory name>`
Edit text files	`vi / vim`	`vi <filename>`
View text files	`less`	`less <filename>`
Clear screen	`clear`	`clear`
Get help	`man`	`man <command>`
Quit	`exit`	`exit`

II.3 Tugas

- Praktekkan beberapa perintah *shell* pada tabel di atas. Gunakan semua kemungkinan pilihan yang tersedia pada masing-masing syntax.
- Gunakan perintah-perintah dasar di atas untuk memodifikasi file / direktori, seperti membuat file / direktori, menghapus, memindah, menduplikasi, dan sebagainya.
- Berikut beberapa perintah linux lebih lengkap:

Command	Description	Typical Command Options	Examples of Use
`$ alias`	Create or display command aliases		`alias list=ls`
`$ alsamixer`	Alter audio volume levels		`alsamixer`
`$ apropos`	Search man pages for specified words/phrases		`apropos "word or phrase"`
`$ apt-cache`	Search, query, and otherwise manipulate the APT database cache (see `apt-get`)	`search`: Search for specified package (regexes may be used; see Chapter 15) `showpkg`: Show information about specified package `depends`: Show package dependencies of specified package, and show other packages that can meet that dependency	`apt-cache search packagename`
`# apt-get`	Multifunction tool use to install, remove, and otherwise administer software packages, according to the APT database	`install`: search for and install specified package from repositories (as specified in `/etc/apt/sources.list`) `update`: Update or build package database by contacting package repositories `upgrade`: Attempt to upgrade all current installed packages with new versions `dist-upgrade`: Attempt to upgrade all currently installed packages, automatically and aggressively resolving package conflicts; often used to upgrade entire distro to new version `remove`: Opposite of `install`; removes packages `clean`: Remove any old package installation files that are stored on hard disk `-f`: Attempt to fix broken package dependencies (used with `install` or `remove`) `--force-yes`: Override any errors and thereby bypass `apt-get`'s protective measures. Dangerous option—use with care!	`apt-get install packagename`

Command	Description	Typical Command Options	Examples of Use
`$ bzip2`	Compress specified file (replaces original file with compressed file and gives it `.bz2` file extension)	`-d`: Decompress specified file `-k`: Don't delete original file `-t`: Test; do a dry run without writing any data	`bzip2 myfile`
`$ bzip2recover`	Attempt recovery of specified damaged `.bz2` file		`bzip2recover myfile.tar.bz2`
`$ cal`	Display calendar for current month (or specified month/year)		`cal 4 2005`
`$ cat`	Display a file on screen or combine and display two files together		`cat myfile`
`$ cd`	Change to specified directory		`cd /usr/bin`
`$ cdparanoia *`	Convert CD audio tracks to hard disk files	`-B`: Batch mode; convert all tracks to individual files `-S`: Set CD read speed (2, 4, 8, 12, and so on; values relate to CD-drive spin speed; used to avoid read errors)	`cdparanoia -S 8 -B`
`# cdrecord *`	Burn audio or CD-R/RW data discs (the latter usually based on an ISO image; see `mkisofs`)	`-dev=`: Specify the drive's device number (can be discovered by running `cdrecord` with the `scanbus` option) `--scanbus`: Scan to see which CD-R/RW drives are present and return device numbers `-speed=`: Specify the write speed (2, 4, 6, 8, and so on) `-v`: Verbose output; obligatory for feedback on `cdrecord`'s progress	`cdrecord dev=0,0,0 -speed=16 -v myfile.iso`
`# cfdisk *`	DANGEROUS! Menu-based disk-partitioning program		`cfdisk /dev/hda`
`# chgrp`	Change group ownership of a file/directory	`-R`: Recursive; apply changes to subdirectories	`chgroup mygroup myfile`
`$ chmod`	Change permissions of a file/directory (where `a`=all, `u`=user, `g`=group; and `r`=read, `w`=write, `x`=executable)	`-R`: Recursive; apply to subdirectories `--reference=`: Copy permissions from specified file	`chmod a+rw myfile`
`$ chown`	Change file ownership to specified username	`-R`: Recursive; apply to subdirectories	`chown username myfile1`
`# chroot`	Change the root of the file system to the specified path		`chroot /home/mydirectory`
`# chvt`	Switch to the specified virtual terminal (equivalent of holding down Ctrl+Alt and pressing F1–F6)		`chvt 3`
`$ clear`	Clears terminal screen and places cursor at top		`clear`

Command	Description	Typical Command Options	Examples of Use
`$ cp`	Copy files	`-r`: Recursive; copy subdirectories and the files therein `-s`: Create symbolic link instead of copying	`cp myfile1 directory/`
`$ crontab`	Edit or display the user's `crontab` file (scheduled tasks)	`-e`: Edit the `crontab` file (create/amend) `-l`: List `crontab` entries `-r`: Delete the `crontab` file `-u`: Specify a user and edit their `crontab` file	`crontab -e`
`$ date`	Display the date and time		`date`
`$ df`	Display free disk space within file system	`-h`: Human readable; display sizes in KB, MB, GB, and TB, as appropriate `-l`: Restrict to local file systems, as opposed to network mounts	`df -h`
`$ diff`	Display differences between specified files	`-a`: Consider all files text files (don't halt when asked to compare binary files) `-i`: ignore lowercase and uppercase differences	`diff myfile1 myfile2`
`$ diff3`	Display differences between three specified files		`diff3 myfile1 myfile2 myfile3`
`$ dig`	Look up IP address of specified domain		`dig mysite.com`
`$ dmesg`	Display kernel message log		`dmesg`
`# dosfsck *`	Check and repair MS-DOS–based file hard disk partition (see also `fsck`)	`-a`: Repair without asking user for confirmation `-r`: Repair file system asking user for confirmation when two or more repair methods are possible `-v`: Verbose; display more information	`dosfsck -rv /dev/hda4`
`# dpkg`	Install, remove, and otherwise administer local installation packages (on your hard disk); see also `apt-get`	`-i`: Install specified package `-r`: Remove (uninstall) specified package `-I`: Show info about specified package `--ignore-depends=packagename.deb`: Don't halt on package dependency issues (dangerous!)	`dpkg -i packagename.deb`
`# dpkg-reconfigure`	Reconfigure an already installed package		`dpkg-reconfigure packagename`

Command	Description	Typical Command Options	Examples of Use
`$ du`	Show sizes of files and folders in kilobytes	`-h`: Human readable; produce output in MB, GB, and TB `-s`: Summary; display totals only for directories rather than for individual files	`du -h /home/myuser`
`$ eject`	Eject a removable storage disk	`-t`: Close an already open tray	`eject /media/dvd-rom`
`$ ex *`	Start a simple text-editor program used principally within shell scripts		`ex myfile.txt`
`$ exit`	Log out of shell (end session)		`exit`
`$ fdformat`	Low-level format a floppy disk (this won't create a file system; see also *mkfs*)		`fdformat /dev/fd0`
`# fdisk *`	DANGEROUS! Hard-disk partitioning program	`-l`: List partition table	`fdisk /dev/hda`
`$ fg`	Brings job running in background to foreground		`fg 1`
`$ file`	Display information about specified file, such as its type		`file myfile`
`$ find *`	Find files by searching directories (starting in current directory)	`-maxdepth`: Specify the number of subdirectories levels to delve into, starting from 1 (current directory) `-name`: Specify name of file to search for `-type`: Specify file types to be returned; `-type d` returns directories and `-type f` returns only files	`find -name "myfile"`
`$ free`	Display information about memory usage	`-m`: Show figures in MB `-t`: Total the columns at bottom of table	`free -m`
`# fsck *`	Check file system for errors (usually run from rescue disc)		`fsck /dev/hda1`
`$ ftp *`	FTP program for uploading/downloading to remote sites		`ftp ftp.mysite.com`
`$ fuser`	Show which processes are using a particular file or file system	`-v`: Verbose; detailed output	`fuser -v myfile`
`$ grep *`	Search specified file for specified text string (or word)	`-i`: Ignore uppercase and lowercase differences `-r`: Recursive; delve into subdirectories (if applicable) `-s`: Suppress error messages about inaccessible files and other problems	`grep "phrase I want to find" myfile.txt`

Command	Description	Typical Command Options	Examples of Use
# groupadd	Create new group		groupadd *mygroup*
# groupdel	Delete specified group		groupdel *mygroup*
$ groups	Display groups the specified user belongs to		groups *myuser*
$ gzip	Compress files and replace original file with compressed version	-d: Decompress specified file -v: Verbose; display degree of compression	gzip *myfile*
# halt	Initiate shutdown procedure, ending all processes and unmounting all disks	-p: Power off system at end of shutdown procedure	halt -p
# hdparm *	DANGEROUS! Tweak or view hard disk settings		hdparm */dev/hda*
$ head	Print topmost lines of text files (default is first 10 lines)	-n: Specify number of lines (such as -n 5)	head *myfile.txt*
$ help	Display list of common BASH commands		help
$ history	Display history file (a list of recently used commands)		history
$ host	Query DNS server based on specified domain name or IP address	-d: Verbose; return more information -r: Force name server to return its cached information rather than query other authoritative servers	host *82.211.81.166*
$ hostname	Display localhost-style name of computer		hostname
$ id	Display username and group info of specified user (or current user if none specified)		id *myuser*
# ifconfig *	Display or configure settings of a network interface (assign an IP address, subnet mask, and activate/deactivate it)	down: Disable interface (used at end of command chain) netmask: Specify a subnet mask up: Enable interface (used at end of command chain)	ifconfig *eth0* *192.168.0.10* netmask *255.255.0.0* up
$ info *	Display info page for specified command		info *command*
# init	Change current run level		init *1*
$ jobs	Display list of jobs running in background		jobs
$ kill	Kill specified process		kill *1433*
$ killall	Kill process(es) that have specified name(s)	-i: Confirm before killing process -v: Verbose; report if and when successful	killall *processnumber*

Command	Description	Typical Command Options	Examples of Use
`$ last`	Display details of recent logins, reboots, and shutdowns		`last`
`$ ldd`	Display system files (libraries) required by specified program		`ldd /usr/bin/program`
`$ less`	Interactively scroll through a text file	`-q`: Quiet; disable beeps when end of file is reached or other error encountered `-i`: Ignore case; make all searches case-insensitive unless uppercase letters are used	`less myfile.txt`
`$ ln`	Create links to specified files, such as symbolic links	`-s`: Create symbolic link (default is hard link)	`ln -s myfile1 myfile2`
`$ lpr`	Print file (send it to the printer spool/queue)	`-V`: Verbose; print information about progress of print job	`lpr myfile.txt`
`$ lpstat`	Display print queue		`lpstat`
`$ ls`	List directory	`-a`: List all files, including hidden files `-d`: List only directory names rather than their contents `-h`: Human readable; print figures in KB, MB, GB, and TB `-l`: Long list; include all details, such as file permissions `-m`: Show as comma-separated list	`ls -h mydirectory`
`# lsmod`	Display currently loaded kernel modules		`lsmod`
`$ lsof`	Display any files currently in use	`-u`: Limit results to files used by specified user	`lsof -u username`
`$ man`	Display specified command's manual		`man command`
`$ md5sum`	Display MD5 checksum (normally used to confirm a file's integrity after download)		`md5sum myfile`
`# mkfs *`	DANGEROUS! Create specified file system on specified device (such as a floppy disk)	`-t`: Specify type of file system	`mkfs -t vfat /dev/fd0`

Command	Description	Typical Command Options	Examples of Use
`$ mkisofs *`	Create ISO image file from specified directory (usually for burning to disc with `cdrecord`)	`-o`: Options; this must appear after command to indicate that command options follow `-apple`: Use Mac OS extensions to make disc readable on Apple computers `-f`: Follow symbolic links and source actual files `-J`: Use Joliet extensions (make ISO compatible with Windows) `-R`: Use Rock Ridge extensions (preferred Linux CD-ROM file system) `-v`: Verbose; display more information (`-vv` for even more info)	`mkisofs -o isoimage.iso -R -J -v mydirectory`
`# modinfo`	Display information about kernel module		`modinfo modulename`
`# modprobe`	Insert specified module into the kernel, as well as any others it relies on	`-k`: Set module's `autoclean` flag so it will be removed from memory after inactivity `-r`: Remove specified module as well as any it relies on to operate	`modprobe modulename`
`$ more`	Interactively scroll through text file (similar to `less`)		`more myfile.txt`
`# mount *`	Mount specified file system at specified location	`-o`: Specify command options, such as `rw` to allow read/write access; various types of file systems have unique commands	`mount /dev/hda4 /mnt`
`$ mv`	Move (or rename) specified files and/or directories	`-b`: Back up files before moving `-v`: Display details of actions carried out	`mv myfile mydirectory/`
`$ netstat *`	Show current network connections		`netstat -a`
`$ nice`	Run specified command with specified priority	`-n`: Specify priority, ranging from the highest priority of –20, to 19, which is the lowest priority	`nice -n 19`
`$ nohup`	Run specified command and continue to run it, even if user logs out		`nohup command`
`$ passwd`	Change user's password		`passwd`
`$ ping`	Check network connectivity between local machine and specified address	`-w`: Exit after specified number of seconds (such as `-w 5`)	`ping mydomain.com`
`$ printenv`	Display all environment variables for current user		`printenv`

Command	Description	Typical Command Options	Examples of Use
`$ ps *`	Display currently running processes	`a`: List all processes (note that command options don't require preceding dash) `f`: Display ownership of processes using tree-style graphics `u`: Limit results to processes running for and started by current user `x`: Include processes in results not started by user but running with the user ID	`ps aux`
`$ pwd`	Display current directory		`pwd`
`# reboot`	Reboot computer		`reboot`
`$ renice`	Change a process's priority while it's running (see `nice`)		`renice 19 10704`
`$ rm`	Delete single or multiple files and/or directories	`-r`: Recursive; delete specified directories and any subdirectories `-f`: Force; don't prompt for confirmation before deleting (use with care!)	`rm -rf mydirectory`
`# rmmod`	Delete module from kernel		`rmmod modulename`
`# route *`	Add and create (or view) entries in routing table (see `ifconfig`)		`route add default gw 192.168.1.1`
`$ runlevel`	Display current run level		`runlevel`
`$ screen *`	Program that runs pseudo shell that is kept alive regardless of current user login	`-ls`: Display list of currently running screen sessions `-R`: Reattach to already running screen session or start new one if none available	`screen`
`$ sftp *`	Secure Shell FTP; like FTP but running over an `ssh` connection (see `ssh`)		`sftp username @192.168.1.14`
`$ shred`	Overwrite data in a file with gibberish, thereby making it irrecoverable	`-u`: Delete file in addition to overwriting `-v`: Verbose; show details of procedure `-f`: Force permissions to allow writing if necessary	`shred -fv myfile`
`$ sleep`	Pause input for the specified period of time (where `s`=seconds, `m`=minutes, `h`=hours, `d`=days)		`sleep 10m`
`$ smbclient *`	FTP-style program with which you can log in to a SMB (Windows)-based file share		`smbclient //192.168.1.1/`

Command	Description	Typical Command Options	Examples of Use
`$ sort`	Sort entries in the specified text file (default is ASCII sort)		`sort myfile.txt -o sorted.txt`
`$ ssh *`	Log in to remote computer using secure shell		`ssh username@ 192.168.1.15`
`$ startx`	Start GUI session (if GUI isn't already running)		`startx`
`$ su`	Temporarily log in as specified user; log in as root if no user specified (provided root account is activated)	`-`: Adopt user's environment variables, such as `$PATH`	`su`
`$ sudo`	Execute specified command with root privileges		`sudo command`
`$ tac`	Display specified text file but in reverse (from last to first line)		`tac myfile.txt`
`$ tail`	Display final lines of specified text file	`-n`: Specify number of lines to display (such as `-n4`)	`tail myfile.txt`
`$ tar *`	Combine specified files and/or directories into one larger file, or extract from such a file	`-c`: Create new archive `-j`: Use `bzip2` in order to compress (or decompress) files `-f`: Specifies filename (must be last in chain of command options) `-r`: Add files to existing archive `-x`: Extract files from existing archive `-z`: Use `gzip` to compress (or decompress) files	`tar -zcf myfile. tar.gz mydirectory`
`$ tee`	Display piped output and also save it to specified file		`ls -lh| tee listing.txt`
`$ top *`	Program that both displays and lets the user manipulate processes		`top`
`$ touch`	Give specified file current time and date stamp; if it doesn't exist, create a zero-byte file with that name		`touch myfile`
`$ tracepath`	Discover and display network path to another host		`tracepath 192.168.1.20`
`$ umask`	Set default permissions assigned to newly created files		`umask u=rwx,g=r,o=`
`# umount`	Unmount a file system		`umount /media/cdrom`
`# useradd`	Add new user	`-m`: Create home directory for user	`useradd -m username`

Command	Description	Typical Command Options	Examples of Use
`# userdel`	Delete all mention of user in system configuration files (effectively deleting the user, although files owned by the user might remain)	`-r`: Remove user's home directory	`userdel -r username`
`$ unalias`	Remove specified `alias`	`-a`: Remove all aliases (use with care!)	`unalias command`
`$ uname`	Display technical information about current system	`-a`: Display all basic information	`uname -a`
`$ unzip`	Unzip a Windows-compatible Zip file	`-l`: Display archive content but don't actually unzip	`unzip myfile.zip`
`$ uptime`	Display uptime for system, as well as CPU load average and logged-in users		`uptime`
`$ vim *`	Text editor program		`vim`
`$ wc`	Count the number of words in a file		`wc myfile.txt`
`$ whatis`	Display one-line summary of specified command		`whatis command`
`$ whereis`	Display information on where a binary command is located, along with its source code and man page (if applicable)	`-b`: Return information only about binary programs	`whereis -b command`
`$ xhost`	Configure which users/systems can run programs on the X server	`+`: When followed by a username and/or system name, gives the user/system permission to run programs on the X server; when used on its own, lets *any* user/system use the X server `-`: Opposite of `+`	`xhost +`
`$ xinit`	Start elementary GUI session (when not already running a GUI)		`xinit`
`$ zip`	Create Windows-compatible compressed Zip files	`-r`: Recursive; includes all subdirectories and files therein `-u`: Updates Zip with specified file `-P`: Encrypts Zip with specified password `-v`: Verbose; display more information `-#`: Set compression level (from 0, which is no compression, to 9, which is highest)	`zip -r myfile.zip mydirectory`
`$ zipgrep`	Searches inside Zip files for specified text string		`zipgrep "search phrase" myfile.zip`

-ooOoo-

PRAKTIKUM III

FILE EDITOR

Tujuan Instruksional Umum:

- Setelah mengikuti praktikum ini, mahasiswa diharapkan mampu menggunakan *file editor* berbasis konsole pada Linux.

Tujuan Instruksional Khusus:

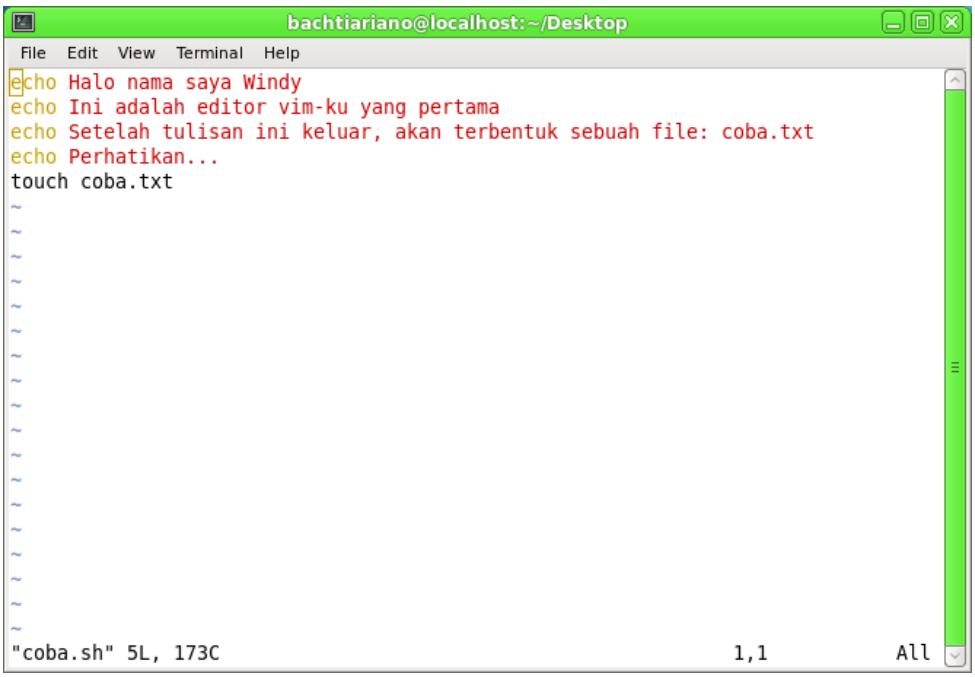
- Mahasiswa mampu mengoperasikan beberapa *file editor* seperti vi / vim, nano, atau gedit.
- Mahasiswa mampu memanfaatkan *file editor* tersebut untuk keperluan pengubahan suatu file.

III.1 Pendahuluan

Jalankan apikasi terminal melalui: menu Application → Accessories → Terminal. Lalu ketiklah perintah:

`vi coba.sh` (membuka file coba.txt menggunakan editor vi) atau
`vim coba.sh` (membuka file coba.txt menggunakan editor vim)

Catatan: Perbedaan penggunaan vi dan vim ada pada file */usr/share/vim/vim72/doc/vi_diff.txt.gz*



Tulislah kalimat seperti pada tampilan di atas. Gunakan panduan pada tabel berikut ini untuk menggunakan editor vi / vim.

Command	Description
Delete Text	
dd	Delete current line
ndd	Delete n number of lines (for example, 5dd will delete five lines)
dw	Delete the current word under the cursor
db	Delete the word before the cursor
D	Delete everything from the cursor to the end of the line
Search	
/	Search forward (type the search text directly after the slash)
?	Search backward
n	Repeat search in a forward direction
N	Repeat search in a backward direction
Cut and Paste	
yy	Copy the current line
nyy	Copy n number of lines into the buffer from the cursor downwards (for example, 5yy copies five lines of text)
p	Paste the contents of the clipboard
Insert Text	
i	Switch to Insert mode at the cursor
o	Switch to Insert mode, placing the cursor below current line
O	Switch to Insert mode, placing the cursor above current line
A	Append text to end of line
Navigation	
$	Move the cursor to the end of the current line
w	Move the cursor to the next word
b	Move the cursor to beginning of the current or previous word
Miscellaneous	
.	Repeat the last command
u	Undo the last command
:w	Save the file
:w!	Save the file and ignore errors such as an existing file with the same filename
:q	Quit vim
:q!	Quit vim and ignore errors such as an unsaved file
:s/word/replacement/	Search from the cursor downwards and replace any instances of the **word** with the **replacement**
:help	View help documentation

III.2 Tugas

- Buatlah *executable file* dengan menyimpan file dengan ekstensi .sh, lalu jalankan file tersebut melalui terminal dengan perintah: **`./namafile`** lalu <enter>.

- Gunakan perintah-perintah yang telah dipelajari pada Praktikum II.

-ooOoo-

PRAKTIKUM IV

PASCAL PROGRAMMING

Tujuan Instruksional Umum:

- Setelah mengikuti praktikum ini, mahasiswa diharapkan mengenal pemrograman tingkat dasar dan dapat mempraktekkannya menggunakan salah satu bahasa pemrograman pada sistem operasi Linux.

Tujuan Instruksional Khusus:

- Mahasiswa mampu mengenal bahasa pemrograman Pascal dan dapat mengoperasikannya menggunakan editor file yang telah dipelajari pada praktikum sebelumnya.
- Mahasiswa mampu menggunakan bahasa pemrograman Pascal untuk memecahkan berbagai permasalahan pemrograman.

IV.1 Pendahuluan

Bahasa pemrograman pascal di rancang oleh Niklaus Wirth seorang ahli komputer dari Swiss pada tahun 1970, kemudian dibakukan pada tahun 1974 sebagai Pascal Standard dengan nama Pascal User manual dan report. Turbo Pascal dikembangkan oleh Borland Internasional, Inc sampai.

Struktur program Pascal terdiri dari :

```
Program Nama _Program; {bagian judul, bersifat optional}
Uses …          {deklarasi unit}
Label …         {deklarasi label, pernyataan ini tidak dianjurkan karena struktur program menjadi
                tidak jelas}
Const …         {definisi konstanta, suatu nilai data yang bersifat tetap selama program
                dijalankan}
Type …          {definisi tipe data di luar tipe data standar Pascal}
Var …           {deklarasi variabel, tempat untuk menyimpan data dalam main memory komputer}

Procedure Nama_Prosedur;        {bagian subprogram prosedur}
Function Nama_Fungsi;           {bagian subprogram fungsi}

Begin  {awal program}
…….;
[statemen-statemen dalam program};
……;
end.
```

IV.2 Tipe Data dalam Pascal

Tipe data sederhana dalam program Pascal mempunyai batasan-batasan sbb:

a. Batasan nilai data pada bilangan bulat / integer

Tipe Data	Batasan Nilai	Ukuran dalam byte
byte	0 … 255	1
shortint	-128 … 127	1
integer	-32768 ... 32757	2
word	0 … 65535	2
longint	-2147483648 … 2147483648	4

b. Batasan nilai data pada bilangan pecahan / real

Tipe Data	Batasan Nilai	Ukuran dalam byte
real	2.9×10^{-39} … 1.7×10^{38}	6
single	1.5×10^{-45} … 3.4×10^{38}	4
double	5.0×10^{-324} … 1.7×10^{308}	8
extended	3.4×10^{-4932} … 1.1×10^{4932}	10
comp	$-2^{63} + 1$ … $2^{63} - 1$	8

c. Data boolean, char, dan string

Tipe Data	Keterangan
boolean	Kondisi TRUE, FALSE
char	Sebuah karakter, misal ‘A’,’B’, ‘1’, ‘^’, dll
string	Merupakan rangkaian karakter, misal : ‘Bapak’, ‘Senin’, dll

IV.2 Struktur Pencabangan dalam Pascal

Pada bahasa pemrograman pascal, terdapat 2 jenis struktur pencabangan yang bisa digunakan, yaitu :

1. Statement IF

a. IF_THEN_

Bentuk umum:

```
IF syarat logika THEN statemen;
```

b. IF_THEN_ELSE_

Bentuk umum:

```
IF syarat logika THEN
   statemen 1
ELSE
   statemen 2;
```

c. IF_THEN_ELSE IF…

Bentuk umum:

```
IF syarat logika THEN
   statemen 1
ELSE IF syarat logika 2 THEN
   statemen 2
   …
ELSE
   statemen n;
```

2. Statement CASE

a. CASE_OF_

Bentuk umumnya adalah:

```
CASE Nama_Var OF
        alternatif_1 : Statemen yang diproses jika alternatif_1 benar;
        alternatif_2 : Statemen yang diproses jika alternatif_2 benar;
        ……
        alternatif_n : Statemen yang diproses jika alternatif_n benar;
End;
```

Pernyataan yang dikerjakan CASE_OF_ ini hanya 1 proses saja, setelah itu keluar dari pernyataan CASE.

b. CASE_OF_ELSE_

Bentuk umumnya adalah :

```
CASE Nama_Var OF
        alternatif_1 : Statemen yang diproses jika alternatif_1 benar;
        alternatif_2 : Statemen yang diproses jika alternatif_2 benar;
        ……
        alternatif_n : Statemen yang diproses jika alternatif_n benar;
ELSE
   Statemen yang diproses jika peluang di atas tidak ada yang benar;
End;
```

Pada bentuk CASE_OF_ELSE_, jika tidak ada alternatif yang memenuhi syarat maka statemen sesudah ELSE yang dikerjakan.

IV.3 Struktur Pengulangan dalam Pascal

Untuk struktur pengulangan terdapat beberapa jenis struktur statement, yaitu:

a. Statemen WHILE_DO_

Bentuk umumnya:

```
WHILE Kondisi DO
Begin
... Statemen ...
End;
```

Statemen ini digunakan untuk mengulang satu / lebih statemen dalam **begin … end** sampai kondisi yang dinyatakan dalam pernyataan **WHILE** tidak terpenuhi.

b. Statemen REPEAT_UNTIL_

Bentuk umumnya:

```
REPEAT
... Statemen ...
UNTIL Kondisi;
```

Dalam statemen ini, proses berulang akan dilaksanakan terus selama kondisi belum terpenuhi.

c. Statemen FOR_TO_DO_(FOR_DOWNTO_DO_)

Bentuk umumnya:

```
FOR var := Bil1 TO Bil2
        Begin
        ... Statemen ...
        End;
```

atau

```
FOR var := Bil1 DOWNTO Bil2
        Begin
        ... Statemen ...
        End;
```

Berbeda dengan WHILE_DO dan REPEAT_UNTIL, jumlah pengerjaan terhadap sejumlah statemen dalam **begin … end** dapat diketahui dengan pasti.

IV.4 Tugas

Buatlah file sumber yang extensi .pas dengan editor kesukaan Anda. Untuk meng-*compile* file tersebut, gunakan perintah: **`fpc <namafile>`** lalu enter.

Kemudian praktekkan contoh-contoh program berikut ini:

- **Contoh 1**

```
Program pertama;
Uses crt;
Begin
   Writeln('Ini program pascal pertamaku ');
   Readln;
End.
```

- **Contoh 2**

```
Program jumlah_kali;
Uses crt;
Var A,B,C,D : integer;
Begin
   A := 5; B := 10;
   C := A + B;
   D := A * B;
   Writeln('Hasil dari ',A,' + ',B,' = ',C);
   Writeln('Hasil dari ',A,' x ',B,' = ',D);
   Readln;
End.
```

- **Contoh 3**

```
Program hari;
Uses crt;
Var kode : integer;
Begin
   Write('Masukkan Kode Hari: ');readln(kode);
   Case kode of
     1 : writeln('Kode hari ',kode,' = Minggu');
     2 : writeln('Kode hari ',kode,' = Senin');
     3 : writeln('Kode hari ',kode,' = Selasa');
     4 : writeln('Kode hari ',kode,' = Rabu');
     5 : writeln('Kode hari ',kode,' = Kamis');
     6 : writeln('Kode hari ',kode,' = Jum at');
     7 : writeln('Kode hari ',kode,' = Sabtu');
   end;
   Readln;
End.
```

- **Contoh 4**

```
Program rata_rata;
Uses crt;
Var i,n,jml,x : integer;
    rata : real;
Begin
   Write('Banyaknya data: ');readln(n);
   Jml := 0;
   For i := 1 to n do
     Begin
       Write('Data ke- ',i);readln(x);
       Jml := jml + x;
     End;
   Rata := jml / n;
   Writeln('Rata – rata : ',rata:6:2);
   Readln;
End.
```

SOAL LATIHAN

1. Buatlah program untuk menghitung panjang sisi miring suatu segitiga siku-siku, gunakan rumus phytagoras : $c^2 = a^2 + b^2$
2. Ubah program hari pada **Contoh 3** di atas menggunakan statement **IF** untuk struktur pencabangannya
3. Ubah program rata_rata pada **Contoh 4** di atas menggunakan statement **WHILE…DO** atau **REPEAT UNTIL** (pilih salah satu) untuk proses pengulangan pemasukan datanya.

-ooOoo-