# Praktikum I
# Perintah Dasar Linux

## I. Tujuan

1. Mengenal sistem operasi GNU/Linux
2. Memahami perintah-perintah dasar GNU/Linux
3. Mampu mengoperasikan GNU/Linux pada mode terminal

## II. Dasar Teori

Command line atau baris perintah adalah instruksi-instruksi yang disediakan oleh OS yang dieksekusi dari terminal dengan cara mengetikkan perintah dan diakhiri “enter”. Meskipun dewasa ini GNU/Linux sudah memiliki desktop environment yang menawan, contohnya GNOME dan KDE, akan tetapi command line tidak bisa ditinggalkan. Command line merupakan cara yang lebih efisien untuk melakukan suatu pekerjaan, oleh karena itu pengguna GNU/Linux masih menggunakan cara ini untuk bekerja. Oleh karena itu pada praktikum kali ini di awali dengan perintah dasar sebagai dasar pada untuk keseluruhan praktikum sistem operasi. Berikut adalah contoh perintah dasar yang sering dipakai oleh pengguna linux.

contoh :

```
praktikum:$  ls -l
total 88
drwxr-xr-x   2 root root  4096 Feb  7 12:23 bin
drwxr-xr-x   3 root root  4096 Feb  6 20:17 boot
drwxr-xr-x  17 root root  3380 Mar 12 17:28 dev
drwxr-xr-x 116 root root  4096 Mar 12 18:05 etc
drwxr-xr-x   3 root root  4096 Feb  6 20:35 home
.................
```

Contoh diatas “ls” merupakan perintah untuk menampilkan konten dari sebuah direktori aktif, dengan opsi “-l” (use a long listing format). Untuk memudahkan pengguna dalam menggunakan perintah baris disediakan pula sebuah bantuan yang berisi informasi lengkap tentang perintah dan opsi-opsinya. Cara untuk mengakses bantuan terhadap sebuah perintah sebagai berikut contoh:

```
praktikum:$ man ls
LS(1)                   User Commands                   LS(1)
NAME
    ls - list directory contents
SYNOPSIS
    ls [OPTION]... [FILE]...
DESCRIPTION
    List  information  about  the FILEs (the current directory by default).
    Sort entries alphabetically if none of -cftuvSUX nor --sort.
    Mandatory arguments to long options are  mandatory  for  short options  too.
    -a, --all do not ignore entries starting with.
    ..............................
```

Banyak sekali baris perintah pada GNU/Linux pada praktikum ini akan dipelajari beberapa yang sering dan sangat berguna jika kita mengoperasikan sistem GNU/Linux.

### Syntax Perintah Linux

Secara umum perintah-perintah Linux dan UNIX memiliki sintaks sebagai berikut:
Perintah [option...] [argumen...]

Keterangan :

- Option merupakan pilihan yang dapat kita gunakan untuk memberikan hasil tertentu dari suatu perintah.
- Argumen merupakan sesuatu yang akan diproses oleh perintah, misalnya nama file atau nama direktori.
- Tanda kurung siku ([ ]) merupakan simbol bahwa option dan argumen tidak harus selalu digunakan dalam menjalankan perintah.
- Tanda titik-titik (&) menandakan bahwa baik option maupun argumen dapat lebih dari satu.

*Note : Semua perintah dalam Linux bersifat case sensitive, artinya huruf besar dan kecil berbeda artinya. Jadi LS akan dianggap berbeda dengan ls.*

**III. Tugas Pendahuluan**

1. Uraikan apa itu runlevel dan bagaiman mengoperasikan, tunjukkan seorang user berada pada suatu runlevel.
2. Jelaskan langkah-langkah dari boot dan start up proses.

**IV. Petunjuk Praktikum**

1. Login ke sistem GNU/Linux kemudian buka terminal.
2. Lakukan pelajari dan lakukan percobaan terhadap perintah-perintah berikut :

**Login**

Untuk masuk kedalam sistem linux, Anda harus melakukan proses login, yaitu dengan cara memasukkan nama user dan password.
Contoh.

```
praktikum-PC login: smuet
Password:
Last login: Rab Agu 27 23:00:32 WIT 2011 on console
Linux praktikum-PC 2.6.24-19-generic #1 SMP Fri Jul 11
23:41:49 UTC 2011 i686
The programs included with the Ubuntu sistem are free
software;
.....
.....
smuet@praktikum-PC:~$
```

**Logout**

Untuk keluar dari user yang sedang login, anda dapat mengetikkan perintah logout
Contoh:

```
smuet@praktikum-PC:~$ logout
```

**Virtual Terminal**

Beberapa user dapat melakukan login pada sebuah PC atau seorang user dapat melakukan beberapa kali login yang sama pada sebuah PC. Hal ini dapat dilakukan dengan mengunakan terminal virtual.
Untuk berganti terminal dugunakan tombol : Alt+Fx, x adalah nomor terminal.
Contoh : jika saat ini anda berada pada mode teks dan pada terminal 1 (F1), maka
untuk pindah ke terminal lainya, tekan tombol:
Alt+F2 : pindah ke terminal 2
Alt+F3 : pindah ke terminal 3
Alt+F4 : pindah ke terminal 4

Alt+F5 : pindah ke terminal 5
Alt+F6 : pindah ke terminal 6
Alt+F7 : pindah ke mode grafik. Pada mode teks, hal ini tidak dapat berfungsi kecuali mode grafik (X window) sudah diaktifkan.

**Mengetahui Posisi Virtual Terminal**
Untuk mengetahi posisi virtual terminal anda dapat menggunakan perintah **tty**.
Contoh:

```
smuet@praktikum-PC:~$ tty
/dev/tty3
```

Ini berarti anda berada pada terminal 3.

**Merestart Sistem**
Untuk merestart sistem , anda dapat menggunakan perintah rebbot dan init 6
Contoh:

```
root@praktikum-PC:~# reboot
root@praktikum-PC:~# init 6
```

**Mematikan Sistem**
Untuk mematikan sistem anda dapat menggunakan perintah shutdown, halt, init 0 atau power off.
Contoh:

```
root@praktikum-PC:~# shutdown
root@praktikum-PC:~# halt
root@praktikum-PC:~# init 0
root@praktikum-PC:~# power off
```

**Membatalkan Perintah**
Anda dapat membatalkan sebuah perintah yang anda berikan pada saat sistem sedang memproses-nya. Untuk membatalkan proses, tekan tombol **Ctrl+c** atau **Ctrl+z.**

**Info**
Membaca dokumentasi dari sebuah perintah.
Format: info perintah-yang-diinginkan
Contoh:

```
smuet@praktikum-PC:~$ info ls
```

*note: Tidak semua distro menyediakan perintah info*

**Whatis**
Digunakan untuk mendapatkan informasi dari perintah secara singkat.
Format: whatis perintah-yang-diinginkan
Contoh:

```
smuet@praktikum-PC:~$ whatis ls
```

*note :Beberapa distro mengharuskan mengetikkan perintah makewhatis sebelum dijalankan, makewhathis berfungsi mengaktifkan database yang akan digunakan oleh whathis.*

**Apropos**
Mencari informasi secara massal. Perintah ini berguna jika anda tidak tahu persis perintah yang dimaksudkan atau jika hanya mengetahui sebagian dari perintah.
Format: apropos perintah-yang-diinginkan

Contoh:

smuet@praktikum-PC:~$ apropos ls

Perintah ini akan menampilkan semua perintah yang mengandung kata huruf ls

**Help Option**

Yang dimaksudkan dengan help option adalah sebuah option yang dapat digunakan untuk mendapatkan informasi singkat dari sebuah perintah.
Format: perintah-yang-diingikan #help
Contoh;

smuet@praktikum-PC:~$ ls #help

**Informasi Sistem**

Menampilkan informasi sistem. Sistem yang dimaksud adalah versi kernel yang digunakan, sistem operasi, platform dan lainnya. Perintah yang digunakan adalah **uname**
format: uname option
Contoh;

smuet@praktikum-PC:~$ uname -r
2.6.24-19-generic

**Melengkapi Perintah**

Command completion atau melengkapi perintah yang diketik adalah sebuah fitur yang disediakan sistem untuk memudahkan user dalam mengetikkan sebuah perintah. Dengan fitur ini user tidak perlu mengetikkan semua kalimat perintah, tetapi cukup mengetikkan beberapa huruf awal saja dari sebuah perintah lalu dengan menekan tombol tab maka sistem akan melengkapi perintah yang diinginkan.
Contoh:

smuet@praktikum-PC:~$ rm [tekn tombol tab]
rm rmic rmiregistry rmt
rmdir rmid rmmod rmt-tar

jika anda ingin menjalankan perintah rmdir, maka tambahkan huruf d pada perintah diatas menjadi:

smuet@praktikum-PC:~$ rmd [tekan tombol tab]

maka secara otomatis huruf rm akan dilengkapi.

**Perintah yang telah dilakukan**

Perintah-perintah yang pernah anda ketikkan sebelumnya masih dapat dilihat dan digunakan kembali. Untuk melihat perintah yang lalu, gunakan tombol panah atas atau bawah. Setelah anda menemukan perintah yang diinginkan, cukup tekan enter untuk menggunakannya. Secara default perintah yang tersimpan dalam history adalah 1000 perintah. untuk melihat history dari perintah, ketikkan perintah history

smuet@praktikum-PC:~$ history

**Mencari Perintah yang telah dilakukan**

Jika history dari perintah terlalu banyak, anda dapat mempercepat pencarian dengan menggunakan perintah : **Ctrl+r**
contoh:
smuet@praktikum-PC:~$ tekan Ctrl dan r, akan muncul :
(reverse-i-search)`#: ketikkan 2-3 karakter dari perintah yang dicari.
(reverse-i-search)`ad#: sudo wifi-radar kemudian tekan enter.

**Menghapus Perintah yang telah dilakukan**
History dari perintah dapat dihapus dengan menggunakan perintah **history -c**
Contoh :

smuet@praktikum-PC:~$ history -c

**Menggunakan Bantuan**
Jika mengalami kesulitan menggunakan perintah yang ada, anda dapat memanfaatkan perintah-perintah bantu yang disediakan oleh linux membantu anda. Perintah yang dapat digunakan: **man, info, whatis, apropos.**

**Menampilkan Tanggal**
Untuk menampilkan tanggal gunakan perintah date atau cal date untuk menampilkan tanggal dan waktu sistem. format: **date [**option...**]**
Contoh:

smuet@praktikum-PC:~$ date
Kam Agu 28 09:31:30 WIT 2008

**cal** untuk menampilkan kalender
format : **cal** [option..]
Contoh :

smuet@praktikum-PC:~$ cal -y

**Menampilkan Identitas Komputer**
Identitas komputer yang dimaksud adalah nama komputer yang sedang digunakan. Perintahnya adalah **hostname.** format: hostname options
Contoh:

smuet@praktikum-PC:~$ hostname
praktikum-PC

**Manajemen File dan Direktori**
**Menampilkan Direktori Aktif**
Direktori aktif adalah letak direktori tempat anda bekerja saat ini. Misalnya jika saat ini anda berada pada direktori data, maka direktori data disebut direktori aktif. Sebenarnya, tanpa menggunakan sebuah perintah pun kita dapat mengetahui posisi direktori aktif. Contohnya, jika terminal anda menunjukkan seperti :

root@praktikum- PC:/home#

maka posisi direktori aktif berada pada direktori home. Tetapi, terkadang model tampilan terminal tidak seperti pada Contoh sehingga menyulitkan untuk mengetahui posisi direktori aktif. Untuk itu digunakan perintah **pwd**
format : **pwd**
Contoh:

root@praktikum-PC:~/Desktop# pwd

**Melihat Isi Direktori**
Anda dapat melihat isi dari direktori aktif atau isi direktori lain dengan menggunakan perintah **ls**.
Format : ls
Contoh:

smuet@praktikum-PC:~$ ls
data Documents flieq~ NetBeansProjects Public Videos
Desktop Examples Music Pictures Templates

**Membuat Direktori**
Anda juga dapat membuat direktori agar memudahkan anda dalam mengatur file-file. Perintah yang digunakn adalah **mkdir**.
Format: **mkdir** [option..] nama-direktori-baru
Contoh:

```
smuet@praktikum-PC:~$ mkdir iso
```

Kita bisa menggunakan perintah berikut yang akan secara otomatis membuat semua direktori yang diperlukan dalam rangka membuat direktori data1.

```
smuet@praktikum-PC:~$ mkdir -p /home/smuet/data/data1
```

*note : secara default sebuah direktori ditandai dengan warna biru.*

**Berpindah Direktori**
Dalam sistem linux banyak sekali terdapat direktori. Anda dapat berpindah-pindah ke direktori-direktori tersebut dengan menggunakn perintah **cd**.
Format: **cd** direktori-yang-dituju
Contoh:

```
smuet@praktikum-PC:~$ cd /boot/grub/
```

*note: Perhatikan bahwa tanda / (slash) tidak selalu harus disertakan. Penggunaanya tanda / disesuai-kan dengan letak direktori*

**Menghapus Direktori**
Direktori dapat dihapus dengan dengan menggunakan perintah **rmdir**
format: rmdir option direktori-yang-akan-dihapus
Contoh:

```
smuet@praktikum-PC:~$ rmdir iso/
```

Jika dalam direktori yang dihapus terdapat file, maka akan terjadi kesalahan dengan pesan ,

```
#rmdir: failed to remove `iso`: Directory not empty #
```

untuk mengatasinya digunakan perintah **rm -rf**.

**Membuat File**
Anda dapat membuat sebuah file dengan perintah touch. File yang dihasilkan dengan perintah ini adalah file kosong yang tidak berisi apa-apa.
Format : touch option nama-file-baru
Contoh:

```
smuet@praktikum-PC:~$ touch fileQ
```

**Mengcopy File**
Jika diperlukan, anda dapat mengcopy sebuah file dengan menggunakan perintah cp.
Format : cp option file-asli file-kopian
Contoh :

```
smuet@praktikum-PC:~$ cp fileQ fileQ_backup
```

**Memindahkan File**
Sebuah file dapat dipindahkan kedirektori lain jika diperlukan. Gunakan perintah mv.
Format : mv option nama-file lokasi-baru
Contoh:

smuet@praktikum-PC:~$ mv fileQ iso/

Perintah tersebut akan memindahkan file yang bernama fileQ kedalam direktori iso. Anda juga dapat memindahkan sebuah fiel sekaligus memberikan nama baru pada file tersebut.
Contoh:

smuet@praktikum-PC:~$ mv fileQ iso/file3

**Menghapus File**
Untuk menghapus file digunakan perintah rm.
Format : rm option nama-file
Contoh:

smuet@praktikum-PC:~$ rm iso/file3

**Menampilkan Isi File**
Isi sebuah file dapat dilihat dengan menggunakan perintah cat.
Format : cat option file-yang-ingin-dilihat
Contoh:

smuet@praktikum-PC:~$ cat /boot/grub/menu.lst

File yang mudah ditampilkan adalah file bejenis teks. Jika anda mencoba menampilkan file biner, maka yang tampak adalah karakter-karakter yang sulit dipahami. Kita bisa membuat file teks baru dengan hanya menggunakan perintah cat dan melakukan redirect dari standard input (keyboard) ke file. Misalnya untuk membuat file data.txt, gunakan perintah sebagai berikut:

smuet@praktikum-PC:~$ cat > data.txt

ketikkan isi file di sini lalu akhiri dengan menekan tombol Ctrl dan D secara bersamaan Anda bisa menggabungkan isi beberapa file menjadi satu file. Misalnya anda beberapa memiliki file dengan nama data.txt & tutorial.txt. Anda ingin mengabungkan seluruh file tersebut pada sebuah file bernama ebook.txt, maka perintah yang digunakan ialah:

smuet@praktikum-PC:~$ cat data.txt tutorial.txt > ebook.txt

**grep**
Menampilkan semua baris yang mengandung pola yang diinginkan:
Contoh :

smuet@praktikum-PC:~$ grep basic ebook.txt

**Perintah more dan less**
**more** atau **less** digunakan untuk menampilkan isi sebuah file. Perbedaan more dan less adalah, more hanya dapat menampilkan secara perlayar dan tidak dapat melihat layar yang sudah ditmpilkan. Sedangkan less dapat menampilkan secara PageUp da PageDown.
Format : more option file
Contoh :

smuet@praktikum-PC:~$ more /boot/grub/menu.lst
.....
#Lebih#(18%)

untuk melihat tampilan selanjutnya tekan tombol space (spasi).
Format : less option file
Contoh:

smuet@praktikum-PC:~$ less /boot/grub/menu.lst
.....
.....
/boot/grub/menu.lst

untuk melihat tampilan selanjutnya tekan tombol:
PageUp-PageDown atau PanahAtas-PanahBawah.

**Mencari File**
Sangat mungkin terjadi anda lupa dimana letak file yang dibutuhkan. Linux menyediakan beberapa perintah yang dapat digunakan untuk mencari file. Perintah yang akan dibahas adalah **find**, **locate**, **which** dan **whereis**.

**Find**
format : **find** lokasi-perkiraan nama-file [option...]
Contoh:

```
smuet@praktikum-PC:~$ find /etc/apache2/ -name *.conf
/etc/apache2/mods-enabled/negotiation.conf
/etc/apache2/mods-enabled/setenvif.conf
/etc/apache2/mods-enabled/mime.conf
/etc/apache2/mods-enabled/alias.conf
/etc/apache2/mods-enabled/dir.conf
/etc/apache2/mods-enabled/cgid.conf
/etc/apache2/mods-enabled/autoindex.conf
```

perintah tersebut akan mencari semua nama file yang berjenis conf. Pencarian akan dilakukan di direktori /etc/apache2/

**Locate**
format : **locate** [option..] nama-file
Contoh:

```
smuet@praktikum-PC:~$ locate fileq
/home/smuet/.icons/black-white_2-
Neon/scalable/actions/filequickprint.png
/usr/share/icons/Tangerine/16◊16/actions/filequickprint.png
/usr/share/icons/Tangerine/22◊22/actions/filequickprint.png
/usr/share/icons/Tangerine/24◊24/actions/filequickprint.png
/usr/share/icons/Tangerine/32◊32/actions/filequickprint.png
```

perintah tersebut akan mencari lokasi file yang bernama fileq. Jika ada pesan kesalahan itu akibat dari database pencarian yang digunakan oleh locate belum diaktifkan. Untuk mengaktifkan database tersebut digunakan perintah updatedb dan harus dijalankan melalui uer root.

Contoh:

```
root@praktikum-PC:~# updatedb
```

**Which**
**which** digunakan khusus untuk mengetahui letak file perintah. Jika anda mencari file biasa, maka which tidak akan menemukannya.
Format : which option nama-perintah
Contoh:

```
smuet@praktikum-PC:~$ which mkdir
/bin/mkdir
```

**Whereis**
Berfungsi sama seperti which. Tetapi hasil perintah whereis juga akan menapilkan letak manualnya. Tersedianya manual tergantung dari distro masing-masing, apakah disertakan atau tidak.
Format : whereis option nama-perintah
Contoh:

```
smuet@praktikum-PC:~$ whereis mkdir
mkdir: /bin/mkdir /usr/share/man/man1/mkdir.1.gz
```

**Izin Akses File**

Setiap file linux memiliki status izin akses file (file permission). Maksudnya setiap file memiliki informasi untuk mengatur siapa saja yang berhak untuk membaca, menjalankan atau mengubah file tersebut. Tujuannya adalah unuk menjaga privasi file, keamanan serta integritas sistem agar tidak terganggu.

**Melihat izin akses file**

Untuk mengetahui izin akses suatu file dapat digunakan perintah **ls** dengan option -l
Contoh :

```
smuet@praktikum-PC:~$ ls -l
-rw-r#r# 1 smuet smuet 206 2008-08-29 20:55 file2
-rwxrwxrwx 1 smuet smuet 73412 2008-08-09 16:11 Firewall.htm
drwxr-xr-x 2 smuet smuet 4096 2008-08-28 21:30 iso
drwxr-xr-x 6 smuet smuet 4096 2008-08-27 00:11 Music
-rwxrwxrwx 1 smuet smuet 1611767 2008-07-10 14:04 my files.odt
```

pada tampilan tersebut, terdapat 9 kolom.
-rw-r-r- : ini adalah izin akses file
1 : link file
smuet : pemilik file
smuet : nama group pemilik file
206 : ukuran file
2008-08-29 : tanggal pembuatan file
20:55 : jam pembuatan/modifikasi file
file2 : nama file
perizinan file dan direktori dibagi atas 3 macam akses, antara lain:
READ(r) membaca fiel atau direktori
WRITE(w) menulis dan menciptakan file atau direktori
EXECUTE(x) mengeksekusi file atau memasuki directori
Kepemilikan file dan direktori dibagi atas 3 macam kepemilikan, antara lain:
Owner (u) yaitu user tertentu
Group (g) yaitu group tertentu
Othera (o) yaitu owner atau group diatas
atau juga bisa digunakan singkatan a untuk mewakili ugo
Untuk izin akses file terdapat 10 digit karakte, yang dibagi menjadi tiga kelompok.

Pada Contoh diatas yaitu:
-rw-r#r#
1 karakter pertama (-), digunakan untuk menentukan tipe file. Tipe yang ada :
- : file biasa
d : direktori
l : link
c : special file
s : socket
p : name piped
b : block device
3 karakter kedua (rw-), digunakan untuk izin akses file terhadap user pemilik file.
3 karakter ketiga (r-), digunakan untuk izin akses file terhadap group pemilik file.
3 karakter keempat (r-), difunakan untuk izin akses terhadap other (user dan group

yang lain).
Dengan demikian, pada Contoh diatas, file dengan nama file2 adalah file biasa, pemilik file mempunyai izin akses read dan write, groupnya mempunyai izin akses read dan other mempunyai izin read.
Management file dan direktori
1. Hak akses file/direktori
Izin akses sebuah file dapat dirubah sesuai dengan kebutuhan. Untuk mengubahnya digunakan perintah chmod.
Format : chmod [ugoa] [=+-] [rwx] file_atau_direktori atau
chmod [angka-perizinan] file-atau-direktori
keterangan:
u : user
g : group
o : other
a : all
= : set sebagai satu-satunya izin yang dimiliki
+ : penambahan izin
# : non-aktifkan suatu izin
r : akses read
w : akses write
x : akses execute
Angka perizinan : owner-group-others dengan akses rwx-rwx-rwx, tiap akses dimisalkna dengan bit 1 jika diberi akses dan bit 0 bila tidak diberi akses lihat dulu izin akses file sebelun dirubah

```
smuet@praktikum-PC:~$ ls -l file2
-rw-r-r- 1 smuet smuet 206 2008-08-29 20:55 file2
```

Contoh 1:
bila owner diberi seluruh akser, group hanya baca, other tidak ada akses sama sekali, maka angka perizinannya : 111-100-000, biner = 7-4-0 desimal ditulis 740

```
smuet@praktikum-PC:~$ chmod 740 file2
```

kemudian lihatlah perubahan perizinan pada file dengan perintah ls- l nama-file

Contoh 2 :

```
smuet@praktikum-PC:~$ chmod ugoa+x file2
```

setelah mendapat izin akses execute, file secara default berubah menjadi hijau.
Kepemilikan File dan Gruop
Untuk keamanan dan privasi, setiap file di linux memiliki identitas kepemilikan(ownership). Dengan adanya identitas ini maka akan jelas siapa pemilik file tersebut.

**Melihat Pemilik File dan Group**
Untuk melihat kepemilikan suatu file dan group, gunakan perintah ls dengan option -l
Contoh:

```
smuet@praktikum-PC:~$ ls -l file2
-rwxr-x-x 1 smuet smuet 206 2008-08-29 20:55 file2
```

Terlihat nama smuet yang pertama adalah pemilik file dan nama arie yang kedua adalah nama groupnya. Secara default group saam seperti nama pemilik file.
Mengubah Pemilik File dan Direktori
Pemilik sebuah file atau direktori dapat diganti menjadi user yang lain. Untuk mengganti digunakan perintah chown
format :chown option pemilik-baru nama-file/direktori

Contoh :

root@praktikum-PC:~# chown klas file2

lihat perubahannya,

root@praktikum-PC:~# ls -l file2
-rwxr-x-x 1 klas smuet 206 2008-08-29 20:55 file2

*note : perintah chown harus dilakukan melalui root. User pengganti sudah harus ada dalam sistem.*

**Mengubah Pemilik Group**

Untuk mengubah pemilik group digunakan perintah chgrp. Perintah ini harus dialkukan melalui root dan grouo pengganti sudah harus ada dalam sistem.
Format: chgrp option group-pengganti nama-file/direktori
Contoh :

root@praktikum-PC:~# chgrp klas file2

lihat kembali perubahannya.

root@praktikum-PC:~# ls -l file2
-rwxr-x#x 1 klas klas 206 2008-08-29 20:55 file2

**Manajemen User**

Anda dapat mengatur user yang akan bekerja dengan sistem linux. Beberapa pengaturan yang akan dibahas adalah:

**Berganti User**

Anda dapat berganti user yang sedang aktif menjadi user lain tanpa harus melakukan logout.
Gunakan perintah **su**
format:
su options nama-user-pengganti
Contoh:

smuet@praktikum-PC:~$
su LAB2
password:

**Membuat User**

Untuk membuat user baru dapat digunakan perintah **adduser** atau **useradd.**
Perintah ini harus dijalankan melalui user root. Login atau bergantilah dari user biasa ke user root.
Format
: **useradd** [option...]
nama-user-baru
: **adduser** [option...] nama-user-baru
Contoh:

root@praktikum-PC:~# useradd LAB2

**Mengganti Password**

Setelah membuat user baru, kita perlu membuat password. Gunakan perintah **passwd**. Perintah ini juga digunakan jika anda ingin menganti password yang sudah ada. Untuk mengganti password user lain, gunakan user root. Jika user ingin mengganti password nya sendiri, tidak diperlukan user root.
Format:
**passwd** [option...] nama-user
Contoh:

```
root@praktikum-PC:~# adduser LAB2
Adding
user `LAB2`
MEnambah grup baru `LAB2` (1001) &
Adding new user `LAB2' (1001) with group `klas` &
The
home directory `/home/LAB2' already exists. Not copying from
`/etc/skel`.
```

Kemudian masukkan password yang diinginkan.beberapa distro menghendaki minimal 6 karakter pasword. Tekan enter setelah mengisi.
Enter
new UNIX password:
Retype
new UNIX password:
passwd:
kata sandi diperbaharui dengan sukses

**Menghapus User**
User yang sudah dibuat juga dapat dihapus. Gunakan perintah **userdel**
Format
: **userdel** [option...] nama-user-yang-akan-dihapus
Contoh

```
smuet@praktikum-PC:~$ sudo userdel -r klas
```

Setiap kali user baru dibuat, secara default sistem akan membuatkan home direktori bagi user tersebut nama home direktori sama dengan nama usernya. Jika anda ingin menghapus user, maka disarankan home direktori juga ikut dihapus. Jika tidak dihapus, bisa menimbulkan kerancuan, “home direktorinya ada, tapi usernya tidak ada?” . Untuk menghapusnya home direktori secara otomatis, tambahkan option -r seperti pada Contoh.

**Menampilkan User**
who digunakan untuk menampilkan user yang login ke sistem.
Format
: who [option]& [file |arg1 arg ]

```
smuet@praktikum-PC:~$ who smuet
tty7 2008-08-31 20:51 (:0)
smuet  pts/0 2008-08-31 21:23 (:0.0)
smuet  pts/1 2008-08-31 21:23 (:0.0)
```

Kolom pertama menunjukkan nama user yang login, kolom kedua menunjukkan terminal line yang digunakan, kolom ketiga menunjukkan waktu login dan kolom keempat menunjukkan domain atau IP asal mereka koneksi, jika kosong berarti mereka main langsung dari console.

**Membuat Group**
Group yang dimaksud adalah kelompok user yang saling bergabung dan mempunyai ketentuan tersendiri di kelompoknya. Setiap kali user baru dibuat, secara default sistem akan membuat sebuah group yang namanya sama dengan nama user tersebut. Selain group yang dibuat secara default oleh sistem, kita juga bisa juga membuat group baru.
Perintah yang digunakan adalah groupadd
format:
**groupadd** [option...] nama-grup-baru

Contoh:

root@praktikum-PC:~# groupadd groupbaru

**Menghapus Group**

Group yang ada dapat dihapus. Gunakan perintah **groupdel**

format:

groupdel

Contoh:

root@praktikum-PC:~# groupdel groupbaru

**V. Tugas**

1. Lakukan beberapa hal berikut:

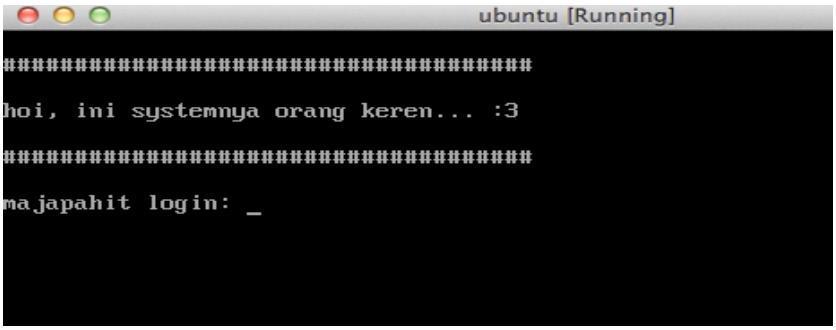
a) extrak dan compres file dengan beberapa format kompresi (.tar,.zip,rar.,.gzip,.bz) dengan perintah tar

b) lakukan instalasi sebuah paket dalam berbagai format .tgz, .deb, .rpm

c) bagaimana caranya mematikan sistem setalah 5 menit?

d) seorang admin ingin pergi makan tapi sebelumnya dia harus mematikan sistem, tunjukkan cara mematikan sistem dengan menyertakan pesan.

e) apa yang akan anda lakukan untuk mendapatkan output seperti gambar 1.1




Gambar 1.1

f) Buatlah pesan selamat datang bagi user yang baru log in

g) Temukan informasi tentang partisi sistem anda

h) Konfigurasi Ip address pada sistem anda.

# Praktikum II
# Shell Programing

**I. Tujuan**

1. memahami konsep shell programing
2. memahami jenis-jenis variabel dalam shell
3. mampu menulis program dengan shell programing
4. Memahami konsep Shell interaktif
5. Mengetahui environment Shell

**II. Dasar teori**

Shell adalah sebuah bahasa penterjemah perintah (command interpreter language) atau sebuah prosesor makro yang menjalankan perintah. Shell juga dapat berarti interpreter pertintah yang menjadi antarmuka antara user dengan utilitas dan bahasa pemrograman. Dengan shell, dapat dibuat sebuah perintah atau file yang berisi perintah-perintah itu sndiri. Perintah baru tersebut mempunyai status yang sama dan di letakkan pada direktori /bin.

shell mengizinkan eksekusi perintah secara synchronously dan asynchronously. Shell menunggu perintah synchronous untuk dilengkapi sebelum menyetujui lebih banyak input, sedangkan perintah asynchronous terus berjalan dalam paralel dengan shell ketika membacanya dan menjalankan perinntah tambahan. Shell juga mengenal adanya redirection. Dengan redirector dapat dilakukan kontrol untuk intput dan output dari perintah yang ada dan juga melakukan kontrol terhadap isinya. Secara default, shell UNIX juga menyediakan perintah-perintah buil-in, seperti pwd, cd, kill, histroy, atau utilitas lain yang terpisah.

Sebagai interface dan command interpreter, shell dapat digunakan secara interaktif maupun noninteraktif. Dangan dua mode tersebut, shell mampu menerima input dari device (keyboard) atau file. Fitur interaktif yang termasuk di dalamnya adalah kontrol job, history, alias, dan editor command line.

Hal lain yang menjadikan shell sangat penting adalah shell menyediakan bahasa pemrograman yang telah disertakan(embeded). Sama halnya dengan bahasa pemrograman tingkat tinggi laennya, interpreter shell juga menyediakan variabel, flow control, quoting dan fungsi.

**Macam-macam shell**

pada UNIX/Linux terdapat berbagai macam shell dengan kelebihan dan kekurangan masing-masing. Dengan banyaknya variasi shell ini, user bebas memilih shell yang digunakan. Meskipun kebanyakan sistem operasi telah menentukan sebuah shell sebagai shell default, tetapi tidak menutup kemungkinan shell lain juga dapat dijalankan. Berikut ini beberapa macam shell yang umum terdapat dalam sistem operasi UNIX/Linux.

- Bourne Shell (/bin/sh)
- Bourne Again Shell(/bin/bash)
- C Shell (/bin/csh)
- Tenex C Shell (/bin/tcsh)
- Tcl shell (/bin/tclsh)
- Korn shell (shell /bin/ksh)
- Public domain korn shell (/bin/pdksh)
- A shell (/bin/ash)
- Z shell (/bin/zsh)

**Mengganti dan menjalankan Shell**

Linux menggunakan bash sebagai shell default, tetapi pengguna bisa mengubah shell default untuk tiap usernya. Untuk melihat shell yang sedang digunakan oleh user bisa dilihat pada file

/etc/passwd.
root:x:0:0:root:/root:/bin/bash
daemon:x:1:1:daemon:/usr/sbin:/bin/sh

Isi dari file tersebut tiap barisnya dibagi menjadi tujuh bagian, dan setiap terakhir digunakan untuk mendefinisikan shell yang digunakan. Dalam contoh di atas user root menggunakan shell Bash sedangkan daemon menggunakan shell bourne shell.
Cara lain yang bisa digunakan untuk melihat shell adalah dengan melihat environmet user dengan menjalankan perintah env. Environment user merupakan lingkungan user yang berisi semua variabel atau ketentuan khusu untuk user tersebut.

$env
.....
SHELL=/bin/bash
.....

**Mengganti Shell**
Ada beberapa cara yang dapat digunakan untuk mengubah shell default, yaitu dengan utilitas userconf, atau dapat juga dengan mengubah pada file /etc/passwd tersebut tetapi harus sebagai root. Contoh mengganti shell dengan utilitas chsh (change shell).

$chsh
Password:
Changing the login shell for praktikum
Enter the new value, or press ENTER for the default
Login Shell [/bin/bash]: /bin/sh

**Menjalankan shell**
sebuah shell dapat dijalankan tanpa harus mengubah default shell-nya. Cara yang digunakan adalah dengan memanggil nama shell pada command prompt. Misalnya, user akan menggunakan shell sh, user tinggal menjalankan shell sh, user tinggal menjalankan sh sehingga prompt akan berubah sesuai dengan promt Bourne shell.
Untuk keluar dan kembali ke shell default, ketik exit atau tekan Ctrl + d.

bash@praktikum:~$ sh
$
$ exit

**Menjalankan script shell**
Untuk menjalankan sebuah script shell, sebaiknya lebih dulu memahami penggunaan path absoluth dan path relatif. Ada dua cara yang digunakan untuk menjalankan sebuah shell script, yaitu

$bash hello.sh

./hello.sh

Untuk dapat menjalankan perintah di atas, file program harus dijalankan sebagai file executable. Untuk memberi atribut eksekusi tersebut, dapat digunakan perintah chmod.

$chmod +x hello.

**VARIABEL**
variabel adalah sebuah kata yang mempunyai nilai. Shell sebagai sebuah interpreter juga menyediakan fasilitas atau kemampuan yang memungkinkan user untuk membuat, mendefinisikan dan menghapus variabel. Sebuah variabel secara umum didefinisikan dengan sintaks berikut :
Nama_variabel=isi variabel

**Macam-macam Variabel**
Ketika sebuah shell dijalankan, akan ada tiga macam variabel yang secara otomatis dipanggil. Variabel-variabel tersebut adalah :

**variabel lokal**

variabel lokal adalah variabel yang ada hanya pada saat masih aktif, dan hanya dikenal di lingkungan itu sendiri, sehingga variabel lokal hanya berlaku pada lingkungan dimana variabel tersebut dibuat.

contoh local variabel

```
#!/bin/bash
        HELLO=Hello
        function hello {
            local HELLO=World
            echo $HELLO
        }
        echo $HELLO
        hello
        echo $HELLO
```

**Variabel lingkungan**

variabel lingkungan adalah variabel yang terdapat dalam shell dan digunakan dalam proses anak yang dijalankan oleh shell tersebut. Variabel lingkungan ini bisa berupa dari variabel lokal yang diekspor. Untuk mengganti variabel lingkungan digunakan perintah export. Contoh berikut adalah mengubah charset lokal menjadi lokal Indonesia.

```
$export LC_LOCAL=id_ID
```

**Variabel shell**

Variabel shell adalah variabel yang ditetapkan oleh shell dan digunakan oleh shell agar berjalan dengan baik. Sebenarnya, variabel ini bisa dimasukkan dalam kategori variabel lingkungan. Contoh variabel ini adalah variabel defaul dari bash, misalnya:
HOME, PWD, PS1 dan PS2.

**Variabel Read-Only**

Variabel read-only adalah variabel yang mempunyai atribut read-only, artinya variabel itu tidak bisa diganti nilainya. Bahkan sebuah variabel tidak bisa dihapus dengan perintah unset jika sebuah variabel diberi atribut read-only.

```
$nama=praktikum
$readonly nama
$nama=os
bash:nama:readonly variable
```

**Quoting**

Quoting adalah mekanisme untuk melindungi metakarakter dari interpretasi sebagai sebuah simbol. Shell juga mempunya beberapa karakter yang difungsikan untuk melindungi metakarkater agar tetap diinterpretasikan sebagai karakter biasa. Ada tiga karakter quoting dalam Shell, yaitu :

1. Backslash ( \ )
2. Petik tunggal ( ' )
3. Petik ganda ( “ )

Contoh quoting dalam Shell :

```
$ echo don\'t miss it
don't miss it
$ echo “don't miss it”
```

don't miss it

Keterangan :
1. Tanda \ menandakan katakter ' yang mengikuti bukan sebuah metakarakter
2. Pengunaan tanda petik double “ juga berfungsi melindungi interpretasi karakter ' sebagai metakarakter

**Metakarakter Dalam Shell**

Metakarakter adalah sebuah karakter yang memiliki arti tertentu. Dalam Shell juga dikenal beberapa metakarakter. Karena metakarakter juga ada dalam Shell maka yang perlu diperhatikan adalah kesalahan dalam penanganan sebuah karakter. Dalam sebuah kasus mencetak sebuah strin di layar monitor, terkadang terjadi kasus dimana dari salah satu karakter dalam string tersebut merupakan metakarater. Karena mengandung metakarakter maka Shell akan menginterpretasikan string tersebut tidak seperti yang diharapkan.

Contoh kasus :

```
$ echo don't miss it 'enter'
```

Keterangan :
1. Tanda ' dinterpretasikan sebagai serangkaian string sehingga Shell akan menunggu sampai tanda ' berikutnya untuk berhenti dan kemudian menampilkannya.

Jika maksudnya adalah untuk mencetak string #don#t miss it# maka yang perlu diperhatikan adalah

```
$ echo don\'t miss it 'enter'
  don't miss it
```

Keterangan :
1. \ merupakan karakter yang meloloskan interpretasi tanda ' yang merupakan metakarakter dalam Shell.
2. Karakter \ dikenal dengan istilah quoting dalam Shell

**Perintah Echo**

Echo adalah perintah untuk menampilkan data yang ada pada argumen ke standard output (stdout), yang dalam hal ini stdout bisa merupakan layar monitor atau juga sebuah file.
Perintah Echo dalam Shell memiliki opsi-opsi untuk membentuk atau memberikan format pada data yang dikeluarkan. Sama halnya dengan pemrograman yang lain misalnya bahasa C. karakter yang digunakan untuk membentuk sebuah format dalam perintah echo biasa disebut “escape sequences character”. Contoh escape sequences character adalah \n yang memiliki arti ganti baris atau baris baru. Untuk bisa menggunakan escape sequences dalam Shell yang perlu diperhatikan adalah bahwa secara default shell tidak menerima escape sequence, namun untuk bisa menggunakannya perlu ditambahkan beberapa opsi yang ada dalam perintah Echo. Berikut tabel opsi dan escape sequence dalam perintah Echo.

**Fungsi**

Fungsi adalah skrip yang berisi kumpulan perintah yang berada diluar program utama. Fungsi biasanya berisi perintah-perintah dalam Shell. Tujuan dari adanya fungsi adalah untuk lebih mengefisiensikan pemangilan sekumpulan perintah yang berulang-ulang pada program yang dibuat.
Di dalam Shell fungsi juga bisa didefinisikan interaktif maupun secara skrip program, dan meskipun didefinisikan secara interaktif, sebuah fungsi juga bisa dipanggil melalui skrip yang dibuat dalam sebuah file dengan catatan fungsi tersebut sudah di export. Setelah melalui mekanisme export ini

sub-shell juga bisa memanggil fungsi tersebut.

Bentuk umum dalam mendefinisikan fungsi dalam BASH Shell adalah sebagai berikut :

```
nama_fungsi () { command; command; }
function nama_fungsi { command; command; }
function nama_fungsi () { command; command; }
```

**Array**

Pada versi BASH 2.x terdapat fungsi untuk mendefinisikan array satu dimensi. Array memungkinkan seorang programmer mengkoleksi daftar beberapa nilai dalam sebuah variabel. Untuk mengektraksi kembali nilai-nilai tersebut dapat dilakukan dengan menyebutkan nama varibel yang diikuti oleh nomer indek array tersebut.

Pendefinisian sebuah array juga bisa dilakukan on the fly(tanpa mendefinisikan terlebih dahulu). Dan tidak ada batasan maksimum dari sebuah array yang dibuat dalam lingkungan BASH Shell. Pada saat sebuah nilai diberikan ke dalam sebuah array yang telah didefinisikan, indek array secara otomatis akan dimulai dari 0, dan bertambah naik 1 sampai semua kumpulan nilai-nilai dimasukkan.

**III. Petunjuk praktikum**

1. mulailah menuslis program dengen shell, misal “hello word”.
2. Cobalah sebagai latihan

Latihan 1

```
$ if grep “root” /etc/passwd
> then
> echo “disini ada user yang bernama root”
> else
> echo “user tersebut tidak ada”
> fi
```

Latihan 2

```
#!/bin/bash
echo “selamat datang $USER”
echo “di shell programming”
```

Latihan 3

```
#!/bin/bash
echo “Shell yang digunakan adalah $SHELL”
echo “saat ini jam `date +%T`”
echo “tanggal `date +%D`”
```

Latihan 4

```
#!/bin/bash
echo “hari ini tanggal `date +%d` bulan `date +%m` tahun `date
+%y`”
echo “:D”
```

Latihan 5

```
#!/bin/bash
clear
echo “nama login anda $LOGNAME”
echo “saat ini anda berada di direktori `pwd`”
```

```
echo “waktu sekarang adalah `date +%T`”
echo “selamat bekerja”
```

Latihan 6

```
#!/bin/bash
clear
echo “komputer anda telah menyala selama `uptime`”
echo “jumlah user yang login sebanyak `who | wc -l` user”
echo “anda login dengan user $LOGNAME”
echo “di shell $SHELL”
```

Latihan 7

```
#!/bin/bash
clear
data='date +%D'
jumlah=`who | wc -l`
echo “tanggal $data ada $jumlah user yang login”
```

Latihan 8

```
#!/bin/bash
user=andi
echo “hai $user i'm glad to meet you”
echo hai $user i\#m glad to meet you
echo “apakah kamu punya teman “special”?”
echo “\”special\” ?? teman apa itu?”
echo “ya pokoknya “special””
```

Latihan 9

```
$ function cetak_selamat {
> echo “Selamat Datang”
> echo “Di Shell Programming”
> }
$ cetak_selamat
Selamat Datang
Di Shell Programming
```

Latihan 10

```
$ function cetak_selamat () {
> echo “Selamat Datang”
> echo “Di Shell Programming”
> }
$ cetak_selamat
Selamat Datang
Di Shell Programming
```

Latihan 11

```
$ data=”halo ini bejo”
$ sdata=${data#*lo}
$ echo $sdata
$ tdata=${data%be*}
$ echo $tdata
```

Latihan 12

```
#!/bin/bash
TITEL=”Membuat Fungsi Sistem Informasi $HOSTNAME”
SAAT_INI=$(date+”%d %T %Z”)
UPD=”Sistem ini di update oleh $USER pada tanggal
$SAAT_INI”
function info_uptime(){
   echo “<h2>informasi uptime</h2>”
   echo “<pre>”
   uptime
   echo “</pre>”
}
cat <<- EOF
<HTML>
<HEAD>
<TITLE>$TITEL</TITLE>
</HEAD>
<BODY>
<H1>$TITEL</H1>
<P>$UPD</P>
$(info_uptime)
</BODY>
</HTML>
EOF
```

Latihan 13

```
#!/bin/bash
let data1 data2 hasil
read --p “masukkan sebuah angka : “
data1=$REPLY
read --p “masukkan sebuah angka lagi : “
data2=$REPLY
((hasil=data1-data2))
echo “hasil dari $data1 - $data2 adalah $hasil “
unset data1 data2 hasil
```

3. analisa kode berikut, apakah sudah benar jika belum benarkan

```
while :
 do
   clear
   echo "-------------------------------------"
   echo " Main Menu "
   echo "-------------------------------------"
   echo "[1] Show Todays date/time"
   echo "[2] Show files in current directory"
   echo "[3] Show calendar"
   echo "[4] Start editor to write letters"
   echo "[5] Exit/Stop"
   echo "======================"
   echo -n "Enter your menu choice [1-5]: "
   read yourch
   case $yourch in
```

```
        1) echo "Today is date , press a key. . ." ; read ;;
        2) echo "Files in pwd" ; la; echo "Press a key. . ." ; read ;;
        3) cal ; echo "Press a key. . ." ; read ;;
        4) vi ;;
        5) exit 0 ;;
        *) echo "Opps!!! Please select choice 1,2,3,4, or 5";
           echo "Press a key. . ." ; read ;;
    esac
done
```

**IV. Tugas**

1. Buat skrip untuk melihat aktifitas jaringan pada sistem anda.
2. Buat skrip untuk merubah input lowercase menjadi uppercase.
3. Cetak variable i = 1 to 20, dengan ketentuan 1..10 jalan pada baground sehingga outputnya :
   11 12 13 14 15 16 17 18 19 20
   1 2 3 4 5 6 7 8 9 10
4. Tuliskan sebuah program dan terdapat kontrol dimana yang bisa mengeksekusi program tersebut hanyalah 'root'.
5. Buat program yang mematikan dirinya sendiri.
6. Buat program yang menghapus file disebuah direktory jika file itu mengandung bad karakter (/[\ +\{\;\"\\\=\?~\(\)\<\>\&*\|\$]/p` )

selamat mengerjakan

# Praktikum 3
# Process dan Thread

**I. Tujauan**

1. mengerti konsep proses dan thread
2. mempu membuat proses dan thread
3. mampu memanajemen proses

**II. Dasar teori**

**proses**

Proses adalah program yang sedang dieksekusi. Setiap kali menjalankan suatu program, Sistem UNIX melakukan suatu fork(), yaitu melakukan beberapa urutan operasi untuk membuat suatu proses konteks dan kemudian mengeksekusi program tersebut dalam konteks yang sudah dibuat. Setiap proses yang dijalankan akan memiliki PID (Process ID).

Kita dapat melihat proses yang sedang berjalan dengan perintah ps. Dengan fork() sistem call akan membuat proses baru yang identik dengan proses induk kecuali proses ID. Proses dicopy ke memory dari proses parent dan strukture proses baru ditangani oleh kernel. Lingkungan, resource limits, umask, controlling terminal, direktori tempat eksekusi, root direktori, signal mask, dan resource proses lainnya digandakan dari proses induk oleh proses fork.

**Thread**

operasi thread meleiputi pembuatan thread, penghentian, sikronisasi, penjadwalan, managemen, dan pengulangan proses.

- Thread tidak mengatur pembuatan thread lain, atau mengetahui sebuah thread laen telah dibuat.
- semua thread dengan sebuah proses saling berbagi pakai address space.
- Thread pada satu proses yang sama saling berbagi pakai:
  - Process instructions
  - Most data
  - open files (descriptors)
  - signals and signal handlers
  - current working directory
  - User and group id
- setiap thread merupakan unik
  - Thread ID
  - set of registers, stack pointer
  - stack for local variables, return addresses
  - signal mask
  - priority
  - Return value: errno
- fungsi pthread memberikan nilai #0# jika OK

contoh membuat dan mengahiri thread

```
#include <stdio.h>
#include <stdlib.h>
#include <pthread.h>

void *print_message_function( void *ptr );
main()
{
    pthread_t thread1, thread2;
    char *message1 = "Thread 1";
```

```
    char *message2 = "Thread 2";
    int  iret1, iret2;
   /* Create independent threads each of which will
execute function */
    iret1 = pthread_create( &thread1, NULL,
print_message_function, (void*) message1);
    iret2 = pthread_create( &thread2, NULL,
print_message_function, (void*) message2);
    /* Wait till threads are complete before main
continues. Unless we  */
    /* wait we run the risk of executing an exit which
will terminate   */
    /* the process and all threads before the threads
have completed.   */
    pthread_join( thread1, NULL);
    pthread_join( thread2, NULL);
    printf("Thread 1 returns: %d\n",iret1);
    printf("Thread 2 returns: %d\n",iret2);
    exit(0);
}
void *print_message_function( void *ptr )
{
    char *message;
    message = (char *) ptr;
    printf("%s \n", message);
}
```

**III. Petunjuk praktikum**

1. latihan

a) Untuk mengetahui berapa jumlah memory yang tersisa dan yang terpakai. Gunakan perintah : 'free --okt'. Perintah tersebut akan menampilkan jumlah byte tentang memory

b) Top juga menampilkan informasi memory secara kontinu. Untuk menggunakannya ketik: 'top'

c) Menggunakan perintah 'at', Perintah at memungkinkan kita untuk mengantrikan eksekusi suatu perintah pada waktu tertentu. Sebagai contoh, untuk membentuk pekerjaan pada jam 2:30 a.m. Untuk menuliskan shcedule pekerjaan dengan 'at'.  yakni dengan menulis perintah baris demi baris dan menekan enter untuk masing-masing perintah dan untuk mengakhiri dapat digunakan <CTRL + D>. Ikuti langkah berikut:

- Ketik : $ date
- Ketik : $ at <waktu yang mendekati waktu saat ini> misal : $ at 17:15
- Akan muncul prompt 'at'
- Ketikkan : $ dmesg > simpanan
- Tekan ENTER kemudian CTRL+D
- Lihatlah daftar antrian dengan mengetikkan : $at #l
- Apabila ada proses yang ingin dibatalkan, bisa dilakukan dengan cara mengetikkan : $ at --d nomorjobid, misal : $at --d 12

d) Untuk mengetahui sebelumnya apakah ada program yang sedang berjalan atau tidak, ketik: $ jobs  Atau untuk melihat proses ID  '$ jobs --p' Atau untuk melihat PID dan apa yang sedang dilakukan '$ jobs --l'

Untuk menjalankan kembali proses yang disuspend di foreground, bisa digunakan perintah

'fg'. Ketik : '$ fg % nomorPIDpadajobs'

Atau untuk proses terakhir yang diproses cukup mengetik '$ fg' . Sekarang kita akan mencoba menjalankan proses di background atau dalam kondisi suspend. Kita akan membuat sebuah script shell yang bertujuan berjalan terus-menerus dan diletakkan di proses background. Buat file dengan nama bgproses.sh di direktori /bin.

Ketik:

```
#! /bin/sh
i=0
while [ true ];
do
i=$(($i+1))
done
echo $i
```

Ubah hak aksesnya agar bisa dieksekusi. Ketik: '$ chmod +x makebgNPM.sh'

Jalankan script yang dibuat. Ketik: '$ makebgNPM.sh' Tekan CTRL+Z untuk melakukan suspend pada proses yang sedang berjalan.

Ketik perintah : '$ ps' Apakah ada proses makebgNPM.sh? berapa PID-nya? Karena sedang disuspend, maka makebgNPM.sh dalam kondisi stopped.

Untuk menjalankan kembali proses makebgNPM.sh, Ketik :

'$ bg % nomorPIDpadajobs'

Lihat menggunakan jobs, apakah makebgNPM.sh telah berjalan kembali. Untuk mematikan secara permanen proses yang sedang berjalan. Kita bisa menggunakan perintah kill. Caranya:

'$ kill -9 <NoPIDmakebgNPM.sh>'

2. cobalah program berikut

```
#include <stdio.h>   /* printf, stderr, fprintf */
#include <unistd.h>  /* _exit, fork */
#include <stdlib.h>  /* exit */
#include <errno.h>   /* errno */

int main(void)
{
  pid_t  pid;

  /* Output from both the child and the parent process
   * will be written to the standard output,
   * as they both run at the same time.
   */
  pid = fork();
  if (pid == 0)
  {
    /* Child process:
     * When fork() returns 0, we are in
     * the child process.
     * Here we count up to ten, one each second.
     */
    int j;
    for (j = 0; j < 10; j++)
    {
      printf("child: %d\n", j);
```

```
        sleep(1);
      }
      _exit(0);  /* Note that we do not use exit() */
    }
    else if (pid > 0)
    {
      /* Parent process:
       * When fork() returns a positive number, we are in
the parent process
       * (the fork return value is the PID of the newly-
created child process).
       * Again we count up to ten.
       */
      int i;
      for (i = 0; i < 10; i++)
      {
        printf("parent: %d\n", i);
        sleep(1);
      }
      exit(0);
    }
    else
    {
      /* Error:
       * When fork() returns a negative number, an error
happened
       * (for example, number of processes reached the limit
).
       */
      fprintf(stderr, "can't fork, error %d\n", errno);
      exit(EXIT_FAILURE);
    }
}
```

**IV. Tugas**

1. Pahamilah contoh kode hello.c sederhana berikut,

```
#include <pthread.h>
#include <stdio.h>
#include <stdlib.h>
#define NUM_THREADS     5

void *PrintHello(void *threadid)
{
  long tid;
  tid = (long)threadid;
  printf("Hello World! It's me, thread #%ld!\n", tid);
  pthread_exit(NULL);
}

int main(int argc, char *argv[])
{
  pthread_t threads[NUM_THREADS];
```

```
  int rc;
  long t;
  for(t=0;t<NUM_THREADS;t++){
    printf("In main: creating thread %ld\n", t);
    rc = pthread_create(&threads[t], NULL, PrintHello, (void *)t);
    if (rc){
      printf("ERROR; return code from pthread_create() is %d\n", rc);
      exit(-1);
      }
    }

  /* Last thing that main() should do */
  pthread_exit(NULL);
}
```

kemudian eksekusi juga dengan beberapa opsi berikut, tuliskan apa yang terjadi.

```
icc -pthread -o hello hello.c
pathcc -pthread -o hello hello.c
pgcc -lpthread -o hello hello.c
gcc -pthread -o hello hello.c
```

2. buat sebuah program yang menunjukkan pembuatan dan termination sebuah thread.
3. buat program satu ekseskusi 100 proses/100 thread.
4. Buatlah sebuah daemon pada waktu tertentu menampilkan pesan “hello”.

selamat mengerjakan

# Praktikum 4
# Modules Kernel dan sistem call

## I. Tujuan

1. Praktikan mengerti kinerja sebuah kernel
2. Praktikan mampu membuat module kernel
3. Praktikan memahami perbedaan module dengan program
4. Praktikan mengerti kinerja sebuah sistem call

## II. Dasar teori

**Kernel** adalah bagian dari sistem yang menangani hardware, mengalokasi sumber daya seperti memory dan CPU, dan yang bertanggung jawab menangani file sistem dan komunikasi jaringan.
**Kernel module** adalah sebuah kode yang dapat disisipkan dan di lepas dari sebuah kernel sesuai keinginan. ketika module di load pada sebuah kernel dapat langsung berfungsi tanpa mereboot sistem. contohnya module untuk sebuah driver, yang mengijinkan sebuah kernel untuk mengakses ke hardware yang tersambung ke sistem. Tanpa menggunakan modules, Untuk membangun sebuah monolithic kernel dengan cara menambahkan functionality, fitur pada kernel image. Disampin ukurannya kernel yang besar untuk mengaktifkan suatu fitur perlu merebook kernel.
Modules yang sudah ada dalam kernel dapat di lihat dengan perintah 'lsmod'. Lsmod adalah perintah yang membaca informasi darii /proc/modules. Sebuah modules yang akan dipasang pada kernel di tangani oleh daemon kmod dengan mengeksekusi modprobe. Modprobe modprobe melewati sebuah string dalam salah satu dari dua bentuk:
" Sebuah nama modul seperti softdog atau ppp.
" seperti char-major-10-30.

Jika modprobe di kenali sebagai generik identifier, terlebih dahulu mencari yang string
dalam file. /etc/modprobe.conf alias char-major-10-30 softdog identifier generik mengacu pada softdog.ko modul.
Kemudian modprobe melihat ke /lib/modules/version/modules.dep, untuk melihat apakah ada modules lain yang harus di load sebelum modules yang di minta diload. File ini dihasilkan oleh depmod -a dan berisi dependensis modules. Contoh msdos.ko memerlukan fat.ko yang perlu di load terlebih dahulu ke dalam kernel. Modules yang di minta memiliki dependensi pada modules yang lain jika terdapat symbol atau variabel yang di butuhkan dan terdapat pada modules lain terakhir, modprobe menggunakan ismod untuk meload prasyarat lainnya pada kernel, kemudia baru meluad modules yang di maksud. Modprobe mengarahkan ismod ke /lib/modules/version/, standart modules direktori.
insmod dimaksudkan mengetahui lokasi modul, sedangkan modprobe secara default mengetahui lokasi modul, tahu bagaimana untuk mengetahui dependensi dan memuat modul dengan urutan yang benar. Jika Anda ingin memuat modul msdos, jalankan terlebih dahulu:
insmod /lib/modules/2.6.11/kernel/fs/fat/fat.ko
insmod /lib/modules/2.6.11/kernel/fs/msdos/msdos.ko atau Modprobe msdos

Insmod membutuhkan full path dan memeasukan modules pada sebelah kanan.
Sedangkan modprobe hanya perlu nama tanpa perlu extensio dan di parsing oleh
/lib/modules/version/modules.dep.

Distribusi linux menyediakan modprobe, ismod, dan depmod sebagai paket yang dipanggil module-init-tools. Pada versi sebelumnya paket ini di panggil modutils. Beberapa distro juga mengeset wrappers yang mengijinkan kedua package untuk di install secara paralel dan cara yang benar sesuai dengan kernel 2.4 dan 2.6. user tidak perlu memikirkan secara detail bagaimana kinerja paket tersebut selama menjalankan versi terbaru dari tool tersebut.

Contoh modules sederhana :

```
/*
 * hello-1.c - The simplest kernel module.
 */
#include <linux/module.h>       /* Needed by all modules */
#include <linux/kernel.h>       /* Needed for KERN_INFO */
int init_module(void)
{
        printk(KERN_INFO "Hello world 1.\n");
        /*
         * A non 0 return means init_module failed; module can't be
loaded.
         */
        return 0;
}
void cleanup_module(void)
{
        printk(KERN_INFO "Goodbye world 1.\n");
}
```

Modul Kernel harus memiliki minimal dua fungsi: sebuah "start" (inisialisasi) fungsi bernama init_module (), yang disebut ketika modul tersebut insmoded ke dalam kernel, dan "end" (cleanup) disebut fungsi cleanup_module () yang sebelum disebut rmmoded.
Kompile kernel modules mengkompile kernel modules berbeda dengan mengkompile program user lainnya. Dengan menggunakan makefiles dan kbuild. Jika mengkompile modules tetapi tidak sesuai dengan official kernel buka file, Linux/Documentation/kbuild/modules.txt

```
obj-m  += hello-1.o
all:
        make -C /lib/modules/$(shell uname -r)/build M=$(PWD) modules
clean:
        make -C /lib/modules/$(shell uname -r)/build M=$(PWD) clean
```

kemudian kompile dengan perintah :

```
make
```

periksa informasi dengan perintah **modinfo**

```
# modinfo hello-1.ko
```

**sistem call**

Tugas sistem operasi adalah membentuk komunikasi antara user dan hardware. Komunikasi tersebut terjadi dalam bentuk sistem call. Melalui sebuah shell sistem operasi akan menangkap perintah dari user dan dikomunikasikan melalui sistem calls. Peran sistem operasi adalah menjadi jembatan user dan hardware. Untuk shell sudah dipelajari pada bab selanjutnya.

*Note : perhatikan versi kernel anda*

**III. Petunjuk praktikum**

1. Cobalah modules hello-2.c berikut dan coba kompile.

```
/*
 * hello-2.c - Demonstrating the module_init() and module_exit() macros.
 * This is preferred over using init_module() and cleanup_module().
```

```
 */
#include <linux/module.h>       /* Needed by all modules */
#include <linux/kernel.h>       /* Needed for KERN_INFO */
#include <linux/init.h>         /* Needed for the macros */
static int __init hello_2_init(void)
{
        printk(KERN_INFO "Hello, world 2\n");
        return 0;
}
static void __exit hello_2_exit(void)
{
        printk(KERN_INFO "Goodbye, world 2\n");
}
module_init(hello_2_init);
module_exit(hello_2_exit);
```

**IV. Tugas**

1. Jelaskan tentang Modules Vs Program.
2. Buat kostum kernel spesifik fungsi/tujuan dari sistem operasi.
3. Buat hello word module dengan beberapa karakteristik.

Tampilkan pesan ketika module akan loading (<<Hello World!!>>) dan ketika unloading module menampilkan pesan (<<Goodbye>>)

4. Buatlah sebuah sistem call dan kemudian tambahkan kedalam kernel.

Selamat mengerjakan

# Praktikum 5
# LFS

**I. Tujuan**

1. Praktikan mengerti pembuatan sebuah sistem operasi.
2. Praktikan mampu membuat sistem operasi linux sendiri.

**II.Dasar Teori**

LFS adalah sebuah metode yang menyediakan instruksi langkah demi langkah untuk membangun sistem linux anda sendiri yang dimulai dari sebuah kode sumber.

Dewasa ini banyak sekali distribusi linux yang siap download dan pakai, akan tetapi ada beberapa keuntungan dari LFS.

- LFS memengajarkan bagaimana sebuah sistem linux bekerja secara inernal
  Membangun sistem secara LFS mengajarkan anda apa saja yang membuat linux bekerja, bagaimana sesuatu berkerja bersama dan saling bergantung satu sama lain. Dan yang paling penting bagaimana membuat sistem sesuai dengan selera dan kebutuhan anda.
- Dengan LFS bisa dibangun sebuah sistem yang simple
  Ketika menginstall regular distribusi, pasti banyak program yang terinstal dan mungkin tidak pernah dipakai. Program tersebut lebih tepatnya menghabiskan space hardisk. Hal seperti ini tidak akan terjadi pada LFS, bahkan bisa membuat sistem hanya dengan 100 MB. LFS sangat berguna jika digunakan untuk membuat sebuah embedded LFS sistem. Misalnya yang dibutuhkan hanya web server apache yang berjalan pada sistem, seharusnya hanya perlu 8 MB dari space hardisk. Bagaimana dengan regular distribusi yang berjalan apache web server?.
- LFS begitu fleksible
  Bisa digambarkan seperti membangun sebuah rumah, apa saja fasilitas dan tata letak bisa dibuat seuasai dengan keinginan.
- LFS menawarkan segi keamanan
  Dengan LFS sebuah sistem dibangun dari sumber, dan semuanya dibawah kontrol sendiri sehingga audit bisa dilakukan. Dengan LFS pula dapat melakukan patch security sendiri tanpa perlu menunggu orang lain untuk melakukakannya seperti yang ada pada regular distribusi.

**II . Tugas**

1. Buatlah sebuah sistem operasi Linux dengan cara LFS.
   Panduan dapat diperoleh dari http://tldp.org/LDP/lfs/LFS-BOOK-6.1.1.pdf
2. Uji sistem yang ada buat pada mesin virtual.
3. Bakar sistem operasi yang anda buat ke cakram pengaya kemudian instal pada sebuah mesin.

***note : perhatikan versi kenel anda***

selamat mengerjakan