

**Nama : Deanissa Sherly Sabilla**

**Kelas / Absen : SIB 1B / 06**

## **-PERTEMUAN 6-**

### **TUGAS PENDAHULUAN :**

Jawablah pertanyaan-pertanyaan di bawah ini :

1. Apa yang dimaksud dengan proses ?
  - Proses adalah program yang sedang dieksekusi. Setiap kali menggunakan utilitas sistem atau program aplikasi dari shell, satu atau lebih proses "child" akan dibuat oleh shell sesuai perintah yang diberikan. Setiap kali instruksi diberikan pada Linux shell, maka kernel akan menciptakan sebuah proses-id. Proses ini disebut juga dengan terminology Unix sebagai sebuah Job.
2. Sebutkan opsi yang dapat diberikan pada perintah ps!
  - ps -fae atau ps -aux
3. Apa yang dimaksud dengan sinyal ? Apa perintah untuk mengirim sinyal ?
  - Sinyal adalah pesan kecil yang dikirim oleh sistem operasi atau proses lain ke proses tertentu untuk memberi tahu atau meminta tindakan tertentu.
  - Perintah untuk mengirim sinyal biasanya adalah kill, "**kill [-nomor sinyal] PID**"
4. Apa yang dimaksud dengan proses foreground dan background pada job control ?
  - **foreground** hanya diperuntukkan untuk satu job pada satu waktu. Job pada foreground akan mengontrol shell -menerima input dari keyboard dan mengirim output ke layar.
  - Job pada **background** tidak menerima input dari terminal, biasanya berjalan tanpa memerlukan interaksi.
5. Apa yang dimaksud perintah-perintah penjadwalan prioritas top, nice, renice.
  - Perintah **top** digunakan untuk memantau proses yang sedang berjalan secara real-time dan mengurutkannya berdasarkan berbagai kriteria, seperti penggunaan CPU atau memori.
  - Perintah **nice** digunakan untuk menentukan prioritas eksekusi proses. Semakin tinggi nilai "nice" yang diberikan pada proses, semakin rendah prioritasnya.
  - Perintah **renice** digunakan untuk mengubah prioritas proses yang sudah berjalan. Ini memungkinkan pengguna untuk menyesuaikan prioritas proses yang sedang berjalan tanpa harus menghentikan dan memulai proses lagi.

**PERCOBAAN:**

1. Login sebagai user.
2. Jalankan Kalkulator
  - Menggunakan perintah 'bc' untuk menjalankan kalkulator

```
deanissa@DESKTOP-9U498E1:~$ bc
bc 1.07.1
Copyright 1991-1994, 1997, 1998, 2000, 2004, 2006, 2008, 2012-2017 Free Software Fo
undation, Inc.
This is free software with ABSOLUTELY NO WARRANTY.
For details type `warranty'.
warranty

bc 1.07.1
Copyright 1991-1994, 1997, 1998, 2000, 2004, 2006, 2008, 2012-2017 Free Software Fo
undation, Inc.

  This program is free software; you can redistribute it and/or modify
  it under the terms of the GNU General Public License as published by
  the Free Software Foundation; either version 3 of the License , or
  (at your option) any later version.

  This program is distributed in the hope that it will be useful,
  but WITHOUT ANY WARRANTY; without even the implied warranty of
  MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
  GNU General Public License for more details.

  You should have received a copy of the GNU General Public License
  along with this program.  If not, write to

    The Free Software Foundation, Inc.
    51 Franklin Street, Fifth Floor
    Boston, MA 02110-1335  USA

5+5
10
```

3. Lakukan percobaan-percobaan di bawah ini kemudian analisa hasil percobaan.
4. Selesaikan soal-soal latihan.

## Percobaan 1 : Status Proses

6. Instruksi `ps` (*process status*) digunakan untuk melihat kondisi proses yang ada. PID adalah Nomor Identitas Proses, TTY adalah nama terminal dimana proses tersebut aktif, STAT berisi S (*Sleeping*) dan R (*Running*), COMMAND merupakan instruksi yang digunakan.

➤ `$ ps`

```
deanissa@DESKTOP-9U498E1:~$ ps
  PID TTY          TIME CMD
   15 tty1      00:00:00 bash
  254 tty1      00:00:00 ps
```

7. Untuk melihat faktor/elemen lainnya, gunakan option `-u` (*user*). %CPU adalah presentasi CPU time yang digunakan oleh proses tersebut, %MEM adalah presentasi system memori yang digunakan proses, SIZE adalah jumlah memori yang digunakan, RSS (*Real System Storage*) adalah jumlah memori yang digunakan, START adalah kapan proses tersebut diaktifkan

`$ ps -u <user>`

8. Mencari proses yang spesifik pemakai. Proses diatas hanya terbatas pada proses milik pemakai, dimana pemakai tersebut melakukan login

➤ `$ ps -u <user>`

```
deanissa@DESKTOP-9U498E1:~$ ps -u deanissa
  PID TTY          TIME CMD
   15 tty1      00:00:00 bash
  255 tty1      00:00:00 ps
```

9. Mencari proses lainnya gunakan opsi `a` (*all*) dan `au` (*all user*)

➤ `$ ps -a`

```
deanissa@DESKTOP-9U498E1:~$ ps -a
  PID TTY          TIME CMD
   15 tty1      00:00:00 bash
  256 tty1      00:00:00 ps
```

➤ `$ ps -au`

```
deanissa@DESKTOP-9U498E1:~$ ps -au
USER      PID %CPU %MEM    VSZ   RSS TTY      STAT START   TIME COMMAND
root         1  0.0  0.0   920    28 tty1    Ss   19:56   0:00 /init
deanissa   15  0.0  0.0  14244  3800 tty1    S   19:56   0:00 bash
deanissa  257  0.0  0.0  15520  1928 tty1    R   20:25   0:00 ps -au
deanissa@DESKTOP-9U498E1:~$
```

## Percobaan 2 : Menampilkan Hubungan Proses Parent dan Child

6. Ketik **ps -eH** dan tekan **Enter**. Opsi **e** memilih semua proses dan opsi **H** menghasilkan tampilan proses secara hierarki. Proses child muncul dibawah proses parent. Proses child ditandai dengan awalan beberapa spasi.

➤ `$ ps -eH`

```
deanissa@DESKTOP-9U498E1:~$ ps -eH
PID TTY          TIME CMD
  1 ?            00:00:00 init
 14 tty1         00:00:00  init
 15 tty1         00:00:00  bash
258 tty1         00:00:00    ps
deanissa@DESKTOP-9U498E1:~$
```

7. Ketik **ps -e f** dan tekan **Enter**. Tampilan serupa dengan langkah 2. Opsi **-f** akan menampilkan status proses dengan karakter grafis (`\` dan `_`)

➤ `$ ps -e f`

```
deanissa@DESKTOP-9U498E1:~$ ps -e f
PID TTY          STAT TIME COMMAND
  1 ?            Ssl  0:00 /init
 14 tty1         Ss   0:00 /init
 15 tty1         S    0:00 \_ -bash
259 tty1         R    0:00 \_ ps -e f
deanissa@DESKTOP-9U498E1:~$
```

8. Ketik **pstree** dan tekan **Enter**. Akan ditampilkan semua proses pada sistem dalam bentuk hirarki parent/child. Proses parent di sebelah kiri proses child. Sebagai contoh proses `init` sebagai parent (*ancestor*) dari semua proses pada sistem. Beberapa child dari `init` mempunyai child. Proses `login` mempunyai i proses `bash` sebagai child. Proses `bash` mempunyai proses child `startx`. Proses `startx` mempunyai child `xinit` dan seterusnya. `$ pstree`

```
deanissa@DESKTOP-9U498E1:~$ pstree
init--init--bash--pstree
  |
  {init}
```

9. Ketik **pstree | grep mingetty** dan tekan **Enter**. Akan menampilkan semua proses `mingetty` yang berjalan pada system yang berupa *console virtual*. Selain menampilkan semua proses, proses dikelompokkan dalam satu baris dengan suatu angka sebagai jumlah proses yang berjalan.

➤ `$ pstree | grep <kalkulator>`

```
deanissa@DESKTOP-9U498E1:~$ pstree | grep <kalkulator>
-bash: syntax error near unexpected token `newline'
```

10. Untuk melihat semua PID untuk proses gunakan opsi **-p**.

➤ `$ pstree -p`

```
deanissa@DESKTOP-9U498E1:~$ pstree -p
init(1)─init(14)─bash(15)─pstree(271)
      │
      └─{init}(7)
```

11. Untuk menampilkan proses dan ancestor yang tercetak tebal gunakan opsi **-h**.

➤ `$ pstree -h`

```
deanissa@DESKTOP-9U498E1:~$ pstree -h
init─init─bash─pstree
  │
  └─{init}
```

### Percobaan 3 : Menampilkan Status Proses dengan Berbagai Format

9. Ketik **ps -e | more** dan tekan **Enter**. Opsi **-e** menampilkan semua proses dalam bentuk 4 kolom : PID, TTY, TIME dan CMD.

➤ `$ ps -e | more`

```
deanissa@DESKTOP-9U498E1:~$ ps -e | more
PID TTY          TIME CMD
  1 ?            00:00:00 init
 14 tty1        00:00:00 init
 15 tty1        00:00:00 bash
273 tty1        00:00:00 ps
274 tty1        00:00:00 more
```

10. Ketik **ps ax | more** dan tekan **Enter**. Opsi **a** akan menampilkan semua proses yang dihasilkan terminal (TTY). Opsi **x** menampilkan semua proses yang tidak dihasilkan terminal. Secara logika opsi ini sama dengan opsi **-e**. Terdapat 5 kolom : PID, TTY, STAT, TIME dan COMMAND.

➤ `$ ps ax | more`

```
deanissa@DESKTOP-9U498E1:~$ ps ax | more
PID TTY      STAT   TIME COMMAND
  1 ?        Ss      0:00 /init
 14 tty1    Ss      0:00 /init
 15 tty1    S       0:00 -bash
275 tty1    R       0:00 ps ax
276 tty1    S       0:00 more
```

11. Ketik **ps -ef | more** dan tekan **Enter**. Opsi **-ef** akan menampilkan semua proses dalam format daftar penuh.

➤ `$ ps -ef | more`

```
deanissa@DESKTOP-9U498E1:~$ ps -ef | more
PID TTY      STAT   TIME COMMAND
 15 tty1    S       0:00 -bash HOSTTYPE=x86_64 LANG=C.UTF-8 PATH=/usr/local/sbin:
/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin:/usr/games:/usr/local/games:/mnt/c/Pro
gram Files/WindowsApps/CanonicalGroupLimited.Ubuntu_2204.3.49.0_x64_79rhkp1fndgsc:
/mnt/c/Windows/system32:/mnt/c/Windows:/mnt/c/Windows/System32/Wbem:/mnt/c/Windows/
System32/WindowsPowerShell/v1.0:/mnt/c/Windows/System32/OpenSSH/:/mnt/c/Program Fi
les/Java/jdk-19/bin:/mnt/c/Program Files/Java/jdk1.8.0_301/bin:/mnt/c/Program Files
/Git/cmd:/mnt/c/Users/TOSHIBA/AppData/Local/Microsoft/WindowsApps:/mnt/c/Users/TOSH
IBA/AppData/Local/Programs/Microsoft VS Code/bin TERM=xterm-256color WSLENV= NAME=D
ESKTOP-9U498E1 HOME=/home/deanissa USER=deanissa LOGNAME=deanissa SHELL=/bin/bash W
SL_DISTRO_NAME=Ubuntu
```

12. Ketik **ps -eo pid, cmd | more** dan tekan **Enter**. Opsi **-eo** akan menampilkan semua proses dalam format sesuai definisi user yaitu terdiri dari kolom PID dan CMD.

➤ `$ ps -eo pid,cmd | more`

```
deanissa@DESKTOP-9U498E1:~$ ps -eo pid,cmd | more
error: unknown user-defined format specifier "pid,cmd"

Usage:
ps [options]

Try 'ps --help <simple|list|output|threads|misc|all>'
or 'ps --help <s|l|o|t|m|a>'
for additional help text.

For more details see ps(1).
```

20. Ketik **ps -eo pid,ppid,%mem,cmd | more** dan tekan **Enter**. Akan menampilkan kolom PID, PPID dan %MEM. PPID adalah proses ID dari proses parent. %MEM menampilkan persentase memory system yang digunakan proses. Jika proses hanya menggunakan sedikit memory system akan ditampilkan 0.

➤ `$ ps -eo pid,ppid,%mem,cmd | more`

```
deanissa@DESKTOP-9U498E1:~$ ps -eo pid, ppid, %mem, cmd | more
error: improper list

Usage:
ps [options]

Try 'ps --help <simple|list|output|threads|misc|all>'
or 'ps --help <s|l|o|t|m|a>'
for additional help text.

For more details see ps(1).
```

**Percobaan 4 : Mengontrol proses pada shell**

1. Gunakan perintah `yes` yang mengirim output `y` yang tidak pernah berhenti

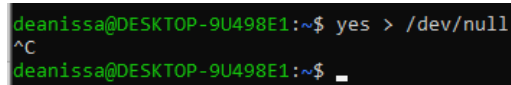
➤ `$ yes`



```
y
y
y
y
y
y
y
y
y
```

2. Belokkan standart output ke `/dev/null`

➤ `$ yes > /dev/null`

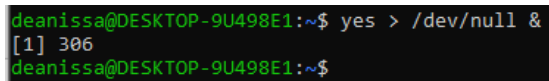


```
deanissa@DESKTOP-9U498E1:~$ yes > /dev/null
^C
deanissa@DESKTOP-9U498E1:~$ _
```

3. Salah satu cara agar perintah `yes` tetap dijalankan tetapi shell tetap digunakan untuk hal yang lain dengan meletakkan proses pada *background* dengan menambahkan karakter `&` pada akhir perintah.

➤ `$ yes > /dev/null &`

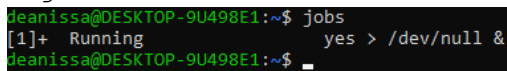
Angka dalam "[ ]" merupakan **job number** diikuti PID.



```
deanissa@DESKTOP-9U498E1:~$ yes > /dev/null &
[1] 306
deanissa@DESKTOP-9U498E1:~$
```

4. Untuk melihat status proses gunakan perintah `jobs`.

➤ `$ jobs`



```
deanissa@DESKTOP-9U498E1:~$ jobs
[1]+  Running                  yes > /dev/null &
deanissa@DESKTOP-9U498E1:~$ _
```

7. Untuk menghentikan job, gunakan perintah `kill` diikuti *job number* atau PID proses. Untuk identifikasi job number, diikuti prefix dengan karakter `%`. Pilih nomor job untuk genome kalkulator

➤ `$ kill %<nomor job>` contoh : `kill %1`



4. Lihat status job setelah diterminasi

➤ `$ jobs`

```
deanissa@DESKTOP-9U498E1:~$ kill %1
deanissa@DESKTOP-9U498E1:~$ jobs
[1]+  Terminated                  yes > /dev/null
```