

# **PRAKTIKUM SISTEM OPERASI**

## **Modul 8: System Call**



**ANNISA DWI PRASTIKA**

**L200210218**

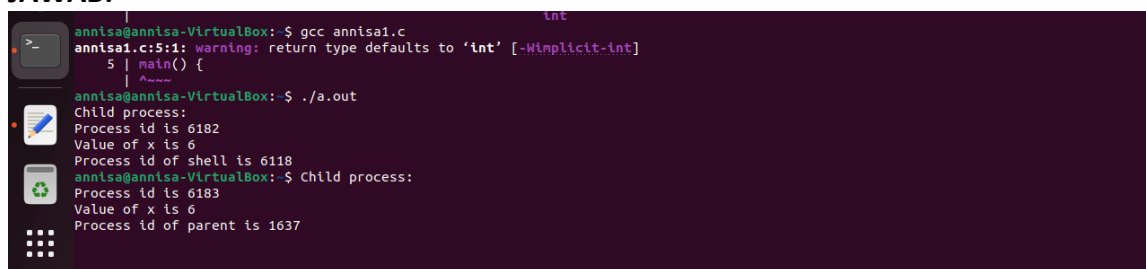
**PRODI TEKNIK INFORMATIKA FAKULTAS KOMUNIKASI DAN INFORMATIKA**

NIM	: L200210218	(Diisi oleh asisten)
Nama	: Annisa Dwi Prastika	
Kelas	: E	Tanda Tangan :
Dosen Penganpu	: Heru Setiya Nugraha, ST, M.Kom	
Tanggal Praktikum	: 6 Desember 2022	
Mata Kuliah	: Praktikum Sistem Operasi	

## LANGKAH KERJA

1. Membuat sebuah 'child process' (proses baru) dengan menggunakan system call 'fork'. Membuat program dengan algoritma sebagai berikut: (contoh program diberikan pada bagian berikutnya).
  - a. Deklarasi sebuah variabel x yang akan diakses bersama antara child proses dan parent proses.
  - b. Membuat sebuah child proses menggunakan system call fork.
  - c. Jika return value bernilai -1, tampilkan teks 'Pembuatan proses GAGAL', dilanjutkan dengan keluar program dengan perintah system call 'exit'.
  - d. Jika return value sama dengan 0 (NOL), Tampilkan teks 'Child Process', tampilkan ID proses dari child proses menggunakan perintah system call 'getpid', tampilkan nilai x, dan tampilkan ID proses parent dengan perintah system call 'getppid'.
  - e. Untuk nilai return value yang lainnya, tampilkan teks 'Parent process', tampilkan ID dari parent proses menggunakan perintah system call getpid, tampilkan nilai x, dan tampilkan ID dari proses shell menggunakan perintah system call getppid.
  - f. Stop

## JAWAB:



```

annisa@annisa-VirtualBox:~$ gcc annisa1.c
annisa1.c:5:1: warning: return type defaults to 'int' [-Wimplicit-int]
  5 | main() {
    | ^~~~~~
annisa@annisa-VirtualBox:~$ ./a.out
Child process:
Process id is 6182
Value of x is 6
Process id of shell is 6118
annisa@annisa-VirtualBox:~$ Child process:
Process id is 6183
Value of x is 6
Process id of parent is 1637

```

2. Menghentikan sementara (block) proses parent sampai dengan proses child selesai, menggunakan perintah system call 'wait'. Membuat program dengan algoritma sebagai berikut, contoh program diberikan pada bagian berikutnya.
  - a. Membuat sebuah child proses menggunakan sytem call 'fork'.

- b. Jika return value bernilai -1, selanjutnya tampilkan teks 'pembuatan proses gagal', dan keluar program dengan menggunakan perintah system call 'exit'.
- c. Jika return value berupa angka positif ( $> 0$ ), 'pause' hentikan sementara 'parent' proses tunggu sampai child proses berakhir dengan menggunakan perintah system call 'wait'. Tampilkan teks 'Parent starts', selanjutnya tampilkan nomor genap mulai dari 0 s/d 10, terakhir tampilkan teks 'Parent end'.
- d. Jika return value bernilai 0 (NOL), tampilkan teks 'Child start', tampilkan nomor ganjil mulai dari 0 s/d 10, selanjutnya tampilkan teks 'child ends'
- e. Stop

#### JAWAB:

```
annisa@annisa-VirtualBox:~$ gcc nomer2.c
nomer2.c:6:1: warning: return type defaults to 'int' [-Wimplicit-int]
  6 | main() {
    | ~~~~~
annisa@annisa-VirtualBox:~$ ./a.out
Child starts
Nomor Ganjil: 1 3 5 7 9
Child ends

Parent starts
Nomor Genap: 2 4 6 8 10
Parent ends
annisa@annisa-VirtualBox:~$
```

3. Loading program yang dapat dieksekusi dalam sebuah 'child' proses menggunakan perintah system call 'exec'. Membuat program dengan algoritma sebagai berikut: (contoh program diberikan pada bagian berikutnya).
  - a. Jika terdapat 3 argumen dalam command-line berhenti (stop).
  - b. Membuat child proses dengan perintah system call 'fork'
  - c. Jika return value adalah -1, selanjutnya tampilkan teks 'Pembuatan proses Gagal', dan keluar program dengan perintah system call exit.
  - d. Jika return value  $> 0$  (positif), selanjutnya hentikan parent-proses sementara hingga child-proses berakhir dengan menggunakan perintah system call wait. Tampilkan teks 'Child berakhir', dan hentikan parent-proses.
  - e. Jika return value sama dengan 0 (NOL), selanjutnya tampilkan teks 'Child starts', load program dari lokasi yang diberikan dalam 'path' ke dalam child-proses, menggunakan perintah system call 'exec'. Jika return value dari perintah 'exec' adalah bilangan negatif, tampilkan error yang terjadi dan stop. Hentikan childproses.
  - f. Stop

#### JAWAB:

```
annisa@annisa-VirtualBox:~$ gcc nomer3.c
nomer3.c: In function 'main':
nomer3.c:21:17: warning: missing sentinel in function call [-Wformat=]
  21 |         i = execl(argv[1], argv[2], 0);
    |         ^
nomer3.c:28:17: warning: implicit declaration of function 'wait' [-Wimplicit-function-declaration]
  28 |         wait(NULL);
    |         ~~~~
annisa@annisa-VirtualBox:~$ ./a.out /bin/ls ls
child process
annisa1.c  annisa8  Desktop  Downloads  nano.4332.save  nomer3.c  nomer5.c  Public  Templates  Videos
annisa.c  a.out    Documents  Music      nomer2.c      nomer4.c  Pictures  snap    'Untitled Document 1'  wait.c
child terminated
annisa@annisa-VirtualBox:~$
```

4. Menampilkan status file menggunakan perintah system call 'stat' Membuat program dengan algoritma sebagai berikut (contoh code ada di bagian berikutnya):
  - a. Gunakan 'nama file' yang diberikan melalui argumen dalam perintah commandline.
  - b. Jika 'nama-file' tidak ada maka stop disini (keluar program)
  - c. Panggil system call 'stat' pada 'nama-file' tersebut yang akan mengembalikan sebuah struktur.
  - d. Tampilkan informasi mengenai st\_uid, st\_blksize, st\_block, st\_size, st\_nlink, etc.
  - e. Ubah waktu dalam st\_time, st\_mtime dengan menggunakan fungsi ctime.
  - f. Bandingkan st\_mode dengan konstanta mode seperti S\_IRUSR, S\_IWGRP, S\_IXOTH dan tampilkan informasi mengenai 'file-permissions'.
  - g. Stop

**JAWAB:**

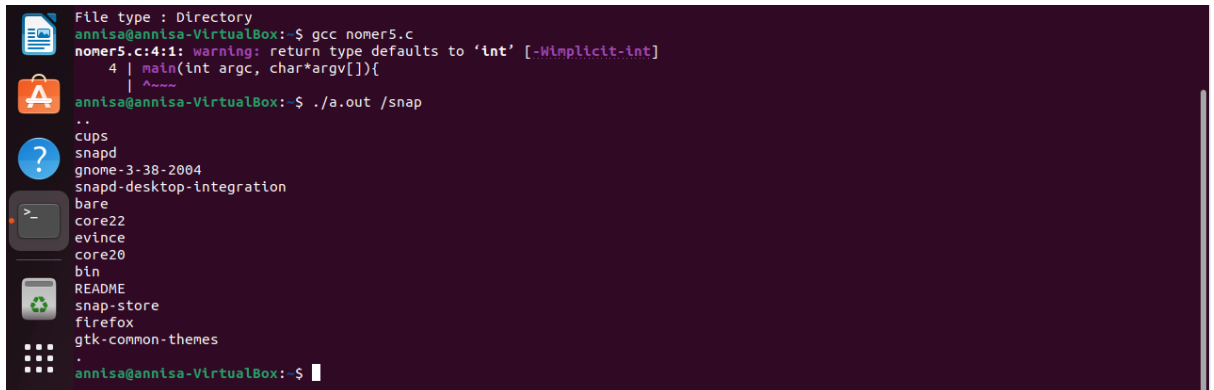
```

annisa@annisa-VirtualBox: ~
annisa@annisa-VirtualBox:~$ gcc nomer4.c
nomer4.c: In function 'main':
nomer4.c:19:31: warning: format '%d' expects argument of type 'int', but argument 2 has type '__blksize_t' {aka 'long int'} [-Wformat=]
19 |         printf("Block size : %d\n", file.st_blksize);
   |                               ^~
   |                               |
   |                               int      __blksize_t {aka long int}
   |                               %ld
nomer4.c:20:36: warning: format '%d' expects argument of type 'int', but argument 2 has type '__blkcnt_t' {aka 'long int'} [-Wformat=]
20 |         printf("Block allocated : %d\n", file.st_blocks);
   |                                ^~
   |                                |
   |                                int      __blkcnt_t {aka long int}
   |                                %ld
nomer4.c:21:29: warning: format '%d' expects argument of type 'int', but argument 2 has type '__ino_t' {aka 'long unsigned int'} [-Wformat=]
21 |         printf("Inode no.: %d\n", file.st_ino);
   |                          ^~
   |                          |
   |                          int      __ino_t {aka long unsigned int}
   |                          %ld
nomer4.c:24:29: warning: format '%d' expects argument of type 'int', but argument 2 has type '__off_t' {aka 'long int'} [-Wformat=]
24 |         printf("File size: %d bytes\n", file.st_size);
   |                          ^~
   |                          |
   |                          int      __off_t {aka long int}
   |                          %ld
nomer4.c:25:33: warning: format '%d' expects argument of type 'int', but argument 2 has type '__nlink_t' {aka 'long unsigned int'} [-Wformat=]
25 |         printf("No. of links : %d\n", file.st_nlink);
   |                          ^~
   |                          |
   |                          int      __nlink_t {aka long unsigned int}
   |                          %ld
nomer4.c:25:33: warning: format '%d' expects argument of type 'int', but argument 2 has type '__nlink_t' {aka 'long unsigned int'} [-Wformat=]
25 |         printf("No. of links : %d\n", file.st_nlink);
   |                          ^~
   |                          |
   |                          int      __nlink_t {aka long unsigned int}
   |                          %ld
annisa@annisa-VirtualBox:~$ ./a.out /snap
User id : 0
Group id : 0
Block size : 4096
Block allocated : 8
Inode no.: 1048577
Last accessed : Tue Aug 9 18:55:27 2022
Last modified: Tue Nov 29 08:38:25 2022
File size: 4096 bytes
No. of links : 14
Permissions : drwxr-xr-x
File type : Directory
annisa@annisa-VirtualBox:~$
  
```

5. Menampilkan isi direktori menggunakan perintah system call 'readdir' Membuat program dengan algoritma sebagai berikut (contoh code ada di bagian berikutnya):

- a. Gunakan 'nama-direktori' yang diberikan sebagai argumen pada command-line.
- b. Jika direktori tidak ditemukan stop, keluar program
- c. Buka direktori menggunakan perintah system call 'opendir' yang akan menghasilkan sebuah struktur.
- d. Baca direktori menggunakan perintah system call 'readdir' yang juga akan menghasilkan struktur data.
- e. Tampilkan d\_name (nama direktori)
- f. Akhiri pembacaan direktori dengan perintah system call 'closedir'.
- g. Stop

#### JAWAB:



```
File type : Directory
annisa@annisa-VirtualBox:~$ gcc nomer5.c
nomer5.c:4:1: warning: return type defaults to 'int' [-Wimplicit-int]
    4 | main(int argc, char*argv[]){
      | ^~~~~
annisa@annisa-VirtualBox:~$ ./a.out /snap
..
cups
snapd
gnome-3-38-2004
snapd-desktop-integration
bare
core22
evince
core20
bin
README
snap-store
firefox
gtk-common-themes
.
annisa@annisa-VirtualBox:~$
```