

**LAPORAN PRAKTIKUM**  
**PRAKTIKUM KE-3:**  
**MANAJEMEN PROSES DI LINUX**



**Disusun oleh:**  
**Nouvella Rahma Fitrah Legarsi**  
**24060124120029**

**PRAKTIKUM SISTEM OPERASI**  
**LAB A2**

**DEPARTEMEN INFORMATIKA**  
**FAKULTAS SAINS DAN MATEMATIKA**  
**UNIVERSITAS DIPONEGORO**  
**2025**

## BAB III

## PEMBAHASAN

### 1.1. Screenshot Program dan Penjelasan

a. Lab1.c

[illegible]

Output :

```
nouvella_legarsi@DESKTOP-GM715F0:~/praktikum3$ vi Lab1.c
nouvella_legarsi@DESKTOP-GM715F0:~/praktikum3$ gcc Lab1.c -o Lab1
nouvella_legarsi@DESKTOP-GM715F0:~/praktikum3$ ./Lab1
Hello World
<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<
I am after forking
    I am process 8983.
<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<
>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
I am after forking
    I am process 8984.
>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
nouvella_legarsi@DESKTOP-GM715F0:~/praktikum3$ _
```

Penjelasan baris kode yang terdapat dalam program di atas :

BARIS KODE	FUNGSI
#include <stdio.h>	Untuk fungsi printf( )
#include <unistd.h>	Untuk fungsi fork( ) dan getpid( )
printf("Hello World\n");	Menampilkan teks sekali sebelum proses dipecah





```

nouvella_legarsi@DESKTOP-GM715F0:~/praktikum3$ vi Lab3.c
nouvella_legarsi@DESKTOP-GM715F0:~/praktikum3$ gcc Lab3.c -o Lab3
nouvella_legarsi@DESKTOP-GM715F0:~/praktikum3$ ./Lab3
Here I am just before first forking statement
#####
Here I am just after first forking statement
#####
Here I am just after first forking statement
Here I am just after second forking statement
        Hello World from process 11527!
Here I am just after second forking statement
        Hello World from process 11528!
Here I am just after second forking statement
        Hello World from process 11529!
Here I am just after second forking statement
        Hello World from process 11530!
nouvella_legarsi@DESKTOP-GM715F0:~/praktikum3$ █

```

Penjelasan baris kode yang terdapat dalam program di atas :

BARIS KODE	FUNGSI
#include <stdio.h>	Untuk fungsi printf( )
#include <unistd.h>	Untuk fungsi fork( ) dan getpid( )
printf("Here I am just before first forking statement\n");	Menampilkan pesan ke layar sebelum pemanggilan fork() pertama
fork();	Membuat proses baru (clone dari proses yang sedang berjalan)
printf("#####\n");	Menampilkan garis pembatas di output
printf("Here I am just after first forking statement\n");	Menampilkan pesan setelah fork pertama, untuk menandai posisi eksekusi
fork();	Membuat proses baru lagi dari setiap proses yang ada
printf("#####\n");	Dicetak oleh <i>parent &amp; child</i>
printf("Here I am just after second forking statement\n");	Menampilkan pesan setelah fork kedua
printf("\t\tHello World from process %d!\n", getpid());	Menampilkan pesan beserta ID proses yang menjalankannya

getpid()	Mengambil ID dari proses yang sedang berjalan
----------	---

d. Lab4.c

```
#include <stdlib.h>
#include <stdio.h>
#include <unistd.h>
#include <sys/wait.h>

void main(void)
{
    int pid;
    int status;

    printf("Hello World!\n");
    pid = fork();
    if (pid == -1)
    {
        perror("bad fork");
        exit(1);
    }
    if (pid == 0) printf("I am the child process.\n");
    else {
        wait(&status);
        printf("I am the parent process.\n");
    }
}
```

Output :

```
nouvella_legarsi@DESKTOP-GM715F0:~/praktikum3$ vi Lab4.c
nouvella_legarsi@DESKTOP-GM715F0:~/praktikum3$ gcc Lab4.c -o Lab4
nouvella_legarsi@DESKTOP-GM715F0:~/praktikum3$ ./Lab4
Hello World!
I am the child process.
I am the parent process.
nouvella_legarsi@DESKTOP-GM715F0:~/praktikum3$
```

Penjelasan baris kode yang terdapat dalam program di atas :

BARIS KODE	FUNGSI
#include <stdlib.h>	Menyediakan fungsi seperti exit() dan utilitas umum
#include <stdio.h>	Untuk fungsi printf( )
#include <unistd.h>	Untuk fungsi fork( ) dan getpid( )

<code>#include &lt;sys/wait.h&gt;</code>	Menyediakan fungsi <code>wait()</code> untuk menunggu proses <i>child</i> selesai
<code>int pid;</code>	Untuk menyimpan nilai hasil dari <code>fork()</code>
<code>int status;</code>	Untuk menyimpan status hasil proses <i>child</i>
<code>printf("Hello World!\n");</code>	Menampilkan pesan ke layar sebelum proses di- <i>fork</i>
<code>pid = fork();</code>	Membuat proses baru ( <i>child</i> ) dari proses <i>parent</i>
<code>if (pid == -1)</code>	Mengecek apakah <code>fork()</code> gagal
<code>perror("bad fork");</code>	Menampilkan pesan error jika <code>fork</code> gagal
<code>exit(1);</code>	Menghentikan program jika <code>fork</code> gagal
<code>if (pid == 0)</code>	Mengecek apakah ini proses <i>child</i>
<code>printf("I am the child process.\n");</code>	Ditampilkan hanya oleh proses <i>child</i>
<code>else</code>	Bagian untuk proses <i>parent</i>
<code>wait(&amp;status);</code>	Membuat <i>parent</i> menunggu sampai proses <i>child</i> selesai
<code>printf("I am the parent process.\n");</code>	Ditampilkan oleh proses <i>parent</i> setelah <i>child</i> selesai

e. Lab5.c

```

#include <stdio.h>
#include <unistd.h>
#include <stdlib.h>
#include <sys/wait.h>

int main(){
    int forkresult;

    printf("%d: I am the parent. Remember my number!\n", getpid());
    printf("%d: I am new going to fork ... \n", getpid());
    forkresult = fork();
    printf("#####\n");
    if (forkresult != 0) {
        printf("%d: My child's pid is %d\n", getpid(), forkresult);
    }
    else
    {
        printf("%d: Hi! I am the child.\n", getpid());
    }
    printf("%d: like father like son. \n", getpid());
}

```

Output :

```

nouvella_legarsi@DESKTOP-GM715F0:~/praktikum3$ vi Lab5.c
nouvella_legarsi@DESKTOP-GM715F0:~/praktikum3$ gcc Lab5.c -o Lab5
nouvella_legarsi@DESKTOP-GM715F0:~/praktikum3$ ./Lab5
12239: I am the parent. Remember my number!
12239: I am new going to fork ...
#####
12239: My child's pid is 12240
12239: like father like son.
#####
12240: Hi! I am the child.
12240: like father like son.
nouvella_legarsi@DESKTOP-GM715F0:~/praktikum3$

```

Penjelasan baris kode yang terdapat dalam program di atas :

BARIS KODE	FUNGSI
#include <stdio.h>	Untuk fungsi printf( )
#include <unistd.h>	Untuk fungsi fork( ) dan getpid( )
#include <stdlib.h>	Menyediakan utilitas umum (misal exit( ))
#include <sys/wait.h>	Menyediakan fungsi wait() untuk menunggu proses child selesai
Int forkresult;	Variabel untuk menyimpan nilai kembalian dari fork()

<code>printf("%d: I am the parent...");</code>	Menampilkan PID proses yang sedang berjalan
<code>getpid()</code>	Mengambil ID proses yang sedang berjalan
<code>printf("I am now going to fork...");</code>	Pesan sebelum memanggil <code>fork()</code>
<code>forkresult = fork();</code>	Membuat proses baru ( <i>child</i> )
<code>printf("#####");</code>	Dicetak oleh <i>parent</i> dan <i>child</i> setelah <i>fork</i>
<code>if (forkresult != 0)</code>	Mengecek apakah ini proses <i>parent</i> (di <i>parent</i> , $\text{forkresult} \neq 0$ )
<code>printf("My child's pid...");</code>	Dicetak oleh <i>parent</i> , menampilkan PID proses <i>child</i>
<code>else</code>	Blok yang dijalankan oleh proses <i>child</i>
<code>printf("Hi! I am the child.");</code>	Dicetak oleh <i>child</i>
<code>printf("like father like son...");</code>	Dicetak oleh kedua proses

f. Lab6.c

```

#include <stdio.h>
#include <unistd.h>
#include <sys/types.h>

int main(void)
{
    int pid;

    printf("I'am the original process with PID %d and PPID %d.\n",
    getpid(), getppid());
    pid = fork();
    printf("#####\n");
    if (pid != 0)
    {
        printf("I'am the parent with PID %d and PPID %d.\n",
        getpid(), getppid());
        printf("My child's PID is %d\n", pid);

    }
    else
    {
        sleep(4);
        printf("I'm the child with PID %d and PPID %d.\n",
        getpid(), getppid());
    }
    printf ("PID %d terminates.\n", getpid());
}

```

Output :

```

nouvelle_legarsi@DESKTOP-GM715F0:~/praktikum3$ vi Lab6.c
nouvelle_legarsi@DESKTOP-GM715F0:~/praktikum3$ gcc Lab6.c -o Lab6
nouvelle_legarsi@DESKTOP-GM715F0:~/praktikum3$ ./Lab6
I'am the original process with PID 12783 and PPID 119.
#####
I'am the parent with PID 12783 and PPID 119.
My child's PID is 12784
PID 12783 terminates.
#####
nouvelle_legarsi@DESKTOP-GM715F0:~/praktikum3$ I'm the child with PID 12784 and PPID 118.
PID 12784 terminates.

```

Penjelasan baris kode yang terdapat dalam program di atas :

BARIS KODE	FUNGSI
#include <stdio.h>	Untuk fungsi printf( )
#include <unistd.h>	Untuk fungsi fork( ) dan getpid( )
#include <sys/wait.h>	Menyediakan fungsi wait() untuk menunggu proses child selesai
int pid;	Variabel untuk menyimpan nilai return dari fork( )

<code>printf("I'm the original...")</code>	Menampilkan PID dan PPID proses awal (sebelum <i>fork</i> )
<code>getpid()</code>	Mengambil ID proses yang sedang berjalan
<code>getppid()</code>	Mengambil ID proses induk
<code>pid = fork();</code>	Membuat proses baru ( <i>child</i> ). Parent mendapat PID <i>child</i> , <i>child</i> mendapat 0
<code>printf("#####");</code>	Dicetak oleh <i>parent</i> dan <i>child</i> setelah <i>fork</i>
<code>if (pid != 0)</code>	Mengecek apakah ini proses <i>parent</i>
<code>printf("I'm the parent...")</code>	Baris ini hanya dijalankan oleh <i>parent</i>
<code>printf("My child's PID...")</code>	Menampilkan PID <i>child</i> , hanya di <i>parent</i>
<code>else</code>	Bagian untuk proses <i>child</i>
<code>sleep(4);</code>	Menunda eksekusi <i>child</i> selama 4 detik
<code>printf("I'm the child...")</code>	Ditampilkan hanya oleh <i>child</i>
<code>printf("PID %d terminates")</code>	Mencetak PID masing-masing proses saat akan berakhir

g. Lab7.c

```

#include <stdio.h>
#include <unistd.h>
#include <stdlib.h>

int main()
{
    int pid;
    pid = fork();
    if (pid != 0)
    {
        while(1)
            sleep(100);
    }
    else
    {
        exit(42);
    }
}

```

```

nouvella_legarsi@DESKTOP-GM715F0:~/praktikum3$ gcc Lab7.c -o zombie
nouvella_legarsi@DESKTOP-GM715F0:~/praktikum3$ ./zombie &
[1] 13131
nouvella_legarsi@DESKTOP-GM715F0:~/praktikum3$ ps
  PID TTY          TIME CMD
  119 pts/0        00:00:10 bash
 13131 pts/0        00:00:00 zombie
 13132 pts/0        00:00:00 zombie
 13134 pts/0        00:00:00 ps
nouvella_legarsi@DESKTOP-GM715F0:~/praktikum3$ kill 13131
nouvella_legarsi@DESKTOP-GM715F0:~/praktikum3$ ps
  PID TTY          TIME CMD
  119 pts/0        00:00:10 bash
 13149 pts/0        00:00:00 ps
[1]+  Terminated                  ./zombie
nouvella_legarsi@DESKTOP-GM715F0:~/praktikum3$

```

Tidak ada output yang keluar karena program memang tidak mencetak apapun. Tujuan dari program ini adalah untuyk mendemonstrasikan zombie process. Penjelasan baris kode yang terdapat dalam program di atas :

BARIS KODE	FUNGSI
#include <stdio.h>	Untuk fungsi printf( )

<code>#include &lt;unistd.h&gt;</code>	Untuk fungsi <code>fork()</code> dan <code>getpid()</code>
<code>#include &lt;sys/wait.h&gt;</code>	Menyediakan fungsi <code>wait()</code> untuk menunggu proses <i>child</i> selesai
<code>int pid;</code>	Variabel untuk menyimpan nilai hasil dari <code>fork()</code>
<code>pid = fork();</code>	Membuat proses baru ( <i>child</i> ). <i>Parent</i> dapat PID <i>child</i> , <i>child</i> dapat 0
<code>if (pid != 0)</code>	Mengecek apakah proses ini adalah <i>parent</i>
<code>while(1)</code>	Loop tanpa henti (infinite loop)
<code>sleep(100);</code>	<i>Parent</i> tidur 100 detik setiap loop agar tidak menghabiskan CPU
<code>else</code>	Bagian khusus untuk proses <i>child</i>
<code>exit(42);</code>	<i>Child</i> langsung berhenti dengan kode keluar 42