

EVALUASI

1. Berdasarkan program **simpleFork.c**, jelaskan bagaimana proses mengetahui dirinya adalah induk atau anak?

Jawab :

Dalam pemanggilan fungsi `fork`, akan dibuat proses baru yang disebut proses anak (*child process*). Fungsi ini akan memberikan dua *return*, yaitu :

- a. Proses anak mendapat *return* nilai 0
- b. Proses induk mendapatkan *return* ID proses anak yang baru

Kemudian dicek nilai PID untuk mengetahui suatu proses termasuk induk atau anak, yaitu apabila PID bernilai kurang dari 0 maka proses `fork` gagal, apabila PID bernilai sama dengan 0 maka terjadi proses anak, sedangkan apabila PID bernilai lebih dari 0 maka terjadi proses induk.

Hasil :

```
Parent
Parent
Parent
Parent
Parent
Parent
Parent
Parent
Parent
Parent
Parent
Parent
Parent
Parent
Parent
Parent
Parent
Parent
Parent
Parent
Parent
Child Complete
sailor_moon@tata-VirtualBox:~$
```

2. Berdasarkan eksekusi **parent.c**, jelaskan tujuan pemanggilan fungsi `exec`!

Jawab :

Fungsi `exec` berguna untuk menggantikan program saat ini dengan program baru dimana fungsi ini akan men-*spawn* proses anak baru. Berdasarkan eksekusi `parent.c` digunakan `exec family`, yaitu `execl` yang digunakan untuk menjalankan `shell` sebagai pelaksana proses anak.

d. `execvp`

Sama seperti `execlp`, `execvp` merupakan program yang dimuat dimana **path-nya dicari** (mencari file executable pada tiap path). `Execvp` merupakan V&P family maka dilakukan *passing* dengan membuat line menjadi array of string dengan pointer variabel `argv`. Variabel `argv` di-*passing* ke `execute` kemudian fungsi `execute` akan memanggil `execvp`.

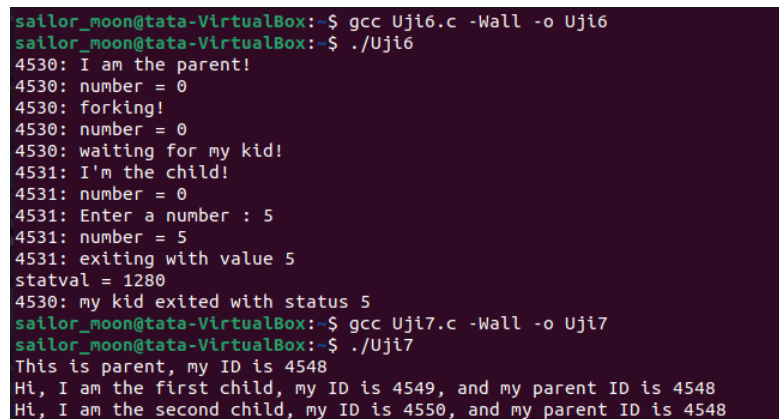
4. Apa fungsi argumen NULL pada argumen `exec`?

Jawab :

Pada `exec` family, fungsi yang memiliki huruf V menerima argumen untuk program yang dieksekusi sebagai NULL terminated array pointer ke string. Elemen terakhir dari array ini harus NULL dimana NULL digunakan untuk terminasi argumen-argumen dalam list.

5. Amati **uji6.c** apakah PID proses berubah ketika `exec` dieksekusi?

Jawab :



```
sailor_moon@tata-VirtualBox:~$ gcc Uji6.c -Wall -o Uji6
sailor_moon@tata-VirtualBox:~$ ./Uji6
4530: I am the parent!
4530: number = 0
4530: forking!
4530: number = 0
4530: waiting for my kid!
4531: I'm the child!
4531: number = 0
4531: Enter a number : 5
4531: number = 5
4531: exiting with value 5
statval = 1280
4530: my kid exited with status 5
sailor_moon@tata-VirtualBox:~$ gcc Uji7.c -Wall -o Uji7
sailor_moon@tata-VirtualBox:~$ ./Uji7
This is parent, my ID is 4548
Hi, I am the first child, my ID is 4549, and my parent ID is 4548
Hi, I am the second child, my ID is 4550, and my parent ID is 4548
```

PID proses tidak berubah ketika `exec` dieksekusi karena `exec` family mengganti kerja suatu proses menjadi proses lain dengan mempertahankan sedikit data aslinya, yaitu PID dan PPID.

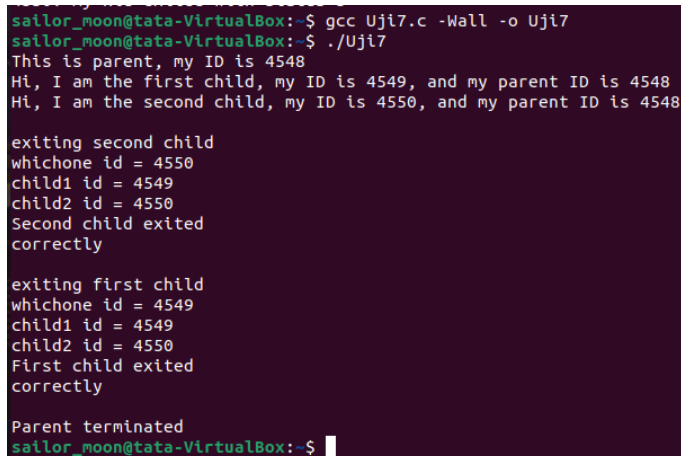
6. Berdasarkan **uji6.c** apakah terbukti bahwa sinyal yang dikirim oleh `exit()` ditangkap oleh `wait()`? Berikan penjelasannya!

Jawab :

Terbukti pada **uji6.c** bahwa sinyal yang dikirim oleh `exit()` ditangkap oleh `wait()`. Variabel `statval` akan menangkap sinyal yang dikirim oleh `child` menggunakan `wait(&statval)`, proses `child` akan terminasi dan `WEXITSTATUS(stataval)` akan dipanggil kemudian meng-extract `exit` status sehingga muncul `exit` status di terminal yang nilainya sama dengan apa yang user inputkan. Hal ini dapat dilihat pada baris terakhir dimana proses `parent` berhasil mencetak nilai kembalian dari proses `child` dengan benar.

7. Berdasarkan **uji7.c** jelaskan siapa `parent` kedua `child` yang dibuat? Jelaskan pula bagaimana `parent` dapat mengenali `child` nya?

Jawab :



```
sailor_moon@tata-VirtualBox:~$ gcc Uji7.c -Wall -o Uji7
sailor_moon@tata-VirtualBox:~$ ./Uji7
This is parent, my ID is 4548
Hi, I am the first child, my ID is 4549, and my parent ID is 4548
Hi, I am the second child, my ID is 4550, and my parent ID is 4548

exiting second child
whichone id = 4550
child1 id = 4549
child2 id = 4550
Second child exited
correctly

exiting first child
whichone id = 4549
child1 id = 4549
child2 id = 4550
First child exited
correctly

Parent terminated
sailor_moon@tata-VirtualBox:~$
```

Parent dari kedua `child` (`child1` dan `child2`), yaitu 4548. Melalui `whichone=(int)wait(&status)`, `parent` dapat mengenali `child`-nya dengan cara mencocokkan `whichone id` dengan `id` dari masing-masing `child`.