Nama : Nashirudin Baqiy

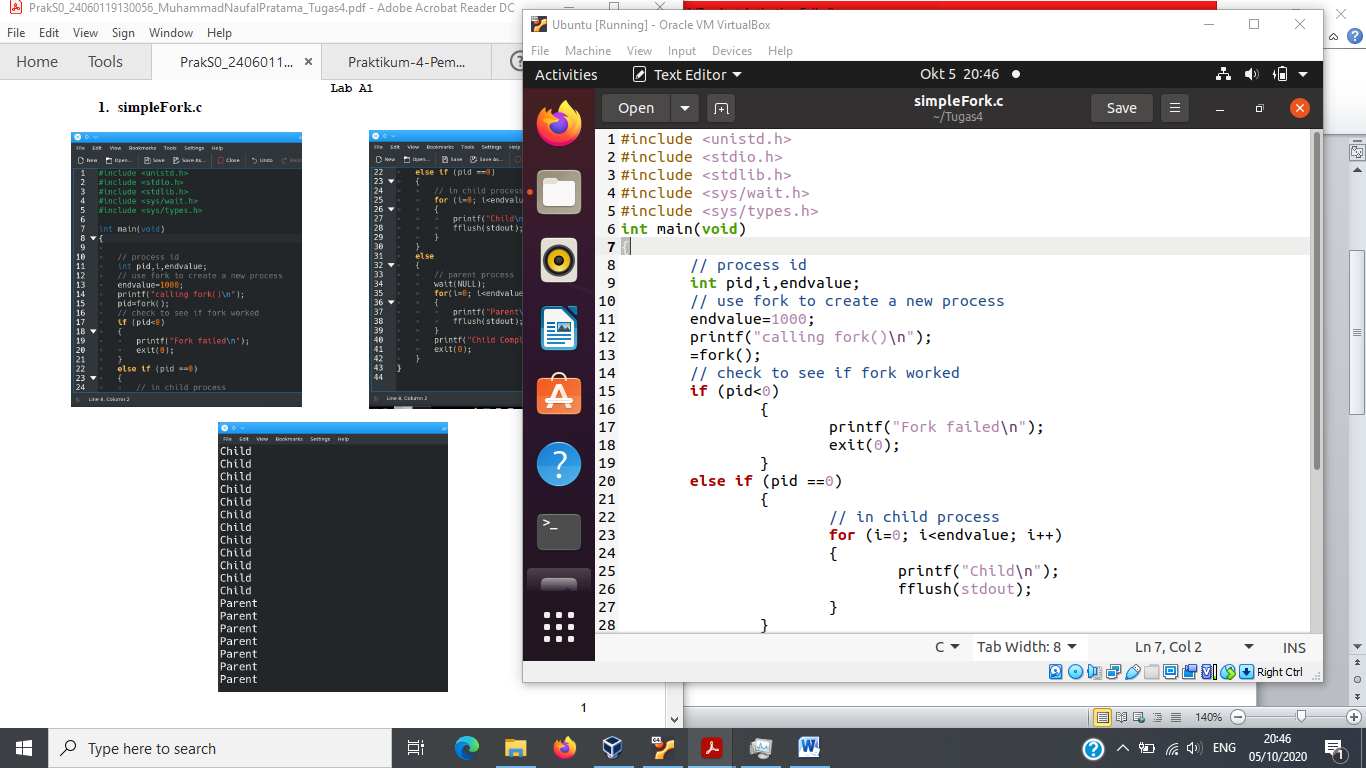
NIM : 24060119130045

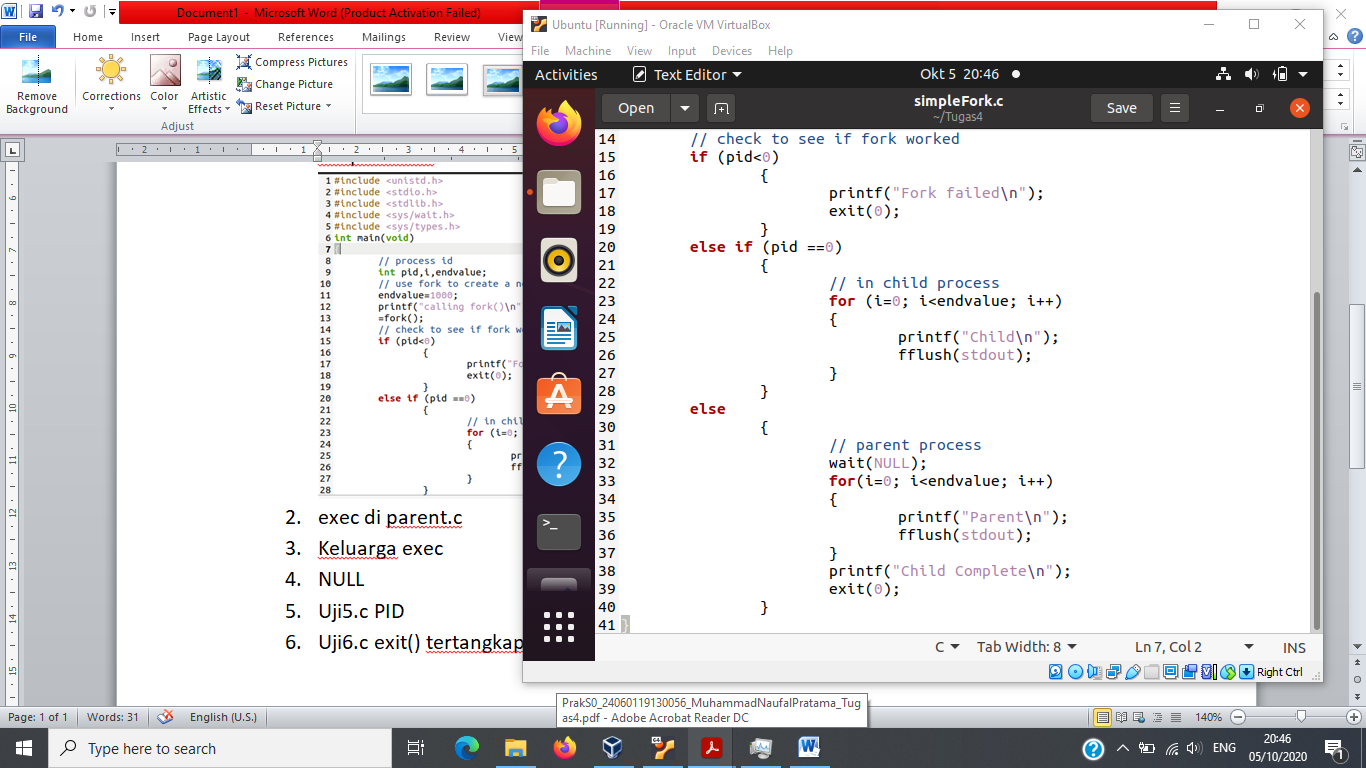
Kelas : A1

**Tugas 4**

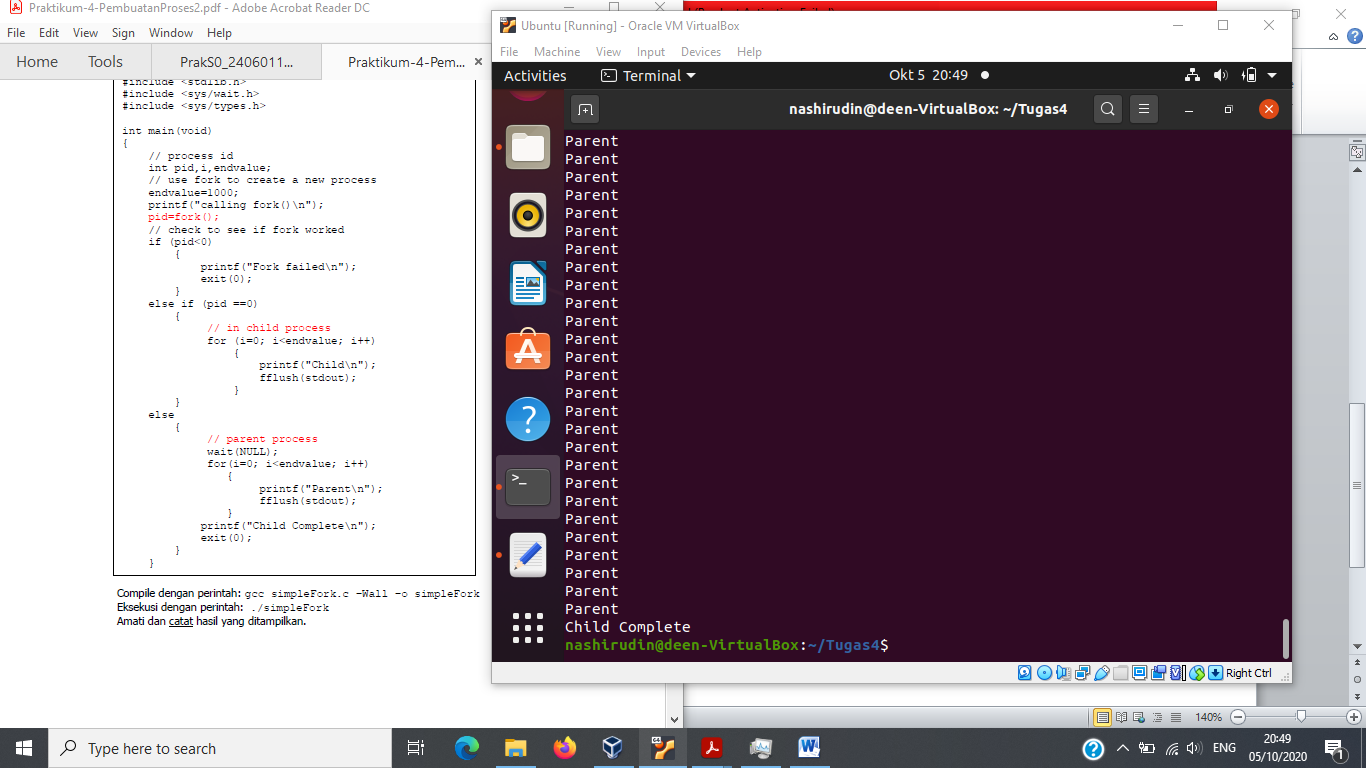
**Praktikum Sistem Operasi**

1. simpleFork.c



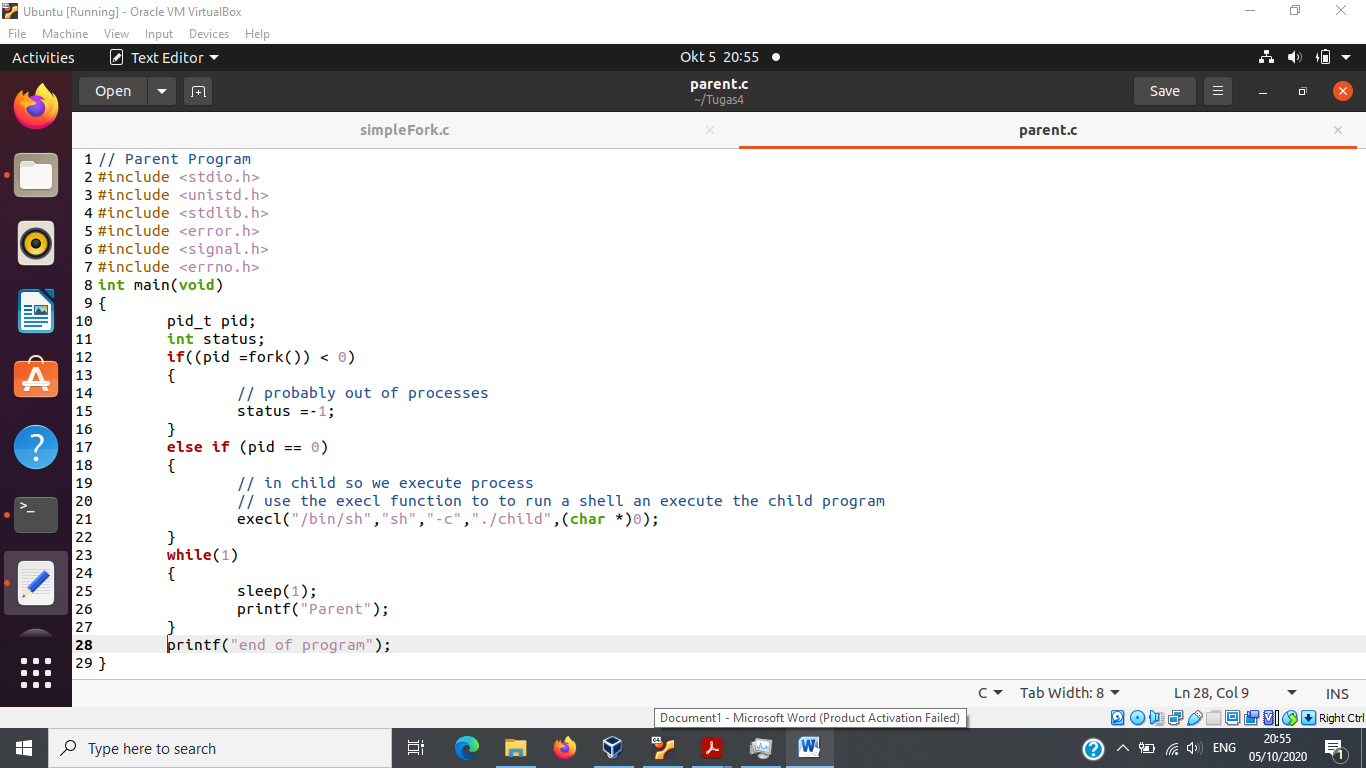


Output:

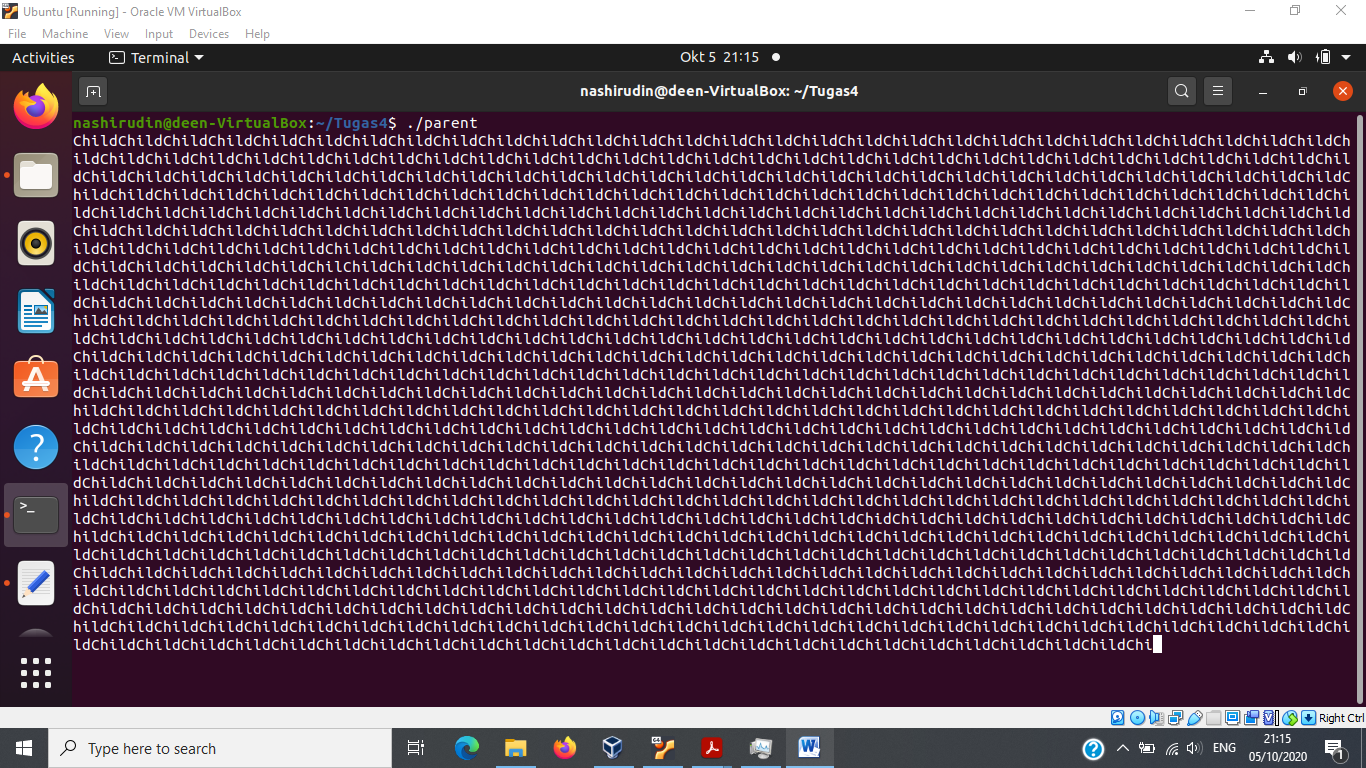


Mengetahui apakah dirinya sendiri sebagai proses parent atau child dari PID hasil forking. Child mengembalikan 0, parent mengembalikan PID childnya. Diketahuilah parent dan childnya, karena parent melakukan wait() yang menunggu proses child selesai.

1. exec di parent.c



Output:



Memastikan proses yang sedang berjalan lalu diganti proses exec. Kita memberi perintah sh yang memiliki fungsi command interpreter dengan flag -c membaca string sebagai command yang kita beri parameter ./child dan ./child dijalankan.

1. Keluarga exec
2. execl

Menjalankan dan mengeluarkan proses dari argument/command yang diberikan dan kita perlu untuk memberikan full path dimana command itu berada. Full path -> command -> flag (bila perlu) -> NULL

1. execpl

Hampir mirip execl sebelumnya, berbeda dibagian path yang diberi. Tidak perlu memberikan full path dari executable binary file tapi menggunakan path environment variable. Jika suatu command sudah terdaftar di path environment variable maka dapat digunakan tanpa menulis full path. Urutan penulisan sama dengan execl.

1. execv

Menerima array sebagai parameter, memerlukan full path agar dapat berjalan. Urutan penulisan sama seperti sebelumnya. Tidak ada NULL di Uji3.c karena NULL dijadikan bentuk terminated array argv[0].

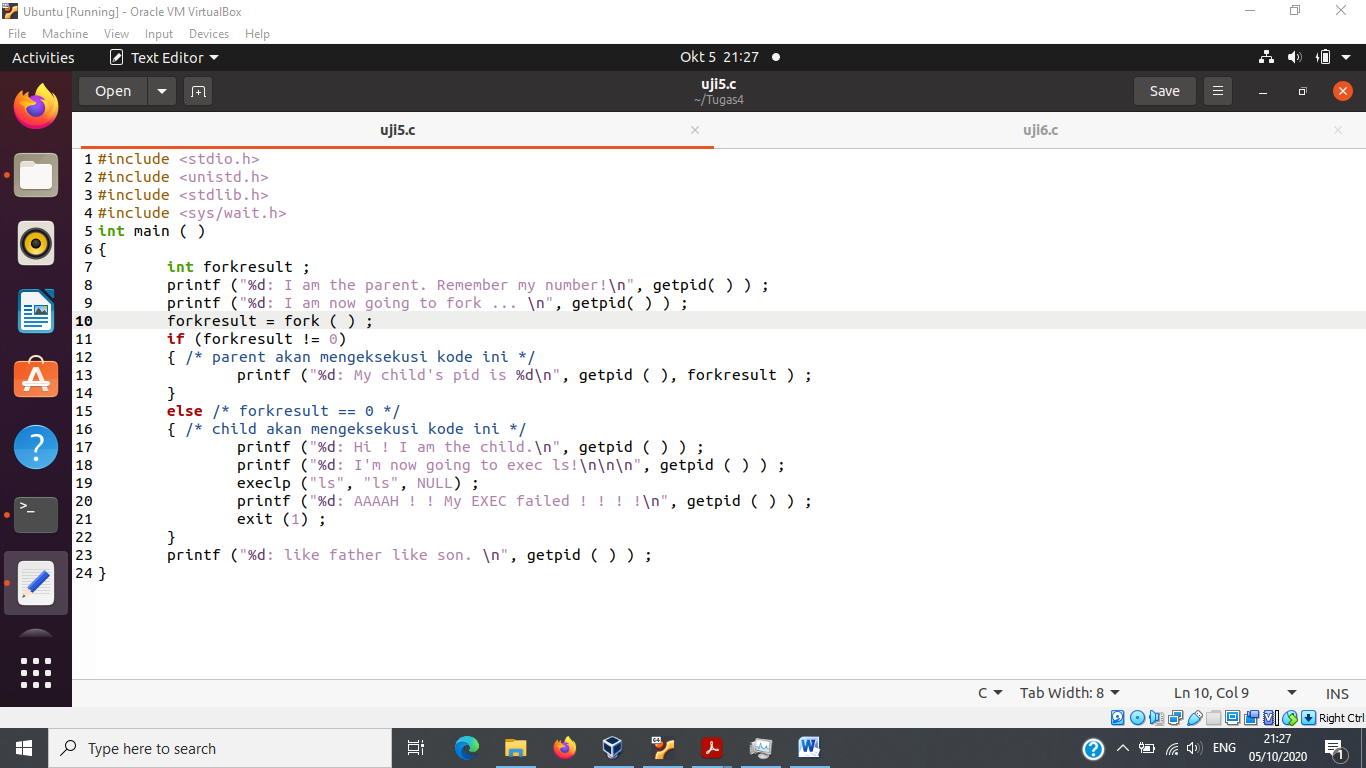
1. execvp

Mirip execv tapi tidak perlu full path, hanya path environment variable. Seperti perbandingan execl dan execlp tapi parameternya array.

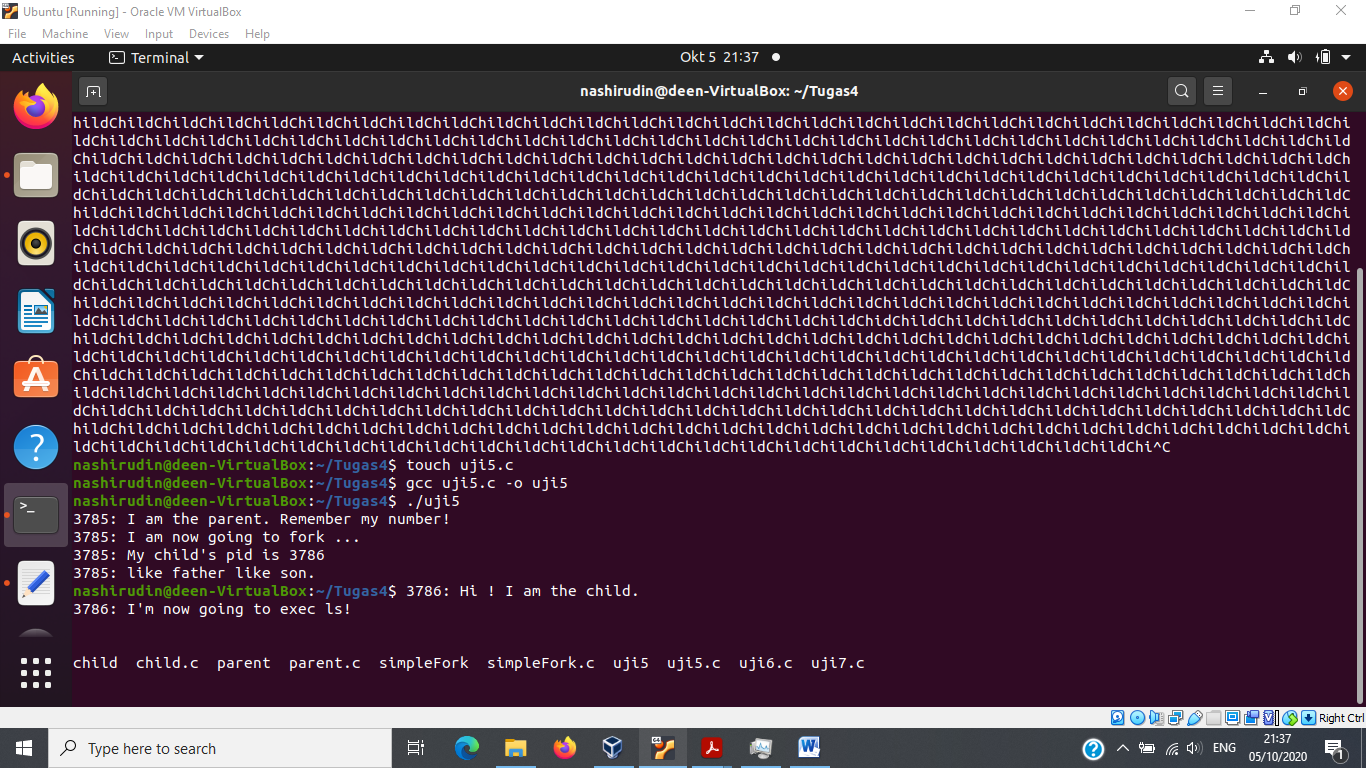
1. NULL

NULL dalam keluarga exec digunakan untuk menghentikan proses yang dijalankan di command exec.

1. Uji5.c PID

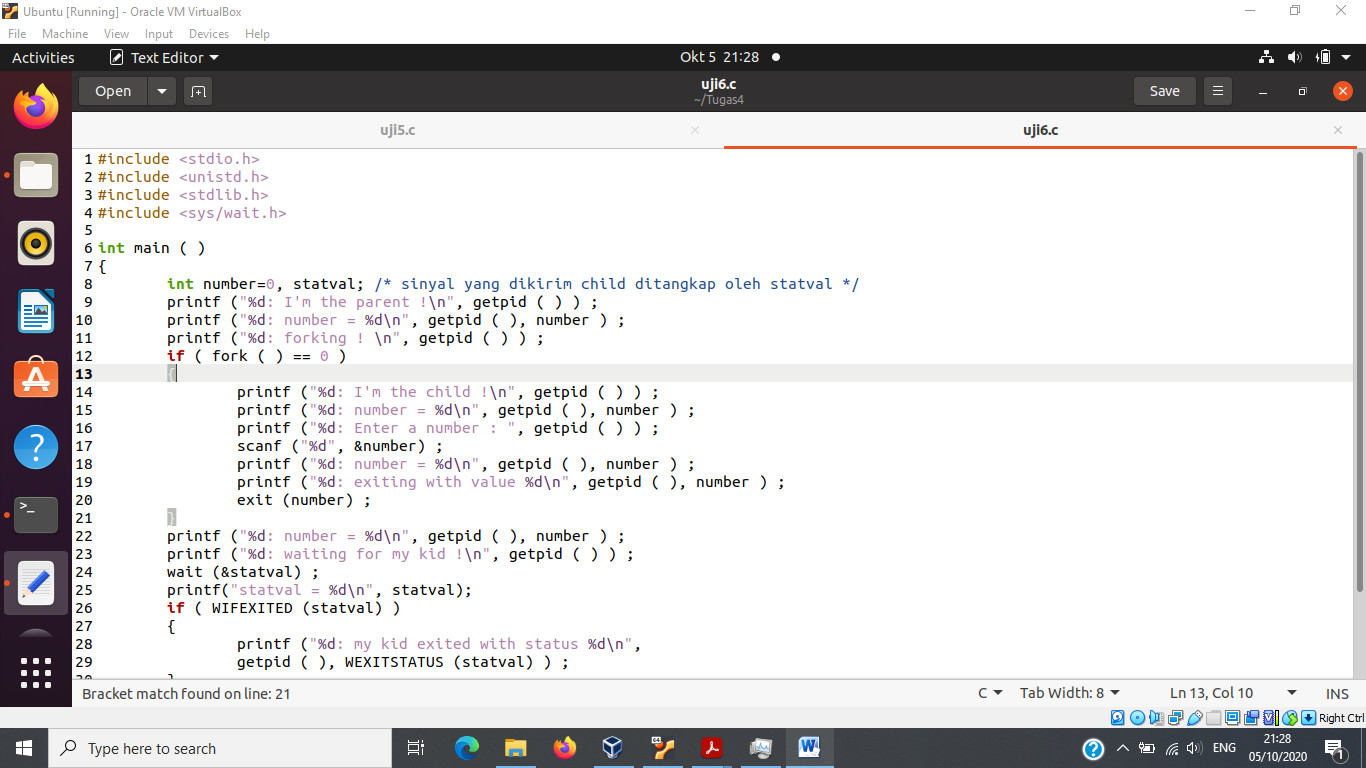


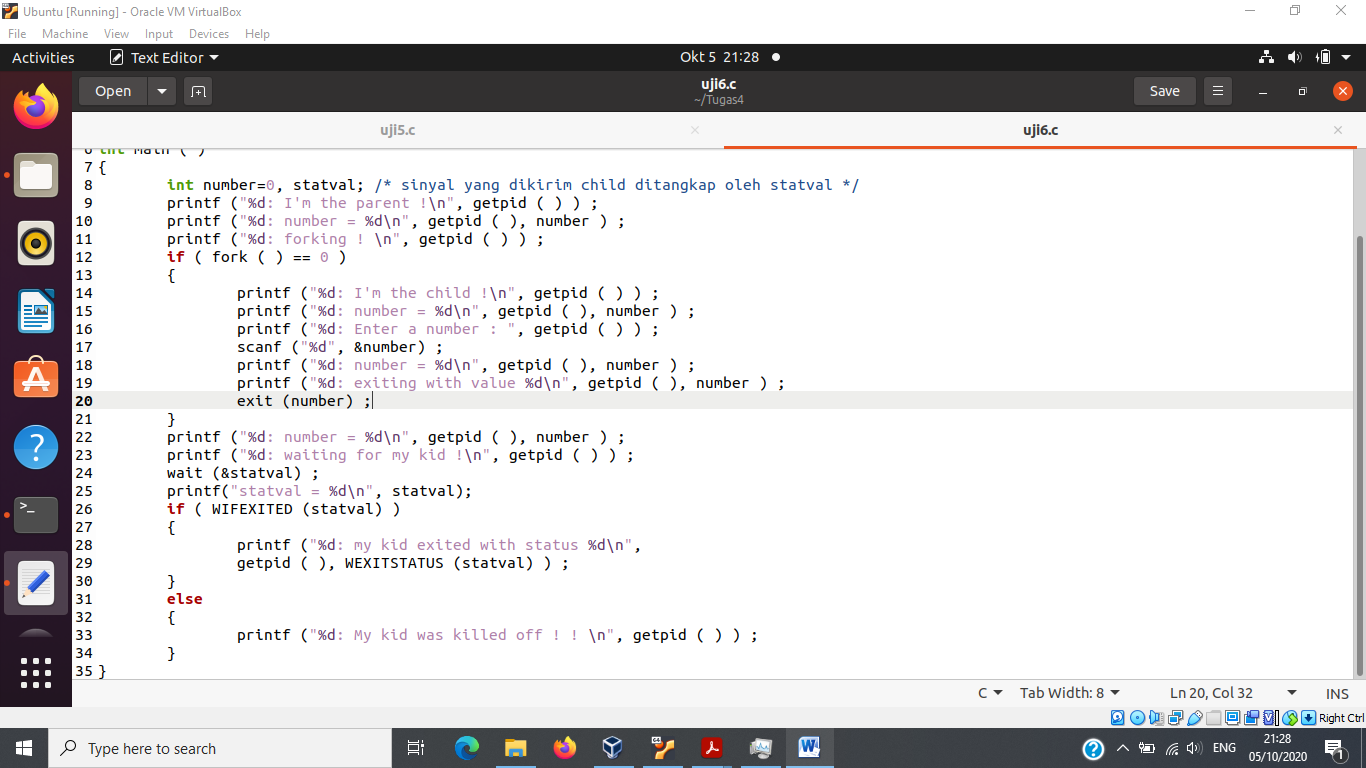
Output:



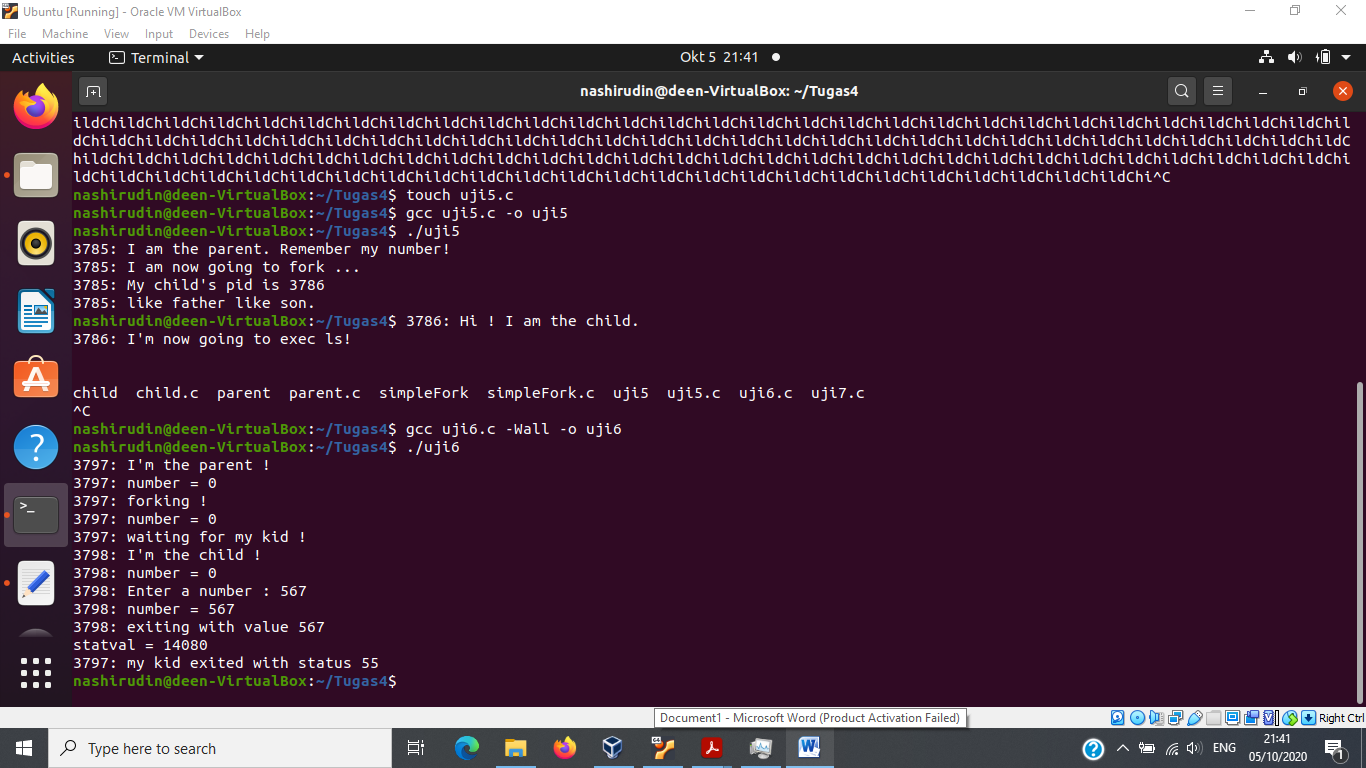
Menggunakan exec untuk mengganti proses yang sedang berjalan yaitu forking. Command exec mengandung perintah ls maka ls dijalankan.

1. Uji6.c exit() tertangkap wait()



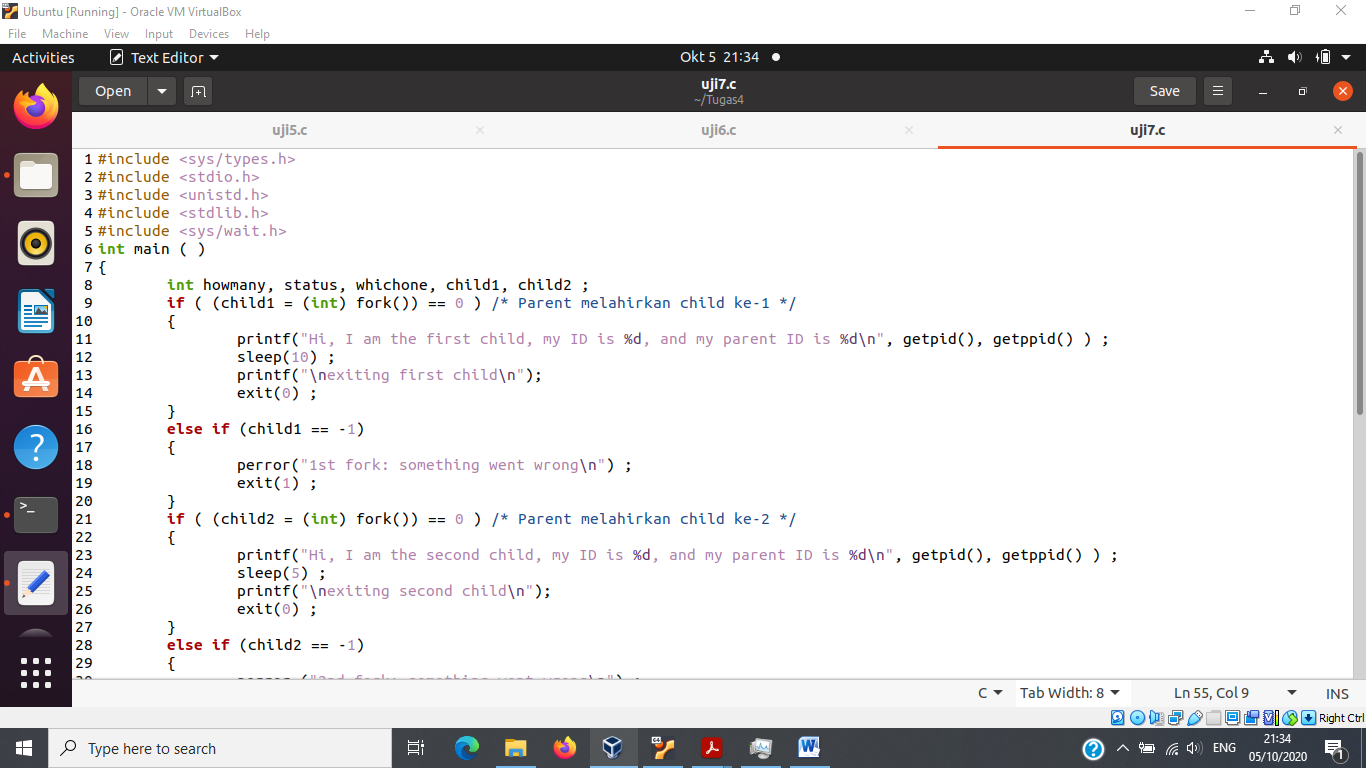


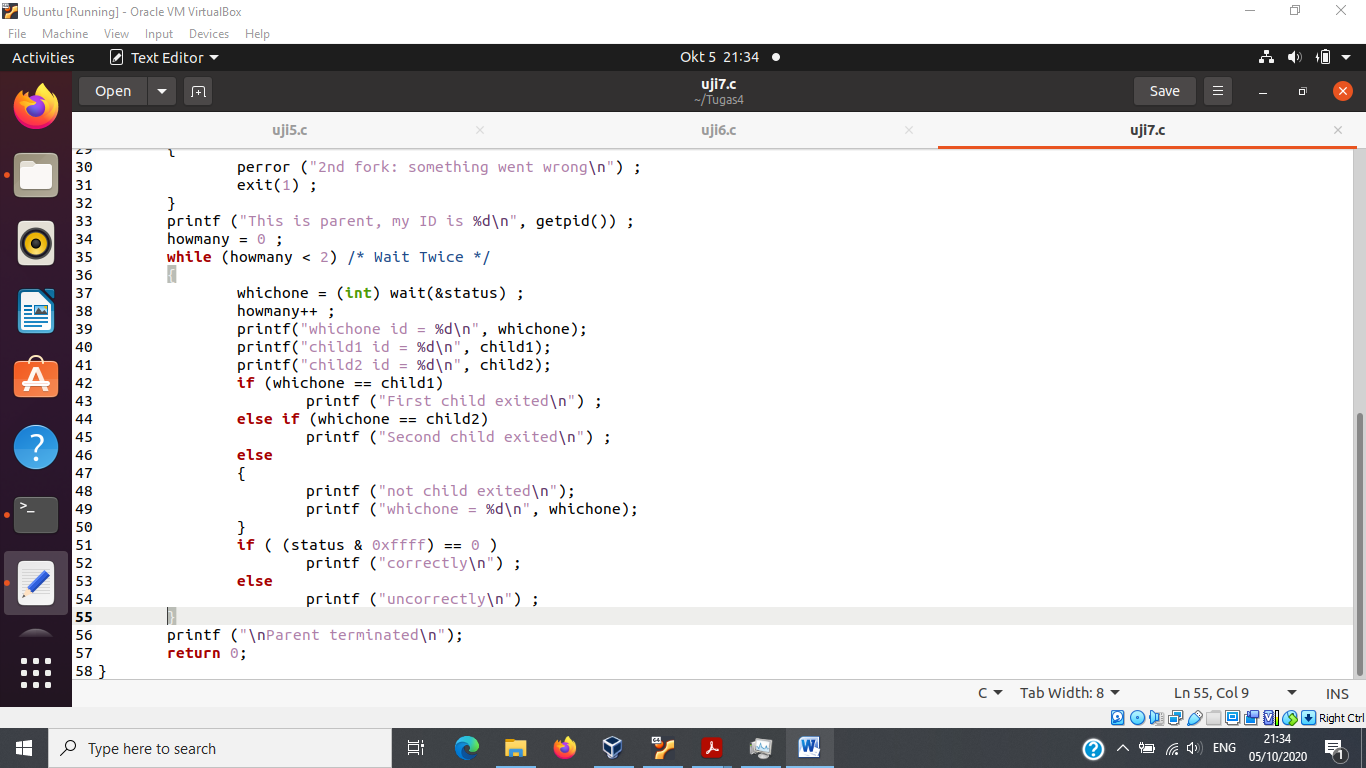
Output:



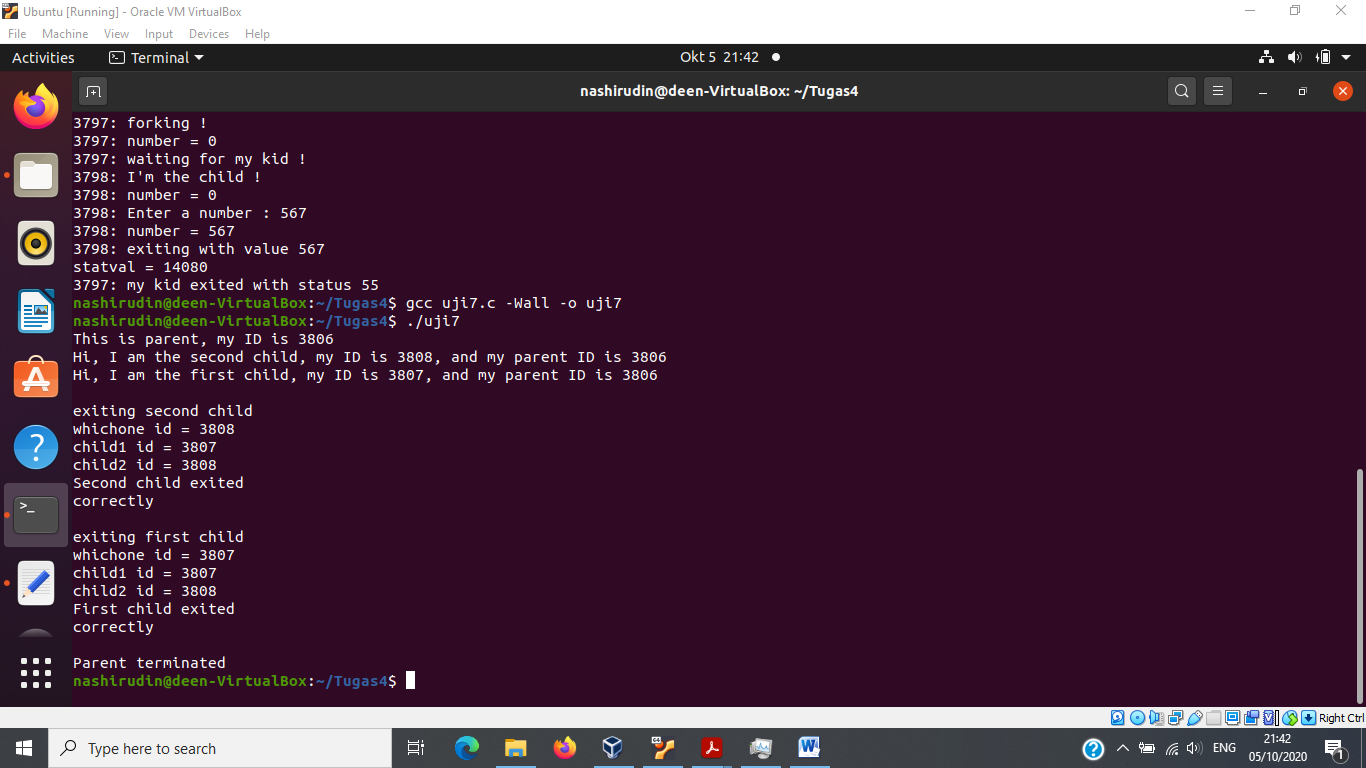
Exit() ditangkap oleh wait(). Kita diminta sebuah angka, angka tersebut menjadi parameter exit() kemudian diberikan kepada wait() sebagai pointer. Statval merupakan angka yang diinput dikali 256. WEXITSTATUS mereturn low order 8bits exit status value dari child. “my kid exited with status 55” merupakan bukti nilai exit() diberikan kepada wait().

1. Uji7.c





Output:



Parent membuat first child dan second child. Second child teroutput pertama karena first child memiliki sleep lebih lama dibanding first child. Child menunggu parent muncul sebelum child muncul karena kedua child sleep. Kemudian parent menunggu child exiting dari child terakhir ke child pertama. Parent mengenali child melalui nilai return yang dihasilkan child. Nilai ini masuk ke exit() kemudian diberikan kepada wait() maka wait memberikan nilai return PID dari child yang akan diterminasi.