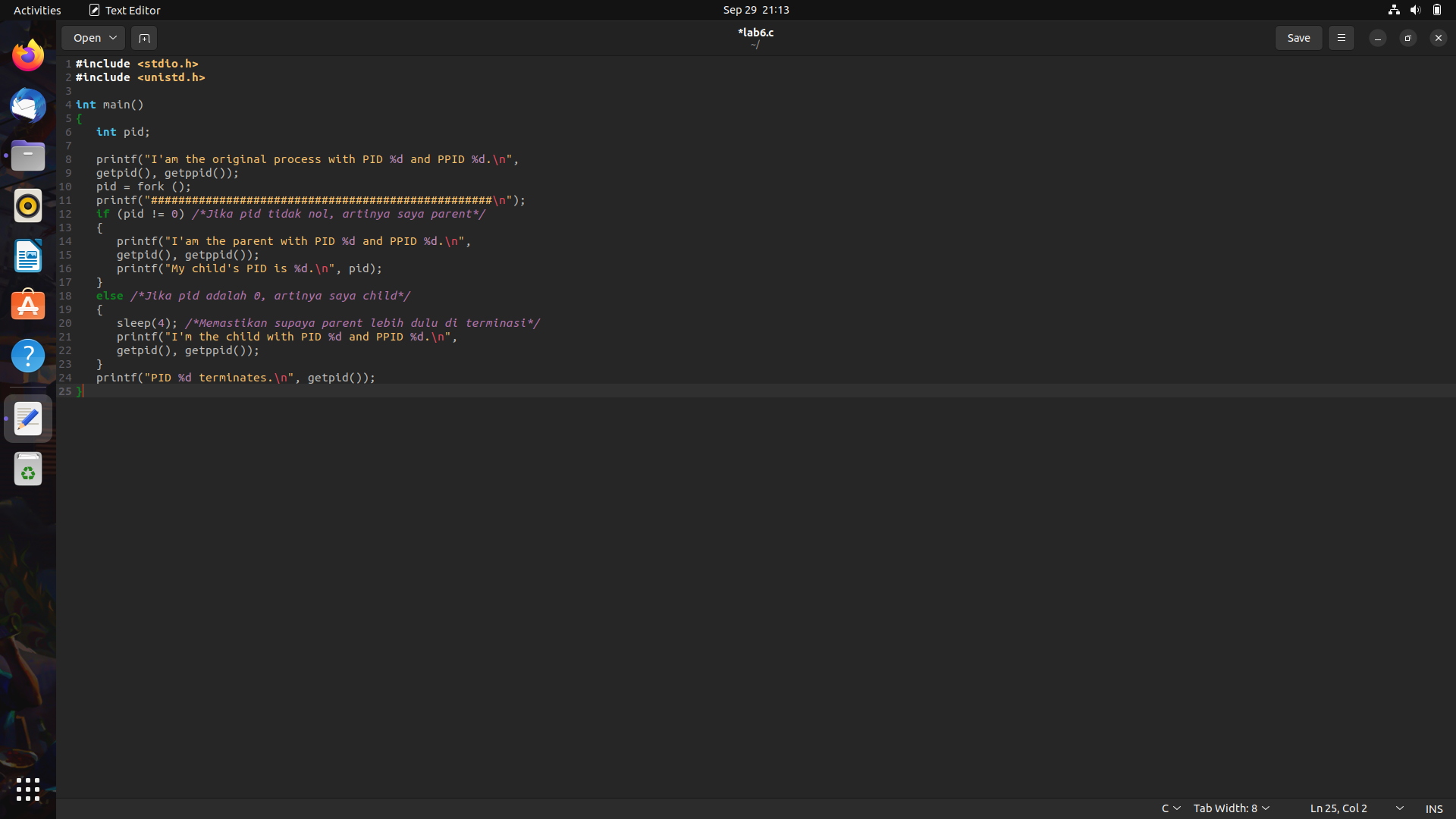
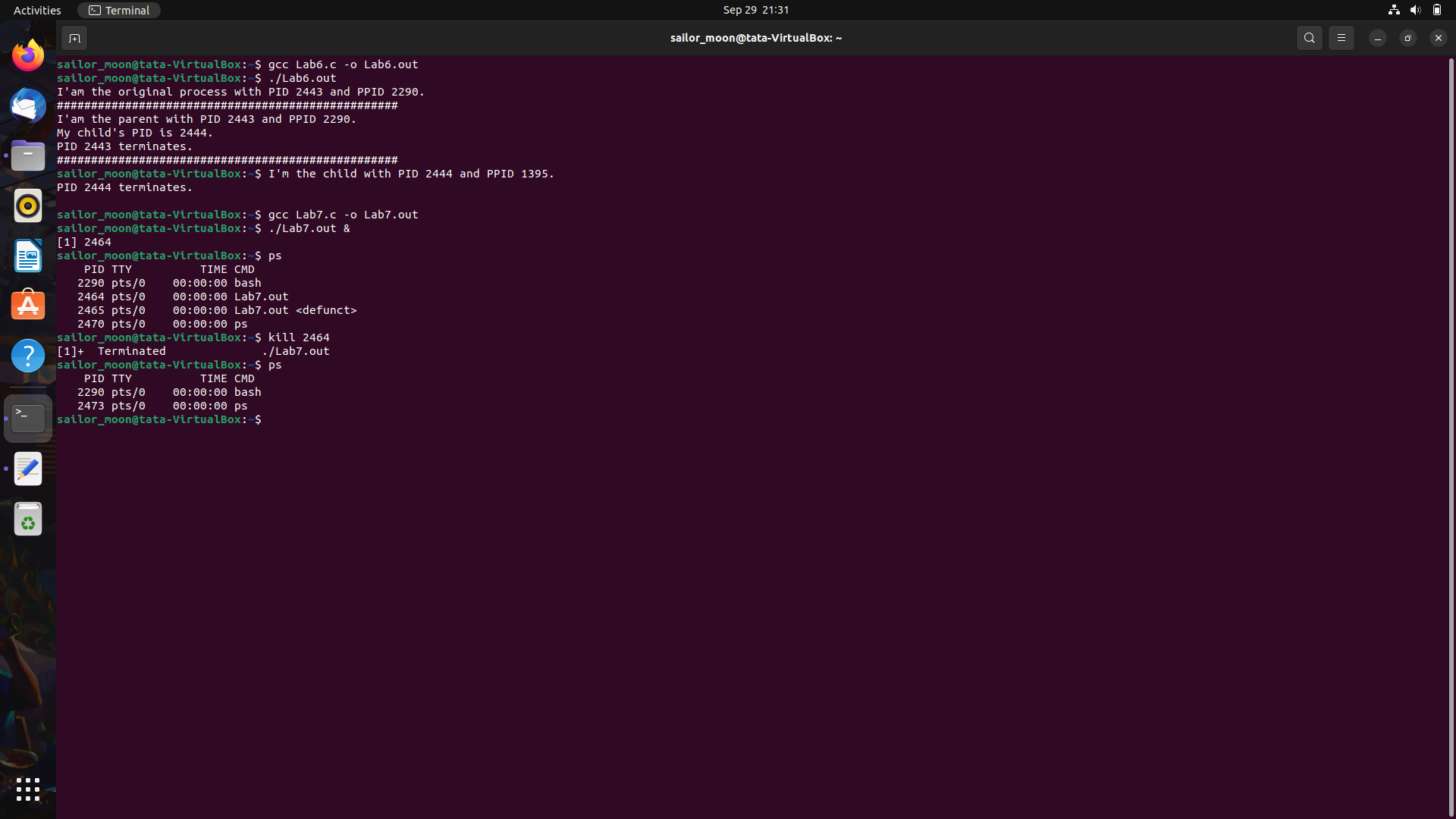
**Proses Orphan**

***Source Code* Lab6.c:**

**Hasil :**

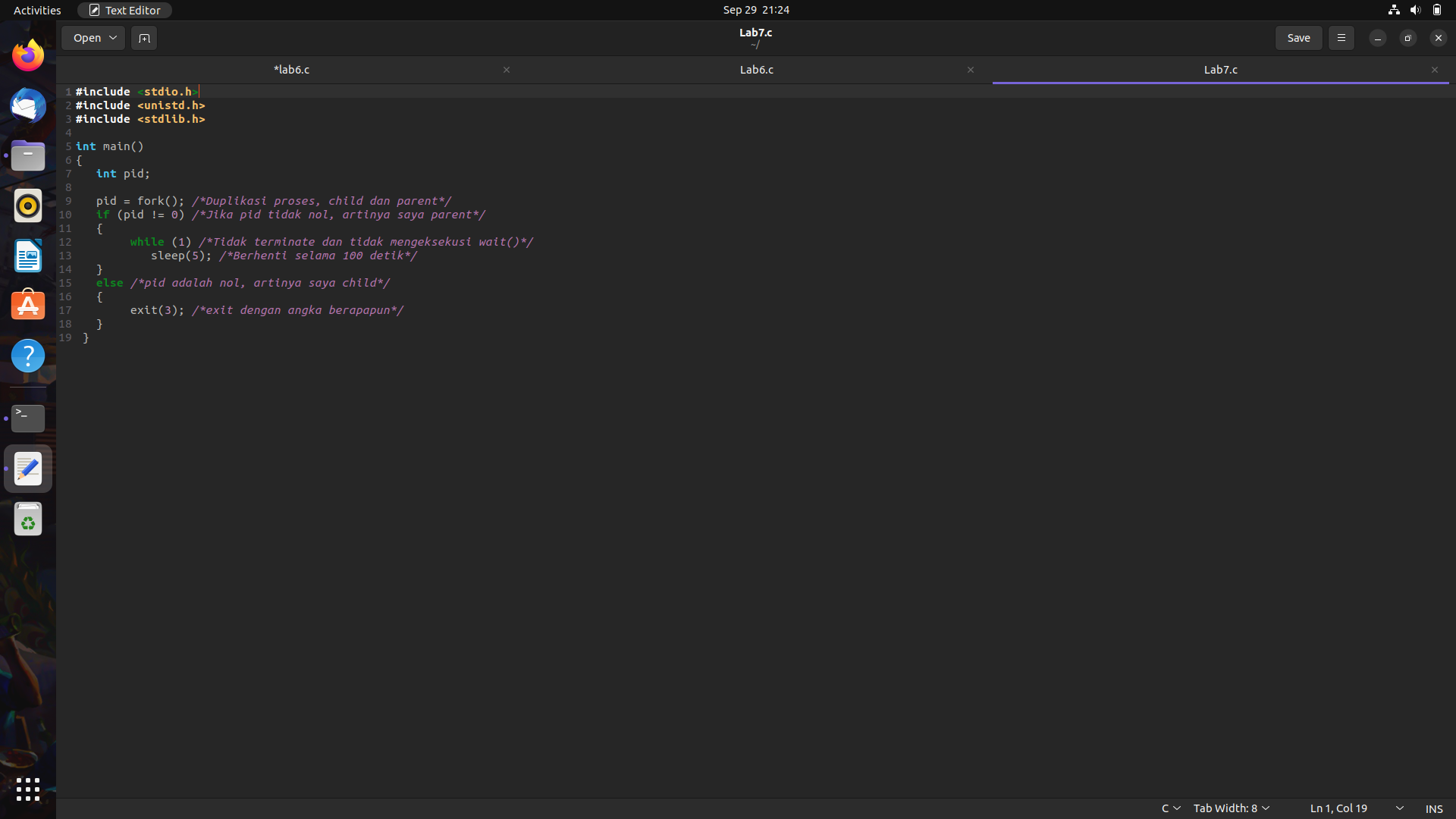
****

**Penjelasan :**

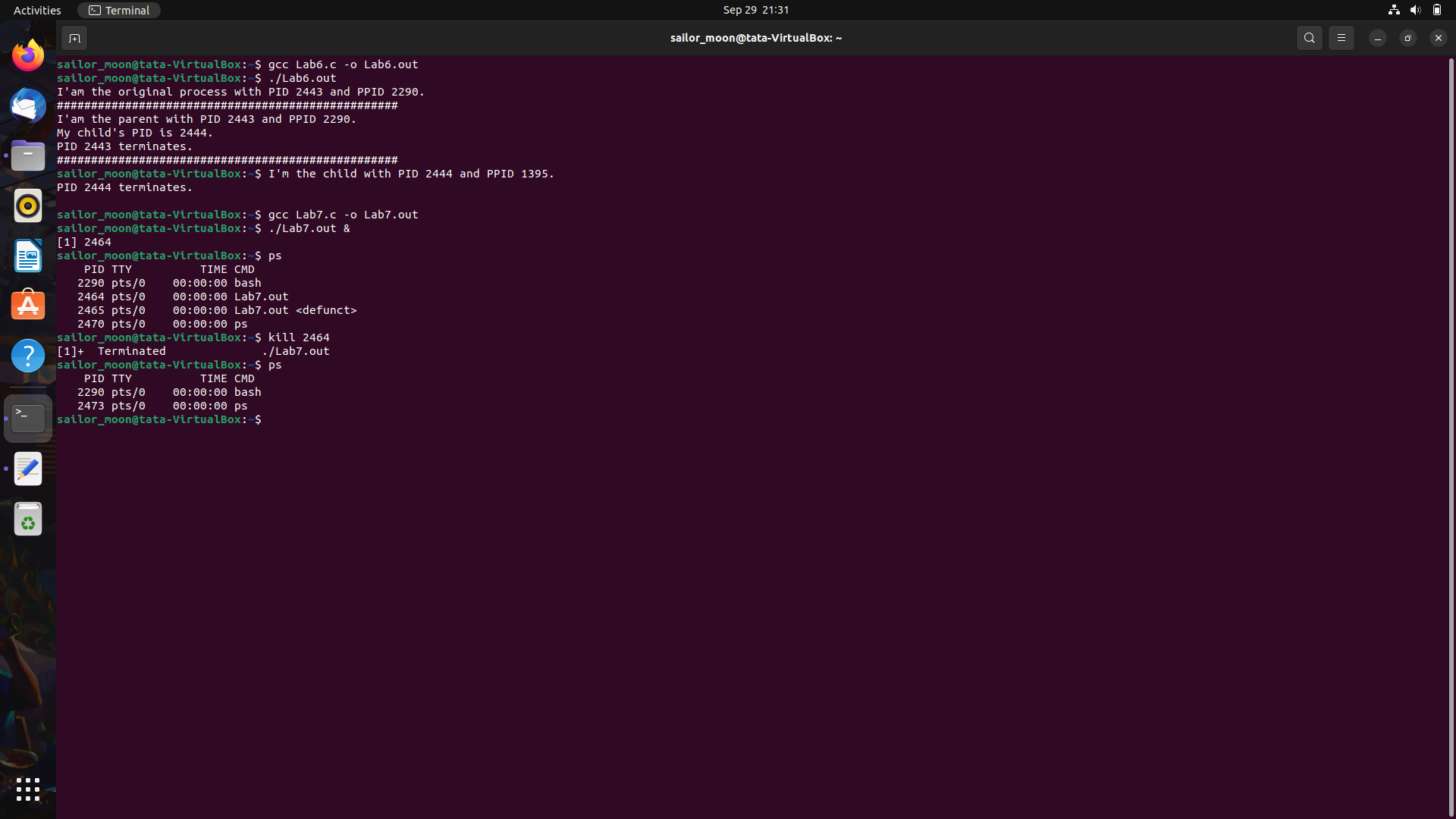
Dalam *source code* diatas terdapat *original process* dengan PID 2443 dan PPID 2290. Kemudian fungsi fork diinisialisasi dalam varibael pid, apabila pid bernilai **BUKAN** 0(nol) maka proses parentakan tetap berjalan dengan nilai PID dan PPID yang sama seperti *original process*. Saat pid dipanggil maka proses childakan dibuat. Setelah proses childdibuat maka pid akan bernilai 0 (nol) sehingga parentdi-terminatemenggunakan fungsi sleep (menangguhkan proses). Karena parent terminatesebelum childmaka childakan menjadi *orphan* dan secara otomatis diadopsi oleh proses *‘init’* asli yang PID-nya adalah 1.

**Proses Zombie**

***Source Code* Lab7.c :**

****

**Hasil :**

****

**Penjelasan :**

Dalam *source code* diatas, fungsi fork diinisialisasi dalam varibael pid untuk menduplikasi proses, child dan parent. Apabila PID bernilai **BUKAN** 0(nol) maka parent dikenai fungsi sleep. Parent tidak terminate (hidup) dan tidak mengeksekusi wait() maka kode pengembalian proses tidak akan pernah diterima dan proses anak menjadi zombie. Sedangkan apabila PID bernilai 0(nol) maka akan di-exit. Hasil run pada terminal menunjukkan bahwa setelah *source code* dieksekusi muncul proses parent dan proses child yang merupakan zombie. Apabila parent di terminate atau di kill maka proses parent dan proses child yang menjadi zombie akan hilang.