**Laporan Latihan Praktikum ke-X  
Sistem Operasi**

**Disusun oleh:**

**Ahmad Fadillah | 121140173**

**Kelas RD | Cluster RD**



**Program Studi Teknik Informatika**

**Jurusan Teknologi Produksi dan Industri**

**Institut Teknologi Sumatera**

**Lampung Selatan**

**2023**

**BAB I  
TEORI DASAR**

**1.1 System Call?**

System call adalah tata cara pemanggilan di program aplikasi untuk memperoleh layanan yang disediakan oleh kernel. System call berupa rutin sistem operasi untuk keperluan tertentu yang spesifik. Bentuk system call beragam, terbanyak berupa rutin prosedur atau fungsi.

Cara system call bekerja adalah sebagai berikut. Pertama-tama, user program akan mensetup argument untuk system call. Salah satu argumen adalah nomor system call. Perlu di catat bahwa semua ini dilakukan secara automatis oleh fungsi library kecuali jika kita menulis menggunakan bahasa assembler. Sesudah semua argumen di setup, program akan menjalankan instruksi "system call". Instruksi ini akan menyebabkan exception: event yang akan menyebabkan processor untuk jump ke satu address dan mulai menjalankan program / code di address tersebut.

Instruksi di alamat yang baru akan menyimpan state user program, menentukan sistem call apa yang kita inginkan, kemudian call fuction tersebut di kernel yang mengimplementasikan system call, setelah selesai maka mengembalikan program state, dan kembali ke user program. Sebuah system call adalah salah satu cara agar function yang di definisikan dalam device driver untuk bisa di panggil. Berikut adalah daftar beberapa panggilan sistem Linux yang umum dan paling banyak digunakan.

• Membuka

• Menutup

• eksekutif

• Menulis

• Membaca

• Lseek

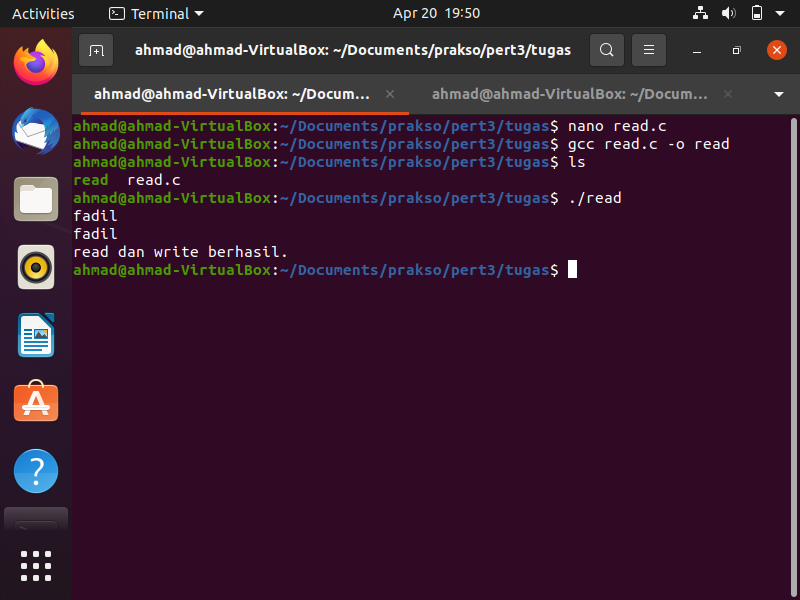
• Pilih

**BAB II  
PEMBAHASAN DAN ANALISIS**

**2.1 Latihan Pertama**

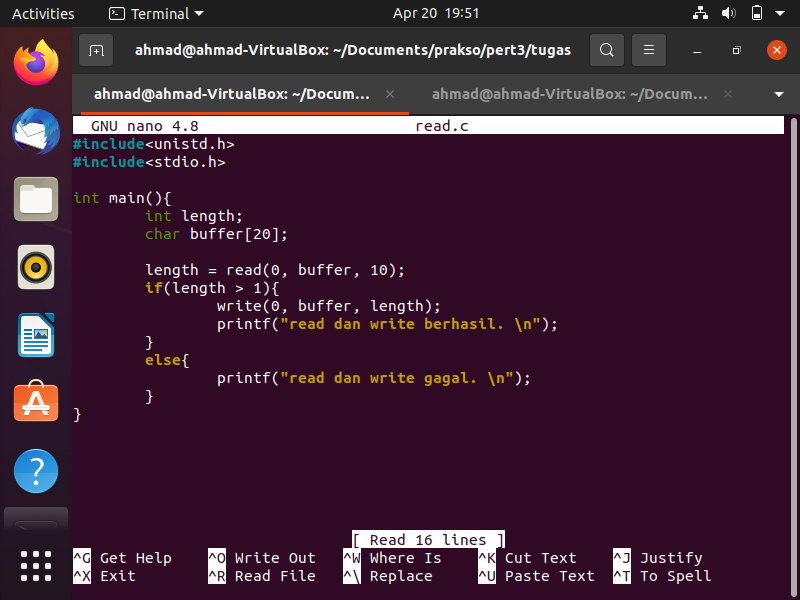
**2.1.1 Langkah Pertama**

Buat file read.c



**2.1.2 Langkah Kedua**

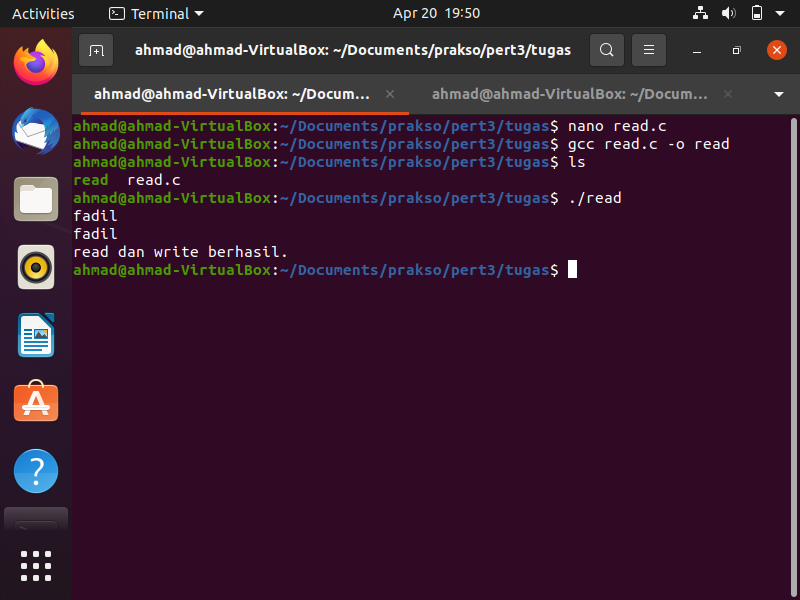
Isi dalam file read.c seperti kodingan dibawah ini lalu CTRL+X



**2.1.3 Langkah Ketiga**

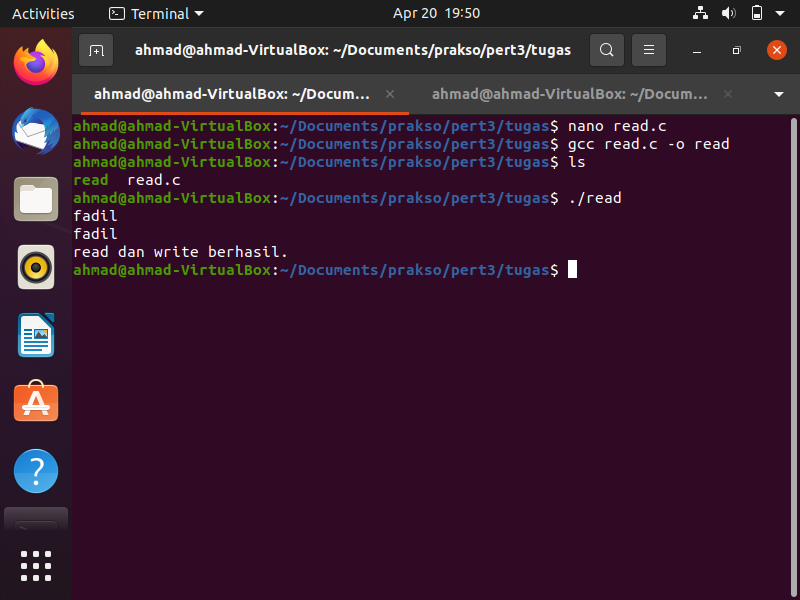
kompile menggunakan gcc dengan command ‘gcc <nama file c> -o <nama file>

lalu cek direktorinya sudah sukses tercompile atau belum



**2.1.4 Langkah Keempat**

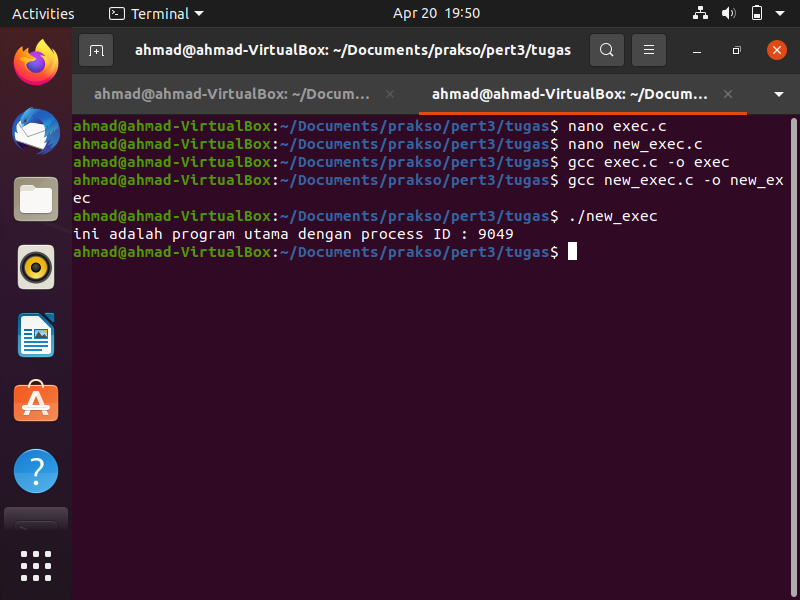
Lalu pilih file yang sudah dicompile dan akan menujukan hasil



**2.2 Latihan Kedua**

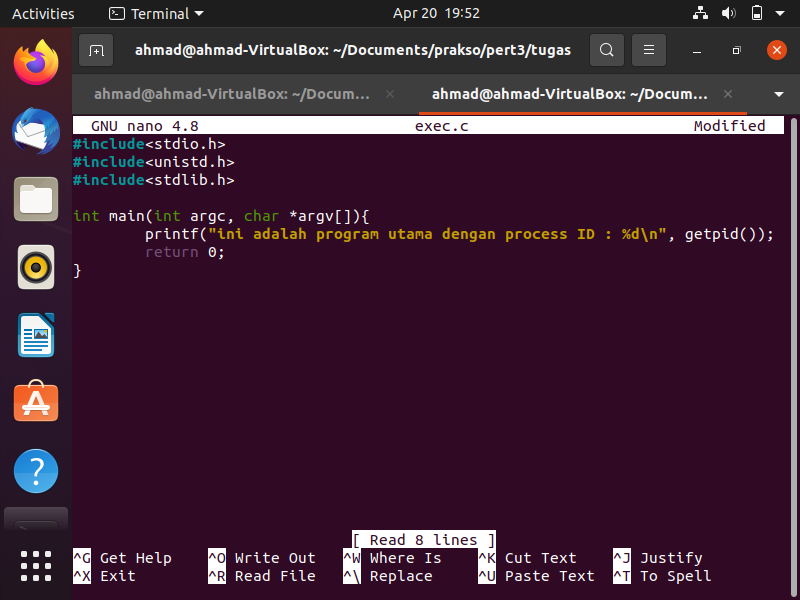
**2.2.1 Langkah Pertama**

Buat dua file .c yang satu bernama ‘exec.c’ dan ‘new\_exec.c’ kedua file ini akan saling berhubungan.



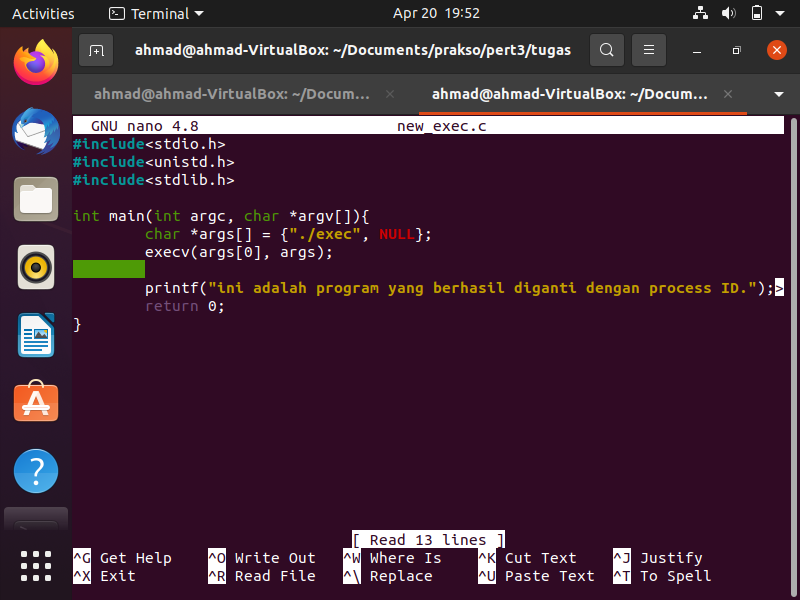
**2.2.2 Langkah Kedua**

tulis isi dari exec.c seperti kode dibawah ini lalu tekan CTRL + X



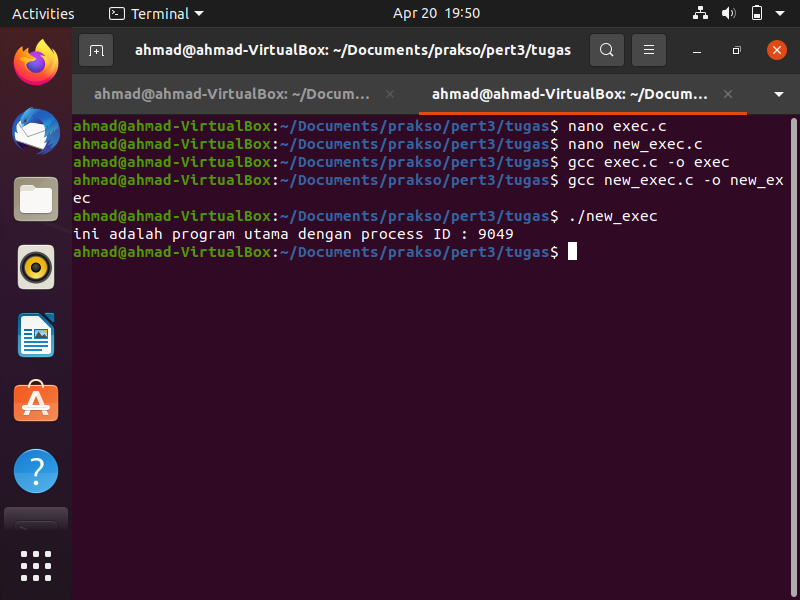
**2.2.3 Langkah Ketiga**

isi juga file new\_exec.c lalu tekan CTRL + X



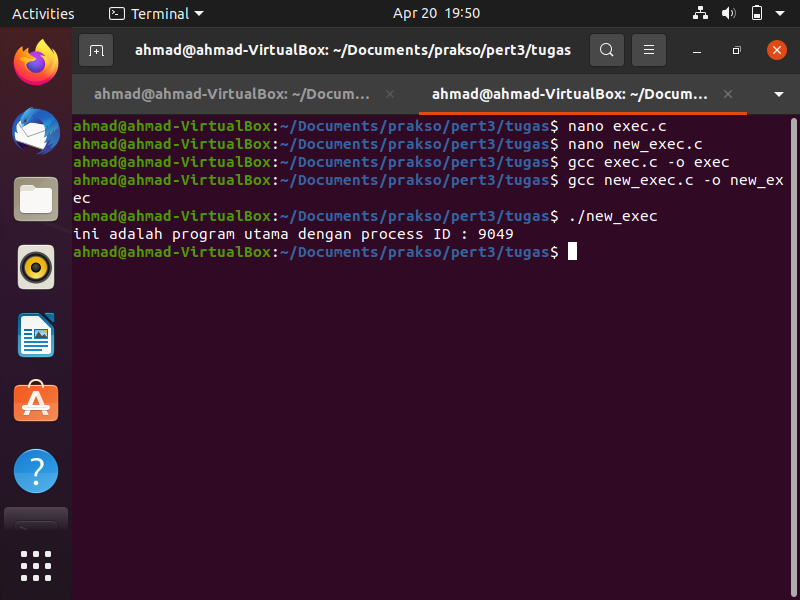
**2.2.4 Langkah Keempat**

kompile kedua file tersebut dengan command yang sama seperti latihan pertama,(namun beda di penamaan filenya saja)



**2.2.4 Langkah Kelima**

Dalam pemanggilan new\_exec, output yang dihasilkan menunjukkan bahwa fungsi tersebut merupakan program utama. Hal ini terjadi karena saat menjalankan perintah gcc, program new\_exec.c berhasil digantikan oleh program exec.c yang telah dijalankan sebelumnya.

****

**BAB III  
KESIMPULAN**

Kesimpulan dari latihan ini adalah pentingnya memahami cara kerja system call pada sistem operasi, terutama pada sistem operasi Linux yang banyak digunakan saat ini. Dengan memahami system call, pengguna dapat lebih memahami proses yang terjadi di balik layar saat menggunakan aplikasi atau mengakses file pada komputer. Selain itu, pengguna juga dapat mengembangkan dan memperbaiki sistem operasi Linux dengan lebih mudah karena sifatnya yang open source.