**LAPORAN PRAKTIKUM**

**SISTEM OPERASI RD**

**MODUL 3**

**Oleh :**

**Rayhan Fadel Irwanto (122140236)**



**Program Studi Teknik Informatika**

**Institut Teknologi Sumatera**

**2024**

# **Daftar Isi**

[**Daftar Isi** 2](#_Toc164805781)

[**1.** **Dasar Teori** 3](#_Toc164805782)

[**2.** **Hasil & Jawaban** 3](#_Toc164805783)

[**3.** **Kesimpulan dan Saran** 10](#_Toc164805784)

# 

# **Dasar Teori**

System call adalah suatu metode yang digunakan oleh program komputer untuk meminta layanan dari kernel sistem operasi tempat program tersebut berjalan. Hal ini dilakukan melalui antarmuka yang telah terdefinisi, biasanya berupa API, yang menyediakan interaksi antara program pengguna dan sistem operasi. Fungsi utama dari system call mencakup berbagai aspek, mulai dari memberikan interface bagi program untuk membuat permintaan layanan tertentu, hingga melindungi sistem dari akses yang tidak sah dengan hak istimewa, serta mengalihkan program ke mode kernel untuk mengakses sumber daya sistem.

Jenis-jenis system call meliputi kontrol proses, manajemen file, pengelolaan perangkat keras, pemeliharaan informasi sistem, komunikasi antar proses dan jaringan, serta manajemen memori. Dalam penggunaannya, system call juga berperan dalam penanganan kesalahan, menyinkronkan akses ke sumber daya bersama, dan menghadapi biaya tambahan akibat pengalihan konteks yang diperlukan saat program berpindah dari mode pengguna ke mode kernel untuk melakukan operasi tertentu.

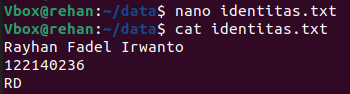
# **Hasil & Jawaban**

**Percobaan 1 Open**

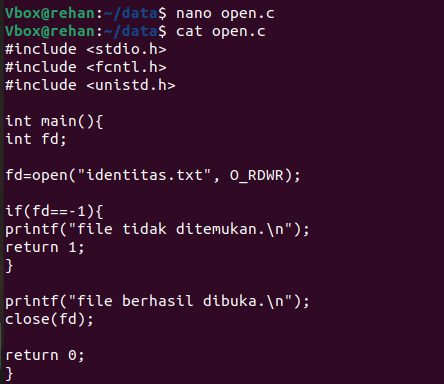
1. Membuat folder “Data” dan mengakses isi folder tersebut



1. Membuat file dengan nama “identitas.txt” dengan isi Nama, Nim, Kelas



1. Membuaf file dengan ektensi C dengan nama “open.c”



1. Lakukan instalasi gcc dengan perintah “sudo apt install gcc” dan jalankan perintah “gcc open.c -o openfile”

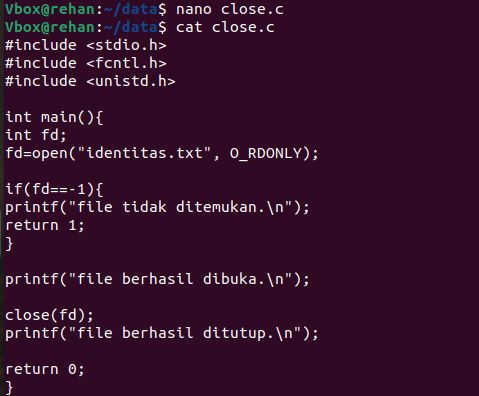
Sudah Install Sebelumnya

1. Lakukan pemanggilan sistem yang telah kita buat dengan perintah “./openfile”. Jika file berhasil dibuka maka akan menampilkan luaran berikut

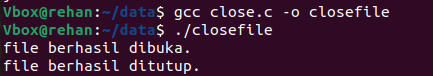


**Perobaan 2 Close**

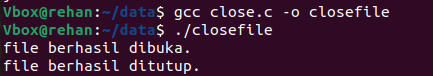
1. Mengakses folder “Data” dan membuaf file dengan ektensi C dengan nama “close.c”



1. Jalankan perintah “gcc close.c -o closefile”

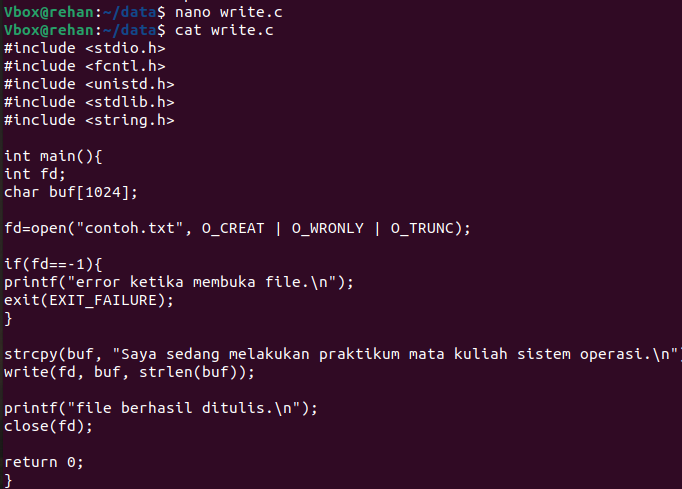


1. Lakukan pemanggilan sistem yang telah kita buat dengan perintah “./closefile”

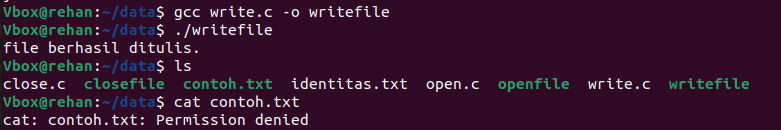


**Percobaan 3 Write**

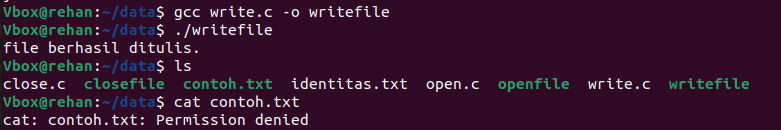
1. Mengakses folder “Data” dan membuaf file dengan ektensi C dengan nama “write.c”



1. Jalankan perintah “gcc write.c -o writefile”.



1. Lakukan pemanggilan sistem yang telah kita buat dengan perintah “./writefile”. Jika file berhasil ditulis maka akan menghasilkan file “contoh.txt”



**Percobaan 4 Delete**

1. Mengakses folder “Data” dan membuaf file dengan ektensi C dengan nama “delete.c”



1. Jalankan perintah “gcc delete.c -o deletefile”

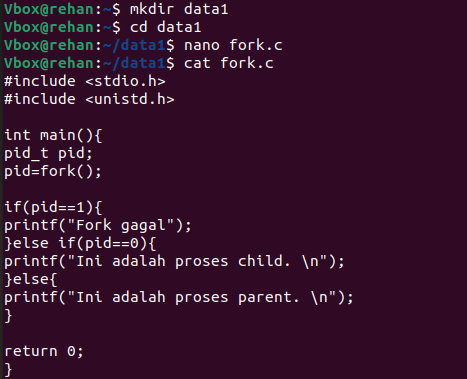


1. Lakukan pemanggilan sistem yang telah kita buat dengan perintah “./deletefile”. Jika file berhasil dihapus maka akan menghasilkan luaran

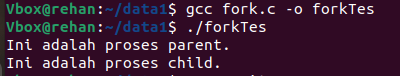


**Percobaan 5 Fork**

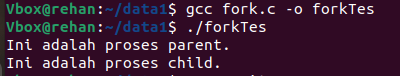
1. Mengakses folder “Data1” dan membuaf file dengan ektensi C dengan nama “fork.c”

****

1. Jalankan perintah “gcc fork.c -o ForkTes”

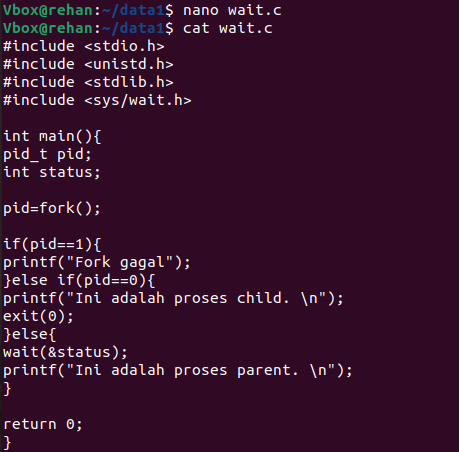


1. Lakukan pemanggilan sistem yang telah kita buat dengan perintah “./Fork.Test”. Jika file berhasil dihapus maka akan menghasilkan luaran

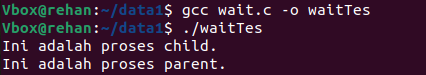


**Percobaan 6 Wait**

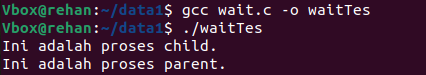
1. Mengakses folder “Data1” dan membuaf file dengan ektensi C dengan nama “wait.c”



1. Jalankan perintah “gcc wait.c -o WaitTes”



1. Lakukan pemanggilan sistem yang telah kita buat dengan perintah “./Wait.Test”. Jika file berhasil dihapus maka akan menghasilkan luaran



# **Kesimpulan dan Saran**

Pemahaman tentang System Call mengungkapkan bahwa System Call berperan sebagai jembatan penting yang menghubungkan program pengguna dengan kernel sistem operasi, memungkinkan program untuk meminta layanan seperti akses file, manajemen proses, pengelolaan perangkat keras, dan komunikasi antar proses. Dengan antarmuka yang terdefinisi dengan baik, interaksi antara program dan sistem operasi dapat dilakukan secara efisien dan aman. Namun, dalam penggunaannya, perlu diperhatikan aspek perlindungan data, hak istimewa, pengelolaan Context Switching untuk mengoptimalkan kinerja sistem, dan penanganan kesalahan agar program dapat merespons kondisi yang tidak diinginkan dengan tepat. Dengan pemahaman yang matang tentang System Call, program dapat berjalan lebih efisien dan dapat diandalkan dalam mengakses sumber daya sistem.