Rodaina Mohamed Mustafa
23101892

OS Assignment 2

Lab 5 code explanation

Example 1:

This program demonstrates how process creation works in Linux using the **fork()** system call. When **fork()** is executed, it creates a new process by duplicating the current one. This results in two processes running the same code: a parent and a child.
The return value of **fork()** helps us distinguish between the two. In the child process, **fork()** returns 0, so it prints: “This is the child process,” along with its process ID using **getpid().** In the parent process, **fork()** returns a positive value (the child’s PID), so it prints: “This is the parent process.” If the return value is negative, it means the fork failed.
So, the program shows how the parent and child processes run simultaneously after **fork()** and how we can identify each one based on the return value.

Example 2:

One background process (job #1, PID 55) that will silently wait for 5 minutes before finishing on its own but we can control it with commands like kill %1 (to stop it)

Example 3:

Here we started a background process, verified it was running with both jobs and ps, killed it cleanly using kill %1, and confirmed it’s dead.

Example 4:

kill -STOP %1: instantly pauses the background job
kill -CONT %1: resumes it exactly where it stopped

EXAMPLE 5:

This program splits the code into two C files. **file1.c** defines the hello() function, and **file2.c** calls it from main(). When the two files are compiled and linked together, the program runs and prints “Hello from file1!”.

EXAMPLE 6:

This C program prints a message. You compile it with gcc simple_program.c -o simple_program to create an executable. Using ldd simple_program lists the libraries the program uses.

Rodaina Mohamed Mustafa
23101892

What is the job of the Linkers?

The **linker** combines multiple object files and libraries into a single executable, resolving references between functions or variables defined in different files so the program can work as a whole.

What is the job of the Loaders?

The **loader** is part of the operating system and is responsible for loading the executable into memory, setting up the program's memory space, and starting its execution. Together, the linker and loader ensure that a C program is correctly built and runs properly.