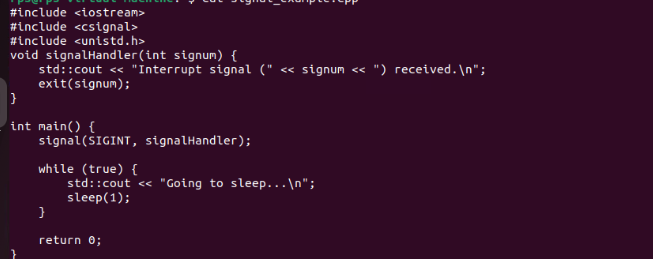
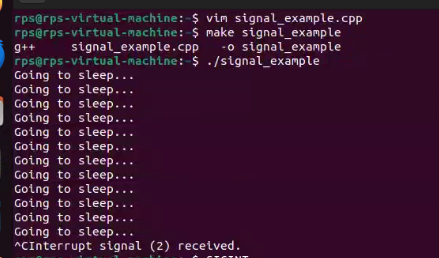
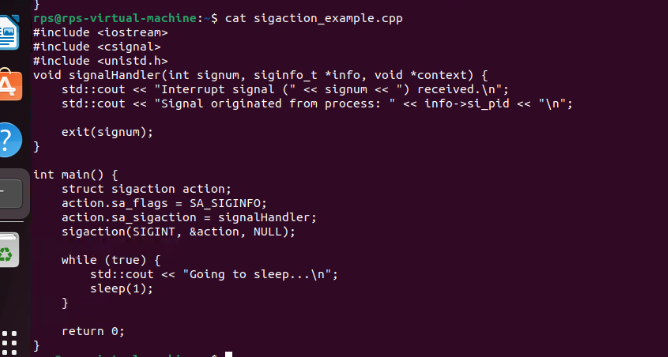
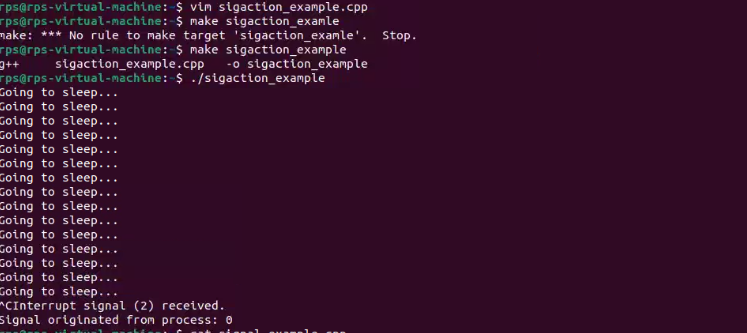
Basic Handling vs. Advanced Control: Implement signal handling using both signal and sigaction (in separate program runs). Observe the behavior. Which API allows for more control over the signal handler? Explain the key difference in a comment within your code.









The sigaction API allows for more control over the signal handler compared to the signal API.

Key diffrence:

// Using `signal` provides basic handling without additional info:

// signal(SIGINT, signalHandler);

// Using `sigaction` provides advanced control and additional info:

// struct sigaction action;

// action.sa\_flags = SA\_SIGINFO;

// action.sa\_sigaction = signalHandler;

// sigaction(SIGINT, &action, NULL);

Summary:

**Signal API**: Provides basic signal handling with a simple handler function.

**sigaction API**: Provides advanced control, allowing handlers to receive additional information and offering more customization options.

Graceful Termination with Signal Handling

Objective: Modify your program to demonstrate graceful termination upon receiving a specific signal (e.g., SIGINT). Within the signal handler, perform any necessary cleanup tasks (e.g., closing files, releasing resources) before exiting the program gracefully.

Implementation:

In your signal handler function, include code to perform cleanup actions. This might involve closing open files, releasing memory, or writing data to disk.

Use exit(0) or similar methods to terminate the program after cleanup is complete.

