#include <stdio.h>

#include <unistd.h>

#include <string.h>

int main() {

int fd[2]; // File descriptors for the pipe

pid\_t pid;

char write\_msg[] = "Hello from parent to child!";

char read\_msg[100];

// Create a pipe

if (pipe(fd) == -1) {

perror("pipe failed");

return 1;

}

// Fork a child process

pid = fork();

if (pid == -1) {

perror("fork failed");

return 1;

}

if (pid == 0) {

// Child process

// Close the unused write end of the pipe

close(fd[1]);

// Read the message from the pipe

read(fd[0], read\_msg, sizeof(read\_msg));

// Print the message received from the parent

printf("Child received: %s\n", read\_msg);

// Close the read end of the pipe

close(fd[0]);

} else {

// Parent process

// Close the unused read end of the pipe

close(fd[0]);

// Write the message to the pipe

write(fd[1], write\_msg, strlen(write\_msg) + 1); // +1 to include null terminator

// Close the write end of the pipe

close(fd[1]);

// Wait for child to finish (optional, but good practice)

wait(NULL);

}

return 0;

}

