```c
#include <stdio.h>
#include <unistd.h>
#include <sys/types.h>
#include <string.h>


#define BUFFER_SIZE 32
#define READ_END     0
#define WRITE_END    1


int main(void)
{
    char write_msg[BUFFER_SIZE] = "You're my child process!";
    char read_msg[BUFFER_SIZE];

    pid_t pid;  // child process id
    int fd[2];  // file descriptors for the pipe

    // Create the pipe.
    if (pipe(fd) == -1) {
        fprintf(stderr,"pipe() failed");
        return 1;
    }

    // Fork a child process.
    pid = fork();
    if (pid > 0) {
        // PARENT PROCESS - write to the pipe.

        // Close the unused READ end of the pipe.
        close(fd[READ_END]);

        // Write to the WRITE end of the pipe.
        write(fd[WRITE_END], write_msg, strlen(write_msg)+1);
        printf("Parent: Wrote '%s' to the pipe.\n", write_msg);

        // Close the WRITE end of the pipe.
        close(fd[WRITE_END]);
    }
    else if (pid == 0) {
        // CHILD PROCESS.

        // Close the unused WRITE end of the pipe - Read from the pipe.
        close(fd[WRITE_END]);

        // Read from the READ end of the pipe.
        read(fd[READ_END], read_msg, BUFFER_SIZE);
        printf("Child: Read '%s' from the pipe.\n", read_msg);

        // Close the READ end of the pipe.
        close(fd[READ_END]);
    }
    else {
```

```
        fprintf(stderr, "fork() failed");
        return 1;
    }

    return 0;
}
```