#include <stdio.h>

#include <stdlib.h>

#include <string.h>

#include <unistd.h>

#include <sys/types.h>

#include <sys/ipc.h>

#include <sys/msg.h>

struct message {

long msg\_type;

char msg\_text[100];

};

int main() {

key\_t key;

int msgid;

key = ftok("message\_queue\_example", 'A');

msgid = msgget(key, 0666 | IPC\_CREAT);

if (msgid == -1) {

perror("msgget");

exit(EXIT\_FAILURE);

}

pid\_t pid = fork();

if (pid == -1) {

perror("fork");

exit(EXIT\_FAILURE);

}

if (pid > 0) { // Parent process

struct message msg\_send;

msg\_send.msg\_type = 1;

printf("Enter a message to send to the child process: ");

fgets(msg\_send.msg\_text, sizeof(msg\_send.msg\_text), stdin);

if (msgsnd(msgid, (void \*)&msg\_send, sizeof(msg\_send.msg\_text), 0) == -1) {

perror("msgsnd");

exit(EXIT\_FAILURE);

}

printf("Message sent to the child process.\n");

wait(NULL);

} else { // Child process

struct message msg\_receive;

if (msgrcv(msgid, (void \*)&msg\_receive, sizeof(msg\_receive.msg\_text), 1, 0) == -1) {

perror("msgrcv");

exit(EXIT\_FAILURE);

}

printf("Message received from the parent process: %s", msg\_receive.msg\_text);

}

if (msgctl(msgid, IPC\_RMID, NULL) == -1) {

perror("msgctl");

exit(EXIT\_FAILURE);

}

return 0;

}