

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ «САМАРСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИМЕНИ АКАДЕМИКА С.П. КОРОЛЕВА  
(САМАРСКИЙ УНИВЕРСИТЕТ)»

Институт Информатики и кибернетики   
Кафедра Программных систем

**ПОЯСНИТЕЛЬНАЯ ЗАПИСКА**  
  
к лабораторной работе №2 по дисциплине «Технологии промышленного программирования»

Обучающийся группы 6232-020302D \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Хорина В.И.

Руководитель \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Баландин А.В.

Самара 2023

СОДЕРЖАНИЕ

[1 Постановка задачи 3](#_Toc147860313)

[2 Результаты работы 4](#_Toc147860314)

[ПРИЛОЖЕНИЕ А Листинг модуля M1 5](#_Toc147860315)

[ПРИЛОЖЕНИЕ B Листинг модуля M2 7](#_Toc147860316)

[ПРИЛОЖЕНИЕ С Листинг модуля M3 9](#_Toc147860317)

1. Постановка задачи

Разработать приложение, состоящее из трех взаимодействующих процессов. Требуется написать три программных модуля – М1, М2, М3. На базе модуля М1 из shell запускается стартовый процесс Р1(М1).

Процесс Р1 создает канал и, используя функцию семейства spawn\*(), запускает процессы Р2(М2) и Р3(М3), передавая им в качестве параметра chid созданного канала, затем переходит в ожидание сообщений по своему каналу.

Процесс Р2 создает свой канал, устанавливает соединение с каналом процесса Р1, отправляет ему сообщение о chid своего канала и переходит в состояние приема сообщений по созданному каналу.

Процесс Р3 создает свой канал, устанавливает соединение с каналом процесса Р1, отправляет ему сообщение о chid своего канала и переходит в состояние приема сообщений по созданному каналу.

Процесс Р1, приняв сообщение от процесса Р2 или Р3 о chid канала, устанавливает соединение и посылает по нему - "Р1 send message to Р?" (? – номер соответствующего процесса), принимает ответ и выдает его на терминал. После такого взаимодействия с Р2 и Р3 процесс Р1 терминируется.

Процесс Р?, получив сообщение от Р1, выдает его на экран терминала, посылает ответ "Р? ОК" процессу Р1 и терминируется.

1. Результаты работы

Результаты работы представлены в виде вывода на консоль сообщений во время выполнения программы.

Изображение выглядит как текст, снимок экрана, Шрифт, документ

Автоматически созданное описание

Рисунок 1 – Результат выполнения программы в консоли

ПРИЛОЖЕНИЕ А  
Листинг модуля M1

**#include** <cstdlib>

**#include** <iostream>

**#include** <stdlib.h>

**#include** <sys/neutrino.h>

**#include** <process.h>

**#include** <string.h>

**using** std::cout;

**using** std::endl;

// Отправляет сообщение по полученному каналу

**void** **sendMessageToP**(**int** chid, **int** pid, **char** \*pX);

**int** **main**(**int** argc, **char** \*argv[]) {

cout << "P1: Запущен" << endl;

// Создание канала

**int** chid = **ChannelCreate**(\_NTO\_CHF\_SENDER\_LEN);

cout << "P1: Канал создан: chid = " << chid << endl;

// Преобразовать число в строку

**char** buffer[20];

**const** **char** \*chidStr = **itoa**(chid, buffer, 10);

// Вызов дочернего процесса P2

**int** pidP2 = **spawnl**( P\_NOWAIT, "/home/host/vika/lab1p2/x86/o/lab1p2", chidStr, NULL);

**if** (pidP2 < 0){

cout << "P1: Ошибка запуска процесса P2" << strerror(pidP2) << endl;

exit(EXIT\_FAILURE);

}

cout << "P1: pid процесса P2 - " << pidP2 << endl;

// Вызов дочернего процесса P2

**int** pidP3 = **spawnl**( P\_NOWAIT, "/home/host/vika/lab1p3/x86/o/lab1p3", chidStr, NULL);

**if** (pidP3 < 0){

cout << "P1: Ошибка запуска процесса P3" << strerror(pidP3) << endl;

exit(EXIT\_FAILURE);

}

cout << "P1: pid процесса P3 - " << pidP3 << endl;

**int** chidP2;

**int** chidP3;

**bool** isWaitP2 = **true**;

**bool** isWaitP3 = **true**;

**while**(isWaitP2 || isWaitP3){

// Буфер приема сообщения

**char** msg[200] = {};

// Информация о сообщении

\_msg\_info info;

// Ссылка на нить клиента

**int** rcvid = **MsgReceive**(chid, msg, **sizeof**(msg), &info);

**if**(rcvid == -1){

cout << "P1: Ошибка MsgReceive" << endl;

}

**if**(info.pid == pidP2){

cout << "P1: Получено сообщение от P2 - " << msg << endl;

**MsgReply**(rcvid, NULL, msg, **sizeof**(msg));

chidP2 = atoi(msg);

isWaitP2 = **false**;

}

**if**(info.pid == pidP3){

cout << "P1: Получено сообщение от P3 - " << msg << endl;

**MsgReply**(rcvid, NULL, msg, **sizeof**(msg));

chidP3 = atoi(msg);

isWaitP3 = **false**;

}

}

sendMessageToP(chidP2, pidP2, (**char** \*)"P2");

sendMessageToP(chidP3, pidP3, (**char** \*)"P3");

cout << "P1: OK" << endl;

**return** EXIT\_SUCCESS;

}

// Отправляет сообщение по полученному каналу

**void** **sendMessageToP**(**int** chid, **int** pid, **char** \*pX){

**char** rmsg[100];

cout << "P1: установление соединения с каналом " << pX << endl;

**int** coidP0 = **ConnectAttach**(0, pid, chid, \_NTO\_SIDE\_CHANNEL, 0);

**if**(coidP0 == -1){

cout << "P1: Ошибка соединения с каналом " << pX << " - " << strerror(coidP0) << endl;

exit(EXIT\_FAILURE);

}

**char** \*smsg = (**char** \*)"Р1 send message to ";

**char** result[100] = {};

strcat(result, smsg);

strcat(result, pX);

cout << "Р1: Посылаю сообщение " << pX << " - " << result << endl;

**int** sendRes = MsgSend(coidP0, result, strlen(result) + 1, rmsg, **sizeof**(rmsg));

**if**(sendRes == -1){

cout << "P1: Ошибка MsgSend при отправки в " << pX << " - " << strerror(sendRes) << endl;

exit(EXIT\_FAILURE);

}

**if**(strlen(rmsg) > 0){

cout << "P1: " << pX << " ответил - " << rmsg << endl;

}

}

ПРИЛОЖЕНИЕ B  
Листинг модуля M2

**#include** <cstdlib>

**#include** <iostream>

**#include** <stdlib.h>

**#include** <sys/neutrino.h>

**#include** <string.h>

**#include** <process.h>

**using** std::cout;

**using** std::endl;

**int** **main**(**int** argc, **char** \*argv[]) {

cout << "P2: Запущен" << endl;

cout << "P2: Параметры: " << "argv[0]= " << argv[0] << endl;

**int** pChid = atoi(argv[0]);

// Id канала для отправки сообщения

cout << "P2: установление соединения с каналом P1" << endl;

**int** coidP1 = **ConnectAttach**(0, **getppid**(), pChid, \_NTO\_SIDE\_CHANNEL, 0);

**if**(coidP1 == -1){

cout << "P2: Ошибка соединения с каналом P1 " << endl;

exit(EXIT\_FAILURE);

}

// Создание канала

**int** chid = **ChannelCreate**(\_NTO\_CHF\_SENDER\_LEN);

cout << "P2: Канал создан: chid = " << chid << endl;

// Преобразовать число в строку

**char** buffer[20];

**const** **char** \*chidStr = **itoa**(chid, buffer, 10);

// буфер ответа

**char** rmsg[200] = {};

// Послать сообщение

cout << "Р2: Посылаю сообщение Р1" << rmsg << endl;

**int** res = MsgSend(coidP1, chidStr, strlen(chidStr) + 1, rmsg, **sizeof**(rmsg));

**if**( res == -1){

cout << "P2: Ошибка MsgSend при отправки в P1" << endl;

exit(EXIT\_FAILURE);

}

cout << "Р2: Ожидаю сообщение от Р1" << endl;

**bool** isWait = **true**;

**while**(isWait){

// Буфер приема сообщения

**char** msg[200] = {};

// Информация о сообщении

\_msg\_info info;

// Ссылка на нить клиента

**int** rcvid = **MsgReceive**(chid, msg, **sizeof**(msg), &info);

**if**(rcvid == -1){

cout << "P2: Ошибка MsgReceive" << endl;

}

//if(info.pid == getppid()){

cout << "P2: Получено сообщение от P1 - " << msg << endl;

**char** \*msgOk = (**char** \*)"P2 OK";

MsgReply(rcvid, NULL, msgOk, strlen(msgOk));

isWait = **false**;

//}

}

cout << "P2: OK" << endl;

**return** EXIT\_SUCCESS;

}

ПРИЛОЖЕНИЕ С  
Листинг модуля M3

**#include** <cstdlib>

**#include** <iostream>

**#include** <stdlib.h>

**#include** <sys/neutrino.h>

**#include** <string.h>

**#include** <process.h>

**using** std::cout;

**using** std::endl;

**int** **main**(**int** argc, **char** \*argv[]) {

cout << "P3: Запущен" << endl;

cout << "P3: Параметры: " << "argv[0]= " << argv[0] << endl;

**int** pChid = atoi(argv[0]);

// Id канала для отправки сообщения

cout << "P3: установление соединения с каналом P1" << endl;

**int** coidP1 = **ConnectAttach**(0, **getppid**(), pChid, \_NTO\_SIDE\_CHANNEL, 0);

**if**(coidP1 == -1){

cout << "P3: Ошибка соединения с каналом P1 " << endl;

exit(EXIT\_FAILURE);

}

// Создание канала

**int** chid = **ChannelCreate**(\_NTO\_CHF\_SENDER\_LEN);

cout << "P3: Канал создан: chid = " << chid << endl;

// Преобразовать число в строку

**char** buffer[20];

**const** **char** \*chidStr = **itoa**(chid, buffer, 10);

// буфер ответа

**char** rmsg[200] = {};

// Послать сообщение

cout << "Р3: Посылаю сообщение Р1" << rmsg << endl;

**int** res = MsgSend(coidP1, chidStr, strlen(chidStr) + 1, rmsg, **sizeof**(rmsg));

**if**( res == -1){

cout << "P3: Ошибка MsgSend при отправки в P1" << endl;

exit(EXIT\_FAILURE);

}

cout << "Р3: Ожидаю сообщение от Р1" << endl;

**bool** isWait = **true**;

**while**(isWait){

// Буфер приема сообщения

**char** msg[200] = {};

// Информация о сообщении

\_msg\_info info;

// Ссылка на нить клиента

**int** rcvid = **MsgReceive**(chid, msg, **sizeof**(msg), &info);

**if**(rcvid == -1){

cout << "P2: Ошибка MsgReceive" << endl;

}

//if(info.pid == getppid()){

cout << "P3: Получено сообщение от P1 - " << msg << endl;

**char** \*msgOk = (**char** \*)"P3 OK";

MsgReply(rcvid, NULL, msgOk, strlen(msgOk));

isWait = **false**;

//}

}

cout << "P3: OK" << endl;

**return** EXIT\_SUCCESS;

}