

# Российский университет дружбы народов

## Факультет физико-математических и естественных наук Лабораторная работа № 14. Именованные каналы

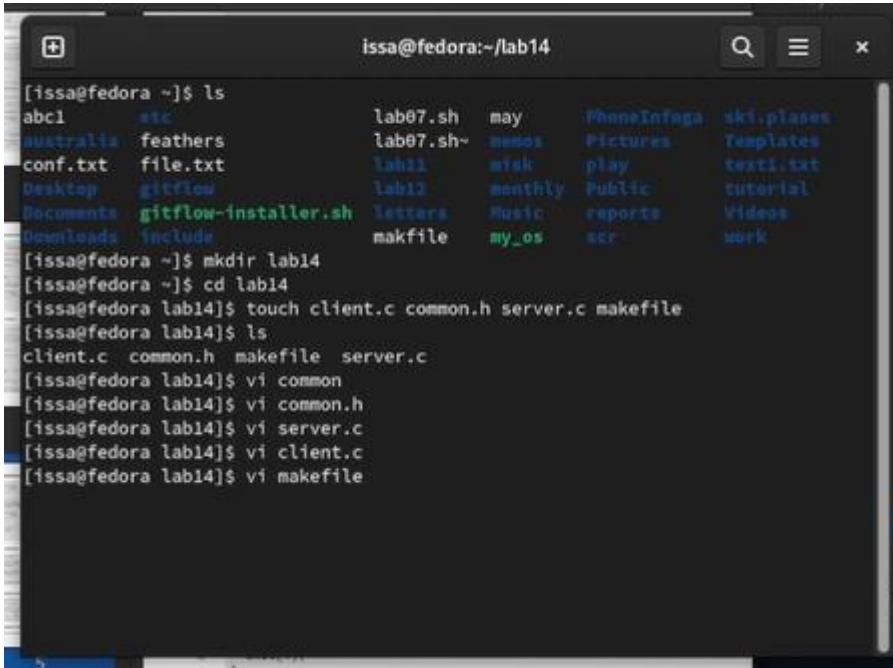
- Имя : исса гадир
- Студенческий билет : 1032218267
- Группа : нфибд-01-21

### Цель работы

*Приобретение практических навыков работы с именованными каналами.*

### выполнения работы

1. Работает не 1 клиент, а несколько (например, два).



```
issa@fedora:~/lab14
[issa@fedora ~]$ ls
abc1      etc          lab07.sh  may       PhoneInfega  ski.places
australia feathers    lab07.sh~ memos     Pictures     Templates
conf.txt  file.txt    lab11    wisk      play         text1.txt
Desktop   gitflow    lab12    monthly  Public       tutorial
Documents gitflow-installer.sh letters   Music      reports     Videos
Downloads include    makfile  my_os     scr         work

[issa@fedora ~]$ mkdir lab14
[issa@fedora ~]$ cd lab14
[issa@fedora lab14]$ touch client.c common.h server.c makefile
[issa@fedora lab14]$ ls
client.c  common.h  makefile  server.c
[issa@fedora lab14]$ vi common
[issa@fedora lab14]$ vi common.h
[issa@fedora lab14]$ vi server.c
[issa@fedora lab14]$ vi client.c
[issa@fedora lab14]$ vi makefile
```

2. Клиенты передают текущее время с некоторой периодичностью (например, раз в пять секунд). Используйте функцию `sleep()` для приостановки работы клиента.

```
0 bytes
+ issa@fedora:~/lab14 — /usr/libexec/vi common.h
#ifndef __COMMON_H__
#define __COMMON_H__

#include <stdio.h>
#include <stdlib.h>

#include <string.h>
#include <errno.h>
#include <sys/types.h>
#include <sys/stat.h>
#include <fcntl.h>

#define FIFO_NAME    "/tmp/fifo"
#define MAX_BUFF     80

#endif

-- INSERT --
2. запустить программу client на другой консоли.
```

```
+ issa@fedora:~/lab14 — /usr/libexec/vi server.c
}

while((n = read(readfd, buff, MAX_BUFF)) > 0)
{
    if(write(1, buff, n) != n)
    {
        fprintf(stderr, "%s: Ошибка вывода (%s)\n",
            __FILE__, strerror(errno));
        exit(-3);
    }
}

close(readfd);

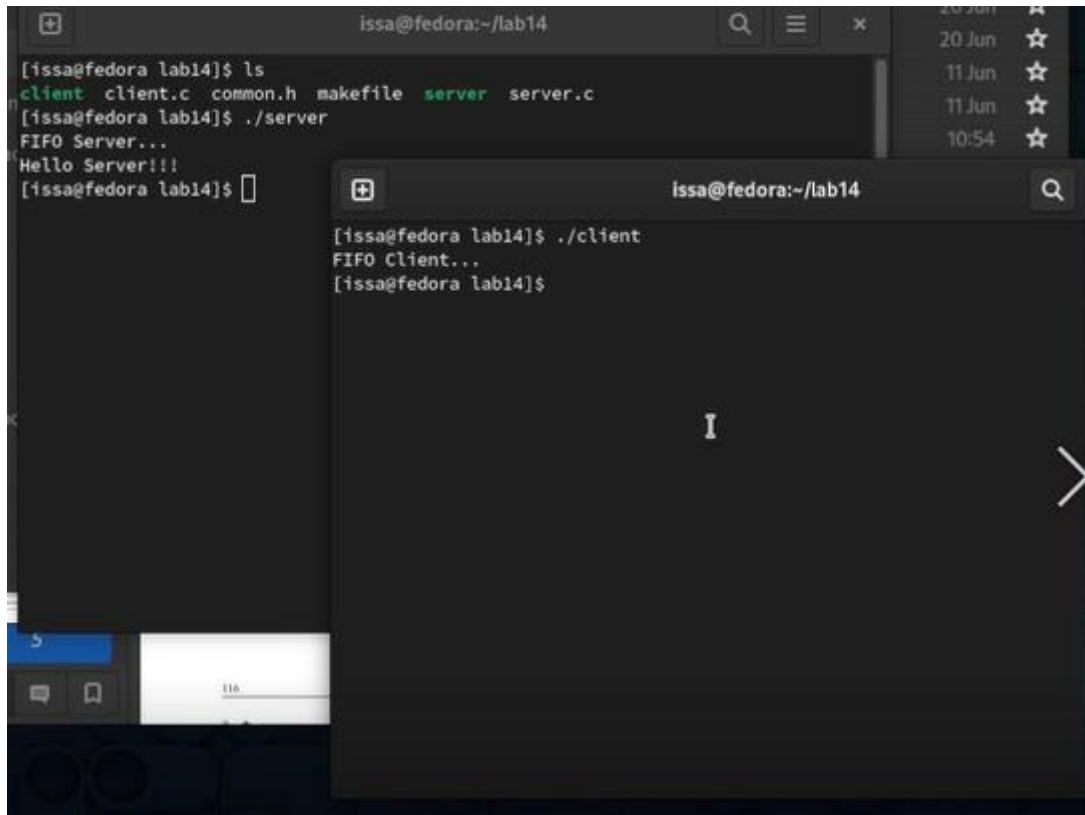
if(unlink(FIFO_NAME) < 0)
{
    fprintf(stderr, "%s: Невозможно удалить FIFO (%s)\n",
        __FILE__, strerror(errno));
    exit(-4);
}

exit(0);

-- INSERT --
```



3. Сервер работает не бесконечно, а прекращает работу через некоторое время (например, 30 сек). Используйте функцию `clock()` для определения времени работы сервера. Что будет в случае, если сервер завершит работу, не закрыв канал?



```
issa@fedora: ~/lab14
[issa@fedora lab14]$ ls
client client.c common.h makefile server server.c
[issa@fedora lab14]$ ./server
FIFO Server...
Hello Server!!!
[issa@fedora lab14]$

[issa@fedora lab14]$ ./client
FIFO Client...
[issa@fedora lab14]$
```

---

## **\*\*контрольные вопросы 🧐\***

### **1. В чем личное отличие именованных каналов от именованных ?**

именованные каналы , в отличие от неименованных , могут использоваться неродственни и процессами . Они дают нам , по сути , 14 же возможности , что и неименование копать , но некоторыми проурисущими обм файлам . Именование каналы используют специальную тались в директории для управлени правами доступа

### **2.возможно ли создание именованного копала из командной строки ?**

Можно

### **3.Возможно ли создание именованного канала и командной строки?**

Можно

**4. Опишите функцию языка C, создающую неименованный канал**

*mknod*

**5. Опишите функцию языка C, создающую именованный канал**

*Mkfifo*

**6. Что будет в случае прочтения из fifo меньшего числа байтов, чем находится в канале? Большого числа байтов?**

- . При чтении меньшего числа Байтов, чем находится в канале или FIFO, возвращается требуемое число байтов, остаток сохраняется для последующих чтений.
- . При чтении большего числа байтов, чем находится в канале или FIFO, возвращается доступное число байтов. Процесс, читающий из канала, должен соответствующим образом обработать ситуацию, когда прочитано меньше, чем заказано.

**7. Аналогично, что будет в случае записи в fifo меньшего числа байтов, чем позволяет буфер? Большого числа байтов?**

- Запись числа байтов, меньшего емкости канала или FIFO, гарантированно атомарно. Это означает, что в случае, когда несколько процессов одновременно записывают в канал, порции данных от этих процессов не перемешиваются.
- При записи большего числа байтов, чем это позволяет канал или FIFO, `write (2)` блокируется до освобождения требуемого места. При этом атомарность операции не гарантируется. Если процесс пытается писать в канал, не открытый ни одним процессом на чтение, процессу генерируется сигнал `SIGPIPE`, а `write (?)` идет установкой ошибки (`errno - EPIPE`) (если процесс установил обработку сигнала `SIGPIPE`, производится обработка по умолчанию процесс завершается)

**8. Могут ли пот и более процесс сон читать или записывать в канал?**

- можно **9. Опишите функцию `write` (тип возвращаемого значения, аргументы и логику работы). Что означает 1 (единица) в вызове этой функции в программе `server.c` (строк 42)**

*Отправить сообщение на сервер*

## **10 - Опишите функцию `strerror`**

*. Строковая функция `strerror` функция языков C / C ++ , транслирующая код ошибки , который обычно хранится в глобальной переменной `errno` , в сообщение об ошибке , понятном человеку*

### **Вывод**

*Я простейшим навыкам разработки , анализа , тестирования и отладки приложений в таких операционных системах , как UNIX / Lin*