

Лабораторная работа 14

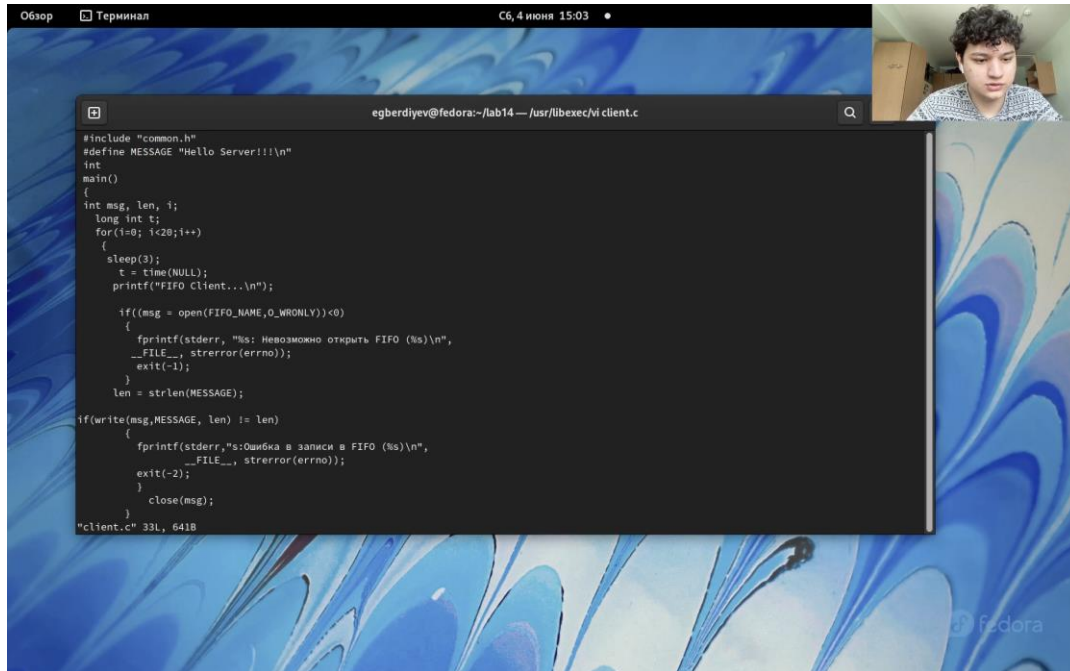
Бердыев Эзиз Группа НФИбд-01-21

Цель работы

Приобретение практических навыков работы с именованными каналами.

Выполнение

1. client.c



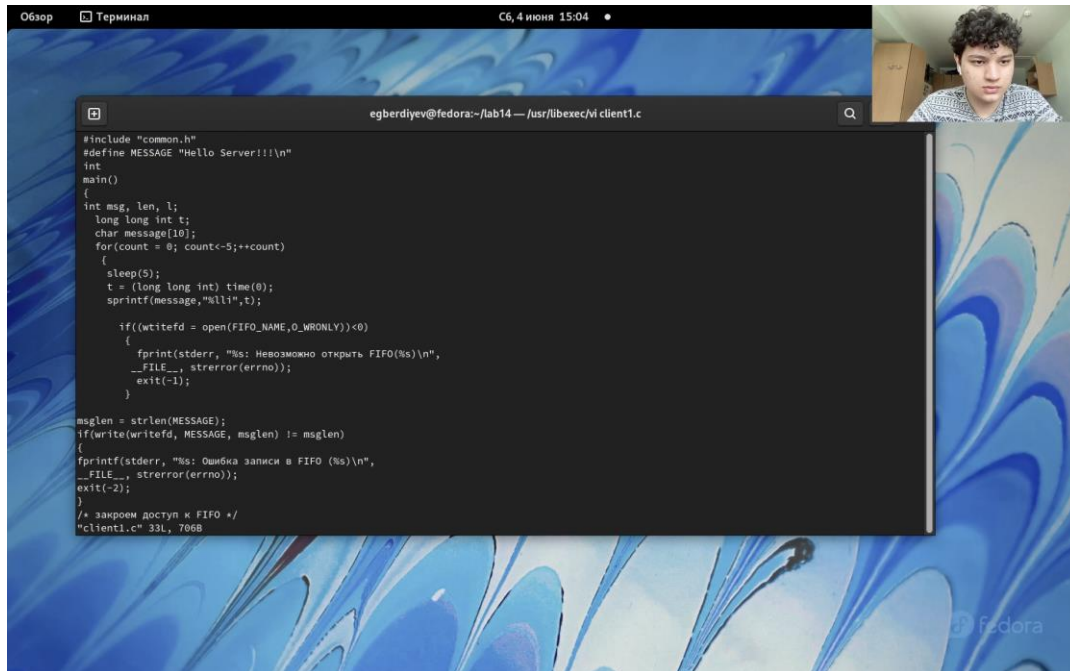
```
egberdiyev@fedora: ~/lab14 — /usr/libexec/vi client.c

#include "common.h"
#define MESSAGE "Hello Server!!!\n"
int
main()
{
    int msg, len, i;
    long int t;
    for(i=0; i<20;i++)
    {
        sleep(3);
        t = time(NULL);
        printf("FIFO client...\n");

        if((msg = open(FIFO_NAME,O_WRONLY))<0)
        {
            fprintf(stderr, "%s: Невозможно открыть FIFO (%s)\n",
                __FILE__, strerror(errno));
            exit(-1);
        }
        len = strlen(MESSAGE);
        if(write(msg,MESSAGE, len) != len)
        {
            fprintf(stderr,"%s:Ошибка в записи в FIFO (%s)\n",
                __FILE__, strerror(errno));
            exit(-2);
        }
        close(msg);
    }
}
"client.c" 33L, 641B
```

client.c.png

2. client1.c



```
egberdiyev@fedora:~/lab14 — /usr/libexec/vi client1.c

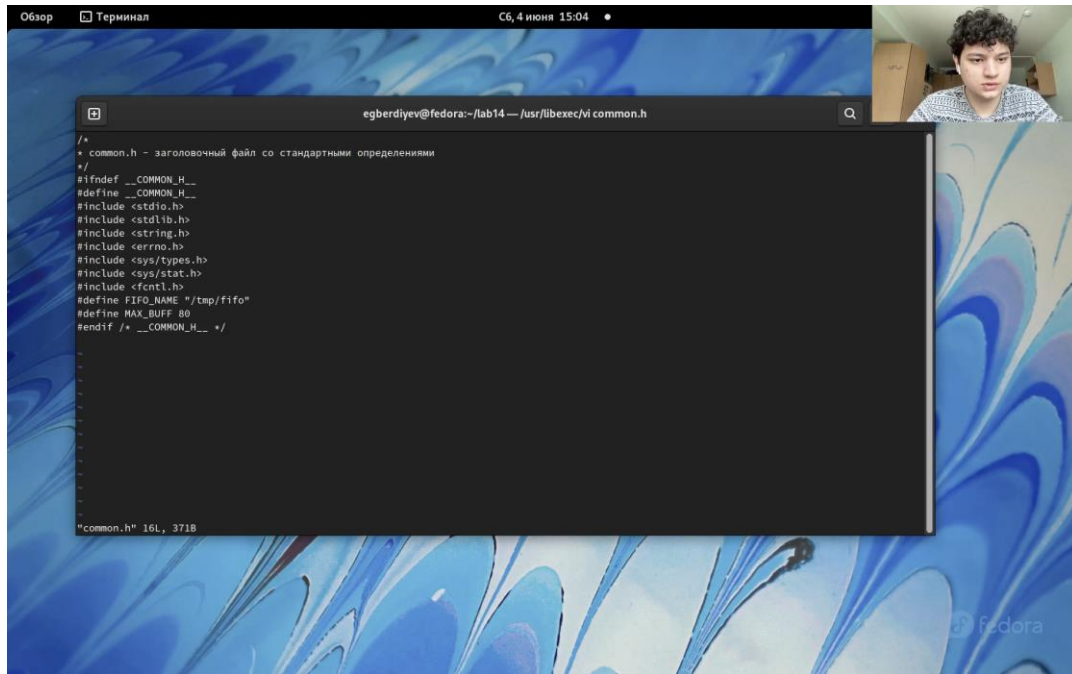
#include "common.h"
#define MESSAGE "Hello Server!!!\n"
int
main()
{
    int msg, len, i;
    long long int t;
    char message[10];
    for(count = 0; count<-5;++count)
    {
        sleep(5);
        t = (long long int) time(0);
        sprintf(message,"%lli",t);

        if((writofd = open(FIFO_NAME,O_WRONLY)<0)
        {
            fprintf(stderr, "%s: Невозможно открыть FIFO(%s)\n",
                __FILE__, strerror(errno));
            exit(-1);
        }

        msglen = strlen(MESSAGE);
        if(write(writofd, MESSAGE, msglen) != msglen)
        {
            fprintf(stderr, "%s: Ошибка записи в FIFO (%s)\n",
                __FILE__, strerror(errno));
            exit(-2);
        }
        /* закроем доступ к FIFO */
        "client1.c" 33L, 706B
```

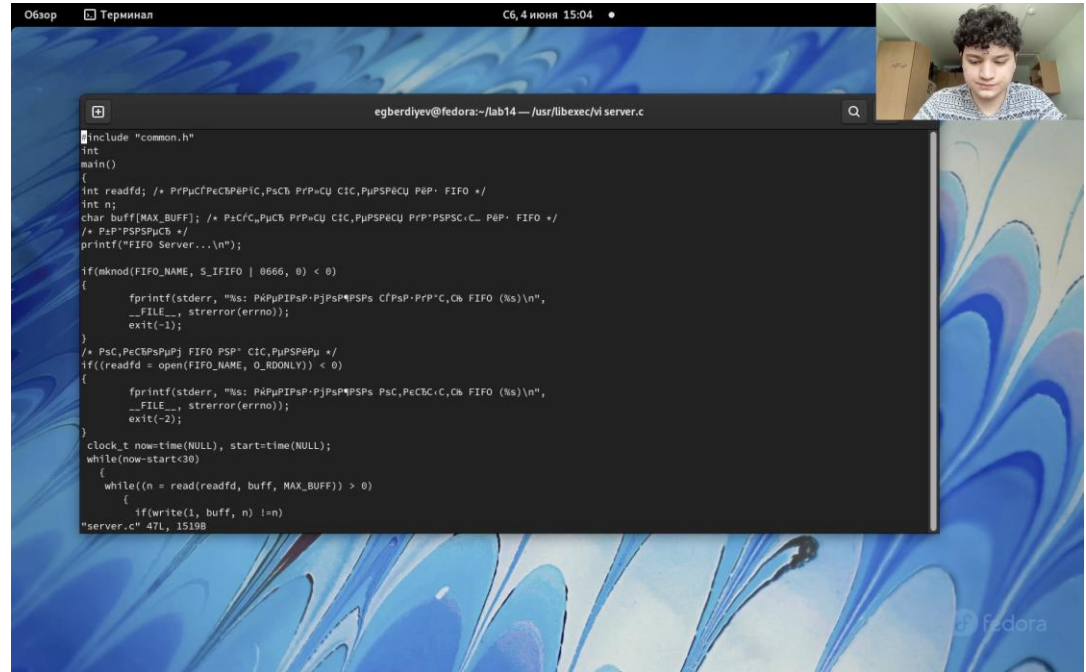
client1.c.png

3. common.h



common.h.png

4. server.c



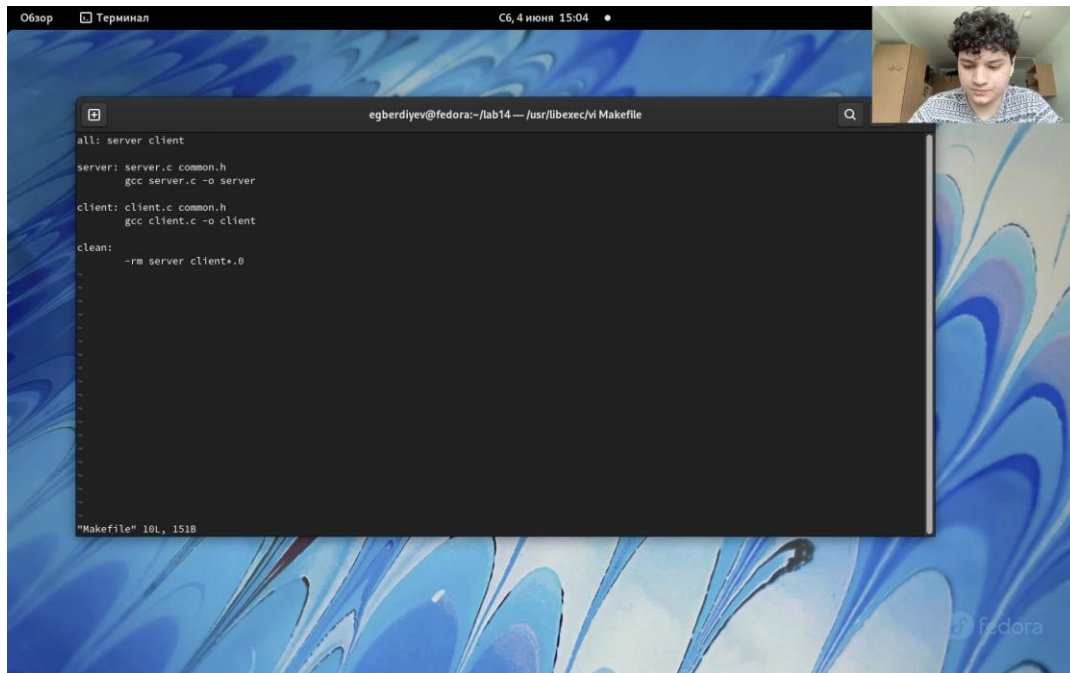
```
Обзор Терминал C6, 4 июня 15:04 • egberdiyev@fedora:~/lab14 — /usr/libexec/vi server.c

#include "common.h"
int
main()
{
    int readfd; /* PpPpCfPcCBPBPIC,PsCB PpP=CJ CJC,PjPSPBPJC PpP= FIFO */
    int n;
    char buff[MAX_BUFF]; /* P=CfC_PjPcB PpP=CJ CJC,PjPSPBPJC PpP=PSPSC_C_ PpP= FIFO */
    /* PpP=PSPSPjPcB */
    printf("FIFO Server...\n");

    if(mknod(FIFO_NAME, S_IFIFO | 0666, 0) < 0)
    {
        fprintf(stderr, "Ns: PpPpPjPpP-PjPpPMPSPs CfPsP-PpP=C,Ch FIFO (Ns)\n",
            __FILE__, strerror(errno));
        exit(-1);
    }
    /* PsC,PcCBPpPjPj FIFO PSP= CJC,PjPSPBPj */
    if((readfd = open(FIFO_NAME, O_RDONLY)) < 0)
    {
        fprintf(stderr, "Ns: PpPpPjPpP-PjPpPMPSPs PsC,PcCB=C,Ch FIFO (Ns)\n",
            __FILE__, strerror(errno));
        exit(-2);
    }
    clock_t now=time(NULL), start=time(NULL);
    while(now-start<30)
    {
        while((n = read(readfd, buff, MAX_BUFF)) > 0)
        {
            if(write(1, buff, n) !=n)
                "server.c" 47L, 1519B
        }
    }
}
```

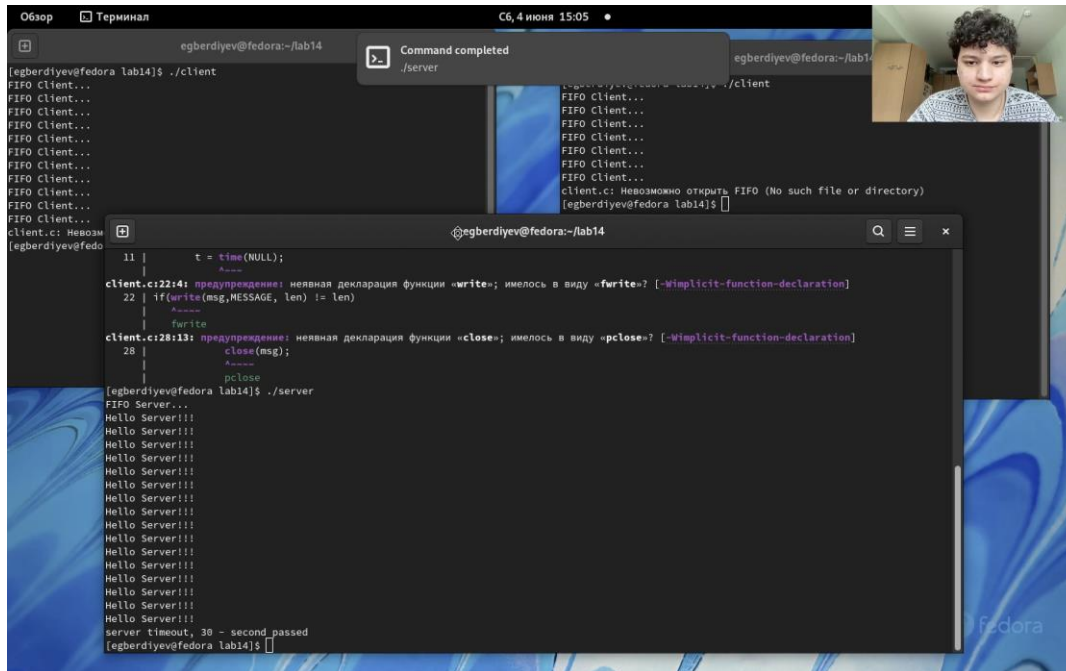
server.c.png

5. Makefile



Makefile.png

6. Результат



```
egberdiyev@fedora:~/lab14
[egberdiyev@fedora lab14]$ ./client
FIFO client...
FIFO client...
FIFO client...
FIFO client...
FIFO client...
FIFO client...
FIFO client...
FIFO client...
FIFO client...
FIFO client...
client.c: Неважно...
[egberdiyev@fedora lab14]$

[egberdiyev@fedora lab14]$ ./server
FIFO Server...
Hello Server!!!
Hello Server!!!
Hello Server!!!
Hello Server!!!
Hello Server!!!
Hello Server!!!
Hello Server!!!
Hello Server!!!
Hello Server!!!
Hello Server!!!
Hello Server!!!
Hello Server!!!
Hello Server!!!
Hello Server!!!
Hello Server!!!
Hello Server!!!
Hello Server!!!
Hello Server!!!
Hello Server!!!
Hello Server!!!
Hello Server!!!
server timeout, 30 - second_passed
[egberdiyev@fedora lab14]$
```

Compiler warnings shown in the terminal:

```
client.c:22:14: предупреждение: неявная декларация функции «write»; имелось в виду «fwrite»? [-Wimplicit-function-declaration]
22 | if(write(msg,MESSAGE, len) != len)
    |      ^~~~~~
client.c:28:13: предупреждение: неявная декларация функции «close»; имелось в виду «pclose»? [-Wimplicit-function-declaration]
28 |     close(msg);
    |     ^~~~~~
```

Результат.png

Контрольные вопросы

1. В чем ключевое отличие именованных каналов от неименованных? Именованные каналы, в отличие от неименованных, могут использоваться неродственными процессами. Они дают вам, по сути, те же возможности, что и неименованные каналы, но с некоторыми преимуществами, присущими обычным файлам. Именованные каналы используют специальную запись в директории для управления правами доступа.

2. Возможно ли создание неименованного канала из командной строки? Да, возможно

3. Возможно ли создание именованного канала из командной строки? Да, возможно

4. Какой результат даст вычисление выражения $\$(10/3)$?
CreatePipe используется для создания неименованного канала.

5. Опишите функцию языка, создающую именованный канал. Функции стандартной библиотеки Си, такие, как `open`, `fread`, `fwrite` и `fclose` позволяют обращаться к именованному каналу.

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. Что будет в случае записи из fifo меньшего числа байтов, чем находится в канале? 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, вызов `write(2)` блокируется до освобождения требуемого места. При этом атомарность операции не гарантируется. Если процесс пытается записать данные в канал, не открытый ни одним

8. Могут ли два и более процессов читать и записывать в канал? Да, при этом один из процессов записывает данные в канал, а другой их считывает.

9. 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

Приобрел практические навыки работы с именованными каналами.