

Отчет по лабораторной работе №14

Павлова Варвара Юрьевна НПИМбд-02-21

Москва, 2022

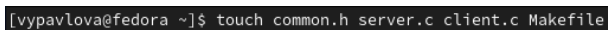
Цель работы

Приобретение практических навыков работы с именованными каналами.

Ход работы

Создание файлов

Создаю необходимые файлы с помощью команды touch. (рис. [-@fig:001])



```
[vypavlova@fedora ~]$ touch common.h server.c client.c Makefile
```

Рис. 1: создание файлов

Написание кода

common.h

В файл common.h добавляю стандартные заголовочные файлы time.h и unistd.h (рис. [-@fig:002])

server.c

В файл server.c добавляю цикл while для контроля за временем работы сервера. Разница между текущим временем и временем начала работы не должна превышать 30 секунд. (рис. [-@fig:003])

server.c

В файл server.c добавляю цикл while для контроля за временем работы сервера. Разница между текущим временем и временем начала работы не должна превышать 30 секунд. (рис. [-@fig:004])

```

/*
 * common.h - заголовочный файл со стандартными определениями
 */

#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>
#include <time.h>
#include <unistd.h>

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

#endif /* __COMMON_H__ */

```

Рис. 2: common.h

```

/*
 * server.c - реализация сервера
 *
 * Чтобы запустить пример, необходимо:
 * 1. запустить программу server на одной консоли
 * 2. запустить программу client на другой консоли
 */

#include "common.h"

int main()
{
    int readfd;
    int n;
    char buff[MAX_BUFF];

    printf("FIFO SERVER...\n");

    if(mknod(FIFO_NAME, S_IFIFO | 0666, 0) < 0)
    {
        fprintf(stderr, "%s: Невозможно создать FIFO (%s)\n",
            __FILE__, strerror(errno));
        exit(-1);
    }

    if((readfd = open(FIFO_NAME, O_RDONLY | O_NONBLOCK)) < 0)
    {
        fprintf(stderr, "%s: Невозможно открыть FIFO (%s)\n",
            __FILE__, strerror(errno));
        exit(-2);
    }

    clock_t start = time(NULL);

    while(time(NULL) - start < 30)
    {
        while((n = read(readfd, buff, MAX_BUFF)) > 0)
        {

```

Рис. 3: server.c

```

clock_t start = time(NULL);

while(time(NULL)-start < 30)

    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);
}

```

Рис. 4: server.c

client.c

В файл client.c добавляю цикл, который отвечает за количество сообщений о текущем времени и команду sleep(5) для приостановки работы клиента на 5 секунд. (рис. [-@fig:005])

```

/*
 * client.c
 *
 * чтобы запустить пример, необходимо:
 * 1. запустить программу server на одной консоли;
 * 2. запустить программу client на другой консоли.
 */
#include "common.h"

int main()
{
    int writefd;
    int msglen;

    printf("FIFO Client...\n");

    int i=0;
    for (i=0; i<4; i++)
    {
        if((writefd = open(FIFO_NAME, O_WRONLY)) < 0)
        {
            fprintf(stderr, "%s: Невозможно открыть FIFO (%s)\n",
                __FILE__, strerror(errno));
            exit(-1);
            break;
        }

        long int ttime=time(NULL);
        char* text=ctime(&ttime);
    }
}

```

Рис. 5: client.c

client.c

В файл client.c добавляю цикл, который отвечает за количество сообщений о текущем времени и команду sleep(5) для приостановки работы клиента на 5 секунд. (рис. [-@fig:006])

Makefile

Файл для сборки - Makefile - не изменяю. (рис. [-@fig:007])

```

    long int ttime=time(NULL);
    char* text=ctime(&ttime);

    msglen = strlen(text);
    if(write(writefd, text, msglen) != msglen)
    {
        fprintf(stderr, "%s: Ошибка записи в FIFO (%s)\n",
            __FILE__, strerror(errno));
        exit(-2);
    }
    sleep(5);
}
close(writefd);
exit(0);
}

```

Рис. 6: client.c

```

all: server client

server: server.c common.h
    gcc server.c -o server

client: client.c common.h
    gcc client.c -o client

clean:
    -rm server client *.o

```

Рис. 7: Makefile

Компиляция

Выполняю команду make all и компилирую необходимые файлы. (рис. [-@fig:008])

```

[vypavlova@fedora ~]$ make all
gcc server.c -o server
gcc client.c -o client

```

Рис. 8: компиляция

Проверка работы

Проверяю работу написанного кода. Открываю 3 консоли и запускаю в одном из них ./server(рис. [-@fig:009]), а в двух других - ./client. (рис. [-@fig:010]) (рис. [-@fig:011])

Выводы

Выполняя данную лабораторную работу я приобрела практические навыки работы с именованными каналами.

```
FIFO SERVER...
Fri Jun 3 15:31:51 2022
Fri Jun 3 15:31:56 2022
Fri Jun 3 15:31:57 2022
Fri Jun 3 15:32:01 2022
Fri Jun 3 15:32:02 2022
Fri Jun 3 15:32:06 2022
Fri Jun 3 15:32:07 2022
Fri Jun 3 15:32:12 2022
□
```

Рис. 9: ./server

```
[vypavlova@fedora ~]$ ./client
FIFO Client...
□
```

Рис. 10: ./client

```
[vypavlova@fedora ~]$ ./client
FIFO Client...
□
```

Рис. 11: ./client