

ОДЕСЬКИЙ НАЦІОНАЛЬНИЙ ПОЛІТЕХНІЧНИЙ УНІВЕРСИТЕТ
ІНСТИТУТ КОМП'ЮТЕРНИХ СИСТЕМ
КАФЕДРА «ІНФОРМАЦІЙНИХ СИСТЕМ»

Лабораторна робота № 11
з дисципліни «Операційні
системи»

Тема «Програмування міжпроцесної та багатопоточної взаємодії»

Виконав:

Студент групи AI-202
Лобко Данііл Віталійович

Одеса-2021

Мета роботи: вивчити особливості обміну інформацією між процесами за допомогою іменованих каналів, керування потоками, а також синхронізацію процесів через семафори та м'ютекси.

Перелік завдань:

2 Завдання

2.1 Робота з іменованими каналами

2.1.1 В домашньому каталозі вашого користувача створіть іменований канал з

використанням команди `mkfifo`:

- назва каналу співпадає з вашим прізвищем у транслітерації
- права доступу до каналу (можна лише читати та писати власнику).

2.1.2 Підключіть до іменованого каналу процес, який буде в нього писати за такими

командами:

- отримати зміст каталогу `/etc`
- отримати назви файлів, які починаються з букви вашого прізвища у транслітерації.

2.1.3 Перейдіть до нового терміналу роботи з ОС Linux та створіть процес, який

буде читати зі створеного раніше каналу.

2.1.4 Поверніться до 1-го терміналу та підключіть до іменованого каналу процес,

який буде в нього писати, архівуючи файл командою `gzip -c < pipe > file1.gz`

де `pipe` – назва вашого каналу, `file1.gz` – назва файлу, який буде створено в результаті архівації

2.1.5 Перейдіть до 2-го терміналу роботи з ОС Linux та створіть процес, який буде

читати зі створеного раніше каналу, архівуючи файл `/etc/passwd`

2.2 Програмування іменованих каналів

Повторіть попереднє завдання, але пункт 2.1.1 виконайте через програмування

іменованого каналу за прикладом з рисунку 1.

2.3 Програмування потоків

За прикладом з рисунку 2 розробіть програму керування потоками, в якій в повідомленнях буде вказано ваше прізвище латиницею.

Виконайте програму за вказаним прикладом.

2.4 Програмування семафорів

За прикладом з рисунку 3 розробіть програму керування семафором, в якій в повідомленнях буде вказано ваше прізвище латиницею.

Виконайте програму в двох терміналах за вказаним прикладом.

Хід роботи

2 Завдання

2.1 Робота з іменованими каналами

2.1.1 В домашньому каталозі вашого користувача створіть іменований канал з

використанням команди `mkfifo`:

- назва каналу співпадає з вашим прізвищем у транслітерації
- права доступу до каналу (можна лише читати та писати власнику).

```
lobko_daniil@vpsj3IeQ:~/lobko_lab_12
[lobko_daniil@vpsj3IeQ lobko_lab_12]$ mkfifo lobko
[lobko_daniil@vpsj3IeQ lobko_lab_12]$ ls -l
total 0
prw-rw-r-- 1 lobko_daniil lobko_daniil 0 May 25 21:25 lobko
[lobko_daniil@vpsj3IeQ lobko_lab_12]$ ls
lobko
[lobko_daniil@vpsj3IeQ lobko_lab_12]$ chmod 744 lobko
[lobko_daniil@vpsj3IeQ lobko_lab_12]$ ls -l
total 0
prwxr--r-- 1 lobko_daniil lobko_daniil 0 May 25 21:25 lobko
[lobko_daniil@vpsj3IeQ lobko_lab_12]$
```

2.1.2 Підключіть до іменованого каналу процес, який буде в нього писати за такими

командами:

- отримати зміст каталогу /etc
- отримати назви файлів, які починаються з букви вашого прізвища у транслітерації.

2.1.3 Перейдіть до нового терміналу роботи з ОС Linux та створіть процес, який

буде читати зі створеного раніше каналу.

```
lobko_daniil@vpsj3IeQ:~/lobko_lab_12
[lobko_daniil@vpsj3IeQ lobko_lab_12]$ mkfifo lobko
[lobko_daniil@vpsj3IeQ lobko_lab_12]$ chmod 744 lobko
[lobko_daniil@vpsj3IeQ lobko_lab_12]$ ls -l
total 0
prwxr--r-- 1 lobko_daniil lobko_daniil 0 May 25 22:09 lobko
[lobko_daniil@vpsj3IeQ lobko_lab_12]$ clear
[lobko_daniil@vpsj3IeQ lobko_lab_12]$ find / -name "l" | ls /etc > lobko
find: '/dev/shm/orahpatch_XE': Permission denied
find: '/sys/kernel/debug': Permission denied
find: '/opt/oracle/oradata': Permission denied
find: '/opt/oracle/product/18c/dbhomeXE/opmn/conf': Permission denied
find: '/opt/oracle/product/18c/dbhomeXE/oui/prov/resources/scripts': Permission denied
find: '/opt/oracle/product/18c/dbhomeXE/oui/bin/resource': Permission denied
find: '/opt/oracle/product/18c/dbhomeXE/xdk': Permission denied
find: '/opt/oracle/product/18c/dbhomeXE/log/vpsj3ieq': Permission denied
find: '/opt/oracle/product/18c/dbhomeXE/log/diag': Permission denied
find: '/opt/oracle/product/18c/dbhomeXE/inventory': Permission denied
find: '/opt/oracle/product/18c/dbhomeXE/cfgtoollogs': Permission denied
find: '/opt/oracle/product/18c/dbhomeXE/network/log': Permission denied
find: '/opt/oracle/product/18c/dbhomeXE/network/trace': Permission denied
find: '/opt/oracle/product/18c/dbhomeXE/data/wallet': Permission denied
find: '/opt/oracle/product/18c/dbhomeXE/rdbms/log/opath': Permission denied
find: '/opt/oracle/orainventory': Permission denied

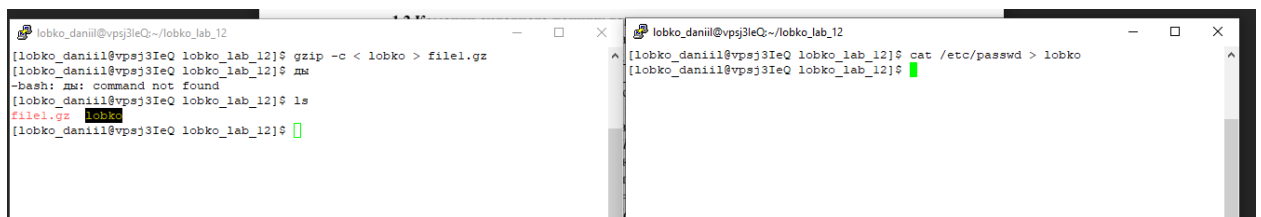
lobko_daniil@vpsj3IeQ:~/lobko_lab_12
sysctl.conf
sysctl.d
systemd
system-release
system-release-cpe
tcsd.conf
terminfo
tmpfiles.d
trusted-key.key
tuned
udev
vconsole.conf
vimrc
virc
vmmail
wgetrc
wpa_supplicant
X11
xdg
xinetd.d
xml
yum
yum.conf
yum.repos.d
[lobko_daniil@vpsj3IeQ lobko_lab_12]$
```

2.1.4 Поверніться до 1-го терміналу та підключіть до іменованого каналу процес,

який буде в нього писати, архівуючи файл командою `gzip -c < pipe > file1.gz` де `pipe` – назва вашого каналу, `file1.gz` – назва файлу, який буде створено в результаті архівації

2.1.5 Перейдіть до 2-го терміналу роботи з ОС Linux та створіть процес, який буде

читати зі створеного раніше каналу, архівуючи файл `/etc/passwd`



```
lobko_daniil@vpsj3IeQ:~/lobko_lab_12
[lobko_daniil@vpsj3IeQ lobko_lab_12]$ gzip -c < lobko > file1.gz
[lobko_daniil@vpsj3IeQ lobko_lab_12]$ mv
-bash: mv: command not found
[lobko_daniil@vpsj3IeQ lobko_lab_12]$ ls
file1.gz lobko
[lobko_daniil@vpsj3IeQ lobko_lab_12]$
```

```
lobko_daniil@vpsj3IeQ:~/lobko_lab_12
[lobko_daniil@vpsj3IeQ lobko_lab_12]$ cat /etc/passwd > lobko
[lobko_daniil@vpsj3IeQ lobko_lab_12]$
```

```
file1.gz lobko
[lobko_daniil@vpsj3IeQ lobko_lab_12]$ mc file1.gz
root:x:0:0:root:/root:/bin/bash
bin:x:1:1:bin:/bin:/sbin/nologin
daemon:x:2:2:daemon:/sbin:/sbin/nologin
adm:x:3:4:adm:/var/adm:/sbin/nologin
lp:x:4:7:lp:/var/spool/lpd:/sbin/nologin
sync:x:5:0:sync:/sbin:/bin/sync
shutdown:x:6:0:shutdown:/sbin:/sbin/shutdown
halt:x:7:0:halt:/sbin:/sbin/halt
mail:x:8:12:mail:/var/spool/mail:/sbin/nologin
operator:x:11:0:operator:/root:/sbin/nologin
games:x:12:100:games:/usr/games:/sbin/nologin
ftp:x:14:50:FTP User:/var/ftp:/sbin/nologin
```

2.2 Програмування іменованих каналів

Повторіть попереднє завдання, але пункт 2.1.1 виконайте через програмування

іменованого каналу за прикладом з рисунку 1.

```
#include <sys/stat.h>
#include <fcntl.h>
#include <string.h>
#include <stdio.h>
#define NAMEDPIPE_NAME "/lobko_task2"
#define BUFSIZE 50

int main (int argc, char ** argv) {
    int fd, len;
    char buf[BUFSIZE];
    if ( mkfifo(NAMEDPIPE_NAME, 0744) ) {
        fprintf(stderr, "Error in mkfifo!");
        return 1;
    }
    printf("%s is created\n", NAMEDPIPE_NAME);
    if ( (fd = open(NAMEDPIPE_NAME, O_RDONLY)) <= 0 ) {
        fprintf(stderr, "Error in open!");
        return 1;
    }
    printf("%s is opened\n", NAMEDPIPE_NAME);
    do {
        memset(buf, '\0', BUFSIZE);
        if ( (len = read(fd, buf, BUFSIZE-1)) <= 0 ) {
            printf("END!");
            close(fd);
            remove(NAMEDPIPE_NAME);
            return 0;
        }
        printf("Incomming message (%d): %s\n", len, buf);
    } while ( 1 );
}
```

```

#include <sys/stat.h>
#include <fcntl.h>
#include <string.h>
#include <stdio.h>
#define NAMEDPIPE_NAME "/lobko_task2"
#define BUFSIZE 50

int main (int argc, char ** argv) {
    int fd, len;
    char buf[BUFSIZE];
    if ( !mkfifo(NAMEDPIPE_NAME, 0744) ) {
        fprintf(stderr, "Error in mkfifo!");
        return 1;
    }
    printf("%s is created\n", NAMEDPIPE_NAME);
    if ( (fd = open(NAMEDPIPE_NAME, O_RDONLY)) <= 0 ) {
        fprintf(stderr, "Error in open!");
        return 1;
    }
    printf("%s is opened\n", NAMEDPIPE_NAME);
    do {
        memset(buf, '\0', BUFSIZE);
        if ( (len = read(fd, buf, BUFSIZE-1)) <= 0 ) {
            printf("END!");
            close(fd);
            remove(NAMEDPIPE_NAME);
            return 0;
        }
        printf("Incomming message (%d): %s\n", len, buf);
    } while ( 1 );
}

```

lobko_daniil@vpsj3IeQ:~/lobko_lab_12

[lobko_daniil@vpsj3IeQ lobko_lab_12]\$ ls -l

total 120

```

-rwxrwxr-x 1 lobko_daniil lobko_daniil 8760 May 26 00:11 2.2
-rw-rw-r-- 1 lobko_daniil lobko_daniil 656 May 26 00:12 2.2.c
-rwxrwxr-x 1 lobko_daniil lobko_daniil 8768 May 26 00:08 2.2_task
-rw-rw-r-- 1 lobko_daniil lobko_daniil 841 May 26 00:10 2.2_task.c
-rwxrwxr-x 1 lobko_daniil lobko_daniil 8640 May 26 00:00 2.3.1
-rw-rw-r-- 1 lobko_daniil lobko_daniil 578 May 26 00:00 2.3.1.c
-rwxrwxr-x 1 lobko_daniil lobko_daniil 8672 May 25 23:57 2.3.2
-rw-rw-r-- 1 lobko_daniil lobko_daniil 920 May 25 23:56 2.3.2.c
-rwxrwxr-x 1 lobko_daniil lobko_daniil 8672 May 25 23:35 2.3_task.c
-rwxrwxr-x 1 lobko_daniil lobko_daniil 8672 May 25 23:26 2.4.c
-rwxrwxr-x 1 lobko_daniil lobko_daniil 8760 May 26 00:18 2_task
-rw-rw-r-- 1 lobko_daniil lobko_daniil 844 May 26 00:19 2_task.c
-rwxrwxr-x 1 lobko_daniil lobko_daniil 8768 May 26 00:03 a.out
-rw-rw-r-- 1 lobko_daniil lobko_daniil 2573 May 25 22:34 file1.gz
prwxr--r-- 1 lobko_daniil lobko_daniil 0 May 25 22:34 lobko
prwxr--r-- 1 lobko_daniil lobko_daniil 0 May 26 00:20 lobko_task2
[lobko_daniil@vpsj3IeQ lobko_lab_12]$

```

2.3 Програмування потоків

За прикладом з рисунку 2 розробіть програму керування потоками, в якій в повідомленнях буде вказано ваше прізвище латиницею.

Виконайте програму за вказаним прикладом.

```
lobko_daniil@vpsj3leQ:~/lobko_lab_12
GNU nano 2.3.1 File: 2.3.1.c

#include <stdio.h>
#include <pthread.h>

main() {
    pthread_t f2_thread, f1_thread;
    void *f2(), *f1();
    int i1 = 10, i2 = 10;
    pthread_create(&f1_thread, NULL, f1, &i1);
    pthread_create(&f2_thread, NULL, f2, &i2);
    pthread_join(f1_thread, NULL);
    pthread_join(f2_thread, NULL);
}

void *f1(int *x) {
    int i,n;
    n = *x;
    for (i=1;i<n;i++) {
        printf("lobko f1: %d\n", i);
        sleep(1);
    }
    pthread_exit(0);
}

void *f2(int *x) {
    int i,n;
    n = *x;
    for (i=1;i<n;i++) {
        printf("lobko f2: %d\n", i);
        sleep(1);
    }
    pthread_exit(0);
}
```



```

lobko_daniil@vpsj3IeQ:~/lobko_lab_12
[lobko_daniil@vpsj3IeQ lobko_lab_12]$ ./2.3.1
lobko f2: 1
lobko f1: 1
lobko f2: 2
lobko f1: 2
lobko f2: 3
lobko f1: 3
lobko f2: 4
lobko f1: 4
lobko f1: 5
lobko f2: 5
lobko f1: 6
lobko f2: 6
lobko f1: 7
lobko f2: 7
lobko f1: 8
lobko f2: 8
lobko f1: 9
lobko f2: 9
[lobko_daniil@vpsj3IeQ lobko_lab_12]$

```

2.4 Програмування семафорів

За прикладом з рисунку 3 розробіть програму керування семафором, в якій в повідомленнях буде вказано ваше прізвище латиницею.

Виконайте програму в двох терміналах за вказаним прикладом.

```

#include <fcntl.h>
#include <sys/stat.h>
#include <semaphore.h>
#include <stdio.h>
#define SEMAPHORE_NAME "/lobko"
int main(int argc, char ** argv) {
    sem_t *sem;
    if ( argc != 2 ) {
        if ( (sem = sem_open(SEMAPHORE_NAME, O_CREAT, 0777, 0)) == SEM_FAILED ) {
            fprintf(stderr, "sem_open error");
            return 1;
        }
        printf("lobko. sem_open. Semaphore is taken.\nWaiting for it to be dropped.\n");
        if (sem_wait(sem) < 0 )
            fprintf(stderr, "sem_wait error");
        if ( sem_close(sem) < 0 )
            fprintf(stderr, "sem_close error");
        return 0;
    }
    else {
        printf("lobko. Dropping semaphore...\n");
        if ( (sem = sem_open(SEMAPHORE_NAME, 0)) == SEM_FAILED ) {
            fprintf(stderr, "sem_open error");
            return 1;
        }
        sem_post(sem);
        printf("lobko. sem_post. Semaphore dropped.\n");
        return 0;
    }
}

```

```
[lobko_daniil@vpsj3IeQ lobko_lab_12]$ gcc 2.3.2.c -O 2.3.2 -lpthread
gcc: error: unrecognized command line option '-O'
[lobko_daniil@vpsj3IeQ lobko_lab_12]$ gcc 2.3.2.c -o 2.3.2 -lpthread
[lobko_daniil@vpsj3IeQ lobko_lab_12]$ ./2.3.2
lobko. sem_open. Semaphore is taken.
Waiting for it to be dropped.
[lobko_daniil@vpsj3IeQ lobko_lab_12]$
```

```
[lobko_daniil@vpsj3IeQ lobko_lab_12]$ ./2.3.2 1
lobko. Dropping semaphore...
lobko. sem_post. Semaphore dropped.
[lobko_daniil@vpsj3IeQ lobko_lab_12]$
```

Висновки: Виконуючи цю лабораторну роботи ми закріпили навички роботи з встановлення віртуальної операційної системи ОС Linux на прикладі програмного забезпечення віртуальної машини Oracle Virtual Box.

Найскладнішим було завдання 7.