

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ДЕРЖАВНИЙ УНІВЕРСИТЕТ “ ОДЕСЬКА ПОЛІТЕХНІКА “
Інститут комп’ютерних систем
Кафедра інформаційних систем

Лабораторна робота № 12
З дисципліни: «Операційні системи»
Тема: «Програмування міжпроцесної та багатопоточної взаємодії»

Виконала:
ст.гр. АІ -204
Сіренко Марія

Перевірили:
Блажко О.А.
Дрозд М.О

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

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.

Виконання завдань:

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).

```
masha@ThinkPad-E14:~/Univer/os/lab_12$ mkfifo sirenko
masha@ThinkPad-E14:~/Univer/os/lab_12$ chmod 600 sirenko
masha@ThinkPad-E14:~/Univer/os/lab_12$ ls -l
total 8
drwxrwxr-x 2 masha masha 4096 tpa 22 08:33 lab12_examples
-rw-rw-r-- 1 masha masha 2364 tpa 22 16:16 lab12_examples.zip
prw----- 1 masha masha    0 tpa 22 17:27 sirenko
masha@ThinkPad-E14:~/Univer/os/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.

```
File Edit View Bookmarks Settings Help
masha@ThinkPad-E14:~/Univer/os/lab_12$ ls /etc |grep "^s" >sirenko
masha@ThinkPad-E14:~/Univer/os/lab_12$
```

2.1.3 Перейдіть до нового терміналу роботи з ОС Linux та створіть процес, який буде читати зі створеного раніше каналу.

```
masha@ThinkPad-E14:~/Univer/os/lab_12$ cat sirenko
samba
sane.d
sddm
security
selinux
sensors3.conf
sensors.d
services
sgml
shadow
shadow-
shells
signond.conf
signon-ui
```

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

```
File Edit View Bookmarks Settings Help
masha@ThinkPad-E14:~/Univer/os/lab_12$ gzip -c <sirenko> file1.gz
masha@ThinkPad-E14:~/Univer/os/lab_12$
```

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`

```
masha@ThinkPad-E14:~/Univer/os/lab_12$ cat /etc/passwd >sirenko | gunzip -c file1.gz
root:x:0:0:root:/root:/bin/bash
daemon:x:1:1:daemon:/usr/sbin:/usr/sbin/nologin
bin:x:2:2:bin:/bin:/usr/sbin/nologin
sys:x:3:3:sys:/dev:/usr/sbin/nologin
sync:x:4:65534:sync:/bin:/bin/sync
games:x:5:60:games:/usr/games:/usr/sbin/nologin
man:x:6:12:man:/var/cache/man:/usr/sbin/nologin
lp:x:7:7:lp:/var/spool/lpd:/usr/sbin/nologin
mail:x:8:8:mail:/var/mail:/usr/sbin/nologin
news:x:9:9:news:/var/spool/news:/usr/sbin/nologin
uucp:x:10:10:uucp:/var/spool/uucp:/usr/sbin/nologin
proxy:x:13:13:proxy:/bin:/usr/sbin/nologin
www-data:x:33:33:www-data:/var/www:/usr/sbin/nologin
backup:x:34:34:backup:/var/backups:/usr/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 "./sirenko"
#define BUFSIZE      50

int main (int argc, char ** argv) {
    int fd, len;
    char buf[BUFSIZE];

    if ( mkfifo(NAMEDPIPE_NAME, 0777) ) {
        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 );
}
```



```
lab12_examples  fifo  fifo.c  file1.gz  lab12_examples.  sirenko
```

```
lab_12: fifo — Konsole
File Edit View Bookmarks Settings Help
masha@ThinkPad-E14:~/Univer/os/lab_12$ gcc fifo.c -o fifo
fifo.c: In function 'main':
fifo.c:27:21: warning: implicit declaration of function 'read'; did you mean 'fread'? [-Wimplicit-function-declaration]
   27 |         if ( (len = read(fd, buf, BUFSIZE-1)) <= 0 ) {
       |                     ^~~~~
       |                     fread
fifo.c:29:13: warning: implicit declaration of function 'close'; did you mean 'pclose'? [-Wimplicit-function-declaration]
   29 |         close(fd);
       |         ^~~~~~
       |         pclose
masha@ThinkPad-E14:~/Univer/os/lab_12$ ./fifo
./sirenko is created
```

```
masha@ThinkPad-E14:~/Univer/os/lab_12$ ls /etc |grep "^s" >sirenko
masha@ThinkPad-E14:~/Univer/os/lab_12$
```

```
pclose
masha@ThinkPad-E14:~/Univer/os/lab_12$ ./fifo
./sirenko is created
./sirenko is opened
Incomming message (49): samba
sane.d
sddm
security
selinux
sensors3.conf

Incomming message (49): sensors.d
services
sgml
shadow
shadow-
shells
sig
Incomming message (49): nond.conf
signon-ui
skel
```

2.3 Програмування потоків

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

Виконайте програму за вказаним прикладом.

```
#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("f1: Sirenko %d\n", i);
        sleep(1);
    }
    pthread_exit(0);
}

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

```
masha@ThinkPad-E14:~/Univer/os/lab_12$ gcc thread.c -o thread -lpthread
```

```
masha@ThinkPad-E14:~/Univer/os/lab_12$ ./thread
```

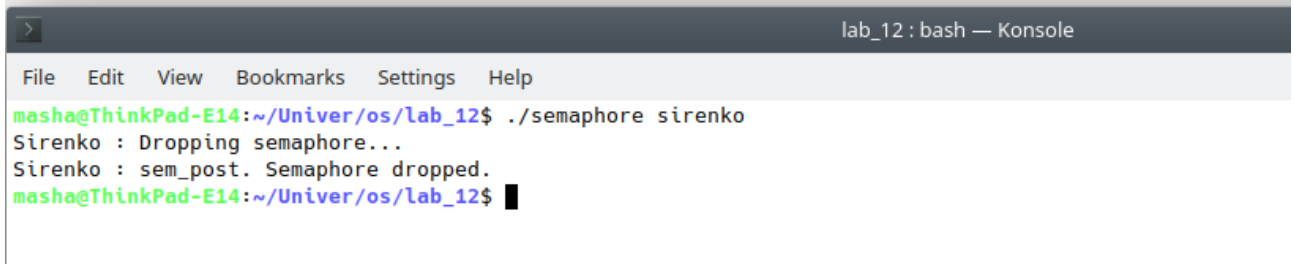
```
f2: Sirenko 1
f1: Sirenko 1
f2: Sirenko 2
f1: Sirenko 2
f2: Sirenko 3
f1: Sirenko 3
f2: Sirenko 4
f1: Sirenko 4
f2: Sirenko 5
f1: Sirenko 5
f2: Sirenko 6
f1: Sirenko 6
f2: Sirenko 7
f1: Sirenko 7
f1: Sirenko 8
f2: Sirenko 8
f1: Sirenko 9
f2: Sirenko 9
```

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.

```
Sirenko : sem_open error masha@ThinkPad-E14:~/Univer/os/lab_12$ ./semaphore
Sirenko : sem_open error masha@ThinkPad-E14:~/Univer/os/lab_12$ gcc semaphore.c -o semaphore -lpthread
masha@ThinkPad-E14:~/Univer/os/lab_12$ ./semaphore
Sirenko : sem_open. Semaphore is taken.
Waiting for it to be dropped.
masha@ThinkPad-E14:~/Univer/os/lab_12$
```



```
masha@ThinkPad-E14:~/Univer/os/lab_12$ ./semaphore sirenko
Sirenko : Dropping semaphore...
Sirenko : sem_post. Semaphore dropped.
masha@ThinkPad-E14:~/Univer/os/lab_12$
```

```
#include <fcntl.h>
#include <sys/stat.h>
#include <semaphore.h>
#include <stdio.h>

#define SEMAPHORE_NAME "/my_named_semaphore"

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, "Sirenko : sem_open error");
            return 1;
        }
        printf("Sirenko : sem_open. Semaphore is taken.\nWaiting for it to be dropped.\n");
        if (sem_wait(sem) < 0 )
            fprintf(stderr, "Sirenko : sem_wait error");
        if ( sem_close(sem) < 0 )
            fprintf(stderr, "Sirenko : sem_close error");
        return 0;
    }
    else {
        printf("Sirenko : Dropping semaphore...\n");
        if ( (sem = sem_open(SEMAPHORE_NAME, 0)) == SEM_FAILED ) {
            fprintf(stderr, "Sirenko : sem_open error");
            return 1;
        }
        sem_post(sem);
        printf("Sirenko : sem_post. Semaphore dropped.\n");
        return 0;
    }
}
```

Висновок : було отримано знання з особливостей обміну інформацією між процесами за допомогою іменованих каналів, керування потоками, а також синхронізацію процесів через семафори та м'ютекси. Жодне з завдань не виявилось складним