

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

Лабораторна робота № 12  
з дисципліни «Операційні  
системи»

**Тема: «Програмування міжпроцесної та багатопоточної взаємодії»**

**Виконала:**

Студентка групи АІ-202  
Гребенік Анжеліка Олександрівна

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.

### **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:

### 2.1 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

```
grebenik_anzhelika@vpsj3IeQ:~  
login as: grebenik_anzhelika  
grebenik_anzhelika@91.219.60.189's password:  
Last login: Tue May 11 02:03:30 2021 from 46.149.53.142  
[grebenik_anzhelika@vpsj3IeQ ~]$ mkfifo grebenik  
[grebenik_anzhelika@vpsj3IeQ ~]$
```

- права доступу до каналу ( можна лише читати та писати власнику).

```
[grebenik_anzhelika@vpsj3IeQ ~]$ chmod o-rwx grebenik  
[grebenik_anzhelika@vpsj3IeQ ~]$ ls -l grebenik  
prw-rw---- 1 grebenik_anzhelika grebenik_anzhelika 0 May 26 00:09 grebenik  
[grebenik_anzhelika@vpsj3IeQ ~]$ chmod g-rwx grebenik  
[grebenik_anzhelika@vpsj3IeQ ~]$ ls -l grebenik  
prw----- 1 grebenik_anzhelika grebenik_anzhelika 0 May 26 00:09 grebenik  
[grebenik_anzhelika@vpsj3IeQ ~]$
```

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.

*Використовуємо команду `ls /etc` для змісту каталога `/etc` та `find / -name "g*"` для пошуку файлів, які починаються з літери прізвища*

```
[grebenik_anzhelika@vpsj3IeQ ~]$ ls /etc | find / -name "g*" > grebenik  
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
```

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

*Використовуючи команду `cat grebenik` отримаємо такий результат*

```

grebenik_anzhelika@vpsj3IeQ:~
/etc/gssproxy/gssproxy.conf
/etc/grub.d
/etc/xdg/autostart/gsettings-data-convert.desktop
/etc/default/grub-oracle-database-preinstall-18c.orabackup
/etc/default/grub-initial.orabackup
/etc/default/grub
/etc/iproute2/group
/etc/gshadow-
/etc/gnupg
/etc/gshadow
/etc/group
/etc/glvnd
/etc/gconf
/etc/gconf/gconf.xml.mandatory
/etc/gconf/gconf.xml.system
/etc/gconf/gconf.xml.defaults
/etc/selinux/targeted/contexts/users/guest_u
/etc/groff
/etc/rwtab.d/gssproxy
/etc/ghostscript
/etc/group-
/etc/security/group.conf
/etc/prelink.conf.d/grub2.conf
[grebenik_anzhelika@vpsj3IeQ ~]$ █

```

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`

```

grebenik_anzhelika@vpsj3IeQ:~
[grebenik_anzhelika@vpsj3IeQ ~]$ gzip -c < grebenik > file1.gz
[grebenik_anzhelika@vpsj3IeQ ~]$ ls
12407 grebenik3.sh lab6.txt
accounts.csv Grebenik_43.csv my_create_file.sh
create Grebenik_4.csv MyOSParam.sh
create.c grebenik_lab2.docx MyOSParam.sh.save
dirl grebenik_lab2.html nohup.out
dir2 grebenik_lab2.pdf Operating-System.-Laboratory-Work-1
$dirname grebenik_lab_3 os.labl.cpl251.html
file1.gz Grebenik_lab5.csv os.labl.utf.html
file1.txt grebenik.sh send_signal
get_signal get_signal send_signal.c
get_signal.c info sirota
Gr info.c sirota.c
grebenik lab5.sh до загальної кількості
grebenik2.sh lab6 найманих працівників"
[grebenik_anzhelika@vpsj3IeQ ~]$ █

grebenik_anzhelika@vpsj3IeQ:~
/etc/grub.d
/etc/xdg/autostart/gsettings-data-convert.desktop
/etc/default/grub-oracle-database-preinstall-18c.orabackup
/etc/default/grub-initial.orabackup
/etc/default/grub
/etc/iproute2/group
/etc/gshadow-
/etc/gnupg
/etc/gshadow
/etc/group
/etc/glvnd
/etc/gconf
/etc/gconf/gconf.xml.mandatory
/etc/gconf/gconf.xml.system
/etc/gconf/gconf.xml.defaults
/etc/selinux/targeted/contexts/users/guest_u
/etc/groff
/etc/rwtab.d/gssproxy
/etc/ghostscript
/etc/group-
/etc/security/group.conf
/etc/prelink.conf.d/grub2.conf
[grebenik_anzhelika@vpsj3IeQ ~]$ cat /etc/passwd > grebenik
[grebenik_anzhelika@vpsj3IeQ ~]$ █

```

*Командою `gunzip -c file1.gz` перевіряємо результат архівування*

```
[grebenik_anzhelika@vpsj3leQ ~]$ gunzip -c 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
nobody:x:99:99:Nobody:/:/sbin/nologin
systemd-network:x:192:192:systemd Network Management:/:/sbin/nologin
dbus:x:81:81:System message bus:/:/sbin/nologin
polkitd:x:999:997:User for polkitd:/:/sbin/nologin
postfix:x:89:89:/:/var/spool/postfix:/sbin/nologin
chrony:x:998:996:/:/var/lib/chrony:/sbin/nologin
sshd:x:74:74:Privilege-separated SSH:/var/empty/sshd:/sbin/nologin
soft:x:1000:1000:/:/home/soft:/sbin/nologin
saslauth:x:997:76:Saslauthd user:/run/saslauthd:/sbin/nologin
mailnull:x:47:47:/:/var/spool/mqueue:/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 "/grebenik2"
#define BUFSIZE 50

int main(void)
{
    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("Incoming message (%d): %s\n", len, buf);
    }while ( 1 );
}

```

```

[grebenik_anzhelika@vpsj3IeQ ~]$ ls
12407      grebenik3.sh      lab6.txt
accounts.csv  Grebenik_43.csv  my_create_file.sh
create      Grebenik_4.csv    MyOSParam.sh
create.c     grebenik_lab2.docx  MyOSParam.sh.save
dir1         grebenik_lab2.html  nohup.out
dir2         grebenik_lab2.pdf   Operating-System.-Laboratory-Work-1
$dirname     grebenik_lab_3      os.lab1.cpl251.html
file1.gz     Grebenik_lab5.csv   os.lab1.utf.html
file1.txt    grebenik.sh         send_signal
get_signal   gret_signal         send_signal.c
get_signal.c info                sirota
Gr           info.c              sirota.c
grebenik     lab12               до загальної кількості
grebenik2    lab5.sh             найманих працівників?
grebenik2.sh lab6

[grebenik_anzhelika@vpsj3IeQ ~]$

```

## 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.

```
grebenik_anzhelika@vpsj3leQ:~  
GNU nano 2.3.1 File: lab12.3_task.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("grebenik 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("grebenik f2: %d\n", i);  
        sleep(1);  
    }  
    pthread_exit(0);  
}
```

```

[grebenik_anzhelika@vpsj3IeQ ~]$ nano lab12.3_task.c
[grebenik_anzhelika@vpsj3IeQ ~]$ gcc lab12.3_task.c -o lab12.3_task -lpthread
[grebenik_anzhelika@vpsj3IeQ ~]$ ./lab12.3_task
grebenik f2: 1
grebenik f1: 1
grebenik f2: 2
grebenik f1: 2
grebenik f2: 3
grebenik f1: 3
grebenik f1: 4
grebenik f2: 4
grebenik f2: 5
grebenik f1: 5
grebenik f2: 6
grebenik f1: 6
grebenik f2: 7
grebenik f1: 7
grebenik f2: 8
grebenik f1: 8
grebenik f2: 9
grebenik f1: 9
[grebenik_anzhelika@vpsj3IeQ ~]$

```

---

## 2.4 Програмування семафорів

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

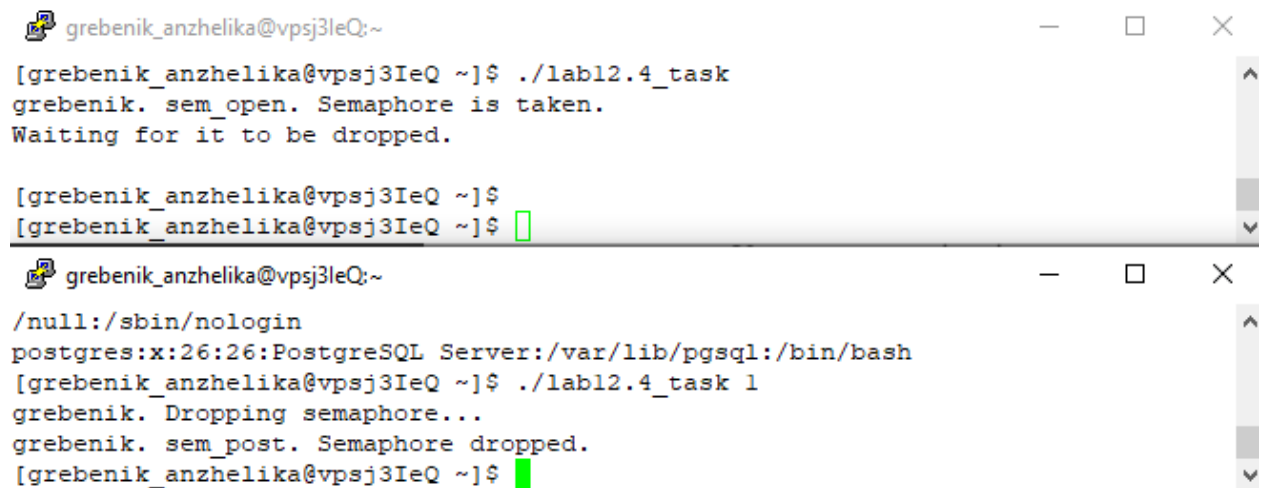
Виконайте програму в двох терміналах за вказаним прикладом.

```

#include <fcntl.h>
#include <sys/stat.h>
#include <semaphore.h>
#include <stdio.h>
#define SEMAPHORE_NAME "/grebenik"
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("grebenik. 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("Dropping semaphore...\n");
        if ( (sem = sem_open(SEMAPHORE_NAME, 0)) == SEM_FAILED ) {
            fprintf(stderr, "sem_open error");
            return 1;
        }
        sem_post(sem);
        printf("grebenik. sem_post. Semaphore dropped.\n");
        return 0;
    }
}

```





The image shows two terminal windows. The top window is titled 'grebenik\_anzhelika@vpsj3IeQ:~' and shows the execution of './lab12.4\_task', which results in 'grebenik. sem\_open. Semaphore is taken.' and 'Waiting for it to be dropped.' The bottom window is titled 'grebenik\_anzhelika@vpsj3IeQ:~' and shows the execution of './lab12.4\_task 1', which results in 'grebenik. Dropping semaphore...', 'grebenik. sem\_post. Semaphore dropped.', and a green cursor.

```
grebenik_anzhelika@vpsj3IeQ:~  
[grebenik_anzhelika@vpsj3IeQ ~]$ ./lab12.4_task  
grebenik. sem_open. Semaphore is taken.  
Waiting for it to be dropped.  
  
[grebenik_anzhelika@vpsj3IeQ ~]$  
[grebenik_anzhelika@vpsj3IeQ ~]$ █
```

---

```
grebenik_anzhelika@vpsj3IeQ:~  
/null:/sbin/nologin  
postgres:x:26:26:PostgreSQL Server:/var/lib/pgsql:/bin/bash  
[grebenik_anzhelika@vpsj3IeQ ~]$ ./lab12.4_task 1  
grebenik. Dropping semaphore...  
grebenik. sem_post. Semaphore dropped.  
[grebenik_anzhelika@vpsj3IeQ ~]$ █
```

Висновок: під час лабораторної роботи ми вивчили особливості обміну інформацією між процесами за допомогою іменованих каналів, керування потоками, а також синхронізацію процесів через семафори та м'ютекси.