МІНІСТЕРСТВО ОСВІТИ ТА НАУКИ УКРАЇНИ
ОДЕСЬКИЙ НАЦІОНАЛЬНИЙ ПОЛІТЕХНІЧНИЙ УНІВЕРСИТЕТ
Інститут комп’ютерних систем
Кафедра інформаційних систем

Варіант №3

Лабораторна робота №12

По дисципліні “Операційні системи”

Тема: «Програмування міжпроцесної та багатопоточної взаємодії»

Виконав:
Студент групи АІ-202
Боднар І.В.
Перевірили:
Блажко О.А

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.

Хід роботи:

1. **Робота з іменованими каналами**

1.1. В домашньому каталозі нашого користувача створюємо іменований канал з використанням команди mkfifo:

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

```
[bodnar_illya@vpsj3IeQ ~]$ mkfifo bodnar
[bodnar_illya@vpsj3IeQ ~]$ chmod 600 bodnar
[bodnar_illya@vpsj3IeQ ~]$ ls -l
total 628
-rw-rw-r-- 1 bodnar_illya bodnar_illya   5366 Mar 23 14:41 accounts.csv
prw------- 1 bodnar_illya bodnar_illya      0 May 25 17:32 bodnar
```

1.2. Підключаємо до іменованого каналу процес, який буде в нього писати за такими командами:

- отримуємо зміст каталогу /etc

```
[bodnar_illya@vpsj3IeQ ~]$ ls /etc > bodnar
[bodnar_illya@vpsj3IeQ ~]$
```

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

- отримуємо назви файлів, які починаються з букви нашого прізвища у транслітерації.

```
[bodnar_illya@vpsj3IeQ ~]$ find b* > bodnar
[bodnar_illya@vpsj3IeQ ~]$
```

```
[bodnar_illya@vpsj3IeQ ~]$ cat bodnar
bodnar
bodnar2.sh
bodnar3.sh
bodnar.csv
bodnar_lab3
bodnar_lab3/illia_2
bodnar_lab3/illia_1
bodnar.sh
[bodnar_illya@vpsj3IeQ ~]$
```

1.3. Переходимо до нового терміналу роботи з ОС Linux та створюємо процес, який буде читати зі створеного раніше каналу.

1.4. Повертаємося до 1-го терміналу та підключаємо до іменованого каналу процес, який буде в нього писати, архівуючи файл командою gzip -c < pipe > file1.gz .

```
[bodnar_illya@vpsj3IeQ ~]$ gzip -c < bodnar > bodnar.gz
[bodnar_illya@vpsj3IeQ ~]$
```

1.5. Переходимо до 2-го терміналу роботи з ОС Linux та створюємо процес, який буде читати зі створеного раніше каналу, архівуючи файл /etc/passwd

```
[bodnar_illya@vpsj3IeQ ~]$ cat /etc/passwd > bodnar
[bodnar_illya@vpsj3IeQ ~]$ gunzip -c  bodnar.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
```

2. Програмування іменованих каналів

2.1. Повторюємо попереднє завдання, але пункт 2.1.1 виконуємо через програмування іменованого каналу за прикладом з рисунку 1.

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

#define NAMEDPIPE_NAME "bodnar"
#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 );
}
```

```
[bodnar_illya@vpsj3IeQ ~]$ gcc 2.c -o 2
[bodnar_illya@vpsj3IeQ ~]$ ./2
bodnar is created
```

3. Програмування потоків

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

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

main() {
    pthread_t bodnar1_thread, bodnar2_thread;
    void *f2(), *f1();
    int i1 = 10, i2 = 10;
    pthread_create(&bodnar1_thread, NULL, f1, &i1);
    pthread_create(&bodnar2_thread, NULL, f2, &i2);
    pthread_join(bodnar1_thread, NULL);
    pthread_join(bodnar2_thread, NULL);
}

void *f1(int *x) {
    int i,n;
    n = *x;
    for (i=1;i<n;i++) {
    printf("bodnar1: %d\n", i);
    sleep(1);
    }
    pthread_exit(0);
}

void *f2(int *x) {
    int i,n;
    n = *x;
    for (i=1;i<n;i++) {
    printf("bodnar2: %d\n", i);
    sleep(1);
    }
    pthread_exit(0);
}
```

3.2. Виконуємо програму за вказаним прикладом.

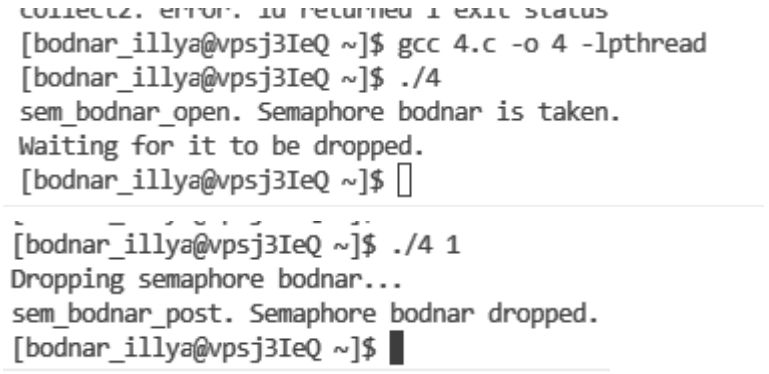
```
[bodnar_illya@vpsj3IeQ ~]$ gcc 3.c -o 3 -lpthread
[bodnar_illya@vpsj3IeQ ~]$ ./3
bodnar1: 1
bodnar2: 1
bodnar1: 2
bodnar2: 2
bodnar1: 3
bodnar2: 3
bodnar1: 4
bodnar2: 4
bodnar1: 5
bodnar2: 5
bodnar1: 6
bodnar2: 6
bodnar1: 7
bodnar2: 7
bodnar1: 8
bodnar2: 8
bodnar1: 9
bodnar2: 9
[bodnar_illya@vpsj3IeQ ~]$
```

4. Програмування семафорів

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

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

#define SEMAPHORE_NAME "/bodnar"

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

4.2. Виконуємо програму в двох терміналах за вказаним прикладом.



Висновок: У процесі виконання лабораторної роботи ми розширили свої знання про особливості обміну інформацією між процесами за допомогою іменованих каналів, керування потоками, а також синхронізацію процесів через семафори та м`ютекси.