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

Лабораторна робота № 12

З дисципліни «Операційні системи»

Тема: «Програмування міжпроцесної та багатопоточної взаємодії»

Варіант 3

Виконала:
Студентка групи АІ-202
Неживих М.О.
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

Завдання 1 Робота з іменованими каналами

В домашньому каталозі 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).

```
[nezhivih_mariya@vpsj3IeQ ~]$ mkfifo nezhivih
[nezhivih_mariya@vpsj3IeQ ~]$ chmod 600 nezhivih
[nezhivih_mariya@vpsj3IeQ ~]$ ls -l
total 672
-rw-rw-r-- 1 nezhivih_mariya nezhivih_mariya    670 Mar 29 22:57 MyOSParam.sh
-rw------- 1 nezhivih_mariya nezhivih_mariya      1 Apr 13 18:54 nano.save
prw------- 1 nezhivih_mariya nezhivih_mariya      0 May 25 20:46 nezhivih
```

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

- отримати зміст каталогу /etc

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

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

```
[nezhivih_mariya@vpsj3IeQ ~]$ ls /etc > nezhivih
```

```
[nezhivih_mariya@vpsj3IeQ ~]$ cat nezhivih
adjtime
aliases
aliases.db
alternatives
anacrontab
asound.conf
audisp
audit
bash_completion.d
bashrc
binfmt.d
centos-release
centos-release-upstream
chkconfig.d
chrony.conf
chrony.keys
cron.d
cron.daily
cron.deny
cron.hourly
```

```
[nezhivih_mariya@vpsj3IeQ ~]$ find n* > nezhivih
```

```
[nezhivih_mariya@vpsj3IeQ ~]$ cat nezhivih
nano.save
nezhivih
nezhivih2.sh
nezhivih3.sh
nezhivih_lab_3
nezhivih_lab_3/maria_2
nezhivih_lab_3/maria_1
nezhivih.sh
nohup.out
```

Повертаємось до 1-го терміналу та підключаємо до іменованого каналу процес, який буде в нього писати, архівуючи файл командою gzip -c < pipe > file1.gz
де pipe – назва нашого каналу, file1.gz – назва файлу, який буде створено в результаті архівації

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

```
[nezhivih_mariya@vpsj3IeQ ~]$ gzip -c < nezhivih > file.gz
```

```
[nezhivih_mariya@vpsj3IeQ ~]$ cat /etc/passwd > nezhivih
```

<u>Завдання 2</u> Програмування іменованих каналів

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

```
2.c                 [----]  8 L:[  1+ 0   1/ 35] *(8    / 760
#include <sys/stat.h>
#include <fcntl.h>
#include <string.h>
#include <stdio.h>

#define NAMEDPIPE_NAME "maria"
#define BUFSIZE 50

int main (int argc, char ** argv) {
    int fd, len;
    char buf[BUFSIZE];
....
    if ( mkfifo(NAMEDPIPE_NAME, 0777) ) {
    fprintf(stderr,"Error in ,kfifo!");
    return 1;
    }
    printf("%s is created\n", NAMEDPIPE_NAME);
....
    if ( (fd = open(NAMEDPIPE_NAME, O_RDONLY)) <= 0 ) {
    fprintf(stderr,"Error in open!",0);
    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 );
    }
```

```
[nezhivih_mariya@vpsj3IeQ ~]$ gcc 2.c -o 2
[nezhivih_mariya@vpsj3IeQ ~]$ .\2
-bash: .2: command not found
[nezhivih_mariya@vpsj3IeQ ~]$ ./2
maria is created
```

<u>Завдання 3</u> Програмування потоків

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

Виконуємо програму за вказаним прикладом.

```
3.c                    [----]  0 L:[  1+13  14/
#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=0;i>n;i++) {
printf("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("f2: %d\n", i);
sleep(1);
}
pthread_exit(0);
}
```

```
[nezhivih_mariya@vpsj3IeQ ~]$ gcc 3.c -o 3 -lpthread
[nezhivih_mariya@vpsj3IeQ ~]$ ./3
f2: 1
f2: 2
f2: 3
f2: 4
f2: 5
f2: 6
f2: 7
f2: 8
f2: 9
```

Завдання 4 Програмування семафорів

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

Виконуємо програму в двох терміналах за вказаним прикладом.

```
4.c                 [----]  2 L:[  1+32  33/ 33] *(786 / 786b) <EOF>
#include <fcntl.h>
#include <sys/stat.h>
#include <semaphore.h>
#include <stdio.h>

#define SEMAPHORE_NAME "/nezhivih"

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_nez_open error");
 return 1;
 }
 printf("sem_nez_open. SEmaphore_nez is taken.\nWaiting for it to be dropped.\n");
 if (sem_wait(sem) < 0 )
 fprintf(stderr,"sem_nez_wait error");
 if  (sem_close(sem) < 0)
 fprintf(stderr,"sem_nez_close error");
 return 0;
 }
 else {
 printf("Dropping semaphore_nez...\n");
 if ( (sem = sem_open(SEMAPHORE_NAME, 0)) == SEM_FAILED ) {
 fprintf(stderr,"sem_nez_open error");
 return 1;
 }
 sem_post(sem);
 printf("sem_nez_post. Semaphore_nez dropped.\n");
 return 0;
 }
 }
```

```
[nezhivih_mariya@vpsj3IeQ ~]$ gcc 4.c -o 4 -lpthread
[nezhivih_mariya@vpsj3IeQ ~]$ ./4
sem_nez_open. SEmaphore_nez is taken.
Waiting for it to be dropped.
```

```
[nezhivih_mariya@vpsj3IeQ ~]$ ./4 1
Dropping semaphore_nez...
sem_nez_post. Semaphore_nez dropped.
```

Висновок: У ході лабораторної роботи було вивчено особливості обміну інформацією між процесами за допомогою іменованих каналів, керування потоками, а також синхронізацію процесів через семафори та м`ютекси.