ДЕРЖАВНИЙ УНІВЕРСИТЕТ «ОДЕСЬКА ПОЛІТЕХНИКА»
ІНСТИТУТ КОМП'ЮТЕРНИХ СИСТЕМ
КАФЕДРА «ІНФОРМАЦІЙНИ СИСТЕМИ»

Лабораторна робота №12

з дисципліни «Операційні системи»

Тема

«Програмуванння міжпроцесної та багатопоточної взаємодії»

Виконав:

Студент групи АІ-202

Узун Михайло

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 помідомленнях буде вказано ваще прізвище латиницею.

Виконайте програму за вказаним прикладом.

2.4 Програмування семафорів

За прикладом з рисунку 3 розробіть програму керування семафором, в якій в

помідомленнях буде вказано ваще прізвище латиницею.

Виконайте програму в двох терміналах за вказаним прикладом.

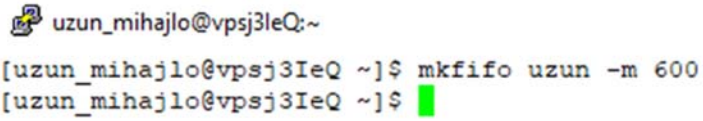
Хід роботи:

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

- 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.

```
[uzun_mihajlo@vpsj3IeQ ~]$ ls /etc > uzun
[uzun_mihajlo@vpsj3IeQ ~]$ find / -name "u*" > uzun
```

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

```
[uzun mihajlo@vpsj3IeQ ~]$ cat uzun
```

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

```
[uzun mihajlo@vpsj3IeQ ~]$ gzip -c < uzun > uzun.gz
```

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

```
[uzun mihajlo@vpsj3IeQ ~]$ cat /etc/passwd > uzun
```

2.2 Програмування іменованих каналів

Повторіть попереднє завдання, але пункт 2.1.1 виконайте 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 "uzun"
#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 );
}
```

1-й терминал

```
[uzun_mihajlo@vpsj3IeQ ~]$ ./fifo
uzun is created
uzun is opened
Incomming message (49): adjtime
aliases
aliases.db
alternatives
anacronta
Incomming message (49): b
asound.conf
audisp
```

2-й терминал

```
[uzun mihajlo@vpsj3IeQ ~]$ ls /etc > uzun
```

1-й терминал

```
[uzun_mihajlo@vpsj3IeQ ~]$ ./fifo
uzun is created
uzun is opened
Incomming message (49): /etc/machine-id
/etc/magic
/etc/mail.rc
/etc/make
Incomming message (49): dumpfile.conf.sample
/etc/man_db.conf
/etc/mke2fs
Incomming message (49): .conf
/etc/motd
```

2-терминал

```
[uzun_mihajlo@vpsj3IeQ ~]$ ls /etc/m* > uzun
```

1-й терминал

```
[uzun_mihajlo@vpsj3IeQ ~]$ gzip -c < ./fifo > file uzun.gz
```

2-й терминал

```
[uzun mihajlo@vpsj3IeQ ~]$ cat /etc/passwd > uzun
```

2.3 Програмування потоків

За прикладом з рисунку 2 розробіть програму керування потоками, в якій в помідомленнях буде вказано ваще прізвище латиницею. Виконайте програму за вказаним прикладом.

```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("f1: Uzun\n");
        sleep(1);
    }
    pthread_exit(0);
}

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

```
[uzun_mihajlo@vpsj3IeQ ~]$ gcc thread.c -o thread -lpthread
[uzun_mihajlo@vpsj3IeQ ~]$ ./thread
f1: Uzun
f2: Uzun
f1: Uzun
f2: Uzun
f1: Uzun
f2: Uzun
f1: Uzun
f2: Uzun
f1: Uzun
f2: Uzun
f1: Uzun
f2: Uzun
f1: Uzun
f2: Uzun
f1: Uzun
f2: Uzun
f1: Uzun
f2: Uzun
f1: Uzun
f2: Uzun
f1: Uzun
f2: Uzun
[uzun_mihajlo@vpsj3IeQ ~]$
```

2.4 Програмування семафорів

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

#define SEMAPHORE_NAME "/uzun_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,"sem_open error");
           return 1;
       }
       printf("sem_open. Semaphore Uzun  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 Uzun...\n");
       if ( (sem = sem_open(SEMAPHORE_NAME, 0)) == SEM_FAILED ) {
            fprintf(stderr,"sem_open error");
            return 1;
       }
       sem_post(sem);
       printf("sem_post. Semaphore Uzun dropped.\n");
       return 0;

    }
}
```

1-й терминал

```
[uzun_mihajlo@vpsj3IeQ ~]$ gcc semaphore.c -o semaphore -lpthread
[uzun_mihajlo@vpsj3IeQ ~]$ ./semaphore
sem_open. Semaphore Uzun  is taken.
Waiting for it to be dropped.
```

2-й терминал

```
[uzun_mihajlo@vpsj3IeQ ~]$ ./semaphore 1
Dropping semaphore Uzun...
sem_post. Semaphore Uzun dropped.
```

Висновок: були вивчині особливості обміну інформацією між процесами за допомогою іменованих каналів, керування потоками, а також синхронізацію процесів через семафори та м`ютекси.