ОДЕСЬКИЙ НАЦІОНАЛЬНИЙ ПОЛІТЕХНІЧНИЙ УНІЕРСИТЕТ

Інститут комп’ютерних систем

Кафедра інформаційних систем

Лабораторна робота №12

З дисципліни «Операційні системи»

Тема: «Програмування міжпроцесної та багатопоточної взаємодії»

Виконав:

Студент групи АІ-202

Полянський М.О.

Перевірили:

Блажко О. А.

Дрозд М.О.

Одесса 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

1. Робота з іменованими каналами

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).

```
[polyanskij_mikola@vpsj3IeQ ~]$ mkfifo polyanskij -m 600
[polyanskij_mikola@vpsj3IeQ ~]$ ls -l polyanskij
prw------- 1 polyanskij_mikola polyanskij_mikola 0 May 31 17:54 polyanskij
[polyanskij_mikola@vpsj3IeQ ~]$
```

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.

```
[polyanskij_mikola@vpsj3IeQ ~]$ ls /etc > polyanskij
[polyanskij_mikola@vpsj3IeQ ~]$ find / -name "p*" > polyanskij
```

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

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

```
[polyanskij_mikola@vpsj3IeQ ~]$ gzip -c < polyanskij > polyanskij1.gz
[polyanskij_mikola@vpsj3IeQ ~]$ cat polyanskij1.gz
```

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

```
[polyanskij_mikola@vpsj3IeQ ~]$ cat /etc/passwd > polyanskij
```

2.1 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 "/home/polyanskij_mikola/polyanskij4"
#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 );
}
```

```
[polyanskij_mikola@vpsj3IeQ ~]$ gcc fifi.c -o fifi
[polyanskij_mikola@vpsj3IeQ ~]$ ./fifi
/home/polyanskij_mikola/polyanskij4 is created
/home/polyanskij_mikola/polyanskij4 is opened
Incomming message (49): l234pol
1.sh
23.sh
accounts.csv
fifi
fifi.c
file1
Incomming message (49): .txt
file.txt
hard_link_1
hard_link_2
Laba2.doc
```

2.2 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.

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

void main() {
    pthread_t f2_thread, f1_thread;
    void* f2(), * f1();
    int x1 = 10, x2 = 10;
    pthread_create(&f1_thread, NULL, &f1, &x1);
    pthread_create(&f2_thread, NULL, &f2, &x2);
    pthread_join(f1_thread, NULL);
    pthread_join(f2_thread, NULL);
}

void* f1(int* x) {
    int i;
    for (i = 1; i < *x; i++) {
        printf("Thread 1 - polyanskij\n");
        sleep(1);
    }
    pthread_exit(0);
}

void* f2(int* x) {
    int i;
    for (i = 1; i < *x; i++) {
        printf("Thread 2 - polyanskij\n");
        sleep(1);
    }
    pthread_exit(0);
}
```

```
[polyanskij_mikola@vpsj3IeQ ~]$ gcc thread.c -o thread -lpthread
[polyanskij_mikola@vpsj3IeQ ~]$ ./thread
Thread 1 - polyanskij
Thread 2 - polyanskij
Thread 1 - polyanskij
Thread 2 - polyanskij
Thread 1 - polyanskij
Thread 2 - polyanskij
Thread 1 - polyanskij
Thread 2 - polyanskij
Thread 1 - polyanskij
Thread 2 - polyanskij
Thread 1 - polyanskij
Thread 2 - polyanskij
Thread 2 - polyanskij
Thread 1 - polyanskij
Thread 2 - polyanskij
Thread 1 - polyanskij
Thread 2 - polyanskij
Thread 1 - polyanskij
```

2.3 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

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

#define SEMAPHORE_NAME "polyanskij_semophore"

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. Polyanskij has taken the semaphore.\nWaiting for it t$
        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("sem_post. Polyanskij dropped Semaphore.\n");
        return 0;

    }
```

```
[polyanskij_mikola@vpsj3IeQ ~]$ nano thread.c
[polyanskij_mikola@vpsj3IeQ ~]$ nano semaphore.c
[polyanskij_mikola@vpsj3IeQ ~]$ gcc semaphore.c -o semaphore -lpthread
[polyanskij_mikola@vpsj3IeQ ~]$ ./semaphore
sem_open. Polyanskij has taken the semaphore.
Waiting for it to be dropped.
[polyanskij_mikola@vpsj3IeQ ~]$
```

```
[polyanskij_mikola@vpsj3IeQ ~]$ ./semaphore 1
Dropping semaphore...
sem_post. Polyanskij dropped Semaphore.
[polyanskij_mikola@vpsj3IeQ ~]$
```

Висновки: ми вивчили особливості обміну інформацією між процесами за допомогою іменованих каналів, керування потоками, а також синхронізацію процесів через семафори та м`ютекси. Всі завдання були досить легкими/