МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ ДЕРЖАВНИЙ УНІВЕРСИТЕТ «ОДЕСЬКА ПОЛІТЕХНІКА»

Інститут комп'ютерних систем Кафедра «Інформаційних систем»

Лабораторна робота №12

3 дисципліни: «Операційні системи»

Тема: «Програмування міжпроцесної та багатопоточної взаємодії»

Виконав:

Студент групи АІ-205

Кучеренко М.М.

Перевірили:

Блажко О.А.

Дрозд М.О.

Мета роботи: вивчити особливості обміну інформацією між процесами за допомогою іменованих каналів, керування потоками, а також синхронізацію процесів через семафори та м'ютекси.

Завдання до виконання

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.

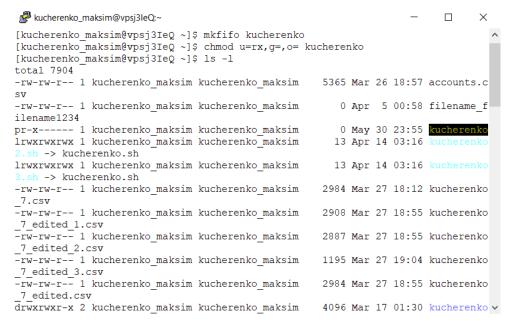
2.4 Програмування семафорів

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.

Скріншоти виконання завдань:

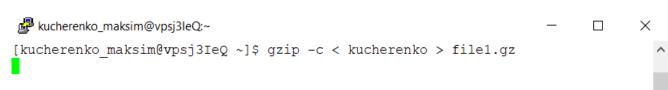
Завдання 2.1.1:



Завдання 2.1.2:



Завдання 2.1.4:



Завдання 2.1.5:

Завдання 2.2:

```
kucherenko_maksim@vpsj3leQ:~
                                                                          X
GNU nano 2.3.1
#include <sys/stat.h>
#include <fcntl.h>
#include <string.h>
#include <stdio.h>
#define NAMEDPIPE NAME "kucherenko 2"
#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.\n");
                return 1;
        printf("%s is created.\n", NAMEDPIPE_NAME);
        if ((fd = open(NAMEDPIPE NAME, O RDONLY)) <= 0) {</pre>
                fprintf(stderr, "Error in open!\n");
                return 1;
        }
        printf("%s is opened\n", NAMEDPIPE_NAME);
        do {
                memset(buf, '\0', BUFSIZE);
                if ((len = read(fd, buf, BUFSIZE - 1)) <= 0) {
                        printf("END!\n");
                        close(fd);
                        remove(NAMEDPIPE NAME);
                        return 0;
                }
                printf("Incoming message (%d): %s\n", len, buf);
        while(1);
```

```
kucherenko_maksim@vpsj3leQ:~
[kucherenko_maksim@vpsj3leQ ~]$ nano chanel.c
[kucherenko_maksim@vpsj3leQ ~]$ gcc -o chanel chanel.c
[kucherenko_maksim@vpsj3leQ ~]$ gcc -o chanel chanel.c
[kucherenko_maksim@vpsj3leQ ~]$ ./chanel
kucherenko_2 is created.
kucherenko_2 is opened
Incoming message (34): bash_completion.d
bashrc
binfmt.d

END!
[kucherenko_maksim@vpsj3leQ ~]$ ]
kucherenko_maksim@vpsj3leQ ~]$ |
kucherenko_mak
```

Завдання 2.3:

```
kucherenko_maksim@vpsj3leQ:~
                                                                                        X
GNU nano 2.3.1
                               File: lab12 task2-3.c
#include <stdio.h>
#include <pthread.h>
int 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);
         return \overline{0};
}
void *f1(int *x)
ſ
         int i, n;
         n = *x;
         for (i = 1; i < n; ++i) \{
                  printf("Kucherenko (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("Kucherenko (f2): %d\n", i);
                  sleep(1);
         }
         pthread_exit(0);
}
```

```
kucherenko_maksim@vpsj3leQ:~
                                                                          \times
[kucherenko maksim@vpsj3IeQ ~]$ nano lab12 task2-3.c
[kucherenko maksim@vpsj3IeQ ~]$ gcc -o lab12 task2-3 lab12 task2-3.c -lpthread
[kucherenko_maksim@vpsj3IeQ ~]$ ./lab12_task2-3
Kucherenko (f2): 1
Kucherenko (f1): 1
Kucherenko (f2): 2
Kucherenko (f1): 2
Kucherenko (f2): 3
Kucherenko (f1): 3
Kucherenko (f2): 4
Kucherenko (f1): 4
Kucherenko (f2): 5
Kucherenko (f1): 5
Kucherenko (f2): 6
Kucherenko (f1): 6
Kucherenko (f2): 7
Kucherenko (f1): 7
Kucherenko (f2): 8
Kucherenko (f1): 8
Kucherenko (f2): 9
Kucherenko (f1): 9
[kucherenko maksim@vpsj3IeQ ~]$
```

Завдання 2.4:

```
kucherenko_maksim@vpsj3leQ:~
                                                                                                         GNU nano 2.3.1
                                     File: laba12 task2-4.c
#include <fcntl.h>
#include <sys/stat.h>
#include <semaphore.h>
#include <stdio.h>
#define SEMAPHORE NAME "/kucherenkos 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, "Kucherenkos sem_open error");
                        return 1;
                printf("sem open. Kucherenkos semaphore is taken.\nWaiting for it to be dropped.\n");
                if (sem_wait(sem) < 0)
                        fprintf(stderr, "Kucherenkos sem wait error.\n");
                if (sem_close(sem) < 0)
                        fprintf(stderr, "Kucherenkos sem close error.\n");
        else {
                printf("Dropping Kucherenkos semaphore...\n");
                if ((sem = sem_open(SEMAPHORE_NAME, 0)) == SEM_FAILED) {
                        fprintf(stderr, "Kucherenkos sem_open error.\n");
                        return 1;
                sem post(sem);
                printf("sem_post. Kucherenkos semaphore dropped.\n");
                return 0;
       }
```

```
kucherenko_maksim@vpsj3leQ:~

[kucherenko_maksim@vpsj3leQ ~]$ nano laba12_task2-4.c

[kucherenko_maksim@vpsj3leQ ~]$ gcc -o laba12_task2-4 laba12_task2-4.c -lpthread

[kucherenko_maksim@vpsj3leQ ~]$ ./laba12_task2-4

sem_open. Kucherenkos semaphore is taken.

Waiting for it to be dropped.

[kucherenko_maksim@vpsj3leQ ~]$ [

kucherenko_maksim@vpsj3leQ ~]$ [

kucherenko_maksim@vpsj3leQ ~]$ [

kucherenko_maksim@vpsj3leQ ~]$ [
```

Висновок: В ході лабораторної роботи були вивчені особливості обміну інформацією між процесами за допомогою іменованих каналів, керування потоками, а також синхронізацію процесів через семафори та м'ютекси.