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

Лабораторна робота № 8

За дисциплиною:”Операційні системи”

Тема: «Програмування керуванням процесами в ОС Unix»

Виконав:
Студент групи АІ-205
Шостак Р.С.
Перевірили:
Блажко О.А.
Дрозд М.О.

Одеса 2021

Мета роботи: отримання навичок в управлінні процесами в ОС Unix на рівні мови програмування С.

2. Завдання

Завдання 1 Перегляд інформації про процес

Створіть С-програму, яка виводить на екран таку інформацію:

− ідентифікатор групи процесів лідера сесії

− ідентифікатор групи процесів, до якої належить процес

− ідентифікатор процесу, що викликав цю функцію

− ідентифікатор батьківського процесу

− ідентифікатор користувача процесу, що викликав цю функцію

− ідентифікатор групи процесу, що викликав цю функцію.

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

int main(void) {
pid_t pid = getpid();
printf("My pid=%d\n",getpid());
printf("My ppid=%d\n",getppid());
printf("My uid=%d\n",getuid());
printf("My gid=%d\n",getgid());
printf("My pgrp=%d\n",getpgrp());
printf("My sid=%d\n",getsid(pid));
return 0;
}
```

```
[shostak_roman@vpsj3IeQ ~]$ gcc -o prim2 prim.c
[shostak_roman@vpsj3IeQ ~]$ ./prim2
My pid=16274
My ppid=15833
My uid=54421
My gid=54427
My pgrp=16274
My sid=15833
[shostak_roman@vpsj3IeQ ~]$
```

Завдання 2 Стандартне породження процесу

Створіть С-програму, яка породжує процес-нащадок. У програмі процес-батько повинен видати повідомлення типу «Parent of Ivanov», а процес-нащадок повинен видати повідомлення типу «Child of Ivanov», де замість слова Ivanov в повідомленні повинно бути ваше прізвище в транслітерації.

```c
#include <stdio.h>
#include <unistd.h>
#include <sys/types.h>

extern char** environ;

int main(void){
    char* echo_args[] = {"echo","I am Echo\n",NULL};
    pid_t pid =fork ();
    if(pid == 0)
    printf("Child of Shostak: pid=%d\n",getpid());
    else{
    printf("Parent of Shostak: pid=%d\n",getpid());
    execve("/bin/echo",echo_args,environ);
    }
    return 0;
}
```

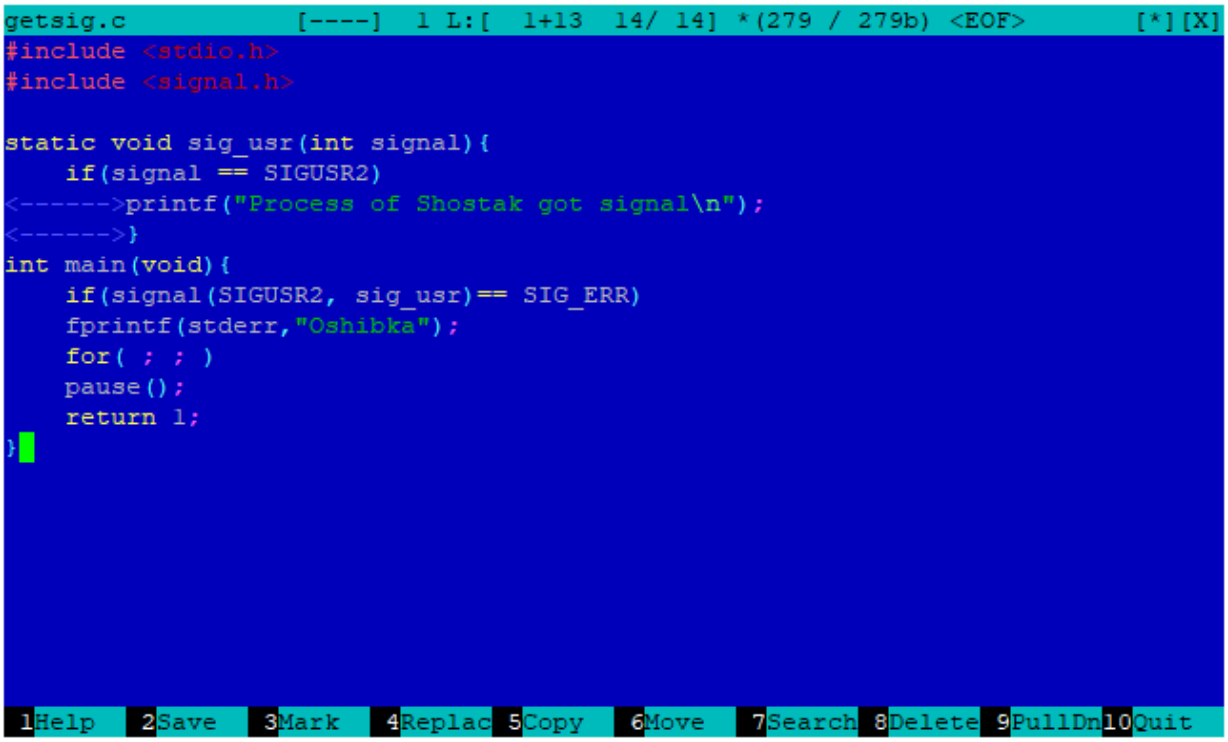
```
 1Help  2Save  3Mark  4Replac 5Copy  6Move  7Search 8Delete 9PullDn10Quit
[shostak_roman@vpsj3IeQ ~]$ gcc -o prim22 prim2.c
[shostak_roman@vpsj3IeQ ~]$ ./prim22
Parent of Shostak: pid=18390
Child of Shostak: pid=18391
I am Echo

[shostak_roman@vpsj3IeQ ~]$
```

Завдання 3 Обмін сигналами між процесами

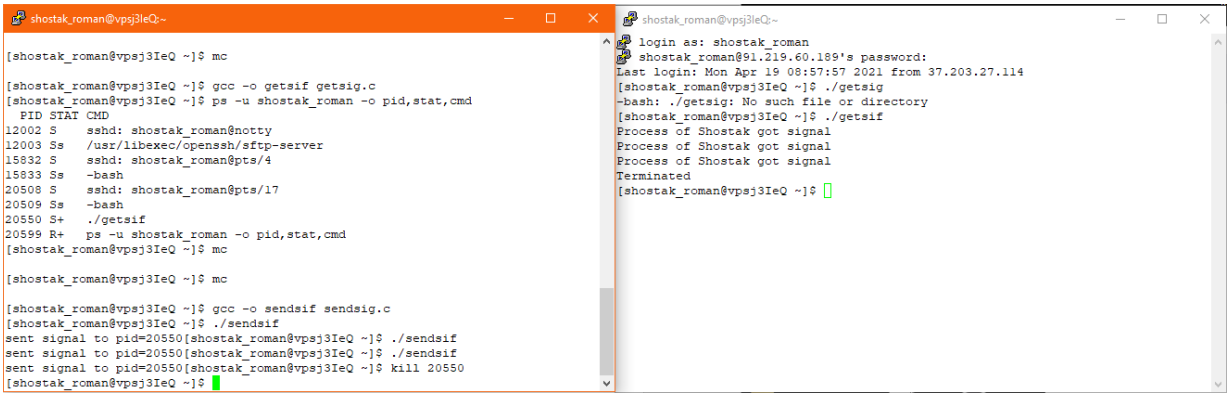
3.1 Створіть С-програму, в якій процес очікує отримання сигналу SIGUSR2 та виводить повідомлення типу «Process of Ivanov got signal» після отримання сигналу, де замість слова Ivanov в повідомленні повинно бути ваше прізвище транслітерації.

Запустіть створену С-програму.



3.2 Створіть С-програму, яка надсилає сигнал SIGUSR1 процесу, запущеному в попередньому пункту завдання.Запустіть створену С-програму та проаналізуйте повідомлення, які виводить перша програма.

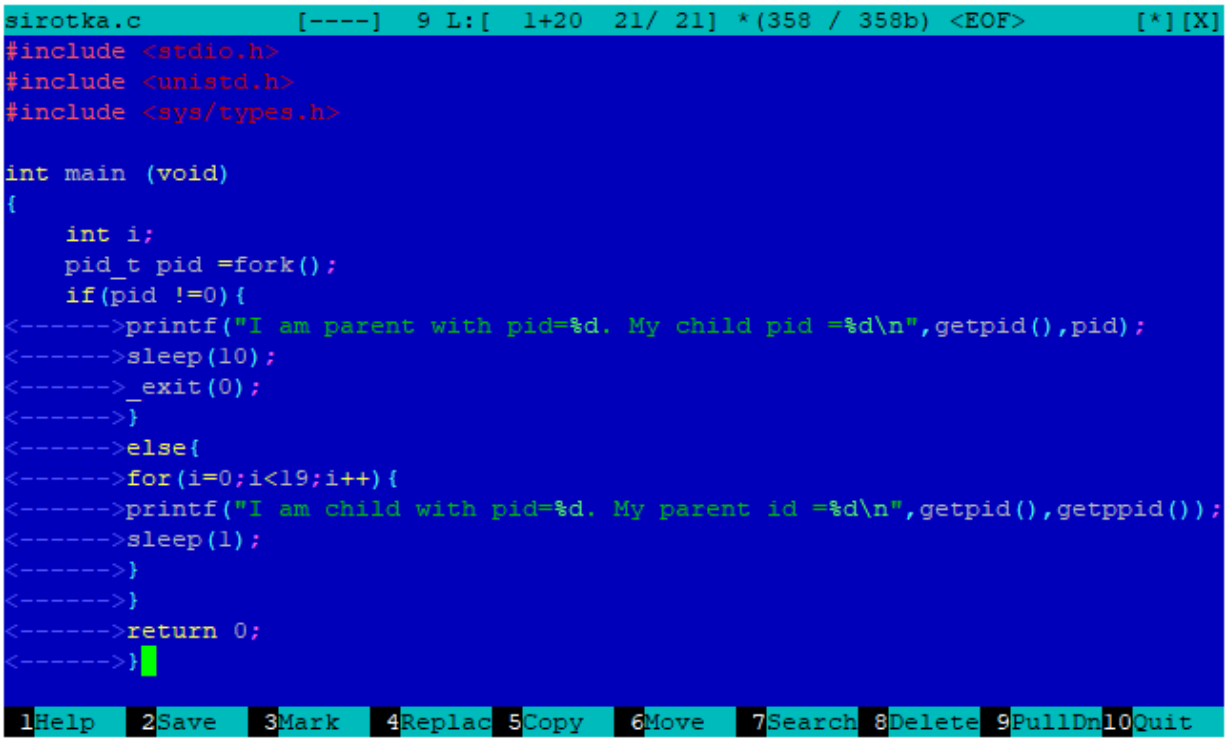
Завершіть процес, запущеному в попередньому пункту завдання.



Завдання 4 Створення процесу-сироти

Створіть С-програму, в якій процес-батько несподівано завершується раніше процесу-нащадку. Процес-батько повинен очікувати завершення n+1 секунд. Процес-нащадок повинен в циклі (2*n+1) раз із затримкою в 1 секунду виводити повідомлення,наприклад,«Parent of Ivanov», за шаблоном як в попередньому завданні, і додатково виводити PPID процесу-батька. Значення n – номер команди студента + номер студента в команді. Перевірте роботу програми, вивчіть вміст таблиці процесів і зробіть відповідні висновки.

```
[shostak_roman@vpsj3IeQ ~]$ gcc -o sirot sirotka.c
[shostak_roman@vpsj3IeQ ~]$ ./sirot
I am parent with pid=22361. My child pid =22362
I am child with pid=22362. My parent id =22361
I am child with pid=22362. My parent id =22361
I am child with pid=22362. My parent id =22361
I am child with pid=22362. My parent id =22361
I am child with pid=22362. My parent id =22361
I am child with pid=22362. My parent id =22361
```


sirotka.c [----] 9 L:[1+20 21/ 21] *(358 / 358b) <EOF> [*][X]
#include <stdio.h>
#include <unistd.h>
#include <sys/types.h>
int main (void)
{
int i;
pid_t pid =fork();
if(pid !=0){
printf("I am parent with pid=%d. My child pid =%d\n",getpid(),pid);
sleep(10);
_exit(0);
}
else{
for(i=0;i<19;i++){
printf("I am child with pid=%d. My parent id =%d\n",getpid(),getppid());
sleep(1);
}
}
return 0;
}
1Help 2Save 3Mark 4Replac 5Copy 6Move 7Search 8Delete 9PullDn 10Quit

//