МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

ОДЕСЬКИЙ НАЦІОНАЛЬНИЙ ПОЛІТЕХНІЧНИЙ УНІВЕРСИТЕТ

ІНСТИТУТ КОМП'ЮТЕРНИХ СИСТЕМ

КАФЕДРА ІНФОРМАЦІЙНИХ СИСТЕМ

Лабораторна робота №8

з дисципліни

“ Операційні системи**”**

Тема

Тема: «Програмування керуванням процесами в ОС Unix»

Виконав:

Султанов А.А

Перевірили:

Блажко О.А

Одеса 2021

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, який викликав цю функцію; − ідентифікатор групи користувача процесу, який викликав цю функцію.

Завдання 2. Стандартне створення процесу Створіть С-програму, яка створює процес-нащадок, породжуючи процес та замінюючи образ процесу. У програмі процес-батько повинен видати повідомлення типу «Parent of Ivanov», а процес-нащадок повинен видати повідомлення типу «Child of Ivanov» через виклик команди echo, де замість слова Ivanov в повідомленні повинно бути ваше прізвище в транслітерації.

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 в транслітерації. Запустіть створену С-програму. 3.2 Створіть С-програму, яка надсилає сигнал SIGUSR2 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. Процеснащадок 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

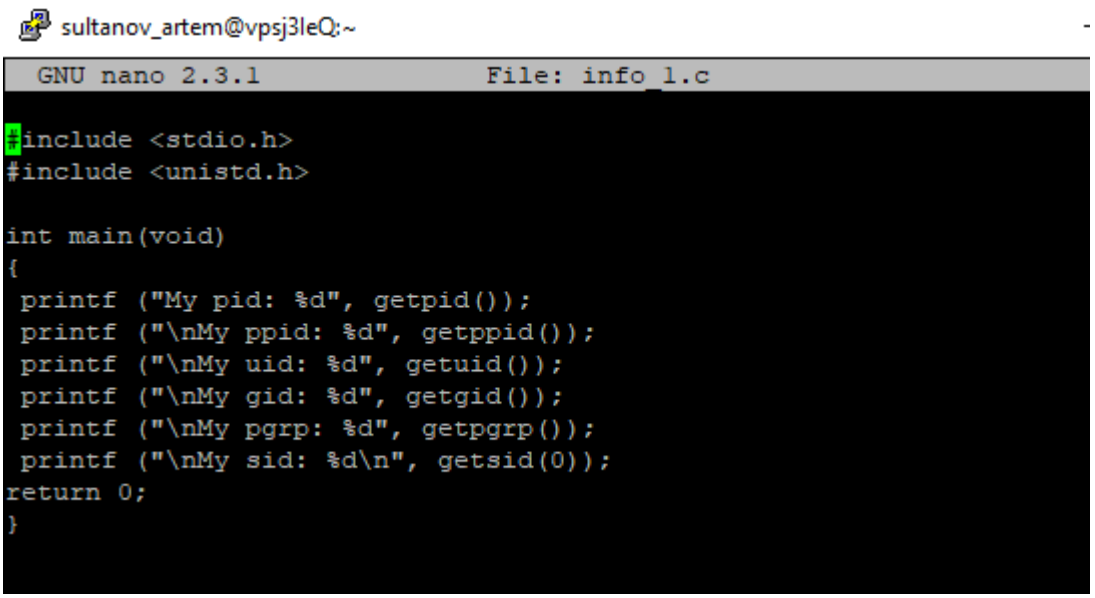
студента в команді. Перевірте роботу програми, вивчіть вміст таблиці процесів і зробіть відповідні висновки.

Завдання 5. Створення процесу-зомбі Створіть С-програму, в якій процес-нащадок несподівано завершується раніше процесу-батька, перетворюється на зомбі, виводячи в результаті повідомлення, наприклад, «I am Zombie-process of Ivanov», за шаблоном як в попередньому завданні. Запустіть програму у фоновому режимі, а в окремому терміналі вивчіть вміст таблиці процесів і зробіть відповідні висновки.

Завдання 6. Попередження створення процесу-зомбі Створіть С-програму, в якій процес-нащадок завершується раніше процесу-батька, але ця подія контролюється процесом-батьком. Процес-нащадок повинен виводити повідомлення, наприклад, «Child of Ivanov is finished», за шаблоном як в попередньому завданні. Процес-батько повинен очікувати (3*n) секунд. Значення n - n – номер команди студента + номер студента в команді. Запустіть програму у фоновому режимі, а в окремому терміналі вивчіть вміст таблиці процесів і зробіть відповідні висновки.

Виконання завдання

Завдання 1.
Перегляд інформації про процес
Я створив 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, який викликав цю функцію;
− ідентифікатор групи користувача процесу, який викликав цю функцію.




```
[sultanov_artem@vpsj3IeQ ~]$ nano info_1.c
[sultanov_artem@vpsj3IeQ ~]$ gcc info_1.c -o info
[sultanov_artem@vpsj3IeQ ~]$ ./info
My pid: 15195
My ppid: 13747
My uid: 54376
My gid: 54382
My pgrp: 15195
My sid: 13747
```

Завдання 2. Стандартне створення процесу я створив С-програму, яка створює процес-нащадок, породжуючи процес та замінюючи образ процесу. У програмі процес-батько повинен видати повідомлення типу «Parent of Ivanov», а процес-нащадок повинен видати повідомлення типу «Child of Ivanov» через виклик команди echo, де замість слова Ivanov в повідомленні повинно бути ваше прізвище в транслітерації.

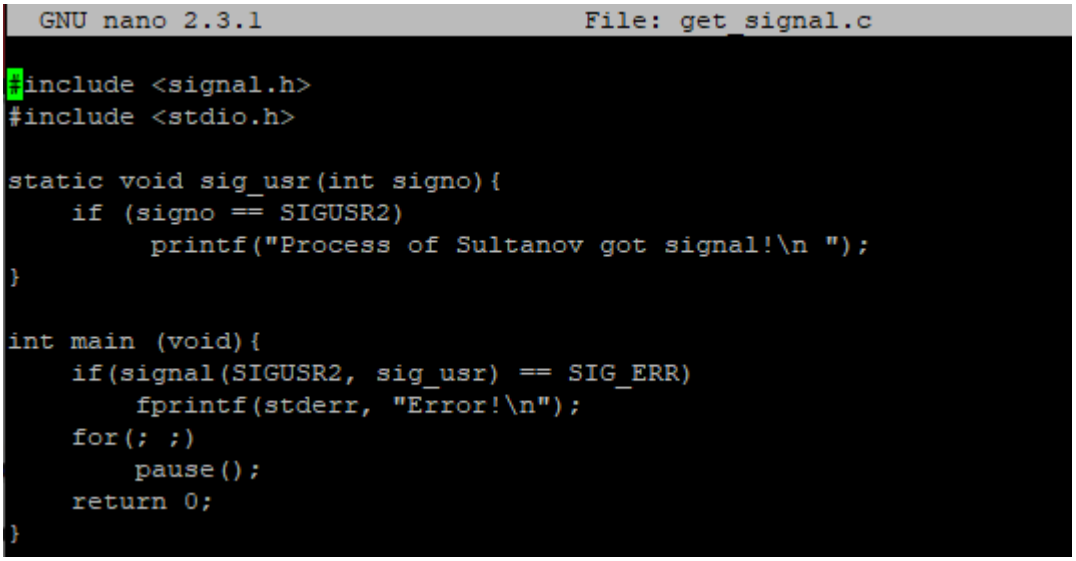
sultanov_artem@vpsj3leQ:~

```
 GNU nano 2.3.1                                File: create_1.c

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

extern char** environ;

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

```
[sultanov_artem@vpsj3IeQ ~]$ gcc create_1.c -o create
[sultanov_artem@vpsj3IeQ ~]$ ./create
Parent pid: 32249
Child pid: 32250
Child of Sultanov!

[sultanov_artem@vpsj3IeQ ~]$
```

Завдання 3. Обмін сигналами між процесами

3.1 Створив 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 в транслітерації. Запустив створену С-програму.



```
 GNU nano 2.3.1                File: get_signal.c

#include <signal.h>
#include <stdio.h>

static void sig_usr(int signo){
    if (signo == SIGUSR2)
         printf("Process of Sultanov got signal!\n ");
}

int main (void){
    if(signal(SIGUSR2, sig_usr) == SIG_ERR)
        fprintf(stderr, "Error!\n");
    for(; ;)
        pause();
    return 0;
}
```

3.2 Створив С-програму, яка надсилає сигнал SIGUSR2 процесу, запущеному в попередньому пункту завдання. Запустив створену С-програму та проаналізуйте повідомлення, які виводить перша програма. Завершив процес, запущеному в попередньому пункту завдання.

Завдання 4. Створення процесу-сироти

Я створив С-програму, в якій процес-батько несподівано завершується раніше процесу-нащадку. Процес-батько повинен очікувати завершення n+1 секунд. Процеснащадок 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. Перевірив роботу програми, вивчив вміст таблиці процесів і зробив відповідні висновки.

```
 GNU nano 2.3.1                File: sirota_artem.c

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

int main (void){
    int i;
    pid_t pid = fork();
    if(pid == 0){
        printf("Sultanov child pid: %d\n", getpid());
        for(i=0; i<18;i++){
        printf("Child of Sultanov pid=%d\n, My parent=%d\n",getpid(),getppid());
         sleep(9);
      }
      }

      else {
         printf("Parent of Sultanov=%d\n", getpid());
         sleep(1);
         _exit(0);
      }
    return 0;
}
```

```
[sultanov_artem@vpsj3IeQ ~]$ gcc sirota_artem.c -o sirota_artem
[sultanov_artem@vpsj3IeQ ~]$ ./sirota_artem
Parent of Sultanov=7952
Sultanov child pid: 7953
Child of Sultanov pid=7953
, My parent=7952
[sultanov_artem@vpsj3IeQ ~]$ Child of Sultanov pid=7953
, My parent=1
Child of Sultanov pid=7953
, My parent=1
Child of Sultanov pid=7953
, My parent=1
```

Висновок: Під час виконання Лабораторної роботи №8 мною були отримані навички в управлінні процесами в ОС Unix на рівні мови програмування С.