

Міністерство освіти і науки України  
Одеський національний політехнічний університет  
Інститут комп'ютерних систем  
Кафедра інформаційних систем

Лабораторна робота №8  
з дисципліни «Операційні Системи»

Тема: «Програмування керуванням процесами в ОС Unix»

Виконав:  
студент групи АІ-204  
Плаксивий Д.В.

Перевірив:  
Блажко О. А.

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

Завдання до виконання

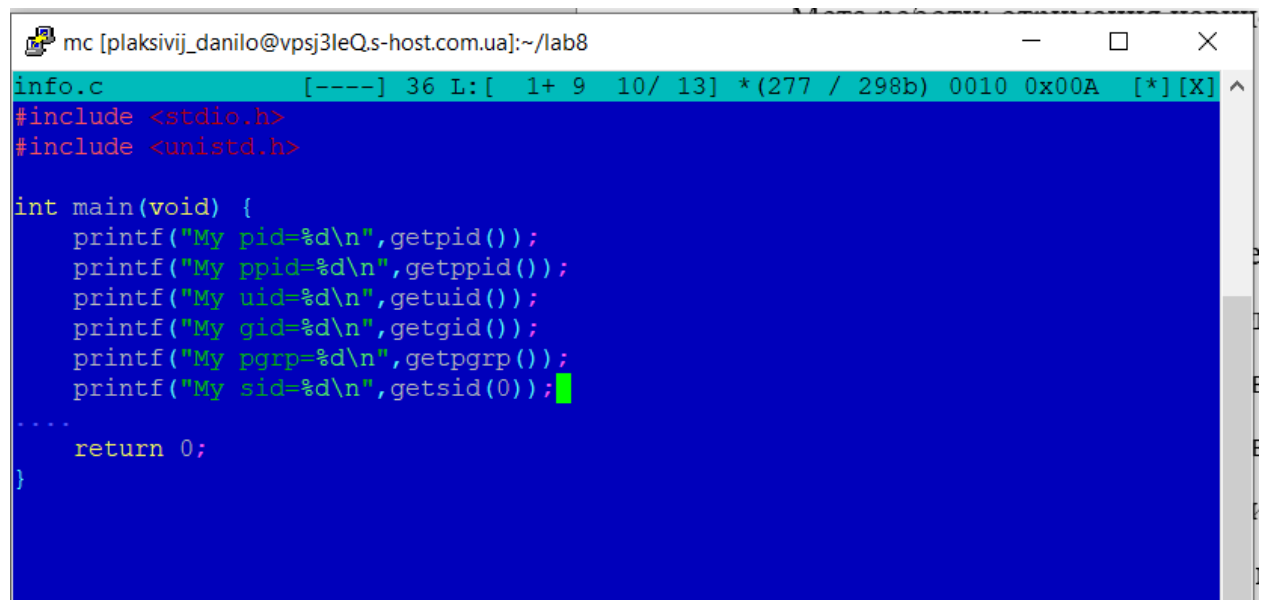
Завдання 1

Перегляд інформації про процес

Створив C-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.

```
[plaksivij_danilo@vpsj3IeQ lab8]$ gcc info.c -o info
[plaksivij_danilo@vpsj3IeQ lab8]$ ./info
My pid=26238
My ppid=24749
My uid=54393
My gid=54399
My pgrp=26238
My sid=24749
[plaksivij_danilo@vpsj3IeQ lab8]$
```



```
mc [plaksivij_danilo@vpsj3IeQ.s-host.com.ua]:~/lab8
info.c [----] 36 L: [ 1+ 9 10/ 13] *(277 / 298b) 0010 0x00A [*] [X] ^
#include <stdio.h>
#include <unistd.h>

int main(void) {
    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(0));
    ....
    return 0;
}
```

Завдання 2

Стандартне створення процесу

Створив C-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.

```
mc [plaksivij_danilo@vpsj3IeQ.s-host.com.ua]:~/lab8
create.c [-----] 71 L: [ 1+14 15/ 19] *(410 / 430b) 0010 0x00A [*] [
#include <stdio.h>
#include <unistd.h>
#include <sys/types.h>

extern char** environ;

int main(void) {
    char* echo_args[] = {"echo", "Child of Plaksivij\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);
<-----> char* echo_args[] = {"echo", "Parent of Plaksivij\n", NULL};
<-----> }
    return 0;
}
```

```
[plaksivij_danilo@vpsj3IeQ lab8]$ gcc create.c -o create
[plaksivij_danilo@vpsj3IeQ lab8]$ ./create
Parent pid=30599
Child pid=30600
Child of Plaksivij
```

### 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.

```
mc [plaksivij_danilo@vpsj3leQ.s-host.com.ua]:~/lab8
getsignal.c [----] 0 L:[ 1+12 13/ 14] *(265 / 276b) 0114 0x072 [*][X] ^
#include <signal.h>
#include <stdio.h>

static void sig_usr(int signo) {
<-----> if (signo == SIGUSR2)
<----->     printf("Process of Plaksivij got signal\n");
}
int main(void) {
    if (signal(SIGUSR2, sig_usr) == SIG_ERR)
<-----> fprintf(stderr, "Error\n");
<-----> for ( ; ; )
<----->     pause();
return 0;
}

[plaksivij_danilo@vpsj3leQ lab8]$ gcc getsignal.c -o getsignal
[plaksivij_danilo@vpsj3leQ lab8]$ ./getsignal

[plaksivij_danilo@vpsj3leQ ~]$ ps -u plaksivij_danilo -o pid,stat,cmd
  PID STAT  CMD
  5292 S      sshd: plaksivij_danilo@pts/4
  5293 Ss      -bash
  5324 S+      /usr/bin/mc -P /tmp/mc-plaksivij_danilo/mc.pwd.5293
  5326 Ss+     bash -rcfile .bashrc
  5690 S      sshd: plaksivij_danilo@pts/44
  5692 Ss      -bash
  5812 S      sshd: plaksivij_danilo@pts/8
  5813 Ss      -bash
  8741 S+      ./getsignal
  9626 R+      ps -u plaksivij_danilo -o pid,stat,cmd
[plaksivij_danilo@vpsj3leQ ~]$
```

```
mc [plaksivij_danilo@vpsj3IeQ.s-host.com.ua]:~/lab8
setsignal.c [-M--] 16 L:[ 1+ 3 4/ 13] *(56 / 179b) 0059 0x03B [*][X] ^
#include <signal.h>
#include <stdio.h>

pid_t pid = 8741;

int main(void)
{
    if (!kill(pid, SIGUSR2))
<----->printf("Send signal\n");
    else
<----->fprintf(stderr, "Error!\n");
return 0;
}

[plaksivij_danilo@vpsj3IeQ lab8]$ gcc setsignal.c -o setsignal
[plaksivij_danilo@vpsj3IeQ lab8]$ ./setsignal
Send signal
[plaksivij_danilo@vpsj3IeQ lab8]$
```

#### Завдання 4

##### Створення процесу-сироти

Створив C-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.

```
mc [plaksivij_danilo@vpsj3IeQs-host.com.ua]:~/lab8
sirota.c [-M--] 41 L:[ 1+15 16/ 22] *(336 / 407b) 0112 0x070 [*][X] ^
#include <stdio.h>
#include <unistd.h>
#include <sys/types.h>

int main(void) {
    int i;
    pid_t pid = fork();
    if (pid == 0) {
<-----> printf("Child pid=%d\n",getpid());
<-----> for (i=0; i<15; i++) {
<-----> printf("I am child. My parent id = %d\n", getppid());
<-----> sleep(7);
<-----> }
<-----> }
    else {
<-----> printf("Parent of Plaksivij pid=%d\n",getppid());
<-----> sleep(7);
<-----> _exit(0);
<-----> }
    return 0;
}

1Help 2Save 3Mark 4Replac 5Copy 6Move 7Search 8Delete 9PullDn10Quit v

[plaksivij_danilo@vpsj3IeQ lab8]$ gcc sirota.c -o sirota
[plaksivij_danilo@vpsj3IeQ lab8]$ ./sirota
Parent of Plaksivij pid=24749
Child pid=3175
I am child. My parent id = 3174
I am child. My parent id = 3174
[plaksivij_danilo@vpsj3IeQ lab8]$ I am child. My parent id = 1
I am child. My parent id = 1
I am child. My parent id = 1
I am child. My parent id = 1
I am child. My parent id = 1
I am child. My parent id = 1
I am child. My parent id = 1
I am child. My parent id = 1
I am child. My parent id = 1
I am child. My parent id = 1
I am child. My parent id = 1
I am child. My parent id = 1
I am child. My parent id = 1
I am child. My parent id = 1
I am child. My parent id = 1
```

Висновок: під час виконання лабораторної роботи н 8 отримав навички в управлінні процесами в ОС Unix на рівні мови програмування C. Найважчим завданням для мене було завдання 3.