

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 ”**  
Варіант 9

Виконала:  
Студентка групи АІ-203  
Мягих А.М.

Перевірив:  
Блажко О.А.

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

## **План:**

### **Завдання 1 Перегляд інформації про процес**

Створіть 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

### **Завдання 2 Стандартне створення процесу**

Створіть 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 команд `printf` та `echo` , де замість слова `Ivanov` в повідомленні повинно бути ваше прізвище в транслітерації.

### **Завдання 3 Обмін сигналами між процесами**

3.1 Створіть C програму, в якій процес очікує отримання сигналу `SIGUSR 2` 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 C програму

3.2 Створіть C програму, яка надсилає сигнал `SIGUSR 2` процесу , запущеному в попередньому пункті завдання. Запустіть створену C програму та проаналізуйте повідомлення , які виводить перша програма.

Завершіть процес , запущеному в попередньому пункті завдання

### **Завдання 4 Створення процесу сироти**

Створіть C програму, в якій процес батько несподівано завершується раніше процесу нащадку. Процес батько повинен очікувати завершення `n` 1 секунд. Процес нащадок повинен в циклі  $(2n + 1)$  раз із затримкою в 1 секунду виводити

повідомлення, наприклад, «Parent of Ivanov », за шаблоном як в попередньому завданні, і додатково виводити RPID 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 C 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 с творення процесу зомбі**

Створіть C 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 номер команди студента + номер студента в команді. Запустіть програму у фоновому режимі, а в окремому терміналі вивчіть вміст таблиці процесів і зробіть відповідні висновки.

### **Хід роботи:**

#### **Завдання 1 Перегляд інформації про процес**

Створіть 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

```
mc [myagkih_arina@vpsj3IeQ.s-host.com.ua]:~
info_1.c [----] 12 L:[ 1+ 4 5/ 12] *(68 / 299b) 0077 0x04D [*] [X] ^
#include <stdio.h>
#include <unistd.h>

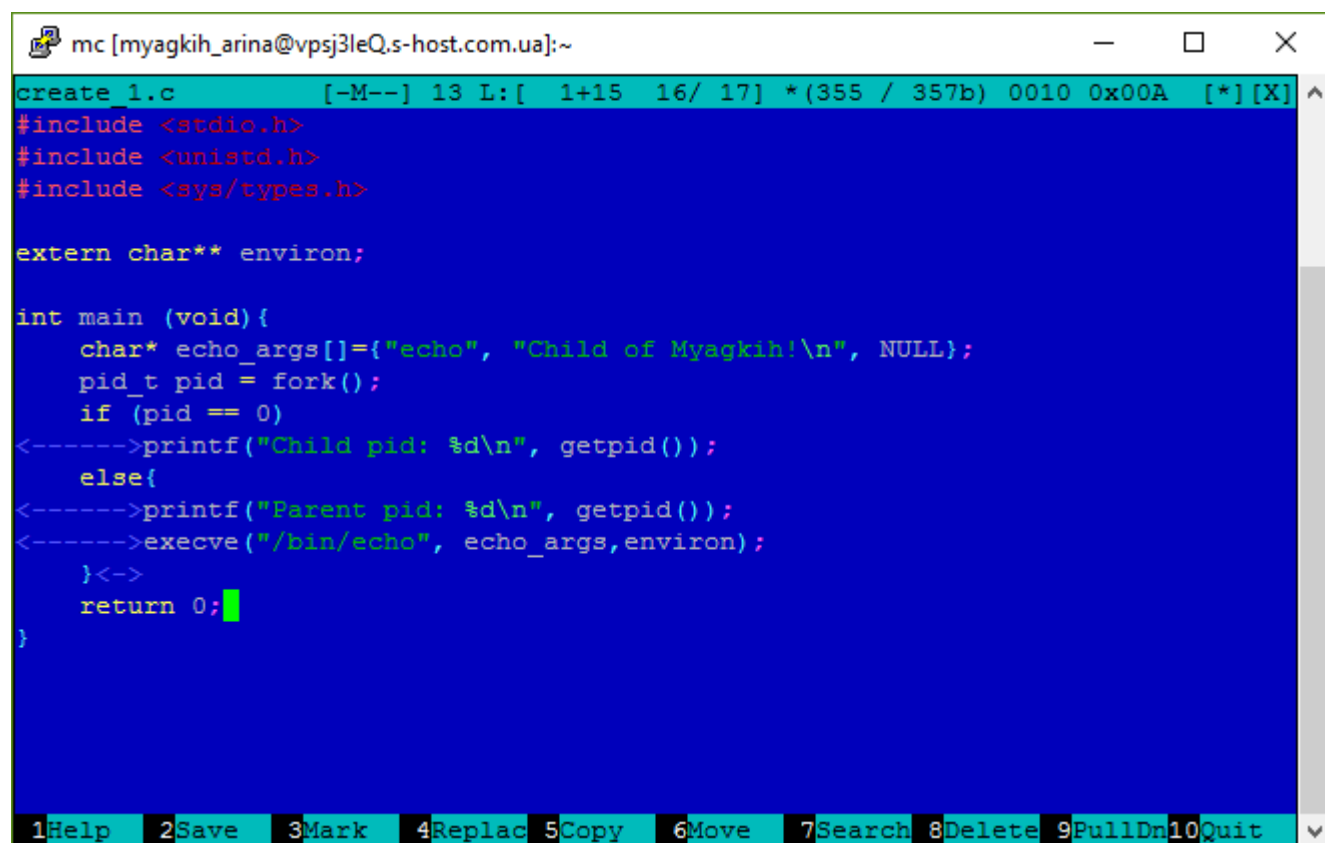
int main(void){
    printf("My pid: %d", getpid());
    printf("\nMy ppid: %d", getppid());
    printf("\nMy uid: %d", getuid());
    printf("\nMy gid: %d", getgid());
    printf("\nMy pgrp: %d", getpgrp());
    printf("\nMy sid: %d\n", getsid(0));
return 0;
}
```

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

```
myagkih_arina@vpsj3IeQ:~
login as:myagkih_arina
myagkih_arina@91.219.60.189's password:
Last login: Mon Apr 19 13:30:29 2021 from 176-119-76-54.broadband.tenet.nikolaev.ua
[myagkih_arina@vpsj3IeQ ~]$ gcc info_1.c -o info
[myagkih_arina@vpsj3IeQ ~]$ ./info
My pid: 19492
My ppid: 19432
My uid: 54367
My gid: 54373
My pgrp: 19492
My sid: 19432
[myagkih_arina@vpsj3IeQ ~]$
```

## Завдання 2 Стандартне створення процесу

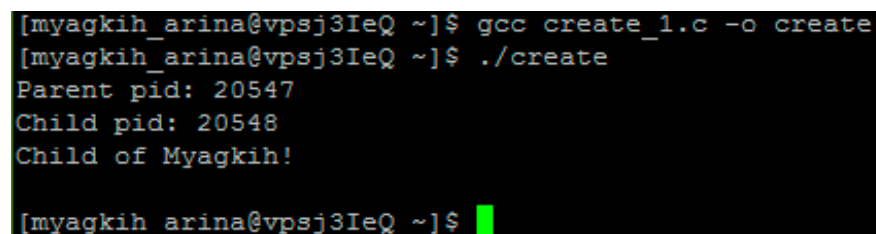
Створіть C- програму, яка створює процес нащадок , породжуючи процес та замінюючи образ процесу . У програмі процес батько повинен видати повідомлення типу « Parent of Ivanov », а процес нащадок повинен видати повідомлення типу «Child of Ivanov » через виклик команд `echo` , де замість слова Ivanov в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



```
mc [myagkih_arina@vpsj3leQ.s-host.com.ua]:~
create_1.c      [-M--] 13 L:[ 1+15 16/ 17] *(355 / 357b) 0010 0x00A  [*] [X] ^
#include <stdio.h>
#include <unistd.h>
#include <sys/types.h>

extern char** environ;

int main (void){
    char* echo_args={"echo", "Child of Myagkih!\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;
}
```

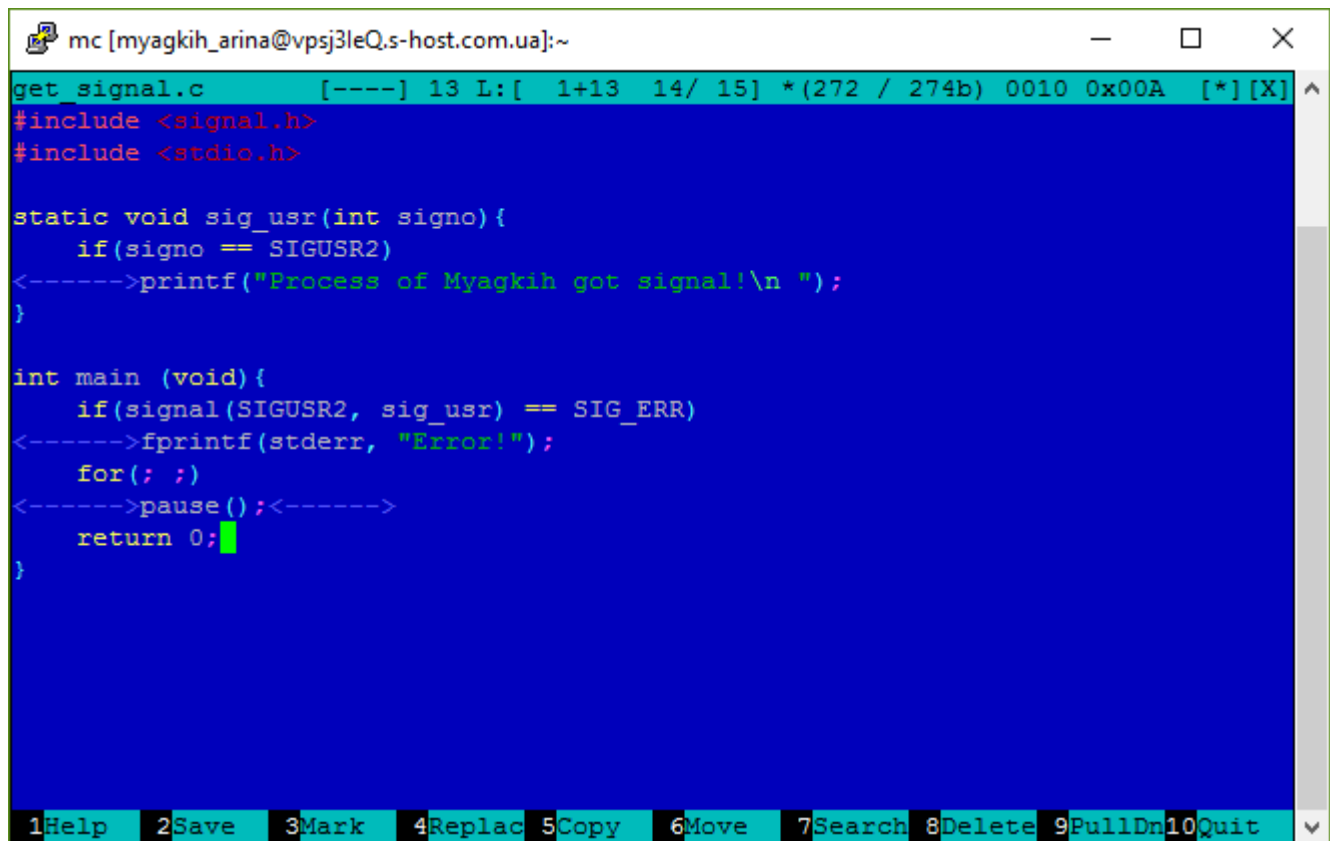


```
[myagkih_arina@vpsj3leQ ~]$ gcc create_1.c -o create
[myagkih_arina@vpsj3leQ ~]$ ./create
Parent pid: 20547
Child pid: 20548
Child of Myagkih!

[myagkih_arina@vpsj3leQ ~]$
```

### Завдання 3 Обмін сигналами між процесами

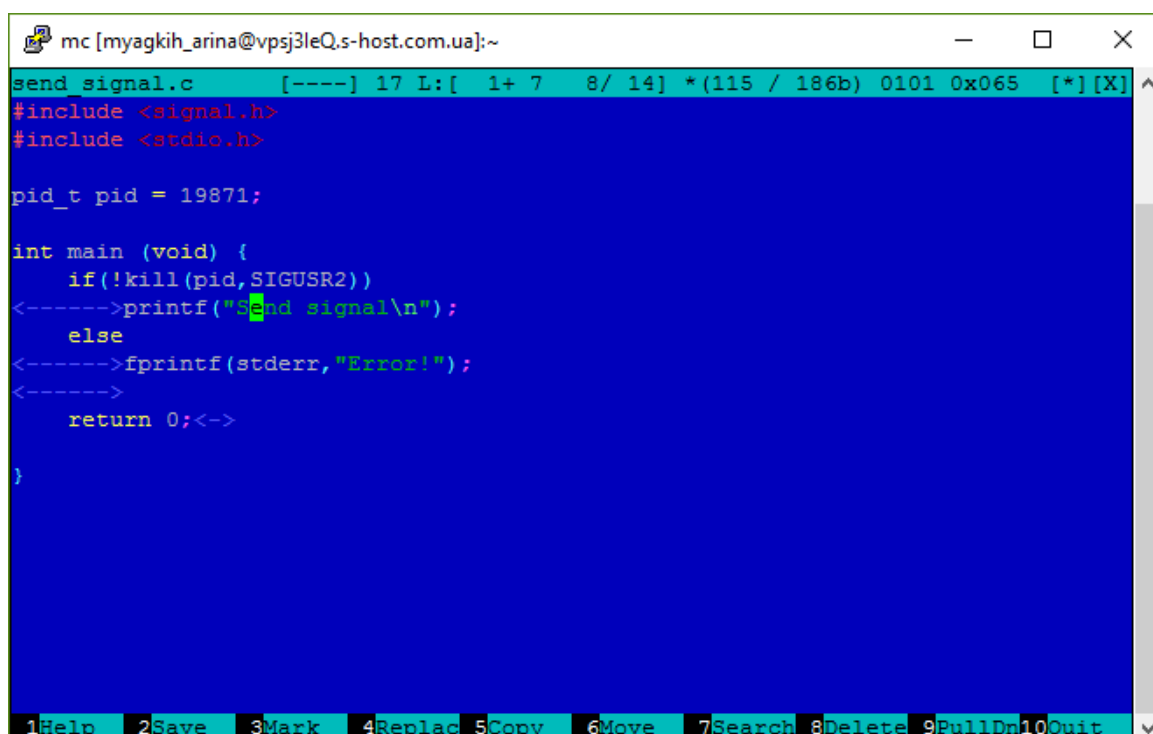
3.1 Створіть С програму, в якій процес очікує отримання сигналу SIGUSR2 та виводить повідомлення типу « Process of Ivanov got signal » після отримання сигналу, де замість слова Ivanov в повідомленні повинно бути ваше прізвище в транслітерації. Запустіть створену С програму



```
mc [myagkih_arina@vpsj3leQ.s-host.com.ua]:~
get_signal.c [----] 13 L:[ 1+13 14/ 15] *(272 / 274b) 0010 0x00A [*][X] ^
#include <signal.h>
#include <stdio.h>

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

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



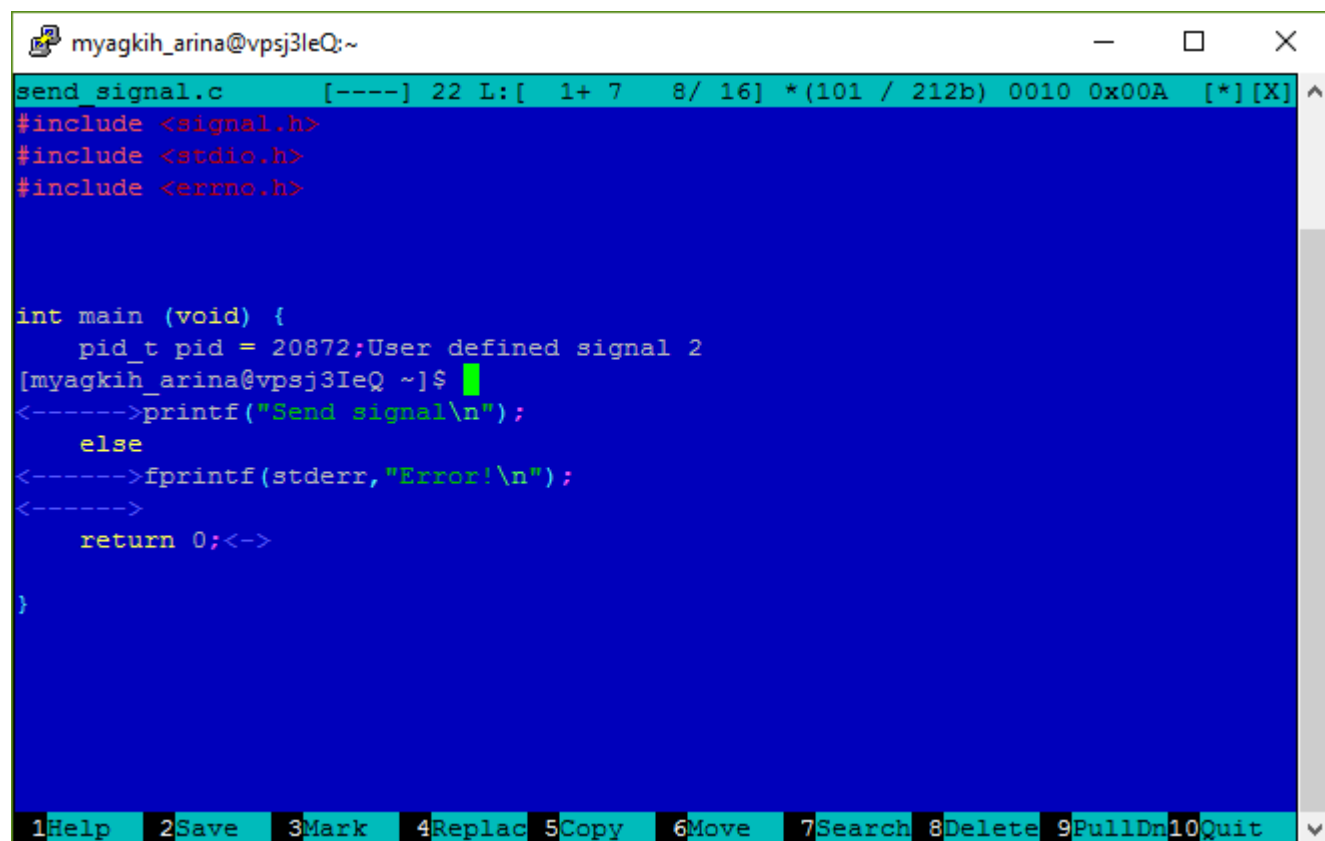
```
mc [myagkih_arina@vpsj3leQ.s-host.com.ua]:~
send_signal.c [----] 17 L:[ 1+ 7 8/ 14] *(115 / 186b) 0101 0x065 [*][X] ^
#include <signal.h>
#include <stdio.h>

pid_t pid = 19871;

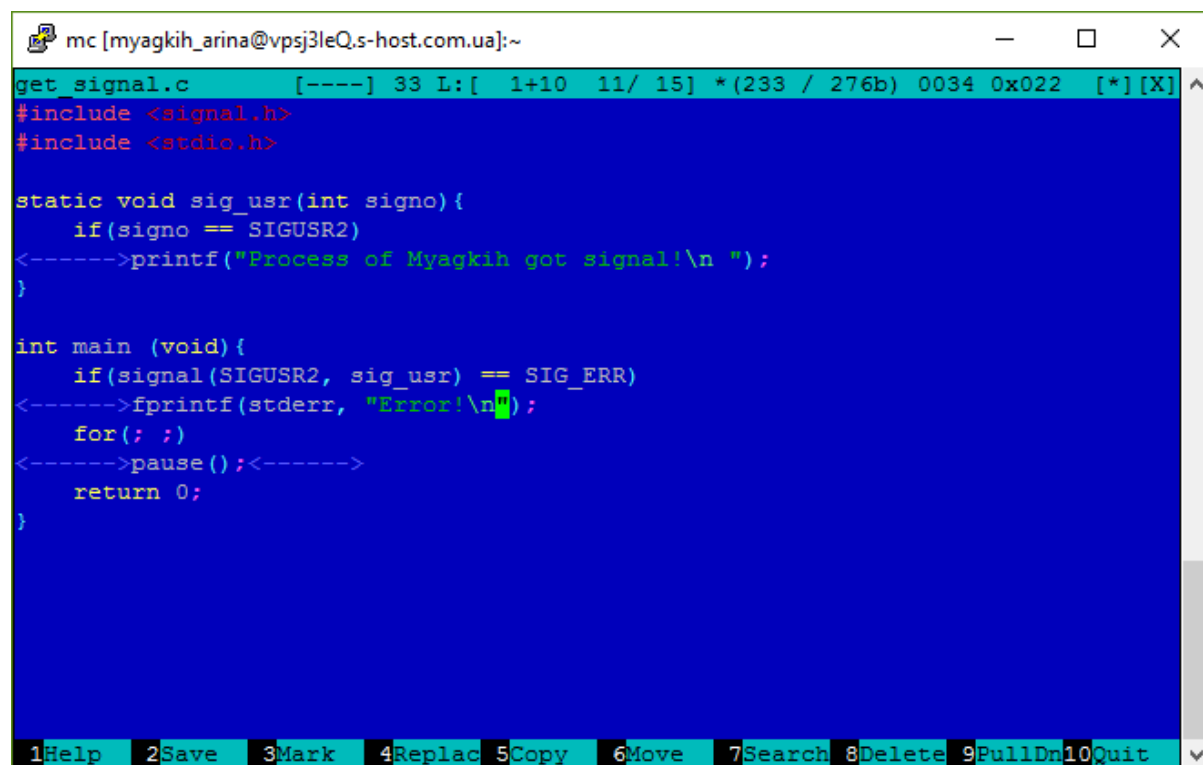
int main (void) {
    if(!kill(pid, SIGUSR2))
<----->printf("Send signal\n");
    else
<----->fprintf(stderr, "Error!");
<----->
    return 0;<->
}
```

3.2 Створіть С програму, яка надсилає сигнал SIGUSR 2 процес у , запущеному в по передньому пункту завдання. Запустіть створену С програму та проаналізуйте повідомлення , які виводить перша програма.

Завершіть процес , запущеному в попередньому пункту завдання



```
myagkih_arina@vpsj3IeQ:~  
send_signal.c [----] 22 L:[ 1+ 7 8/ 16] *(101 / 212b) 0010 0x00A [*][X]  
#include <signal.h>  
#include <stdio.h>  
#include <errno.h>  
  
int main (void) {  
    pid_t pid = 20872; User defined signal 2  
[myagkih_arina@vpsj3IeQ ~]$  
<----->printf("Send signal\n");  
    else  
<----->fprintf(stderr, "Error!\n");  
<----->  
    return 0; <-->  
}
```



```
mc [myagkih_arina@vpsj3IeQ.s-host.com.ua]:~  
get_signal.c [----] 33 L:[ 1+10 11/ 15] *(233 / 276b) 0034 0x022 [*][X]  
#include <signal.h>  
#include <stdio.h>  
  
static void sig_usr(int signo){  
    if(signo == SIGUSR2)  
<----->printf("Process of Myagkih got signal!\n ");  
}  
  
int main (void){  
    if(signal(SIGUSR2, sig_usr) == SIG_ERR)  
<----->fprintf(stderr, "Error!\n");  
    for(;;)  
<----->pause(); <----->  
    return 0;  
}
```

```
[myagkih_arina@vpsj3IeQ ~]$ ps -u myagkih_arina -o pid,stat,cmd
  PID STAT  CMD
19431 S      sshd: myagkih_arina@pts/30
19432 Ss     -bash
20785 S      sshd: myagkih_arina@pts/2
20786 Ss     -bash
20872 S+     /usr/bin/mc -P /tmp/mc-myagkih_arina/mc.pwd.20786
20874 Ss+    bash -rcfile .bashrc
21438 T      ./get_signal
21866 R+     ps -u myagkih_arina -o pid,stat,cmd

[myagkih_arina@vpsj3IeQ ~]$ gcc send_signal.c -o send_signal
[myagkih_arina@vpsj3IeQ ~]$ ./send_signal
Send signal
[myagkih_arina@vpsj3IeQ ~]$ gcc get_signal.c -o get_signal
[myagkih_arina@vpsj3IeQ ~]$ ./get_signal
```

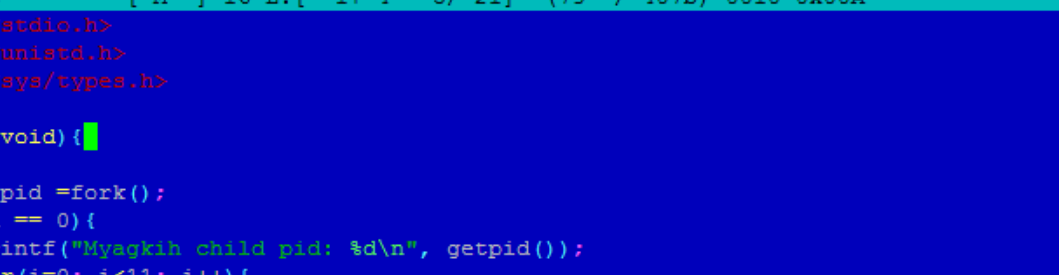
#### Завдання 4 Створення процесу сироти

Створіть C програму, в якій процес батько несподівано завершується раніше процесу нащадку. Процес батько повинен очікувати завершення  $n$  1 секунд. Процес нащадок повинен в циклі  $(2n + 1)$  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.





The screenshot shows a terminal window with the following content:

```

mc [myagkih_arina@vpsj3leQ.s-host.com.ua]:~
sirota_1.c      [-M--] 16 L:[ 1+ 4 5/ 21] *(79 / 407b) 0010 0x00A      [*] [X] ^
#include <stdio.h>
#include <unistd.h>
#include <sys/types.h>

int main (void){
    int i;
    pid_t pid =fork();
    if(pid == 0){
<----->printf("Myagkih child pid: %d\n", getpid());
<----->for(i=0; i<11; i++){
<----->    printf("Child of Myagkih pid: %d\n, my parent: \n", getpid(), getppid());
<----->    sleep(6);
<----->}
    }
    else{
<----->printf("Parent of Myagkih pid: %d\n", getpid());
<----->sleep(1);
<----->_exit(0);.
    }
    return 0;
}

```

At the bottom of the terminal, there is a menu bar with the following options: 1Help, 2Save, 3Mark, 4Replac, 5Copy, 6Move, 7Search, 8Delete, 9FullDn, 10Quit.

```
myagkih_arina@vpsj3IeQ:~  
.ua  
[myagkih_arina@vpsj3IeQ ~]$ gcc sirota_1.c -o sirota  
[myagkih_arina@vpsj3IeQ ~]$ ./sirota  
Child of Myagkih pid: 23501  
, my parent:  
Myagkih child pid: 23502  
Child of Myagkih pid: 23502  
, my parent:  
Child of Myagkih pid: 23501  
, my parent:  
Child of Myagkih pid: 23502  
, my parent:  
Child of Myagkih pid: 23502  
, my parent:  
Child of Myagkih pid: 23501  
, my parent:  
Child of Myagkih pid: 23502  
, my parent:  
Child of Myagkih pid: 23501  
, my parent:  
Child of Myagkih pid: 23502  
, my parent:  
Child of Myagkih pid: 23501  
, my parent:  
Child of Myagkih pid: 23502  
, my parent:  
Child of Myagkih pid: 23501  
, my parent:  
Child of Myagkih pid: 23502  
, my parent:  
Child of Myagkih pid: 23501  
, my parent:
```

**Висновок:** під час виконання лабораторної роботи було отримано навички в управлінні процесами ОС Unix на рівні мови програмування C