Одесский национальный политехнический университет
Институт компьютерных систем
Кафедра информационных систем

Лабораторная работа № 14
По дисциплине: «Операционные системы»
Тема: «Основы управления правами доступа к файловой системе»

Выполнила:
Студентка группы АИ-205
Колдунова Т. А.
Проверил:
Блажко О.А.

Одесса 2020

Цель работы: получение навыков в управлении процессами в ОС Unix на уровне языка программирования С

План работы.

1 Теоретические сведения

1.1 Расширенный функционал команд оболочки ОС интерфейса командной строки и утилит командной строки

1.2 Редактирование каталогов и файлов файловой системы

1.3 Перенаправление потоков данных

1.4 Конвейеризация команд

1.5 Команды оболочки и утилиты командной строки по обработке текста

1.6 Конфигурация работы оболочки

2 Задание к исполнению

3 Требования к оформлению протокола выполнения лабораторной работы

Решение:

```
SilverBlock — koldunova_tetyana@vpsj3IeQ:~ — ssh koldunova_tetyana@91.219.60.1...
-bash: warning: setlocale: LC_CTYPE: cannot change locale (UTF-8): No such file
or directory
[[koldunova_tetyana@vpsj3IeQ ~]$ nano info.c                                    ]
[[koldunova_tetyana@vpsj3IeQ ~]$ gcc -o info info.c                             ]
info.c: In function 'main':
info.c:4:15: error: 'LC_ALL' undeclared (first use in this function)
     setlocale(LC_ALL, "rus");
               ^
info.c:4:15: note: each undeclared identifier is reported only once for each fun
ction it appears in
[[koldunova_tetyana@vpsj3IeQ ~]$ nano info.c                                    ]
[[koldunova_tetyana@vpsj3IeQ ~]$ gcc -o info info.c                             ]
[[koldunova_tetyana@vpsj3IeQ ~]$ gcc -o info.c                                  ]
gcc: fatal error: no input files
compilation terminated.
[[koldunova_tetyana@vpsj3IeQ ~]$ gcc -o info info.c                             ]
[[koldunova_tetyana@vpsj3IeQ ~]$ ./info                                         ]
Ідентифікатор групи процесів лідера сесії 11214
Ідентифікатор групи процесів, до якої належить процес 20038
Ідентифікатор процесу, що викликав цю функцію 11214
Ідентифікатор батьківського процесу 11214
Ідентифікатор користувача процесу, що викликав цю функцію 54415
Ідентифікатор групи процесу, що викликав цю функцію 54421
[koldunova_tetyana@vpsj3IeQ ~]$
```

```c
#include <stdio.h>
#include <unistd.h>
#include <locale.h>
int main(int argc, const char * argv[]) {
    setlocale(LC_ALL, "rus");
    pid_t pgrp = getpgrp();
    pid_t pid = getppid();
    pid_t sid = getsid(pid);
    pid_t ppid = getppid();
    uid_t uid = getuid();
    gid_t gid = getgid();

    printf("Ідентифікатор групи процесів лідера сесії %d\n", sid);
    printf("Ідентифікатор групи процесів, до якої належить процес %d\n", pgrp);
    printf("Ідентифікатор процесу, що викликав цю функцію %d\n", pid);
    printf("Ідентифікатор батьківського процесу %d\n", ppid);
    printf("Ідентифікатор користувача процесу, що викликав цю функцію %d\n", uid);
    printf("Ідентифікатор групи процесу, що викликав цю функцію %d\n", gid);
    return 0;
}
```

```
[[koldunova_tetyana@vpsj3IeQ ~]$ ./fork
Parent procces: pid = 20284
Child procces: pid = 20285
Proccess of Koldunova Tanya got signal
```

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

extern char** environ;

int main(int argc, const char * argv[]) {

    char* echo_args[] =  {"echo", "Proccess of Koldunova Tanya got signal\n", NULL};
    pid_t pid = fork();
    if (pid == 0) {
        printf("Child procces: pid = %d\n", getpid());
    } else {
        printf("Parent procces: pid = %d\n", getpid());
        execve("/bin/echo", echo_args, environ);
    }
    return 0;
}
```

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


static void sig_usr(int signo) {
    if (signo == SIGUSR2) {
        printf("i got SIGUSR2\n");
    }
}

int main(int argc, const char * argv[]) {
    if (signal(SIGUSR2, sig_usr) == SIG_ERR) {
        fprintf(stderr, "Error");

    }
    for ( ; ; ) {
        pause();
    }
    return 0;
```

```
^G Get Help  ^O WriteOut  ^R Read File ^Y Prev Page ^K Cut Text  ^C Cur Pos
^X Exit      ^J Justify   ^W Where Is  ^V Next Page ^U UnCut Text^T To Spell
[[koldunova_tetyana@vpsj3IeQ ~]$ gcc -o get_signal get_signal.c
[[koldunova_tetyana@vpsj3IeQ ~]$ ./get_signal
```

```
[Error[koldunova_tetyana@vpsj3IeQ ~]$ gcc -o get_signal2 get_signal2.c
[[koldunova_tetyana@vpsj3IeQ ~]$ ./get_signal2
Error[koldunova_tetyana@vpsj3IeQ ~]$
```

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

pid_t pid = 18144;

int main(void) {
    if (!kill(pid, SIGUSR2))
        printf("Send signal go pid=%d", pid);
    else
        fprintf(stderr, "Error");
    return 1;
}
```

```
#include <stdio.h>
#include <signal.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(),getpp$
        sleep(2);
        _exit(0);
    } else {
        for(i = 0; i<5; i++) {
            printf("I am child with pid=%d. My parent id=%d", getpid(), getppid$
            sleep(1);
        }
    }
    return  0;

^G Get Help  ^O WriteOut  ^R Read File ^Y Prev Page ^K Cut Text  ^C Cur Pos
^X Exit      ^J Justify   ^W Where Is  ^V Next Page ^U UnCut Text^T To Spell
[[koldunova_tetyana@vpsj3IeQ ~]$ ./task4
I am parent with pid = 25102. My child pid = 11214
[koldunova_tetyana@vpsj3IeQ ~]$ I am child with pid=25103. My parent id=25102I a
m child with pid=25103. My parent id=25102I am child with pid=25103. My parent i
d=1I am child with pid=25103. My parent id=1I am child with pid=25103. My parent
 id=1
```

Вывод: В данной лабораторной мы получили навыки в управлении процессами в ОС Unix на уровне языка программирования С