

Лабораторная работа №12

Программирование в командном процессоре ОС UNIX. Расширенное программирование

Аскеров А.Э.

24 апреля 2023

Российский университет дружбы народов, Москва, Россия

Вступление

Изучить основы программирования в оболочке ОС UNIX. Научиться писать более сложные командные файлы с использованием логических управляющих конструкций и циклов.

Выполнение лабораторной работы

1. Напишем командный файл, реализующий упрощённый механизм семафоров.

Командный файл должен в течение некоторого времени t_1 дожидаться освобождения ресурса, выдавая об этом сообщение, а дождавшись его освобождения, использовать его в течение некоторого времени $t_2 < t_1$, также выдавая информацию о том, что ресурс используется соответствующим командным файлом (процессом). Запустим командный файл в одном виртуальном терминале в фоновом режиме, перенаправив его вывод в другой (`> /dev/tty#`, где `#` — номер терминала куда перенаправляется вывод), в котором также запущен этот файл, но не фоновом, а в привилегированном режиме. Доработаем программу так, чтобы имелась возможность взаимодействия трёх и более процессов.

Создадим командный файл, напишем в нём программу, дадим файлу право на исполнение, запустим его.

```
[aeaskerov@fedora lab12]$ vi lab121
[aeaskerov@fedora lab12]$ chmod +x lab121
[aeaskerov@fedora lab12]$ ./lab121
flock: requires file descriptor, file or directory
file was unlocked
flock: requires file descriptor, file or directory
file was unlocked
```

Рис. 1: Создание командного файла, право доступа и запуск

Приведём саму программу.

```
lockfile="./locking.file"

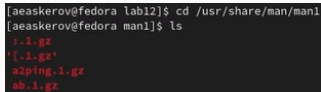
exec {fn}>"$lockfile"
if test -f "$lockfile"
then
    while [ 1 != 0 ]
    do
        if flock -n ${fn}
        then
            echo "file was locked"
            sleep 4
            echo "unlocking"
            flock -u ${fn}

        else
            echo "file was unlocked"
            sleep 3
        fi
    done
fi
```

Рис. 2: Программа один

2. Реализуем команду `man` с помощью командного файла. Изучим содержимое каталога `/usr/share/man/man1`. В нем находятся архивы текстовых файлов, содержащих справку по большинству установленных в системе программ и команд. Каждый архив можно открыть командой `less` сразу же просмотрев содержимое справки. Командный файл должен получать в виде аргумента командной строки название команды и в виде результата выдавать справку об этой команде или сообщение об отсутствии справки, если соответствующего файла нет в каталоге `man1`.

Изучим содержимое каталога `/usr/share/man/man1`.



```
[aeaskerov@fedora lab12]$ cd /usr/share/man/man1
[aeaskerov@fedora man1]$ ls
i.1.gz
' [.1.gz'
a2ping.1.gz
ab.1.gz
```

Рис. 3: Содержимое каталога `/usr/share/man/man1`

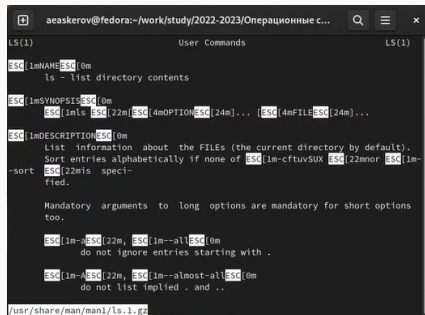
Создадим командный файл, напишем в нём программу, дадим файлу право на исполнение, запустим его. Например, посмотрим информацию о команде ls.

```
[aeaskerov@fedora lab12]$ vi lab122  
[aeaskerov@fedora lab12]$ chmod +x lab122  
[aeaskerov@fedora lab12]$ ./lab122 -c ls
```

Рис. 4: Создание командного файла два, право доступа и запуск

Задание 2

Покажем результат работы командного файла.



```
aeaskerov@fedora:~/work/study/2022-2023/Операционные с...
LS(1)                                User Commands                                LS(1)

ESC[1mNAMEESC[0m
ls - list directory contents

ESC[1mSYNOPSISESC[0m
ESC[1mls ESC[22m[ESC[4mOPTIONESC[24m]... [ESC[4mFILEESC[24m]...

ESC[1mDESCRIPTIONESC[0m
List information about the FILES (the current directory by default).
Sort entries alphabetically if none of ESC[1m-cftuvSUX ESC[22mnor ESC[1m-
-sort ESC[22mis speci-
fied.

Mandatory arguments to long options are mandatory for short options
too.

ESC[1m-mESC[22m, ESC[1m--allESC[0m
do not ignore entries starting with .

ESC[1m-AESC[22m, ESC[1m--almost-allESC[0m
do not list implied . and ..

/usr/share/man/man1/ls.1.gz
```

Рис. 5: Результат работы

Приведём саму программу.

```
command=""

while getopts :c: opt
do
case $opt in
    c)command="$OPTARG";;
esac
done

if test -f "/usr/share/man/man1/$command.1.gz"
then less /usr/share/man/man1/$command.1.gz
else
echo "No such command!"
fi
```

Рис. 6: Программа два

- Используя встроенную переменную `$RANDOM`, напомним командный файл, генерирующий случайную последовательность букв латинского алфавита. Учтём, что `$RANDOM` выдаёт псевдослучайные числа в диапазоне от 0 до 32767.

Создадим командный файл, напомним в нём программу, дадим файлу право на исполнение, запустим его. Видим, что последовательности генерируются.

```
[aeaskerov@fedora lab12]$ vi lab123
[aeaskerov@fedora lab12]$ chmod +x lab123
[aeaskerov@fedora lab12]$ ./lab123
ea jj
[aeaskerov@fedora lab12]$ ./lab123
ia bj
[aeaskerov@fedora lab12]$ ./lab123
da bi h
```

Рис. 7: Создание командного файла три, право доступа и запуск

Приведём саму программу.

```
echo $RANDOM | tr '0-9' 'a-zA-Z'
```

Рис. 8: Программа три

Заключение

Изучены основы программирования в оболочке ОС UNIX. Приобретён навык написания более сложных командных файлов с использованием логических управляющих конструкций и циклов.