

Лабораторная работа 12

Программирование в командном процессоре ОС UNIX.

Расширенное программирование

Korshunova Polina

2022, 28 May 2022

RUDN University, Moscow, Russian Federation

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

1. Написать 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). Запустить командный файл в одном виртуальном терминале в фоновом режиме, перенаправив его вывод в другой, в котором также запущен этот файл, но не фоновом, а в привилегированном режиме. Доработать программу так, чтобы имелась возможность взаимодействия трёх и более процессов.

2. Реализовать команду `man` с помощью командного файла. Изучите 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`.

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

Преимущества и недостатки Bash:

Многие языки программирования намного удобнее и понятнее для пользователя. Например, Python более быстр, так как компилируется байтами. Однако главное преимущество Bash – его повсеместное распространение. Более того, Bash позволяет очень легко работать с файловой системой без лишних конструкций (в отличие от других языков программирования). Но относительно таких bash очень жжат. То есть, например, C имеет гораздо более широкие возможности для разработчика.

Выполнение лабораторной работы

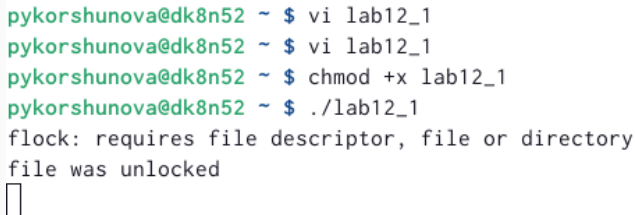
1. Я написала командный файл, реализующий упрощенный механизм семафоров.

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

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

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

Затем я добавила право на исполнение файла и выполнила его.

A terminal window with a light gray title bar. The prompt is 'pykorshunova@dk8n52 ~ \$'. The commands and their outputs are: 'vi lab12_1' (no output), 'vi lab12_1' (no output), 'chmod +x lab12_1' (no output), and './lab12_1' (output: 'flock: requires file descriptor, file or directory' and 'file was unlocked'). A cursor is visible on the line following the last output.

```
pykorshunova@dk8n52 ~ $ vi lab12_1
pykorshunova@dk8n52 ~ $ vi lab12_1
pykorshunova@dk8n52 ~ $ chmod +x lab12_1
pykorshunova@dk8n52 ~ $ ./lab12_1
flock: requires file descriptor, file or directory
file was unlocked
█
```

Рис. 2: Результат выполнения скрипта 1.

2. Я просмотрела содержимое каталога `/usr/share/man/man1`.

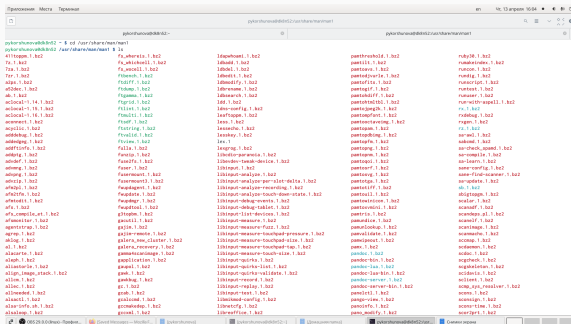


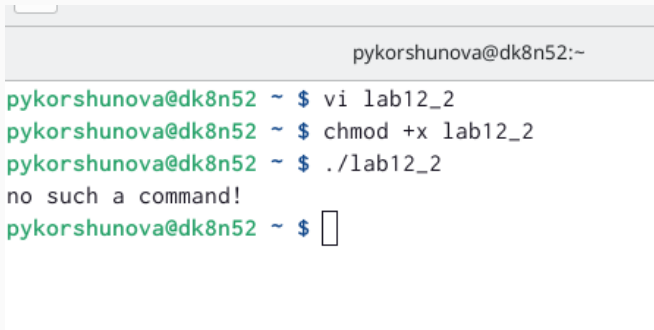
Рис. 3: Просмотр каталога `/usr/share/man/man1`.

Выполнение лабораторной работы

Я написала командный файл, позволяющий реализовать команду man с помощью команды less, которая выдает содержимое справки по команде.

```
pykorshunova@dk8n52:~  
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 a command!"  
fi  
~
```

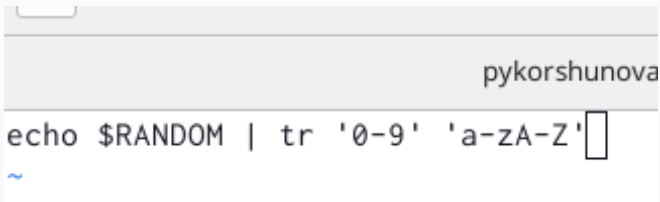
Затем я добавила право на исполнение файла и выполнила его.

A terminal window with a light gray title bar and a white background. The title bar contains the text 'pykorshunova@dk8n52:~'. The terminal shows a series of commands and their outputs. The prompt is 'pykorshunova@dk8n52 ~ \$'. The first command is 'vi lab12_2'. The second command is 'chmod +x lab12_2'. The third command is './lab12_2', which results in the output 'no such a command!'. The prompt returns to 'pykorshunova@dk8n52 ~ \$' followed by a cursor.

```
pykorshunova@dk8n52 ~ $ vi lab12_2
pykorshunova@dk8n52 ~ $ chmod +x lab12_2
pykorshunova@dk8n52 ~ $ ./lab12_2
no such a command!
pykorshunova@dk8n52 ~ $
```

Рис. 5: Результат выполнения скрипта 2.

3. Я написала командный файл, который генерировал случайную последовательность букв латинского алфавита, для этого я использовала встроенную переменную \$RANDOM.

A terminal window with a light gray title bar. The title bar contains the text 'pykorshunova' on the right side. The terminal has a white background. The prompt 'pykorshunova' is shown in a gray font. Below the prompt, the command 'echo \$RANDOM | tr '0-9' 'a-zA-Z'' is entered in a black font. A blue cursor is positioned at the end of the command. A small blue tilde '~' is visible on the line below the command.

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

Рис. 6: Скрипт к заданию 3.

Затем я добавила право на исполнение файла и выполнила его.

```
pykorshunova@dk8n52 ~ $ vi lab12_3
pykorshunova@dk8n52 ~ $ chmod +x lab12_3
pykorshunova@dk8n52 ~ $ ./lab12_3
dbaec
pykorshunova@dk8n52 ~ $
```

Рис. 7: Результат выполнения скрипта 3.

В ходе выполнения лабораторной работы я изучила основы программирования в оболочке ОС UNIX, а также научилась писать более сложные командные файлы с использованием логических управляющих конструкций и циклов.