

# Лабораторная работа №12

Курс “Операционные Системы”

---

Шилоносов Д.В., НКАбд-03-22

29 апреля 2023

Российский университет дружбы народов, Москва, Россия

## Информация

---

- Шилоносов Данил Вячеславович
- студент группы НКАбд-03-22
- кафедры Компьютерные и информационные науки
- Российский университет дружбы народов
- 1132221810@pfur.ru

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 (`> /dev/tty#`, где `#` — номер терминала куда перенаправляется вывод), 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.

## Выполнение лабораторной работы

---



## Исходный код скрипта №1

```
1 #!/bin/bash
2
3 resource="semaphore.txt"
4 t1=5
5 t2=10
6
7 # установка блокировки на ресурсе
8 exec 200>"$resource"
9 flock -x 200
10
11 # ожидаем освобождения ресурса
12 echo "Waiting for resource..."
13 sleep $t1
14
15 # использование ресурса
16 echo "Using resource..."
17 echo "Process $$ is using the resource" > "$resource"
18 sleep $t2
19
20 # освобождение ресурса
21 flock -u 200
22 echo "Resource released" I
```

Данный скрипт реализует упрощенный механизм семафоров на основе файловой блокировки. Он дожидается освобождения ресурса, выдавая об этом сообщение, затем использует его, выдавая информацию о том, что ресурс используется соответствующим процессом. Ресурс защищен блокировкой на файле `semaphore.txt`. Скрипт можно запустить в фоновом режиме и перенаправить его вывод в другой терминал.

# Результат выполнения скрипта №1

dvshilonosov@dvshilonosov:~/scripts/script1 — tmux attach-session -t 3	
<pre>/dev/pts/8 [dvshilonosov@dvshilonosov script1]\$ ./run_in_all_terminals.sh Запуск скрипта в терминале /dev/pts/0 [sudo] пароль для dvshilonosov: Запуск скрипта в терминале /dev/pts/1 Запуск скрипта в терминале /dev/pts/2 Запуск скрипта в терминале /dev/pts/3 Запуск скрипта в терминале /dev/pts/4 Запуск скрипта в терминале /dev/pts/5 Запуск скрипта в терминале /dev/pts/6 Запуск скрипта в терминале /dev/pts/7 Запуск скрипта в терминале /dev/pts/8 Запуск скрипта в терминале /dev/pts/9 [dvshilonosov@dvshilonosov script1]\$ Waiting for resource... Using resource... Resource released</pre>	<pre>[dvshilonosov@dvshilonosov script1]\$ tty /dev/pts/9 [dvshilonosov@dvshilonosov script1]\$ Waiting for resource...</pre>
<pre>[dvshilonosov@dvshilonosov script1]\$ tty /dev/pts/7 [dvshilonosov@dvshilonosov script1]\$ Waiting for resource... Using resource... Resource released</pre>	<pre>[dvshilonosov@dvshilonosov script1]\$ tty /dev/pts/6 [dvshilonosov@dvshilonosov script1]\$ Waiting for resource... Using resource... Resource released</pre>

```
1 #!/bin/bash
2
3 if [ -z "$1" ]; then
4     echo "Использование: $0 <название команды>"
5     exit 1
6 fi
7
8 command_name="$1"
9 man_directory="/usr/share/man"
10
11 # Указываем путь к каталогу с man-страницами и вызываем стандартную команду man
12 MANPATH="$man_directory" man "$command_name"
```

Этот скрипт предоставляет пользователю возможность получить справку для команды, указанной в качестве аргумента. Он использует стандартную команду `man` с указанием пути к каталогу с `man`-страницами через переменную окружения `MANPATH`.

Скрипт проверяет, указано ли имя команды в качестве аргумента. Если аргумент не указан, выводит справочное сообщение и завершает работу. Запоминает указанное имя команды и путь к каталогу `man`-страниц. Устанавливает значение переменной окружения `MANPATH` равным пути к каталогу `man`-страниц. Вызывает стандартную команду `man` с именем команды в качестве аргумента.

## Результат выполнения скрипта №2

```
dvshilonosov@dvshilonosov:~/scripts/script2 — /bin/bash ./my_man.sh less
```

LESS(1) General Commands Manual

**NAME**

less - opposite of more

**SYNOPSIS**

```
less -?
less --help
less -V
less --version
less [-[+]aABcCdeEfFgGiIJKLmMnNqQrRsSuUVwWX~]
      [-b space] [-h lines] [-j line] [-k keyfile]
      [-{oO} logfile] [-p pattern] [-P prompt] [-t tag]
      [-T tagsfile] [-x tab,...] [-y lines] [-[z] lines]
      [-# shift] [+{+}cmd] [--] [filename]...
```

(See the OPTIONS section for alternate option syntax with long option names.)

**DESCRIPTION**

less is a program similar to more(1), but which allows backward movement in the file as well as forward movement. It reads the entire input file before starting, so with large input files it starts up faster than text editors like vi (on some systems), so it can run on a variety of terminals. There is even limited support for hardcopy terminals. Commands which should be printed at the top of the screen are prefixed with a caret.)

Commands are based on both more and vi. Commands may be preceded by a decimal number, called N in the description of some commands, as indicated.

## Исходный код скрипта №3

```
1 #!/bin/bash
2
3 # Задайте количество букв в последовательности
4 sequence_length=10
5
6 # Создайте пустую строку
7 random_string=""
8
9 # Цикл для генерации последовательности
10 for ((i=0; i<$sequence_length; i++)); do
11     # Генерируйте случайное число от 0 до 25 с использованием $RANDOM
12     random_number=$((($RANDOM % 26))
13
14     # Преобразуйте случайное число в соответствующую букву латинского алфавита
15     random_letter=$(printf "\\$(printf '%03o' $((random_number + 65)))")
16
17     # Добавьте случайную букву к строке
18     random_string+="$random_letter"
19 done
20
21 # Выведите случайную последовательность
22 echo "$random_string"
```

Скрипт генерирует случайную последовательность латинских букв заданной длины. Он использует переменную `$RANDOM` для получения псевдослучайных чисел, преобразует их в соответствующие буквы и объединяет их в строку. Затем выводит эту строку на экран.



## Результат выполнения скрипта №3



```
[dvshilonosov@dvshilonosov script3]$ ./script.sh  
CONZYJQWEU  
[dvshilonosov@dvshilonosov script3]$ ./script.sh  
VYOXAJOBBU  
[dvshilonosov@dvshilonosov script3]$ ./script.sh  
DRWUIAEWRA  
[dvshilonosov@dvshilonosov script3]$ ./script.sh  
JULMJSNZLE  
[dvshilonosov@dvshilonosov script3]$ ./script.sh  
HQLZTRMVPA  
[dvshilonosov@dvshilonosov script3]$ ./script.sh  
TGWEROIEKR  
[dvshilonosov@dvshilonosov script3]$ ./script.sh  
KQONUVJIUZ  
[dvshilonosov@dvshilonosov script3]$ ./script.sh  
XRPCBDZWKH  
[dvshilonosov@dvshilonosov script3]$ ./script.sh
```

## Результаты

---

Были получены практические навыки написания более сложные командных файлов с использованием логических управляющих конструкций и циклов.