

# **Лабораторная работа-13**

**Программирование в командном процессоре ОС UNIX. Расширенное  
программирование**

Доленко Дарья Васильевна НБИбд-01-21

# Содержание

1	Цель работы	4
2	Выполнение лабораторной работы	5
3	(рис. 3.1)	8
4	Вывод	9
5	Ответы на контрольные вопросы:	10

## Список иллюстраций

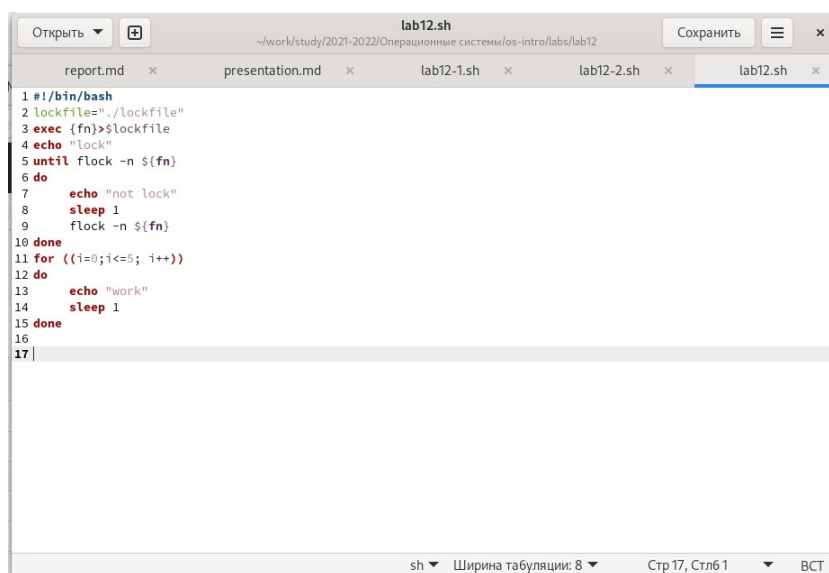
2.1	Текст программы . . . . .	5
2.2	Текст программы . . . . .	6
2.3	Результат . . . . .	7
2.4	Текст программы . . . . .	7
3.1	Создание файлов, наделение их необходимыми правами и запуск.	8

# 1 Цель работы

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

## 2 Выполнение лабораторной работы

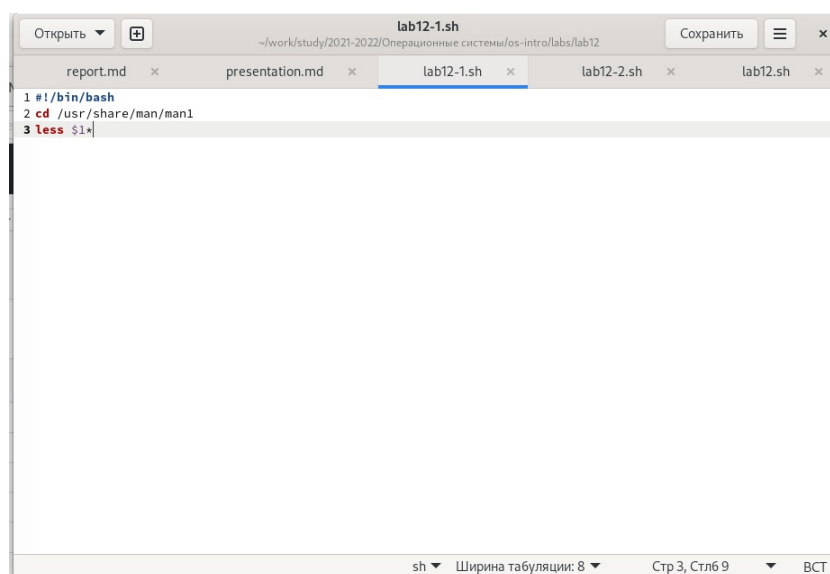
Написала командный файл, реализующий упрощённый механизм семафоров. Командный файл должен в течение некоторого времени  $t_1$  дожидаться освобождения ресурса, выдавая об этом сообщение, а дождавшись его освобождения, использовать его в течение некоторого времени  $t_2 < t_1$ , также выдавая информацию 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. Доработала программу так, чтобы имела возможность взаимодействия трёх и более процессов. (рис. 2.1)



```
1 #!/bin/bash
2 lockfile="/lockfile"
3 exec {fn}>$lockfile
4 echo "lock"
5 until flock -n ${fn}
6 do
7     echo "not lock"
8     sleep 1
9     flock -n ${fn}
10 done
11 for ((i=0; i<=5; i++))
12 do
13     echo "work"
14     sleep 1
15 done
16
17
```

Рис. 2.1: Текст программы

Реализовала 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 просмотреть 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`. (рис. 2.2 2.3)



The image shows a terminal window titled "lab12-1.sh" with a path of "~/work/study/2021-2022/Операционные системы/os-intro/labs/lab12". The terminal has several tabs: "report.md", "presentation.md", "lab12-1.sh" (active), "lab12-2.sh", and "lab12.sh". The command history shows: 1 `#!/bin/bash`, 2 `cd /usr/share/man/man1`, and 3 `less $1*`. The status bar at the bottom indicates "sh", "Ширина табуляции: 8", "Стр 3, Стлб 9", and "ВСТ".

Рис. 2.2: Текст программы



### 3 (рис. 3.1)

```
[dvdolenko@fedora lab12]$ touch lab12.sh lab12-1.sh lab12-2.sh
[dvdolenko@fedora lab12]$ chmod +x *.sh
[dvdolenko@fedora lab12]$ ./lab12.sh
lock
work
work
work
work
work
work
work
[dvdolenko@fedora lab12]$ ./lab12-1.sh less
[dvdolenko@fedora lab12]$ ./lab12-2.sh

10 random words:
ccbcggcfc
1
cbbbcddbc
2
jhbcfdgcb
3
```

Рис. 3.1: Создание файлов, наделение их необходимыми правами и запуск.



## 4 Вывод

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

## 5 Ответы на контрольные вопросы:

1. Команда `getopts` является встроенной командой командной оболочки `bash`, предназначенной для разбора параметров сценариев. Она обрабатывает исключительно однобуквенные параметры как с аргументами, так и без них и этого вполне достаточно для передачи сценариям любых входных данных.
2. При генерации имен используют метасимволы:  
произвольная (возможно пустая) последовательность символов; `?` один произвольный символ; `[...]` любой из символов, указанных в скобках перечислением и/или с указанием диапазона; `cat f*` выдаст все файлы каталога, начинающиеся с `"f"`; `cat f` выдаст все файлы, содержащие `"f"`; `cat program.?` выдаст файлы данного каталога с однобуквенными расширениями, скажем `"program.c"` и `"program.o"`, но не выдаст `"program.com"`; `cat [a-d]*` выдаст файлы, которые начинаются с `"a"`, `"b"`, `"c"`, `"d"`. Аналогичный эффект дадут и команды `"cat [abcd]"` и `"cat [bdac]"`.
3. Операторы `&&` и `||` являются управляющими операторами. Если в командной строке стоит `command1 && command2`, то `command2` выполняется в том, и только в том случае, если статус выхода из команды `command1` равен нулю, что говорит об успешном ее завершении. Аналогично, если командная строка имеет вид `command1 || command2`, то команда `command2` выполняется тогда, и только тогда, когда статус выхода из команды `command1` отличен от нуля.

4. Оператор `break` завершает выполнение ближайшего включающего цикла или условного оператора, в котором он отображается.
5. Команда `true` всегда возвращает ноль в качестве выходного статуса для индикации успеха. Команда `false` всегда возвращает не-ноль в качестве выходного статуса для индикации неудачи. Во всех управляющих конструкциях в качестве логического значения используется код возврата из программы, указанной в качестве условия. Код возврата 0 – истина, любое другое значение – ложь. Программа `true` – всегда завершается с кодом 0, `false` – всегда завершается с кодом 1.
6. Введенная строка означает условие существования файла `mans/i.$s`
7. Цикл `While` выполняется до тех пор, пока указанное в нем условие истинно. Когда указанное условие становится ложным - цикл завершается. Цикл `Until` выполняется до тех пор, пока указанное в нем условие ложно.