

Презентация к 14 лабораторной работе

операционные системы

Четвергова М.В.

10 мая 2024

Российский университет дружбы народов, Москва, Россия

- Четвергова Мария Викторовна
- Студентка 1 курса НПИбд-02-23
- Российский университет дружбы народов
- 1132232886@pfur.ru

Цель работы

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

Задание

Выполнить 3 задания, направленных на создание скриптовых файлов на языке `bash`.

Выполнение лабораторной работы

Задание 1

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.

Задание 1

```
foot
[mvchetvergova@mvchetvergova ~]$ cd os-intro/labs/lab14
[mvchetvergova@mvchetvergova lab14]$ ls
presentation  report
[mvchetvergova@mvchetvergova lab14]$ touch task1.sh
[mvchetvergova@mvchetvergova lab14]$ chmod +x task1.sh
[mvchetvergova@mvchetvergova lab14]$ bash task1.sh
```

Рис. 1: Создание и настройка доступа необходимых файлов для задания

Задание 1

```
foot
GNU nano 7.2                                     task1.sh
#!/bin/bash

lockfile="./lock.file"
exec {fn}>${lockfile}

while test -f "$lockfile"
do
if flock -n ${fn}
then
    echo "File is blocked"
    sleep 5
    echo "File is unlocked"
    flock -u ${fn}
else
    echo "File is blocked"
    sleep 5
fi
done
```

Рис. 2: Скрипт программы для задания 1

Задание 1

[illegible]

Рис. 3: Работа программы

Задание 2

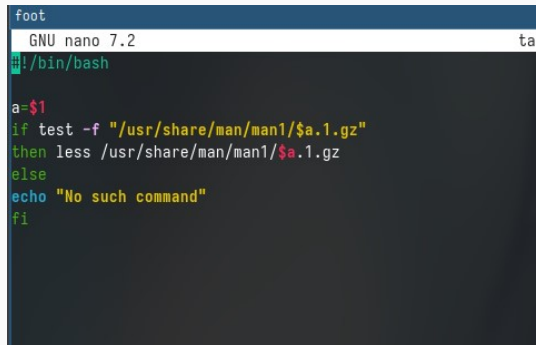
Реализовать команду `man` с помощью командного файла. Изучите содержимое каталога `/usr/share/man/man1`. В нем находятся архивы текстовых файлов, содержащих справку по большинству установленных в системе программ и команд. Каждый архив можно открыть командой `less` сразу же просмотрев содержимое справки. 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`.

Задание 2

```
lchvetvergova@lchvetvergova: ~$ ls -ls /usr/share/man/man1
[evchvetvergova@evchvetvergova ~]$ ls /usr/share/man/man1
.:1.gz
'.1.gz'
a2ping.1.gz
abrt.1.gz
abrt-action-analyze-backtrace.1.gz
abrt-action-analyze-c.1.gz
abrt-action-analyze-cpp-locals.1.gz
abrt-action-analyze-oops.1.gz
abrt-action-analyze-python.1.gz
abrt-action-analyze-vmcore.1.gz
abrt-action-analyze-vulnerability.1.gz
abrt-action-analyze-xorg.1.gz
abrt-action-check-oops-for-hw-error.1.gz
abrt-action-find-bodhi-update.1.gz
abrt-action-generate-backtrace.1.gz
abrt-action-generate-core-backtrace.1.gz
abrt-action-list-dsos.1.gz
abrt-action-notify.1.gz
abrt-action-save-package-data.1.gz
abrt-action-trim-files.1.gz
abrt-applet.1.gz
abrt-auto-reporting.1.gz
abrt-bodhi.1.gz
abrt-cli.1.gz
abrt-dump-journal-core.1.gz
abrt-dump-journal-oops.1.gz
abrt-dump-journal-xorg.1.gz
abrt-dump-oops.1.gz
abrt-dump-xorg.1.gz
abrt-handle-upload.1.gz
abrt-harvest-pstoreoops.1.gz
abrt-harvest-vmcore.1.gz
abrt-merge-pstoreoops.1.gz
abrt-server.1.gz
abrt-watch-log.1.gz
ac.1.gz
aconnect.1.gz
acyclic.1.gz
addr2line.1.gz
afa2afa.1.gz
afa2pi.1.gz
afa2tfa.1.gz
albatross.1.gz
aleph.1.gz
mtx-unicode.1.gz
mtx-unzip.1.gz
mtx-update.1.gz
mtx-vscode.1.gz
mtx-watch.1.gz
mtx-youless.1.gz
atype.1.gz
aupdf.1.gz
ausixflr.1.gz
ausixtex.1.gz
autagen-inspect.1.gz
autagen-pony.1.gz
autool.1.gz
av.1.gz
avxattr.1.gz
wzip.1.gz
named-checkzone.1.gz
named-compilezone.1.gz
named-nzd2nzf.1.gz
name1.1.gz
nano.1.gz
nc.1.gz
ncat.1.gz
ndctl.1.gz
ndctl-activate-firmware.1.gz
ndctl-check-labels.1.gz
ndctl-check-namespace.1.gz
ndctl-clear-errors.1.gz
ndctl-create-namespace.1.gz
ndctl-destroy-namespace.1.gz
ndctl-disable-dimm.1.gz
ndctl-disable-namespace.1.gz
ndctl-disable-region.1.gz
ndctl-enable-dimm.1.gz
ndctl-enable-namespace.1.gz
ndctl-enable-region.1.gz
ndctl-freeze-security.1.gz
ndctl-init-labels.1.gz
ndctl-inject-error.1.gz
ndctl-inject-smart.1.gz
ndctl-list.1.gz
ndctl-load-keys.1.gz
ndctl-monitor.1.gz
ndctl-read-infoblock.1.gz
```

Рис. 4: Создание и настройка доступа необходимых файлов для задания

Задание 2



The image shows a terminal window with a dark background. At the top, a blue header bar contains the text 'foot'. Below it, a white bar shows 'GNU nano 7.2' on the left and 'ta' on the right. The main area of the terminal is dark and contains a script written in a light green monospaced font. The script starts with a shebang line, followed by an assignment, an if-then-else block, and ends with a 'fi' statement.

```
#!/bin/bash

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

Рис. 5: Скрипт программы для задания 2

Задание 2

```
foot
ESC[4mLSESC[24m(1)                                User Commands
ESC[4mLSESC[24m(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 specified.

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

ESC[1m-eESC[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 ..

ESC[1m--authorESC[0m
with ESC[1m-lESC[22m, print the author of each file

ESC[1m-bESC[22m, ESC[1m--escapeESC[0m
print C-style escapes for nongraphic characters

ESC[1m--block-sizeESC[22m=ESC[4mSIZEESC[0m
with ESC[1m-lESC[22m, scale sizes by SIZE when printing them; e.g., '--block-size=M'; see SIZE format below

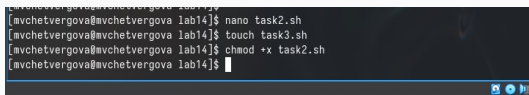
ESC[1m-BESC[22m, ESC[1m--ignore-backupESC[0m
do not list implied entries ending with ~

ESC[1m-c ESC[22mwith ESC[1m-lESC[22m: sort by, and show, ctime (time of last change of file status information); with ESC[1m-lESC[22m: show
ctime and sort by name; otherwise: sort by ctime, newest first

ESC[1m-C ESC[22mlist entries by columns
```

Рис. 6: Работа программы

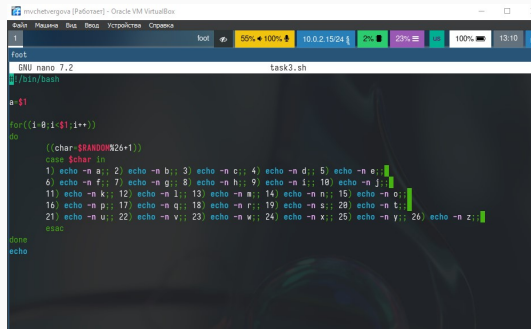
3. Используя встроенную переменную \$RANDOM, напишите командный файл, генерирующий случайную последовательность букв латинского алфавита. Учтите, что \$RANDOM выдаёт псевдослучайные числа в диапазоне от 0 до 32767.

A terminal window with a dark background and light blue text. The prompt is [mvchetvergova@mvchetvergova lab14]\$. The commands entered are: nano task2.sh, touch task3.sh, and chmod +x task2.sh. The cursor is at the end of the last command line.

```
[mvchetvergova@mvchetvergova lab14]$ nano task2.sh
[mvchetvergova@mvchetvergova lab14]$ touch task3.sh
[mvchetvergova@mvchetvergova lab14]$ chmod +x task2.sh
[mvchetvergova@mvchetvergova lab14]$
```

Рис. 7: Создание и настройка доступа необходимых файлов для задания

Задание 3



```
mvchetyergova [Работаю] - Oracle VM VirtualBox
Файл Машина Вид Ввод Устройство Справка
Foot 55% 100% 10.0.2.15/24 2% 23% us 100% 13:10
GNU nano 7.2 task3.sh
#!/bin/bash
a=$1
for((i=0;i<$1;i++))
do
  ((char=$((RANDOM%26+1)))
  case $char in
    1) echo -n a;; 2) echo -n b;; 3) echo -n c;; 4) echo -n d;; 5) echo -n e;;
    6) echo -n f;; 7) echo -n g;; 8) echo -n h;; 9) echo -n i;; 10) echo -n j;;
    11) echo -n k;; 12) echo -n l;; 13) echo -n m;; 14) echo -n n;; 15) echo -n o;;
    16) echo -n p;; 17) echo -n q;; 18) echo -n r;; 19) echo -n s;; 20) echo -n t;;
    21) echo -n u;; 22) echo -n v;; 23) echo -n w;; 24) echo -n x;; 25) echo -n y;; 26) echo -n z;;
    esac
  done
  echo
```

Рис. 8: Скрипт программы для задания 3

Задание 3

```
[mvchetvergova@mvchetvergova lab14]$ chmod +x task3.sh
[mvchetvergova@mvchetvergova lab14]$ nano task3.sh
[mvchetvergova@mvchetvergova lab14]$ bash task3.sh 4
1knu
[mvchetvergova@mvchetvergova lab14]$ bash task3.sh 10
ylbfavvvec
[mvchetvergova@mvchetvergova lab14]$ bash task3.sh 132
htasxexzhvydfnulgvsedprqryqadavpzwltqitahgziltaihrubwylvcifnwgumnpggdzhosnvnbdazyzfvzowuugmogzdqljggpciarntfvda]hw
vkkakqgbnkookww
[mvchetvergova@mvchetvergova lab14]$
```

Рис. 9: Работа программы

Выводы

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