

Программирование в командном процессоре ОС UNIX.

Расширенное программирование

Ыбырай Роза

Содержание

Цель работы	1
Задание	1
Выполнение лабораторной работы.....	1
Первое задание	1
Второе задание	3
Третье задание.....	7
Выводы	9

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

Выполнение лабораторной работы

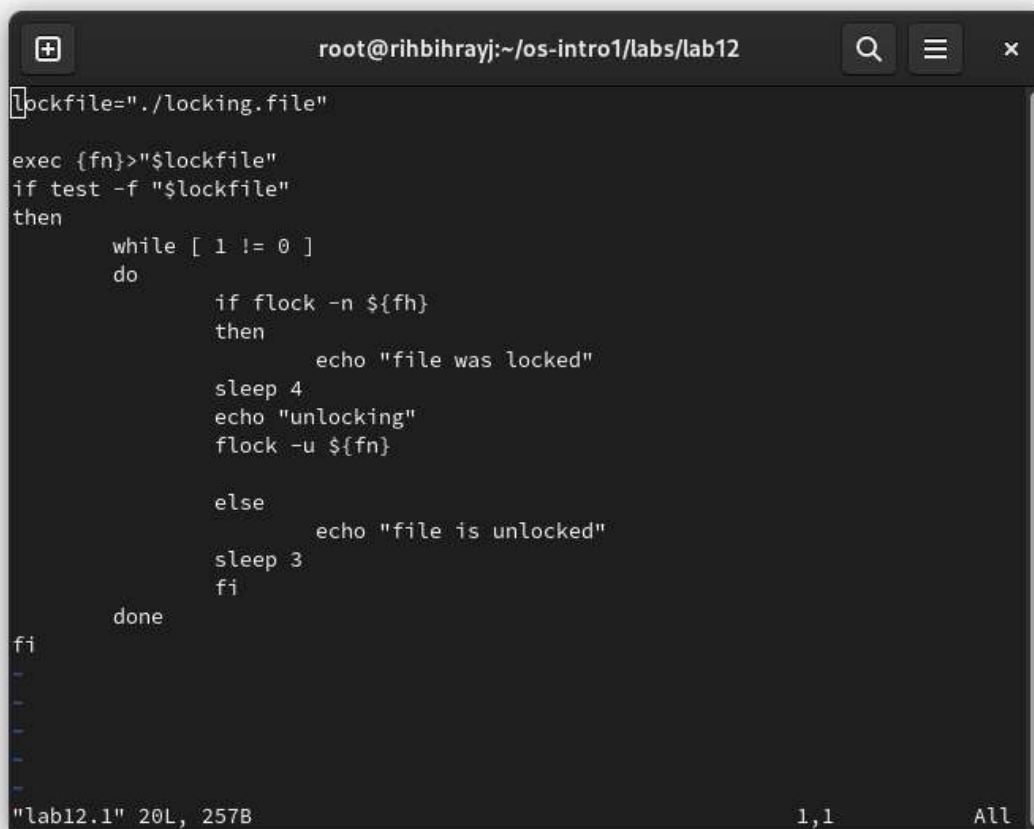
Первое задание

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.

```
[root@rihbihrayj lab12]# vi lab12.1
[root@rihbihrayj lab12]# chmod +x lab12.1
[root@rihbihrayj lab12]# ./lab12.1
flock: requires file descriptor, file or directory
file is unlocked
flock: requires file descriptor, file or directory
file is unlocked
flock: requires file descriptor, file or directory
file is unlocked
flock: requires file descriptor, file or directory
file is unlocked
flock: requires file descriptor, file or directory
file is unlocked
```

Рис.1.



```
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 is unlocked"
            sleep 3
            fi
        done
    fi
fi

"lab12.1" 20L, 257B 1,1 All
```

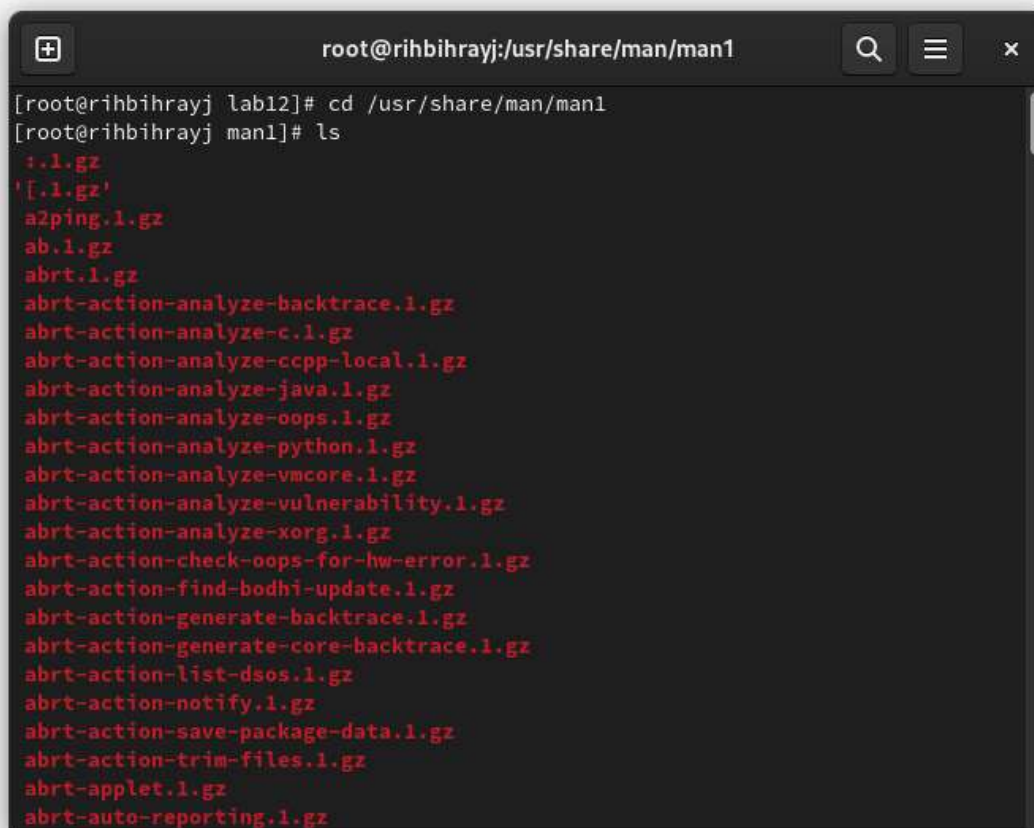
Рис.2.

```
[rozaybyrai@rihbihrayj ~]$ sudo -i
[sudo] пароль для rozaybyrai:
[root@rihbihrayj ~]# cd os-intro1/labs/lab12
[root@rihbihrayj lab12]# ls
lab12.1  locking.file  presentation  report
[root@rihbihrayj lab12]#
```

Рис.3.

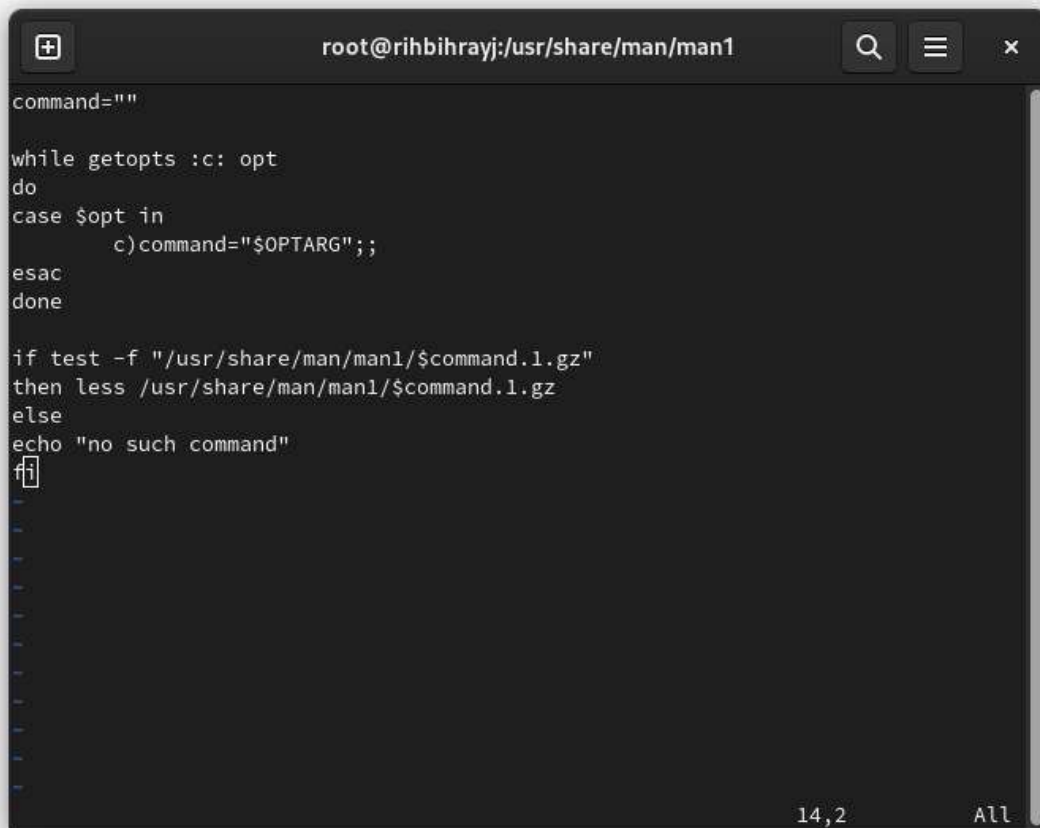
Второе задание

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`.

A terminal window with a dark background and light text. The title bar at the top shows the path 'root@rihbihrayj:/usr/share/man/man1' and standard window controls (search, menu, close). The terminal content shows a sequence of commands and their output. The user enters 'cd /usr/share/man/man1' and then 'ls'. The output of 'ls' is a long list of files, all ending in '.gz', including '1.gz', 'a2ping.1.gz', 'ab.1.gz', 'abrt.1.gz', and various 'abrt-action-' prefixed files like 'abrt-action-analyze-backtrace.1.gz' through 'abrt-action-trim-files.1.gz', as well as 'abrt-applet.1.gz' and 'abrt-auto-reporting.1.gz'.

```
root@rihbihrayj:/usr/share/man/man1
[root@rihbihrayj lab12]# cd /usr/share/man/man1
[root@rihbihrayj man1]# ls
1.gz
a2ping.1.gz
ab.1.gz
abrt.1.gz
abrt-action-analyze-backtrace.1.gz
abrt-action-analyze-c.1.gz
abrt-action-analyze-ccpp-local.1.gz
abrt-action-analyze-java.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
```

Puc.4.



A terminal window with a dark background and light text. The title bar at the top reads 'root@rihbihrayj:/usr/share/man/man1'. The script content is as follows:

```
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
```

The cursor is positioned at the end of the last line. At the bottom right of the terminal, the coordinates '14,2' and the word 'All' are visible.

Puc.5.

```

root@rihbihrayji:/usr/share/man/man1
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-aESC[22m, ESC[1m--allESC[0m
do not ignore entries starting with .

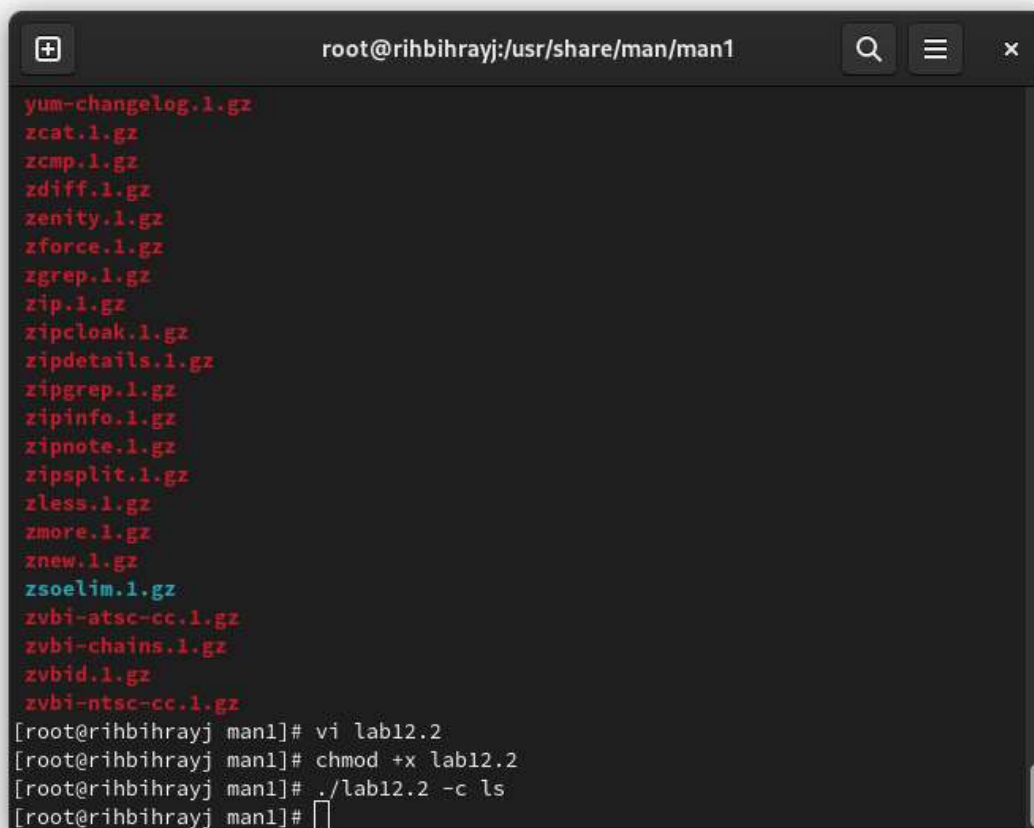
ESC[1m-AESC[22m, ESC[1m--almost-allESC[0m
do not list implied . and ..

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

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

```

Рис.6.

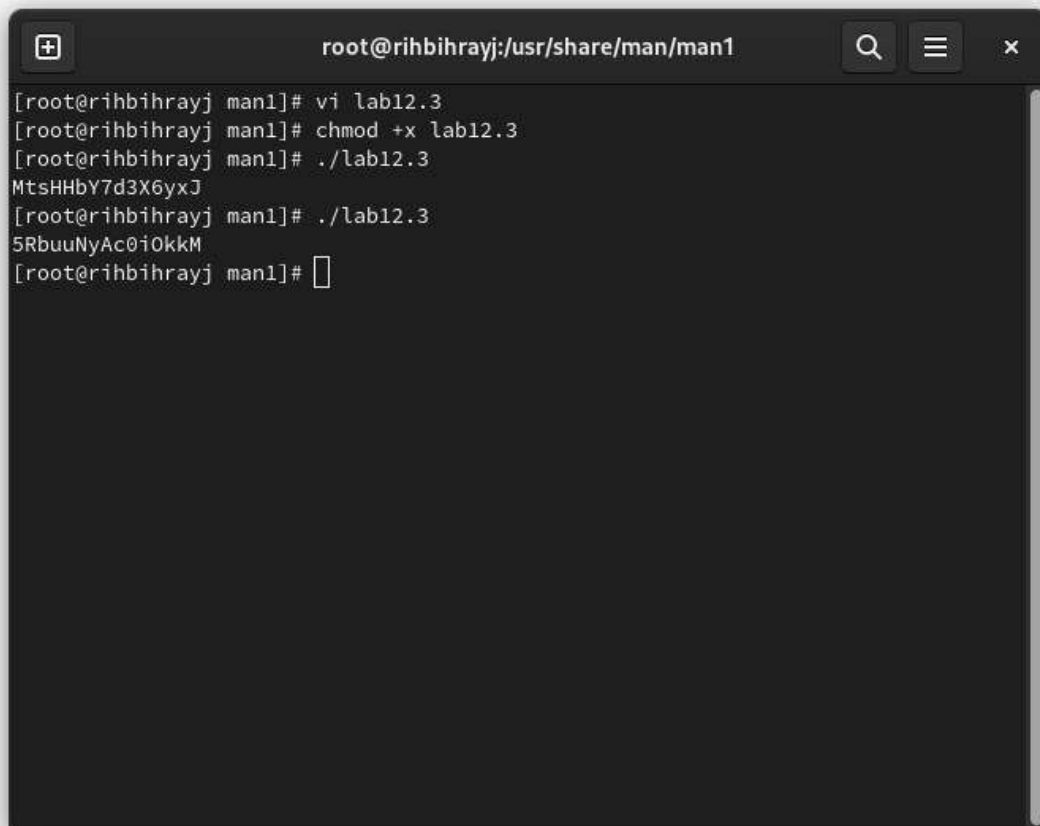
A terminal window with a dark background. The title bar shows 'root@rihbihrayj:/usr/share/man/man1'. The terminal displays a list of files in red text: yum-changelog.1.gz, zcat.1.gz, zcmp.1.gz, zdiff.1.gz, zenity.1.gz, zforce.1.gz, zgrep.1.gz, zip.1.gz, zipcloak.1.gz, zipdetails.1.gz, zipgrep.1.gz, zipinfo.1.gz, zipnote.1.gz, zipsplit.1.gz, zless.1.gz, zmore.1.gz, znew.1.gz, zsoelim.1.gz, zvbi-atsc-cc.1.gz, zvbi-chains.1.gz, zvid.1.gz, and zvbi-ntsc-cc.1.gz. Below the list, four commands are entered in white text: 'vi lab12.2', 'chmod +x lab12.2', './lab12.2 -c ls', and the prompt returns to the shell.

```
root@rihbihrayj:/usr/share/man/man1
yum-changelog.1.gz
zcat.1.gz
zcmp.1.gz
zdiff.1.gz
zenity.1.gz
zforce.1.gz
zgrep.1.gz
zip.1.gz
zipcloak.1.gz
zipdetails.1.gz
zipgrep.1.gz
zipinfo.1.gz
zipnote.1.gz
zipsplit.1.gz
zless.1.gz
zmore.1.gz
znew.1.gz
zsoelim.1.gz
zvbi-atsc-cc.1.gz
zvbi-chains.1.gz
zvid.1.gz
zvbi-ntsc-cc.1.gz
[root@rihbihrayj man1]# vi lab12.2
[root@rihbihrayj man1]# chmod +x lab12.2
[root@rihbihrayj man1]# ./lab12.2 -c ls
[root@rihbihrayj man1]#
```

Рис.7.

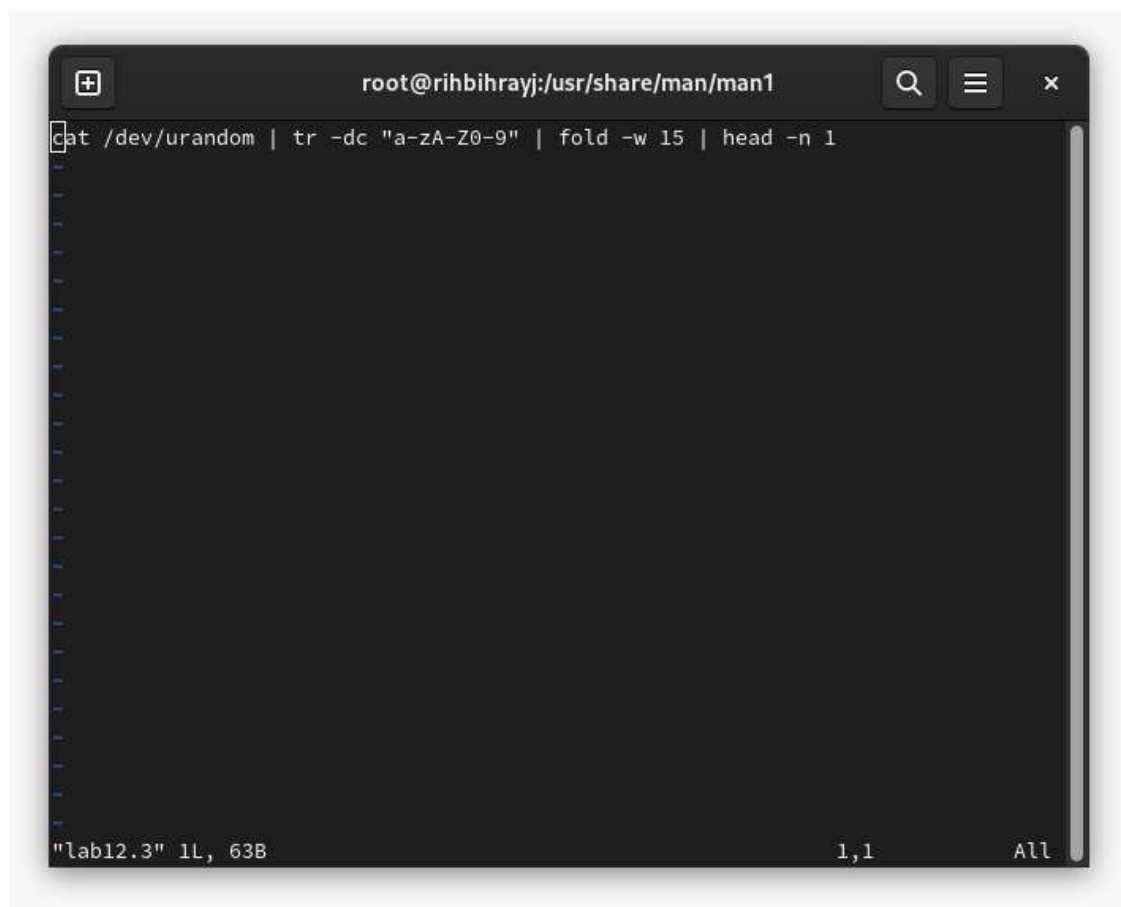
Третье задание

- 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.



```
root@rihbihrayj:/usr/share/man/man1
[root@rihbihrayj man1]# vi lab12.3
[root@rihbihrayj man1]# chmod +x lab12.3
[root@rihbihrayj man1]# ./lab12.3
MtsHHbY7d3X6yxJ
[root@rihbihrayj man1]# ./lab12.3
5RbuuNyAc0i0kkM
[root@rihbihrayj man1]#
```

Puc.8.

A terminal window with a dark background. The title bar shows 'root@rihbihrayj:/usr/share/man/man1'. The command 'cat /dev/urandom | tr -dc "a-zA-Z0-9" | fold -w 15 | head -n 1' is entered at the prompt. The terminal is mostly empty, with a vertical scrollbar on the right. At the bottom, status information reads '"lab12.3" 1L, 63B', '1,1', and 'All'.

```
root@rihbihrayj:/usr/share/man/man1
cat /dev/urandom | tr -dc "a-zA-Z0-9" | fold -w 15 | head -n 1

"lab12.3" 1L, 63B 1,1 All
```

Рис.9.

Выводы

В процессе выполнения данной лабораторной работы научились писать более сложные командные файлы с использованием логических управляющих конструкций и циклов.

...