

# **Лабораторная работа №14**

**Дисциплина: Операционные системы**

Жибицкая Евгения Дмитриевна

# Содержание

<b>1</b>	<b>Цель работы</b>	<b>5</b>
<b>2</b>	<b>Задание</b>	<b>6</b>
<b>3</b>	<b>Выполнение лабораторной работы</b>	<b>8</b>
<b>4</b>	<b>Контрольные вопросы</b>	<b>14</b>
<b>5</b>	<b>Выводы</b>	<b>16</b>

## Список иллюстраций

3.1	Создание файла 14-1.sh . . . . .	8
3.2	Программа 1 . . . . .	9
3.3	Запуск файла 14-1.sh . . . . .	10
3.4	Создание файла 14-2.sh . . . . .	10
3.5	Программа 2 . . . . .	11
3.6	Результат запуска файла 14-2.sh . . . . .	12
3.7	Файл 14-3.sh . . . . .	12
3.8	Запуск файла 14-3.sh . . . . .	13

## **Список таблиц**

# 1 Цель работы

Изучение основ программирования в оболочке ОС UNIX. Приобретение практических навыков в написании более сложных командных файлов с использованием логических управляющих конструкций и циклов.

## 2 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.
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`.
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.

### 3 Выполнение лабораторной работы

Для написания программы необходимо создать файл, открыть его в любом редакторе(желательно emacs, для удобной работы с подсветкой кода), наделить правом на исполнение(рис. 3.1).

```
edzhibitskaya@edzhibitskaya ~]$ touch 14-1.sh
edzhibitskaya@edzhibitskaya ~]$ gedit 14-1.sh
edzhibitskaya@edzhibitskaya ~]$ chmod +x 14-1.sh
edzhibitskaya@edzhibitskaya ~]$ ./14-1.sh
```

Рис. 3.1: Создание файла 14-1.sh

Далее реализуем код для первого задания - освобождения ресурса, выдавая об этом сообщение, также выдавать 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. Иметь возможность взаимодействия трёх и более процессов. (рис. 3.2).



```
#!/bin/bash
= "./lock.file"
exec {fn} >$lockfile

while t -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. 3.2: Программа 1

```
#!/bin/bash
= "./lock.file"
exec {fn} >$lockfile

while t -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
```

Запустим программу(./14-1.sh), посмотрим на результат (рис. 3.3).

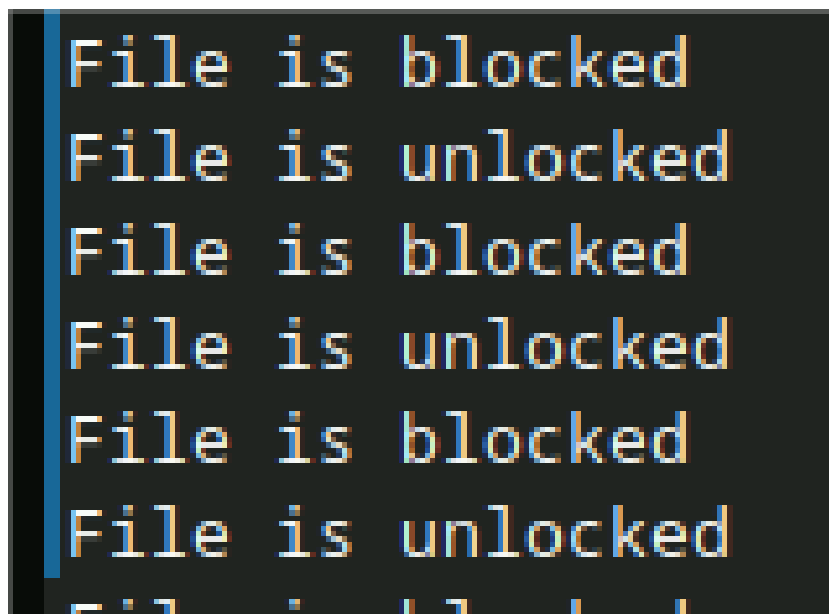


Рис. 3.3: Запуск файла 14-1.sh

Перейдем к реализации второй программы. Также создадим файл(рис. 3.4) и напомним сам код - команда `man` с использованием архивных файлов (рис. 3.5).

```
[edzhibitskaya@edzhibitskaya ~]$ touch 14-2.sh
[edzhibitskaya@edzhibitskaya ~]$ chmod +x 14-2.sh
[edzhibitskaya@edzhibitskaya ~]$ emacs 14-2.sh
[edzhibitskaya@edzhibitskaya ~]$ gedit 14-2.sh
[edzhibitskaya@edzhibitskaya ~]$ emacs 14-2.sh
[edzhibitskaya@edzhibitskaya ~]$ ./14-2.sh ls
```

Рис. 3.4: Создание файла 14-2.sh

```
#!/bin/bash
x=$1
if [ -f /usr/share/man/man1/$x.1.gz ]
then
    gunzip -c /usr/share/man/man1/$1.1.gz | less
else
    echo "No information about this command"
fi
```

Рис. 3.5: Программа 2

```
#!/bin/bash
x=$1
if [ -f /usr/share/man/man1/$x.1.gz ]
then
    gunzip -c /usr/share/man/man1/$1.1.gz | less
else
    echo "No information about this command"
fi
```

Посмотрим на вывод программы - man ls (рис. 3.6).

```
foot
.\\" DO NOT MODIFY THIS FILE! It was generated by help2man 1.48.5.
.TH LS "1" "January 2024" "GNU coreutils 9.3" "User Commands"
.SH NAME
ls \- list directory contents
.SH SYNOPSIS
.B ls
[\fI\,OPTION\|\fR]... [\fI\,FILE\|\fR]...
.SH DESCRIPTION
.\" Add any additional description here
.PP
List information about the FILEs (the current directory by default).
Sort entries alphabetically if none of \fB\--cftuvSUX\|fR nor \fB\--sort\|fR is specified.
.PP
Mandatory arguments to long options are mandatory for short options too.
.TP
\fB\--a\|fR, \fB\--all\|fR
do not ignore entries starting with .
.TP
\fB\--A\|fR, \fB\--almost-all\|fR
do not list implied . and ..
.TP
\fB\--author\|fR
with \fB\--l\|fR, print the author of each file
.TP
\fB\--b\|fR, \fB\--escape\|fR
print C\--style escapes for nongraphic characters
.TP
\fB\--block-size\|fR=\fI\,SIZE\|\fR
with \fB\--l\|fR, scale sizes by SIZE when printing them;
e.g., '\fB\--block-size=M\|fR'; see SIZE format below
.TP
\fB\--B\|fR, \fB\--ignore-backups\|fR
do not list implied entries ending with ~
.TP
:
█
```

Рис. 3.6: Результат запуска файла 14-2.sh

Далее напишем скрипт для 3 программы. Также создадим файл и напишем код - генерация случайной последовательности букв латинского алфавита (рис. 3.7).

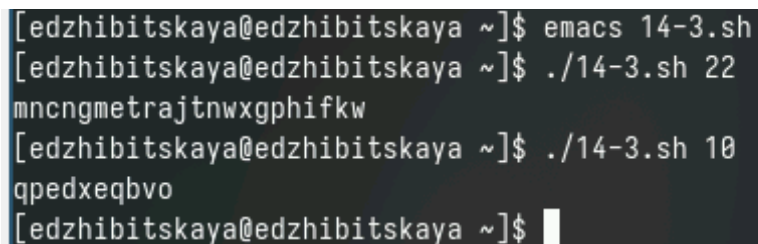
```
foot
[edzhbitskaya@edzhbitskaya ~]$ touch 14-3.sh
[edzhbitskaya@edzhbitskaya ~]$ chmod +x 14-3.sh
[edzhbitskaya@edzhbitskaya ~]$ emacs 14-3.sh

14-3.sh - GNU Emacs at edzhbitskaya
File Edit Options Buffers Tools Sh-Script Help
[Icons] Save Undo [Icons] C
#!/bin/bash
x=$1
for ((i=0; i<x; i++))
do
  ((char=$((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. 3.7: Файл 14-3.sh

```
#!/bin/bash
x=$1
for ((i=0; i<x; i++))
do
    ((char=$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 y;;
    esac
done
echo
```

Запустим файл, проверим корректность вывода (рис. 3.8).



```
[edzhibitskaya@edzhibitskaya ~]$ emacs 14-3.sh
[edzhibitskaya@edzhibitskaya ~]$ ./14-3.sh 22
mncngmetrajtnwxgphifkw
[edzhibitskaya@edzhibitskaya ~]$ ./14-3.sh 10
qpexeqbvo
[edzhibitskaya@edzhibitskaya ~]$
```

Рис. 3.8: Запуск файла 14-3.sh

## 4 Контрольные вопросы

1. Необходимо использовать двойные квадратные скобки для условия в цикле while.

```
while [[ $1 != "exit" ]]
```

2. Для объединения нескольких строк в одну можно использовать оператор конкатенации в bash - символ +
3. Seq возвращает последовательность чисел, можно реализовать используя цикл for или питоновский range() при программировании на bash.
4. Выражение  $\$(10/3)$  даст результат 3, так как в bash целочисленное деление возвращает только целую часть от деления.
5. Основные отличия zsh от bash включают в себя более продвинутый и гибкий автодополнитель, расширенные возможности настройки интерфейса и мощные встроенные функции. Знание bash хорошо переносится на zsh, но zsh имеет больше возможностей.
6. Синтаксис конструкции for ((a=1; a <= LIMIT; a++)) верен, это цикл for с заданным началом, пределом и шагом.
7. Язык bash отличается от других языков программирования, таких как Python или C++, тем что он предназначен для автоматизации задач в командной строке операционной системы. Его преимущества включают встроенные утилиты для обработки текста, удобные средства работы

с файлами и переменными окружения. Недостатки включают ограниченные возможности работы с более сложными структурами данных и ограниченные возможности взаимодействия с графическим интерфейсом.

## 5 Выводы

В ходе работы было изучено программирование в оболочке ОС UNIX. Приобретены навыки в написании более сложных командных файлов с использованием логических управляющих конструкций и циклов, а также реализовано 3 программы в данной оболочке.