

# **Отчёт по лабораторной работе №14**

**Операционные системы**

Бережной Иван Александрович

# 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>7</b>
3.1	Написание первого командного файла . . . . .	7
3.2	Написание второго командного файла . . . . .	9
3.3	Написание третьего командного файла . . . . .	10
<b>4</b>	<b>Выводы</b>	<b>12</b>
	<b>Список литературы</b>	<b>13</b>

# Список иллюстраций

3.1	Первый скрипт . . . . .	8
3.2	Запуск первого скрипта . . . . .	8
3.3	Второй скрипт . . . . .	9
3.4	Запуск второго скрипта . . . . .	9
3.5	Проверка работы скрипта . . . . .	10
3.6	Третий скрипт . . . . .	11
3.7	Запуск третьего скрипта . . . . .	11

## **Список таблиц**

# 1 Цель работы

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

## 2 Задание

1. Написать командный файл, реализующий упрощённый механизм семафоров
2. Реализовать команду `map` с помощью командного файла
3. Написать командный файл, генерирующий случайную последовательность букв латинского алфавита

## **3 Выполнение лабораторной работы**

### **3.1 Написание первого командного файла**

Создадим файл в терминале и присвоим ему право на исполнение. Напишем скрипт, который в течение некоторого времени дожидается освобождения ресурса, выдавая об этом сообщение, а дождавшись его освобождения, использует его в течение некоторого времени, также выдавая информацию о том, что ресурс используется соответствующим командным файлом (рис. 3.1). Запустим скрипт и проверим результат (рис. 3.2).




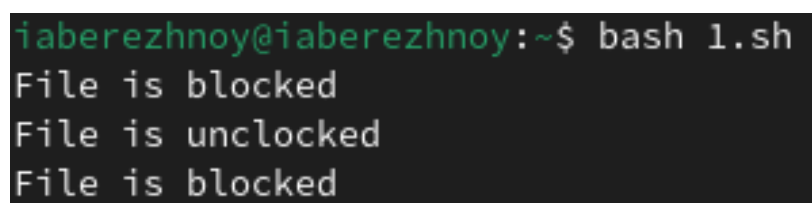
```
Открыть ▾  1.sh  
~/  
  
lockfile="./lock.file"  
exec {fn}>$lockfile  
  
while test -f "$lockfile"  
do  
  if flock -n ${fn}  
  then  
    echo "File is blocked"  
    sleep 5  
    echo "File is unlocked"  
    flock -u ${fn}  
  else  
    echo "File is blocked"  
    sleep 5  
  fi  
done
```

Рис. 3.1: Первый скрипт



```
iaberezhnoy@iaberezhnoy:~$ bash 1.sh  
File is blocked  
File is unlocked  
File is blocked
```

Рис. 3.2: Запуск первого скрипта



## 3.2 Написание второго командного файла

Создадим второй командный файл и также дадим ему право на исполнение. Реализуем команду `man` скриптом, который будет получать в виде аргумента командной строки название команды и в виде результата выдавать справку об этой команде или сообщение об отсутствии справки (рис. 3.3). Запустим файл (рис. 3.4) и получим такой результат (рис. 3.5).

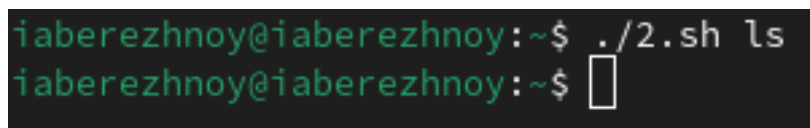


```
Открыть ▾ + 2.sh ~/
1.sh

#!/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 "There is no such command"
fi
```

Рис. 3.3: Второй скрипт



```
iaberezhnoy@iaberezhnoy:~$ ./2.sh ls
iaberezhnoy@iaberezhnoy:~$
```

Рис. 3.4: Запуск второго скрипта

```
ESC[4mL$ESC[24m(1) User Commands ESC[4mL$ESC[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[22mmor ESC[1m--sort ESC[22mis specified.

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

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-backupsESC[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

ESC[1m--colorESC[22m[=ESC[4mWHENESC[24m]
color the output WHEN; more info below

ESC[1m-dESC[22m, ESC[1m--directoryESC[0m
list directories themselves, not their contents

ESC[1m-DESC[22m, ESC[1m--diredESC[0m
generate output designed for Emacs' dired mode

ESC[1m-f ESC[22mlist all entries in directory order
```

Рис. 3.5: Проверка работы скрипта

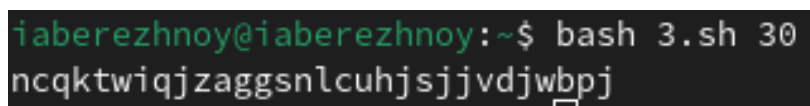
### 3.3 Написание третьего командного файла

Теперь напишем командный файл, генерирующий случайную последовательность букв латинского алфавита (рис. 3.6). Запустим файл (рис. 3.7)



```
Открыть ▾  3.sh  
~/  
  
#!/bin/bash  
  
a=$1  
  
for ((i=0; i<$a; i++))  
do  
    ((char=$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. 3.6: Третий скрипт



```
iaberezhnoy@iaberezhnoy:~$ bash 3.sh 30  
ncqkwtwiqjzaggsnlcuhsjjvdjwbpj
```

Рис. 3.7: Запуск третьего скрипта

## 4 Выводы

В ходе выполнения лабораторной работы мы изучили основы программирования в оболочке ОС UNIX и научились писать более сложные командные файлы с использованием логических управляющих конструкций и циклов.

## **Список литературы**