

Лабораторная работа №12

Дисциплина: операционные системы

пронякова Ольга максимовна

Содержание

1	Цель работы	5
2	Задание	6
3	Выполнение лабораторной работы	8
4	Выводы	13
	Список литературы	14

Список иллюстраций

3.1	Код программы	9
3.2	Выполнение команды	9
3.3	Код программы	10
3.4	Выполнение команды	10
3.5	Выполнение команды	11
3.6	Код программы	11
3.7	Выполнение команды	12

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. 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

Пишу 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). Запускаю 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. Дорабатываю программу так, чтобы имелась возможность взаимодействия трёх и более процессов (рис. 3.1) (рис. 3.2).


```
#!/bin/bash

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

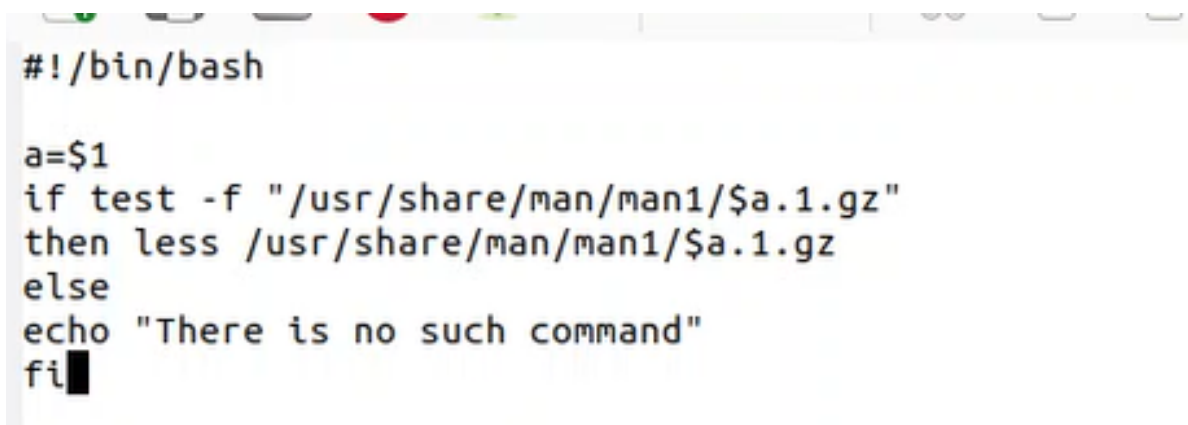
while test -f "$lockfile"
do
if flock -n ${fn}
then
    echo "File is blocked"
    sleep 5
    echo "File is unlocked"
    flock -u ${fn}
else
    echo "file is blocked"
    sleep 5
fi
done
```

Рис. 3.1: Код программы

```
olga@ompronyakova:~$ emacs lab12_1
olga@ompronyakova:~$ bash lab12_1
File is blocked
File is unlocked
File is blocked
File is unlocked
File is blocked
File is unlocked
File is blocked
```

Рис. 3.2: Выполнение команды

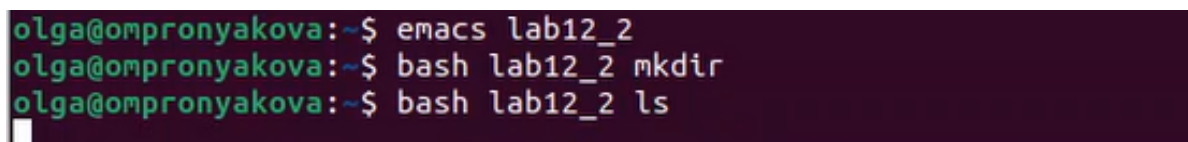
Реализовываю команду `man` с помощью командного файла. Изучаю содержимое каталога `/usr/share/man/man1`. В нем находятся архивы текстовых файлов, содержащих справку по большинству установленных в системе программ и команд. Каждый архив можно открыть командой `less` сразу же просмотрев содержимое справки. 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` (рис. 3.3) (рис. 3.4) (рис. 3.5).

A screenshot of a code editor showing a shell script. The script starts with a shebang line `#!/bin/bash`. It then assigns the first argument to a variable `a` using `a=$1`. An `if` statement checks if a file exists at `/usr/share/man/man1/$a.1.gz`. If it exists, it runs `less /usr/share/man/man1/$a.1.gz`. Otherwise, it echoes the message "There is no such command". The script ends with `fi`.

```
#!/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: Код программы

A screenshot of a terminal window with a dark background. It shows three lines of commands being executed by a user named 'olga' on a machine named 'ompronyakova'. The first line is `emacs lab12_2`, the second is `bash lab12_2 mkdir`, and the third is `bash lab12_2 ls`.

```
olga@ompronyakova:~$ emacs lab12_2
olga@ompronyakova:~$ bash lab12_2 mkdir
olga@ompronyakova:~$ bash lab12_2 ls
```

Рис. 3.4: Выполнение команды

```
olga@ompronyakova:~$ man 1 ls
.\" DO NOT MODIFY THIS FILE! It was generated by help2man 1.47.3.
.TH LS "1" "February 2022" "GNU coreutils 8.32" "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 sp
ecified.
.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
/usr/share/man/man1/ls.1.gz
```

Рис. 3.5: Выполнение команды

Используя встроенную переменную \$RANDOM, пишу командный файл, генерирую- щий случайную последовательность букв латинского алфавита(рис. 3.6) (рис. 3.7).

```
olga@ompronyakova:~$ bash lab12_2 fhfh
There is no such command
olga@ompronyakova:~$
```

Рис. 3.6: Код программы

```
#!/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. 3.7: Выполнение команды

4 Выводы

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

Список литературы

1. Лабораторная работа №12