

Лабораторная работа № 12

Андрианова Марина
Георгиевна
RUDN University, Moscow,
Russian Federation
NEC–2022, 27 May

Цель работы

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

Выполнение лабораторной работы

- 1). Создала файл `s1.sh` и написала соответствующий скрипт: 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.

Задача 1

Проверяем работу написанного скрипта (команда `“./s1.sh 2 5”`), предварительно добавив право на исполнение файла(команда `“chmod +x s1.sh”`). Скрипт работает корректно(рис.1).

```
[mgandrianova@fedora ~]$ chmod +x s1.sh
[1]+  Завершён          emacs
[mgandrianova@fedora ~]$ ./s1.sh 2 5
Ожидание
Ожидание
Выполнение
Выполнение
Выполнение
Выполнение
Выполнение
[mgandrianova@fedora ~]$
```

Рис.1: Проверка скрипта

Изменение скрипта

После этого изменяем скрипт так, чтобы его можно было выполнять в нескольких терминалах. Проверила его работу(команда `“./s1.sh 3 5 Ожидание > /dev/pts/2 &”` и команда `“./s1.sh 3 5 Ожидание > /dev/tty2”`). При этом ни одна из команд не сработала, выводя сообщение “Отказано в доступе”. При этом скрипт работает корректно(рис.2).

```
[mgandrianova@fedora ~]$ chmod +x s1.sh
[1]+  Завершён          emacs
[mgandrianova@fedora ~]$ ./s1.sh 3 5 Ожидание > /dev/pts/2 &
[1] 4032
bash: /dev/pts/2: Отказано в доступе
[1]+  Выход 1           ./s1.sh 3 5 Ожидание > /dev/pts/2
[mgandrianova@fedora ~]$ ./s1.sh 3 5 Ожидание > /dev/pts/1 &
[1] 4041
bash: /dev/pts/1: Отказано в доступе
[1]+  Выход 1           ./s1.sh 3 5 Ожидание > /dev/pts/1
[mgandrianova@fedora ~]$ ./s1.sh 3 5 Ожидание > /dev/tty2
```

Рис.2: Проверка скрипта

2). Реализовала команду `man` с помощью командного файла. Изучила содержимое каталога `/usr/share/man/man1`: сначала перешла в него командой `cd /usr/share/man/man1`, а затем вывела его содержимое (команда `ls`) (рис.3). В нем находятся архивы текстовых файлов, содержащих справку по большинству установленных в системе программ и команд.

Задача 2

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`.

```
[mgandrianova@fedora ~]$ cd /usr/share/man/man1
[mgandrianova@fedora man1]$ ls
:~.1.gz
'~.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-core.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-install-debuginfo.1.gz
abrt-action-list-dsos.1.gz
abrt-action-notify.1.gz
abrt-action-perform-ccpp-analysis.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
```

Рис.3: Переход в каталог и вывод его содержимого

Задача 2

Для данной задачи я создала файл `s2.sh` и написала соответствующий скрипт.

Проверяем работу написанного скрипта (команды `./s2.sh touch` и `./s2.sh rm`), предварительно добавив право на исполнение файла(команда `chmod +x s2.sh`)(рис.4). Скрипт работает корректно(рис.5).

```
[mgandrianova@fedora ~]$ chmod +x s2.sh  
[1]+  Завершён          emacs  
[mgandrianova@fedora ~]$ ./s2.sh touch  
[mgandrianova@fedora ~]$ ./s2.sh rm
```

Рис.4: Предоставление права на исполнение и проверка скрипта

```
mgandrianova@fedora:~ — /bin/bash ./s2.sh touch

.\" DO NOT MODIFY THIS FILE! It was generated by help2man 1.47.3.
.TH TOUCH "1" "March 2022" "GNU coreutils 8.32" "User Commands"
.SH NAME
touch \- change file timestamps
.SH SYNOPSIS
.B touch
[\fI,OPTION]\fR... \fI,FILE\fR...
.SH DESCRIPTION
.\" Add any additional description here
.PP
Update the access and modification times of each FILE to the current time.
.PP
A FILE argument that does not exist is created empty, unless \fB -c\fR or \fB -h\fR
is supplied.
.PP
A FILE argument string of \- is handled specially and causes touch to
change the times of the file associated with standard output.
.PP
Mandatory arguments to long options are mandatory for short options too.
.TP
\fB -a\fR
change only the access time
.TP
\fB -c\fR, \fB -no\fR-create\fR
do not create any files
.TP
\fB -d\fR, \fB -date\fR=\fI,STRING\fR
parse STRING and use it instead of current time
.TP
\fB -f\fR
:
```

Рис.5: Результат после введения команды “./s2.sh touch”

3). Создала файл `s3.sh` и написала соответствующий скрипт. Используя встроенную переменную `$RANDOM`, написала командный файл, генерирующий случайную последовательность букв латинского алфавита. Учтём, что `$RANDOM` выдаёт псевдослучайные числа в диапазоне от 0 до 32767. Добавила право на исполнение файла(команда `“chmod +x s3.sh”`)(рис.6). Скрипт работает корректно(рис.6).

```
[mgandrianova@fedora ~]$ chmod +x s3.sh
[1]+  Завершён      emacs
[mgandrianova@fedora ~]$ ./s3.sh 5
dsxkx
[mgandrianova@fedora ~]$ ./s3.sh 5
tkuio
[mgandrianova@fedora ~]$ ./s3.sh 100
slzurshlbrkwhwsmtjcodfaqdwcnujhvoqrisgcvuaftdkxawfdoomypnmzhkassiidjxqolbrxulcgpueyraymdbdtraqvfw
[mgandrianova@fedora ~]$ ./s3.sh 1
n
[mgandrianova@fedora ~]$ ./s3.sh 1
x
[mgandrianova@fedora ~]$
```

Рис.6: Проверка скрипта

Выводы

Я изучила основы программирования в оболочке ОС UNIX и научилась писать более сложные командные файлы с использованием логических управляющих конструкций и циклов.