

Отчет по лабораторной работе №12

Ничипорова Елена

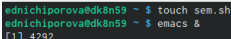
23-05-22

РУДН, Москва

Отчет

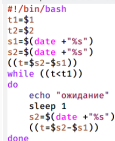
- Изучить основы программирования в оболочке ОС UNIX.
Научиться писать более сложные командные файлы с использованием логических управляющих конструкций и циклов.

1. Написала 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. Для данной задачи я создала новый файл (рис. 1) и написала соответствующий скрипт (рис. 2)



```
ednichiporova@dk8n59 ~ $ touch sem.sh
ednichiporova@dk8n59 ~ $ emacs &
[1] 4292
```

Figure 1: Создание файла



```
#!/bin/bash
t1=$1
t2=$2
s1=$(date +%s)
s2=$(date +%s)
((t=s2-s1))
while ((t<t1))
do
    echo "ожидание"
    sleep 1
    s2=$(date +%s)
    ((t=s2-s1))
done
```

- Далее я проверила работу написанного скрипта, перед этим я предоставила право на исполнение данного файла командой “chmod +x sem.sh”. Скрипт работает корректно(рис. 3)

```
ednichiporova@dk8n59 ~ $ chmod +x sem.sh
ednichiporova@dk8n59 ~ $ ./sem.sh 2 3
ожидание
ожидание
выполнение
выполнение
выполнение
ednichiporova@dk8n59 ~ $
```

Figure 3: Проверка скрипта №1

После этого я изменила скрипт так, чтобы его можно было выполнять в нескольких терминалах (рис. 4) (рис. 5) и проверила его работу, но у меня не получилось проверить скрипт, так как было отказано в доступе (рис. 6)

```
#!/bin/bash
function ogidania
{
    s1=$(date +%s")
    s2=$(date +%s")
    ((t=$s2-$s1))
    while ((t<t1))
    do
        echo "Ожидание"
        sleep 1
        s2=$(date +%s")
        ((t=$s2-$s1))
    done
}
function vipolnenie
{
    s1=$(date +%s")
    s2=$(date +%s")
    ((t=$s2-$s1))
    while ((t<t2))
    do
        echo "Выполнение"
        sleep 1
        s2=$(date +%s")
        ((t=$s2-$s1))
    done
}
t1=$1
t2=$2
command=$3
while true
do
    if [ "$command" == "Выход" ]
    then
        echo "Выход"
        exit 0
    fi
    if [ "$command" == "Ожидание" ]
    then
        ogidania
    fi
    if [ "$command" == "Выполнение" ]
    then
        vipolnenie
    fi
done
```

Figure 4: Измененный скрипт №1

2. Реализовала команду `man` с помощью командного файла. Изучила содержимое каталога `/usr/share/man/man1`. (рис. 7) 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`.

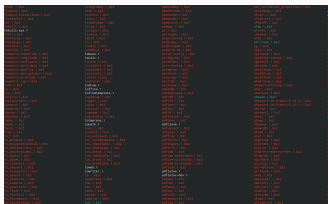


Figure 7: Реализация команды `man`

- Для данной задачи я создала новый файл (рис. 8) и написала соответствующий скрипт(рис. 9)

```
ednichiporova@dkn59 ~ $ touch man.sh  
ednichiporova@dkn59 ~ $ emacs #
```

Figure 8: создание нового файла

```
#!/bin/bash  
a=$1  
if [ -f /usr/share/man/man1/${a}.bz2 ]  
then  
    bunzip2 -c /usr/share/man/man1/${a}.bz2 | less  
else  
    echo "справки по данной команде нет"  
fi
```

Figure 9: скрипт №2

- Проверила работу скрипта, предварительно добавив право на исполнение файла. Скрипт работает корректно (рис. 10) (рис. 11) (рис. 12) (рис. 13)

```
ednichiporova@dk8n59 ~ $ ./man.sh mkdir
ednichiporova@dk8n59 ~ $
ednichiporova@dk8n59 ~ $ ./man.sh rm
ednichiporova@dk8n59 ~ $ ./man.sh cat
```

Figure 10: проверка работы скрипта №2

```
.. DO NOT MODIFY THIS FILE! It was generated by helpgen 1.10.0
.. man.sh v1.0 "man.sh: man page generator"
.. NAME
.. man.sh - make directories
.. SYNOPSIS
.. man.sh
.. FILE OPTIONS/PRE... \FILE_DIRECTORY/VFILE...
.. DESCRIPTION
.. V- Add any additional description here.
..
.. Create the DIRECTORY(ies), if they do not already exist.
..
.. Mandatory arguments to long options are mandatory for short options too.
..
.. FILE-OPTION, \FILE-Mode[FILE-Mode]/FILE
.. and file mode (as in chmod), not syntax \-umask.
..
.. FILE-OPTION, \FILE-Context[FILE-Context]/FILE
.. no error if existing, make parent directories as needed
..
.. FILE-OPTION, \FILE-Verbs[FILE-Verbs]
.. print a message for each created directory
..
.. FILE-OPTION
.. set SELinux security context of each created directory
.. to the default type
..
.. FILE-OPTION[FILE-Context]/FILE
.. like FILE-OPTION, or if FILE is specified then set the SELinux
.. or SMACK security context to FILE
..
.. FILE-OPTION
.. display this help and exit
..
.. FILE-OPTION
.. output version information and exit
..
.. AUTHOR
.. Written by David Kennerly.
..
.. "REPORTING BUGS"
.. Get coreutils online help: <https://www.gnu.org/software/coreutils/>
..
.. Report any translation bugs to <https://translationproject.org/team/>
..
.. "SEE ALSO"
.. mkdir(1)
..
.. FULL DOCUMENTATION <https://www.gnu.org/software/coreutils/mkdir>
..
.. or available locally via: info '(man coreutils) mkdir invocation'
..
.. Packaged by section (8.10-1) (p40)
..
```

Выполнение

```
% DO NOT MODIFY THIS FILE! It was generated by helpman 1.47.3.
% On "08_1996 March 28th" Joe Caramazza & JZ "User Commands"
% =====
r/v/r - remove files or directories
rm/rmP/rmPS/rmPSI/rmPSIS/rmPSISL
    r      : recursive
    v/r   : option(s)/file(s)
    rm     : remove file(s)
    rmP/rmPS/rmPSI/rmPSIS/rmPSISL : same as rm except that it removes each specified file. By default, it does not remove directories.
    If the VPA-V/R or VPA/V-interactively-remove/r option is given,
    there are more than three files or the VPA-V/R, VPA-V/R/,
    or VPA/V-recursive/r are given, then
        r prompts the user for whether to proceed with the entire operation. If
the response is not affirmative, the entire command is aborted.
    Otherwise, if the file is unwritable, standard input is a terminal, and
VPA-V-R or VPA/V-force/r option is not given, or the
VPA-V/R/ or VPA/V-interactively-keep/r option is given,
        r prompts the user for whether to remove the file. If the response is
not affirmative, the file is skipped.
% =====
rm OPTIONS
Remove (unlink) the FILE(s).
?r
VPA-V/R, VPA/V-force/r
Unlink nonexistent files and arguments, never prompt.
VPA-V/R
prompt before every removal.
?r
VPA-V/R
Prompt before removing more than three files, or
when removing recursively; less intrusive than VPA-/R/,
while still giving protection against most mistakes
?r
VPA-V-interactively-keep/r, VPA/V-R, VPA/V
prompt according to WHEN never, once (VPA-V/R), or
always (VPA-V/R/); without menu, prompt always.
```

[illegible]

3. Используя встроенную переменную \$RANDOM, написала командный файл, генерирующий случайную последовательность букв латинского алфавита. Для этого создала новый файл и написала соответствующий скрипт (рис. 14)

```
#!/bin/bash
n=$1
for (( i=0; i< $n; i++ ))
do
    (( char=$((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
```

Figure 14: скрипт №3

- Проверила работу данного скрипта, предварительно добавив право на исполнение. Скрипт работает корректно (рис. 15)

```
ednichiporova@dk8n59 ~ $ chmod +x random.sh
ednichiporova@dk8n59 ~ $ ./random.sh 5
wzqdu
ednichiporova@dk8n59 ~ $ ./random.sh 12
vbelpzcoqycf
```

Figure 15: проверка работы скрипта №3

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