

Лабораторная работа 12

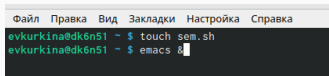
Куркина Евгения Вячеславовна

¹RUDN University, Moscow, Russian Federation

Лабораторная работа 12

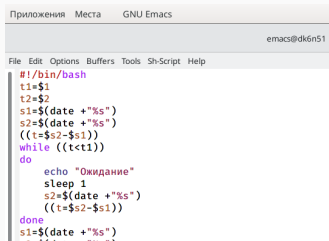
Цель данной лабораторной работы — Изучить основы программирования в оболочке ОС UNIX. Научиться писать более сложные командные файлы с использованием логических управляющих конструкций и циклов.

Создала отдельный файл sem.sh(рис. 1). Написала командный файл, который реализует упрощенный механизм семафоров. Команда должна дожидаться освобождения ресурса, выдавая об этом сообщение, а затем использовать его в течении некоторого времени, также выдавая информацию об этом (рис. 2)



```
Файл  Правка  Вид  Закладки  Настройка  Справка
evkurkina@dk6n51 ~ $ touch sem.sh
evkurkina@dk6n51 ~ $ emacs &
```

Figure 1: Создание файла, переход в emacs



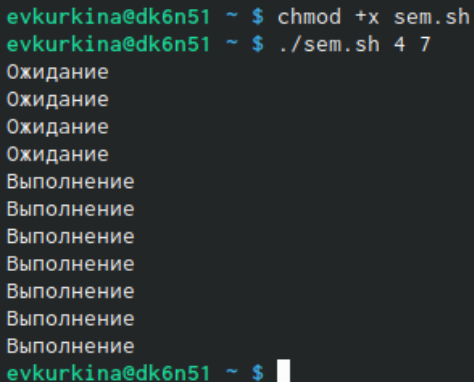
```
Приложения  Места  GNU Emacs
emacs@dk6n51

File  Edit  Options  Buffers  Tools  Sh-Script  Help

#!/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
s1=$(date +%s)
```

Шаг 1.2

Затем дала права на выполнение, а затем проверила результат работы файла (рис. 3).

A terminal window with a dark background and light green text. The prompt is 'evkurkina@dk6n51 ~ \$'. The first command is 'chmod +x sem.sh'. The second command is './sem.sh 4 7'. The output consists of eight lines: four 'Ожидание' (Waiting) and four 'Выполнение' (Execution), alternating. The prompt returns at the end.

```
evkurkina@dk6n51 ~ $ chmod +x sem.sh
evkurkina@dk6n51 ~ $ ./sem.sh 4 7
Ожидание
Ожидание
Ожидание
Ожидание
Выполнение
Выполнение
Выполнение
Выполнение
Выполнение
Выполнение
evkurkina@dk6n51 ~ $
```

Figure 3: Права на выполнение и результат работы файла

Изменила текст скрипта так, чтобы его можно было выполнять сразу в нескольких терминалах (рис. 4)(рис. 5).




```
File Edit Options Buffers Tools Sh-Script Help
#!/bin/bash
function ogidanie
{
    t1=$1
    t2=$2
    s1=$(date +%s)
    s2=$(date +%s)
    ((t=$2-$s1))
    while ((t<t1))
    do
        echo "Ожидание"
        sleep 1
        s2=$(date +%s)
        ((t=$2-$s1))
    done
}
function vypolnenie
{
    s1=$(date +%s)
    s2=$(date +%s)
    ((t=$2-$s1))
    while ((t<t2))
    do
        echo "Выполнение"
        sleep 1
        s2=$(date +%s)
        ((t=$2-$s1))
    done
}
t1=$1
t2=$2
command=$3
while true
do
    if [ "$command" == "Выход" ]
    then
        echo "Выход"
        exit 0
    fi
    if [ "$command" == "Ожидание" ]
    then
        ogidanie
```

Figure 4: Измененный текст скрипта ч1



```
fi
if [ "$command" == "Ожидание" ]
then ogidanie
fi
```

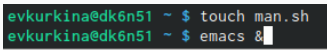
Затем проверила его работу (нет прав доступа для данной команды) (рис. 6).



```
evkurkina@dk6n51 ~ $ ./sem.sh 2 3 Ожидание > /dev/pts/1 &  
[2] 12528  
evkurkina@dk6n51 ~ $ bash: /dev/pts/1: Отказано в доступе  
█
```

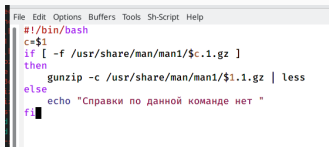
Figure 6: Проверка командного файла

Далее создала файл `man.sh` (рис. 9). Написала текст скрипта, который получает в виде аргумента название команды, и выдавать справку об этой команде или сообщение о том, что справка отсутствует (рис. 10).



```
evkurkina@dk6n51 ~ $ touch man.sh
evkurkina@dk6n51 ~ $ emacs &
```

Figure 9: Создание нового файла и переход в emacs



```
File Edit Options Buffers Tools Sh-Script Help
#!/bin/bash
c=$1
if [ -f /usr/share/man/man1/${c}.1.gz ]
then
  gunzip -c /usr/share/man/man1/${c}.1.gz | less
else
  echo "Справки по данной команде нет "
fi
```

Figure 10: Текст командного файла

Затем дала права на исполнение и проверила работу командного файла, получив справку о команде `rm` (рис. 11) (рис. 12).

```
evkurkina@edk6n51 ~ $ chmod +x man.sh
evkurkina@edk6n51 ~ $ ./man.sh rm
```

Figure 11: Права доступа, команды проверки файла

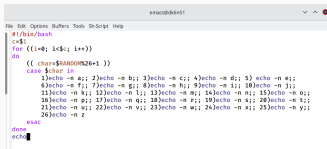
```
GNU rm: This manual page is not for the rm command.
It was generated by help2man 1.47.3.
To see "rm" "March 2020" GNU coreutils 8.32 "User Commands"
in NAME
rm -V, remove files or directories
in SYNOPSIS
rm [OPTION]... FILE...
in DESCRIPTION
This manual page
documents the GNU version of
rm.
rm
Removes each specified file. By default, it does not remove
directories.
If the -V/-V or -V/-V-interactive/-V/-V option is given,
and there are more than three files or the -V/-V, -V/-V,
or -V/-V-recursive/-V/-V are given, then
rm
prompts the user for whether to proceed with the entire operation. If
the response is not affirmative, the entire command is aborted.
Otherwise, if a file is unwritable, standard input is a terminal, and
the -V/-V or -V/-V-recursive/-V/-V option is not given, or the
-V/-V or -V/-V-interactive/-V/-V option is given,
rm
prompts the user for whether to remove the file. If the response is
not affirmative, the file is skipped.
in OPTIONS
-m
Remove (unlink) the FILE(s).
-V/-V, -V/-V, -V/-V-recursive/-V/-V
Ignore nonexistent files and arguments, never prompt
or
-V/-V/-V
Prompt before every removal
or
-V/-V/-V
Prompt once before removing more than three files, or
when removing recursively; less intrusive than -V/-V/-V,
while still giving protection against most mistakes
-m
-V/-V-interactive/-V/-V, -V/-V, -V/-V
Prompt according to MCH: never, once (-V/-V/-V), or
always (-V/-V/-V); without MCH, prompt always
or
-V/-V/-V/-V/-V/-V/-V/-V/-V/-V
When removing a hierarchy recursively, skip any
directory that is on a file system different from
that of the corresponding command line argument
or
-V/-V/-V/-V/-V/-V/-V/-V/-V/-V
Do not touch -V/-V specially
Index, 1-41
```

Figure 12: Справка по команде

Создала новый файл с названием random.sh (рис. 13). Написала текст командного файла, которой с помощью встроенной переменной \$RANDOM генерирует случайную последовательность букв латинского алфавита (рис. 14).

```
evkurkina@dk6n51 ~ $ touch random.sh
evkurkina@dk6n51 ~ $ emacs &
```

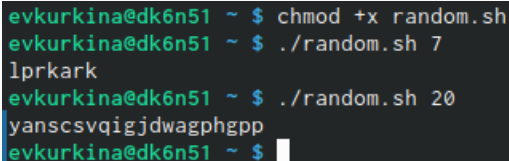
Figure 13: Создание нового файла для скрипта 3



```
File Edit Options Buffers Tools Shell Script Help
~/bin/bash
c=$1
for ((i=0; i<$c; i++))
do
  (( chars=$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: Текст скрипта3

Дала права доступа на исполнение и затем проверила работу скрипта (рис. 15).

A terminal window with a dark background and light green text. The prompt is 'evkurkina@dk6n51 ~ \$'. The first command is 'chmod +x random.sh'. The second command is './random.sh 7', which outputs 'lprkark'. The third command is './random.sh 20', which outputs 'yanscsvqigjdwagphgpp'. The prompt is now 'evkurkina@dk6n51 ~ \$' with a white cursor.

```
evkurkina@dk6n51 ~ $ chmod +x random.sh
evkurkina@dk6n51 ~ $ ./random.sh 7
lprkark
evkurkina@dk6n51 ~ $ ./random.sh 20
yanscsvqigjdwagphgpp
evkurkina@dk6n51 ~ $
```

Figure 15: Права доступа результат проверки исполнения файла

Во время выполнения данной лабораторной работы я изучила основы программирования в оболочке ОС UNIX. Научилась писать более сложные командные файлы с использованием логических управляющих конструкций и циклов.

