

# Отчет по лабораторной работе №12

Алмазова Елизавета Андреевна

# **ОТЧЕТ ПО ЛАБОРАТОРНОЙ РАБОТЕ №12**

# Цель работы и задание

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

Задание:

1. Написать командный файл, реализующий упрощённый механизм семафоров. Командный файл должен в течение некоторого времени  $t_1$  дожидаться освобождения ресурса, выдавая об

# Ход работы

1. Написала командный файл `semaphor.sh`, реализующий упрощённый механизм семафоров. Командный файл, получая в качестве аргумента `t1` и `t2`, в течение некоторого времени `t1` дожидается освобождения ресурса, выдавая об этом сообщение, а дождавшись его освобождения, использует его в течение некоторого времени `t2`, также выдавая информацию о том, что ресурс используется соответствующим командным файлом (процессом) (рис.1).

```

1 #!/bin/bash
2
3 t1=$1                #time of waiting
4 t2=$2                #time of using resource
5 s1=$(date +%s")      #s1, s2 - timers
6 s2=$(date +%s")
7 t=$s2-$s1            #passed time
8 while ((t < t1))
9 do
10     echo "Waiting"    #printing waiting
11     sleep 1
12     s2=$(date +%s")
13     t=$s2-$s1
14 done
15 s1=$(date +%s")
16 s2=$(date +%s")
17 t=$s2-$s1            #restart time
18 while ((t+t1 < t2))
19 do
20     echo "Using resource" #printing using
21     sleep 1
22     s2=$(date +%s")
23     t=$s2-$s1
24 done
25

```

Рисунок 1 - semaphor.sh.

2. Проверила работу файла. Запустила командный файл `semaphor.sh` в одном виртуальном терминале в фоновом режиме, перенаправив его вывод в другой (`> /dev/tty1`), в котором также запущен файл `semaphorprivileged.sh` в привилегированном режиме (рис.2).

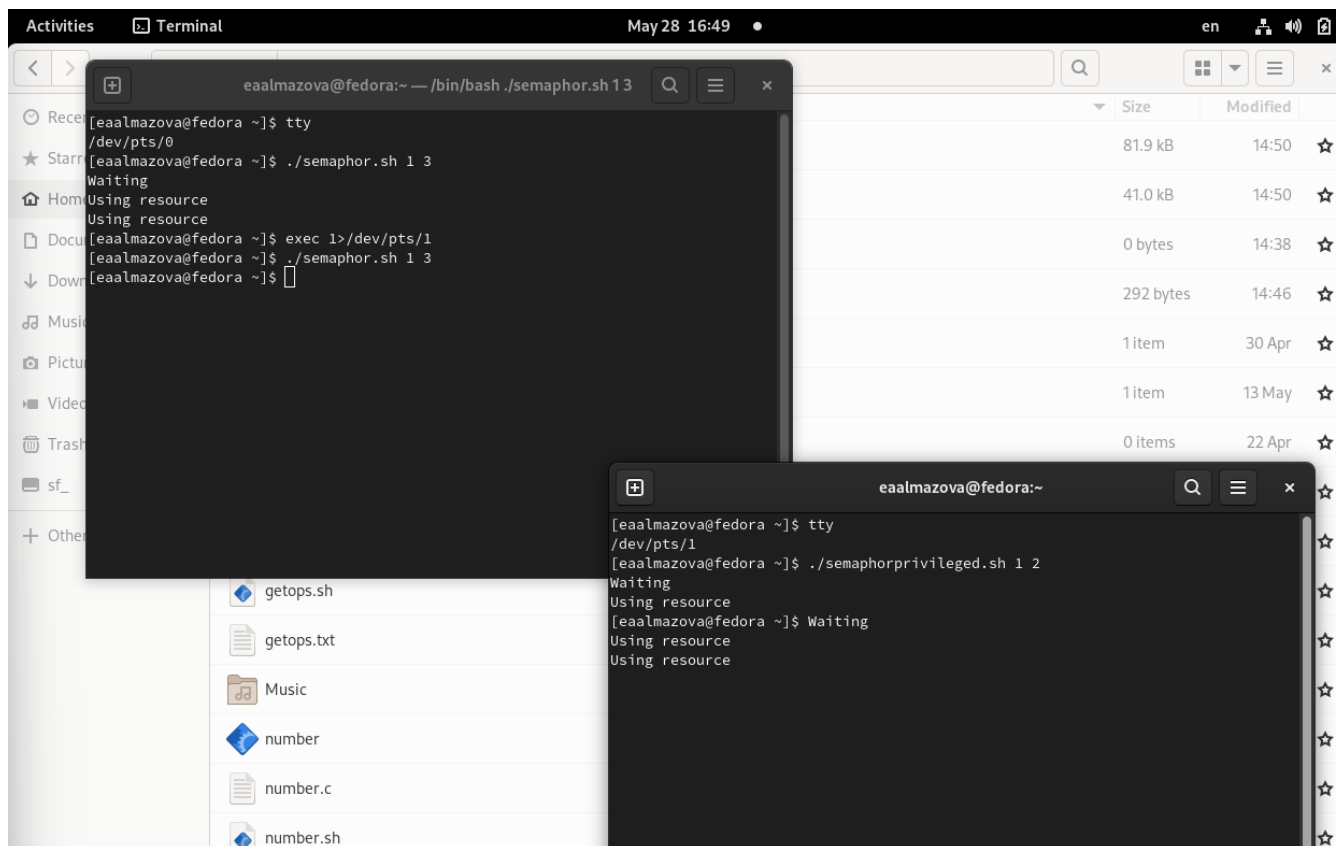


Рисунок 2 - Проверка работы файла и перенаправление вывода.

3. Доработала программу `semaphor2.sh` так, чтобы имелась возможность взаимодействия трёх и более процессов (рис.3). Проверила работу программы (рис.4).



```

1 #!/bin/bash
2
3 t1=$1          #time to wait
4 t2=$2          #time to use resource
5 N=$3           #number of terminals
6 i=0            #iterator of terminals
7
8 while (( i < N )); do
9     exec 1>/dev/pts/$i          #changing output to terminal #i
10    s1=$(date +%s")
11    s2=$(date +%s")
12    t=$s2-$s1                    #setting time
13    while (( t < t1 )); do
14        echo "$i waiting"        #terminal #i waiting
15        sleep 1
16        s2=$(date +%s")
17        t=$s2-$s1
18    done
19    s1=$(date +%s")
20    s2=$(date +%s")
21    t=$s2-$s1+$t1                #resetting time
22    while (( t < t2 )); do
23        echo "$i using resource" #terminal #i using resource
24        sleep 1
25        s2=$(date +%s")
26        t=$s2-$s1
27    done
28    i=$((i+1))                  #changing iterator
29 done

```

Рисунок 3 - semaphore2.sh.



4. Реализовала команду `man` с помощью командного файла `man.sh`. Изучила 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 получает в виде аргумента командной строки название команды и в виде результата выдает справку об этой команде или сообщение об отсутствии справки, если

```
1 #!/bin/bash
2
3 command=$1                                #command name
4 if [ -f /usr/share/man/man1/$command.1.gz ]    #looking for manual
5 then
6     gunzip -c /usr/share/man/man1/$command.1.gz | less    #unarchivating if manual exists
7 else
8     echo "There is no manual"                                #message if there is no manual
9 fi
```

Рисунок 5 - man.sh.

```
eaalmazova@fedora:~ — /bin/bash ./man.sh man

'\'' t
.\'' ** The above line should force tbl to be a preprocessor **
.\'' Man page for man
.\''
.\'' Copyright (C) 1994, 1995, Graeme W. Wilford. (Wilf.)
.\'' Copyright (C) 2001-2019 Colin Watson.
.\''
.\'' You may distribute under the terms of the GNU General Public
.\'' License as specified in the file COPYING that comes with the
.\'' man-db distribution.
.\''
.\'' Sat Oct 29 13:09:31 GMT 1994  Wilf. (G.Wilford@ee.surrey.ac.uk)
.\''
.pc
.TH MAN 1 "2021-02-08" "2.9.4" "Manual pager utils"
.SH NAME
man \- an interface to the system reference manuals
.SH SYNOPSIS
.\'' The general command line
.B man
.RI [\|] "man options" \|]
.RI [\|[\| section \|]
.IR page \ \|.\|.\|.\|]\| \|.|\|.|\|.|\&
.\'' The apropos command line
.br
.B man
.B \-k
.RI [\|] "apropos options" \|]
.I regexp
\&.\|.\|.\|.\&
.\'' The --global-apropos command line
.br
.B man
.B \-K
.RI [\|] "man options" \|]
.RI [\| section \|]
.IR term \ \|.\|.\|.\&
.\'' The whatis command line
.br
.B man
.B \-f
.RI [\| whatis
.IR options \|]
.I page
```

Рисунок 6 - Проверка работы файла.

5. Используя встроенную переменную \$RANDOM, написала командный файл random.sh, генерирующий случайную последовательность букв латинского алфавита, получая в качестве аргумента ее длину (рис.7). Проверила работу файла (рис.8).

```

1 #!/bin/bash
2
3 char=$1          #number of symbols
4 i=0              #iterator
5 while (( i < char )); do
6     (( letter= $RANDOM%26+1 ))          #getting letter number
7     case $letter in                    #printing letter
8         1) echo -n a;; 2) echo -n b;; 3) echo -n c;; 4) echo -n d;; 5) echo -n e;; 6) echo -n f;; 7) echo -n g;; 8) echo -n h;; 9) echo -n i;; 10) e
-n j;; 11) echo -n k;; 12) echo -n l;; 13) echo -n m;; 14) echo -n n;; 15) echo -n o;; 16) echo -n p;; 17) echo -n q;; 18) echo -n r;; 19) echo -n s
20) echo -n t;; 21) echo -n u;; 22) echo -n v;; 23) echo -n w;; 24) echo -n x;; 24) echo -n y;; 26) echo -n z;;
9         esac
10     i=$((i+1))                        #changing iterator
11 done
12 echo                                  #printing enter

```

Рисунок 7 - random.sh.

```
[eaalmazova@fedora ~]$ ./random.sh 11  
fwcjbsdpl  
[eaalmazova@fedora ~]$ ./random.sh 3  
frr
```

Рисунок 8 - Проверка работы файла.



# Выводы

В ходе 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.

Спасибо за внимание!