

Презентация по лабораторной работе №12

Сячинова Ксения Ивановна, НПМбд-02-21

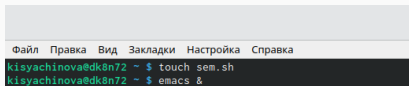
Российский Университет Дружбы Народов

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. Напишем командный файл, реализующий упрощённый механизм семафоров. Командный файл должен будет 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). Для этого создадим файл: `sem.sh` и напишем скрипт. (рис. 1), (рис. 2)

A screenshot of a terminal window with a light gray title bar and a dark gray menu bar. The menu bar contains the following items: 'Файл', 'Правка', 'Вид', 'Закладки', 'Настройка', and 'Справка'. The terminal text shows a user named 'kisyachinova@dk8n72' in a shell prompt '~ \$' executing the command 'touch sem.sh'. The next line shows the user executing 'emacs &'.

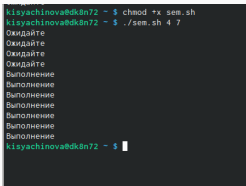
```
Файл  Правка  Вид  Закладки  Настройка  Справка
kisyachinova@dk8n72 ~ $ touch sem.sh
kisyachinova@dk8n72 ~ $ emacs &
```

Figure 1: Создание файла

```
1 #!/bin/bash
2 t1=$1
3 t2=$2
4 s1=$(date +%s)
5 s2=$(date +%s)
6 ((t=s2-s1))
7 while ((t < t1))
8 do
9     echo "Ожидайте"
10    sleep 1
11    s2=$(date +%s)
12    ((t=s2-s1))
13 done
14 s1=$(date +%s)
15 s2=$(date +%s)
16 ((t=s2-s1))
17 while ((t < t2))
18 do
19     echo "Выполнение"
20    sleep 1
21    s2=$(date +%s)
22    ((t=s2-s1))
23 done
```

Figure 2: Скрипт 1

Далее проверим работу написанного скрипта с помощью команды “./sem.sh 4 7”, но перед этим добавим право на выполнение командой “chmod +x sem.sh”. Скрипт работает верно. (рис. 3)

A terminal window with a dark background. The prompt is 'kisyachinova@dk8n72 ~'. The first command entered is '\$ chmod +x sem.sh'. The second command is './sem.sh 4 7'. The output consists of four 'Ожидайте' (Waiting) lines, followed by five 'Выполнение' (Execution) lines, and ends with the prompt 'kisyachinova@dk8n72 ~ \$' and a cursor.

```
kisyachinova@dk8n72 ~ - $ chmod +x sem.sh
kisyachinova@dk8n72 ~ - $ ./sem.sh 4 7
Ожидайте
Ожидайте
Ожидайте
Ожидайте
Выполнение
Выполнение
Выполнение
Выполнение
Выполнение
Выполнение
kisyachinova@dk8n72 ~ - $
```

Figure 3: Проверка скрипта 1

Далее изменим скрипт так, чтобы его можно было выполнять в нескольких терминалах.(рис. 4)

```
1 #!/bin/bash
2 function ogidanie
3 {
4     s1=$(date +%s)
5     s2=$(date +%s)
6     ((t=s2-s1))
7     while ((t < t1))
8     do
9         echo "Ожидание"
10        sleep 1
11        s2=$(date +%s)
12        ((t=s2-s1))
13    done
14 }
15 function vipolnenie
16 {
17     s1=$(date +%s)
18     s2=$(date +%s)
19     ((t=s2-s1))
20     while ((t < t2))
21     do
22         echo "Выполнение"
23         sleep 1
24         s2=$(date +%s)
25         ((t=s2-s1))
26     done
27 }
28 t1=$1
29 t2=$2
30 command=$3
31 while true
32 do
33     if [ "$command" == "Выход" ]
34     then
35         echo "Выход"
36         exit 0
37     fi
38     if [ "$command" == "Ожидание" ]
39     then
40         ogidanie
41     fi
42     if [ "$command" == "Выполнение" ]
43     then
44         vipolnenie
45     fi
46     echo "Следующее действие: "
47     read command
```

Figure 4: Изменённый скрипт

Проверим работу с помощью команды “./sem.sh 2 3 Ожидание>/dev/pts/1 &”. Проверить работа данного скрипта не удалось, так как было отказано в доступе.(рис. 5)

```
[1]+ Выход 1 ./sem.sh 2 3 Ожидание > /dev/pts/1
kisyachinova@dk8n72 ~ $ ./sem.sh 2 3 Ожидание>/dev/pts/2 &
[1] 9930
kisyachinova@dk8n72 ~ $ bash: /dev/pts/2: Отказано в доступе

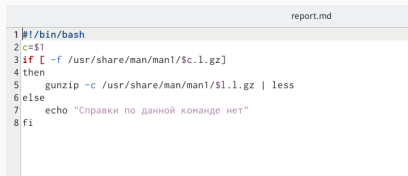
[1]+ Выход 1 ./sem.sh 2 3 Ожидание > /dev/pts/2
kisyachinova@dk8n72 ~ $ ./sem.sh 2 3 Выполнение > /dev/pts/1 &
[1] 9956
kisyachinova@dk8n72 ~ $ bash: /dev/pts/1: Отказано в доступе

[1]+ Выход 1 ./sem.sh 2 3 Выполнение > /dev/pts/1
kisyachinova@dk8n72 ~ $ ./sem.sh 2 3 Выполнение > /dev/pts/2 &
[1] 9959
kisyachinova@dk8n72 ~ $ bash: /dev/pts/2: Отказано в доступе

[1]+ Выход 1 ./sem.sh 2 3 Выполнение > /dev/pts/2
kisyachinova@dk8n72 ~ $
```

Figure 5: Проверка изменённого скрипта

Далее я создала файл “man.sh”, в котором буду писать следйущий скрипт. (рис. 7)



```
report.md
1 #!/bin/bash
2 c=$1
3 if [ -f /usr/share/man/man1/${c}.1.gz ]
4 then
5     gunzip -c /usr/share/man/man1/${c}.1.gz | less
6 else
7     echo "Справки по данной команде нет"
8 fi
```

Figure 7: Скрипт 2

Проверим работу скрипта, дадим ему право на выполнение “chmod +x man.sh” и проверим с помощью команды “./man.sh ls”, “./man.sh mkdir”(рис. 8)

```
kisyachinova@dk8n72 ~ $ chmod +x man.sh
kisyachinova@dk8n72 ~ $ ./man.sh ls
Справки по данной команде нет
kisyachinova@dk8n72 ~ $ ./man.sh mkdir
Справки по данной команде нет
kisyachinova@dk8n72 ~ $
```

Figure 8: Проверка скрипта 2

3. Используем встроенную переменную \$RANDOM, напишем командный файл, генерирующий случайную последовательность букв латинского алфавита. Учтём, что \$RANDOM выдаёт псевдослучайные числа в диапазоне от 0 до 32767. Создадим файл для написания третьего скрипта (random.sh).(рис. 9)

```
#!/bin/bash
k=$1
for (( i=0; i<$k; 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 9: Скрипт 3

Затем даём право на выполнение и с помощью команды “./random 5” и “./random 12” проверяем выполнение работы.(рис. 10)

```
kisyachinova@dk8n72 ~ $ chmod +x random.sh
kisyachinova@dk8n72 ~ $ ./random.sh 5
qhvbj
kisyachinova@dk8n72 ~ $ ./random.sh 12
tzazpzfjwgtz
kisyachinova@dk8n72 ~ $ █
```

Figure 10: Проверка 3 скрипта

Выводы

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