

Программирование в командном процессоре ОС UNIX. Расширенное программирование

Световидова Полина НБИбд-04-22

26 апреля, 2023, Москва, Россия

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

Задание

Задание

1. Написать 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 (процессом). Запустить командный файл в одном виртуальном терминале в фоновом режиме, перенаправив его вывод в другой (`> /dev/tty#`, где # — номер терминала куда перенаправляется вывод), в котором также запущен этот файл, но не фоновом, а в привилегированном режиме. Доработать программу так, чтобы имелась

Выполнение лабораторной работы

Выполнение лабораторной работы

Создаю файл для выполнения работы и написания кода

```
[root@10 ~]# touch 12lab.sh  
[root@10 ~]# chmod +x 12lab.sh
```

Рис. 1: создание файла

написание кода по заданию в emacs

```
1  #!/bin/bash
2  t1=$1
3  t2=$2
4  s1=$(date +%S)
5  s2=$(date +%S)
6  ((t=$s2-$s1))
7  while ((t<t1))
8  do
9      echo "ожидание"
10     sleep 1
11     s2=$(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
24
```

проверяю написанную мной программу

```
[root@10 ~]# ./12lab.sh  
[root@10 ~]#
```

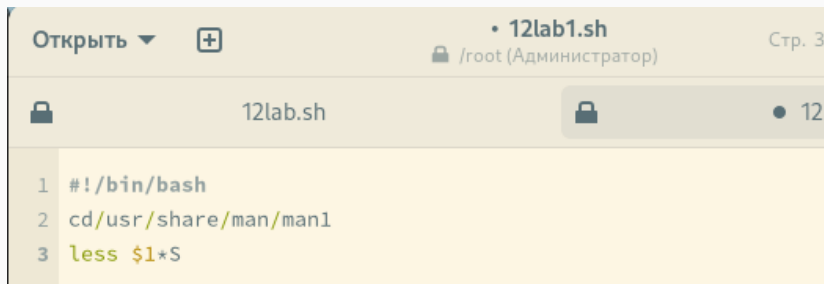
Рис. 3: проверка программы

создаю новый файл для реализации команды `map`

```
[root@10 ~]# ./12lab.sh  
[root@10 ~]# touch 12lab1.sh  
[root@10 ~]# chmod +x 12lab1.sh  
[root@10 ~]#
```

Рис. 4: создание нового файла

пишу сам код для реализации задумки



```
Открыть ▾ + • 12lab1.sh Стр. 3
🔒 /root (Администратор)
🔒 12lab.sh 🔒 • 12
1 #!/bin/bash
2 cd/usr/share/man/man1
3 less $1*S
```

Рис. 5: код для man

проверяю новую программу на работу

```
[root@10 ~]# ./12lab1.sh less
```

Рис. 6: проверка кода

```
root@10:/usr
LESS(1) General Commands Manual LESS(1)

NAME
    less - opposite of more

SYNOPSIS
    less -?
    less --help
    less -V
    less --version
    less [-[+ ]aAbCcDeEfFgGiIJKLmMnNqQrRsSuUVwWxX-]
        [-b space] [-h lines] [-j line] [-k keyfile]
        [-{oo} logfile] [-p pattern] [-P prompt] [-t tag]
        [-T tagfile] [-x tab,...] [-y lines] [-{z} lines]
        [-# shift] [+{+}cmd] [--] [filename]...

    (See the OPTIONS section for alternate option syntax with long option
    names.)

DESCRIPTION
    Less is a program similar to more(1), but which allows backward move-
    ment in the file as well as forward movement. Also, less does not have
    to read the entire input file before starting, so with large input
    files it starts up faster than text editors like vi(1). Less uses
    termcap (or terminfo on some systems), so it can run on a variety of
    terminals. There is even limited support for hardcopy terminals. (On
    a hardcopy terminal, lines which should be printed at the top of the
    screen are prefixed with a caret.)

    Commands are based on both more and vi. Commands may be preceded by a
    decimal number, called N in the descriptions below. The number is used
    by some commands, as indicated.

COMMANDS
    In the following descriptions, ^X means control-X. ESC stands for the
    ESCAPE key; for example ESC-v means the two character sequence "ES-
    CAPE", then "v".

    h or H Help: display a summary of these commands. If you forget all
    the other commands, remember this one.

    SPACE or ^V or f or ^F
    Scroll forward N lines, default one window (see option -z be-
    low). If N is more than the screen size, only the final screen-
    ful is displayed. Warning: some systems use ^V as a special
    Manual page less(1) line 1 (press h for help or q to quit)
```

Рис. 7: man less

создаю новый текстовый файл и делаю его исполняемым

```
[root@10 ~]# cd  
[root@10 ~]# touch 12lab2.sh  
[root@10 ~]# chmod +x 12lab2.sh  
[root@10 ~]#
```

Рис. 8: создание нового файла

пишу код, генерирующий случайную последовательность
букв латинского алфавита

```
1  #!/bin/bash
2  M=10
3  c=1
4  d=1
5  echo
6  echo "10 случайных слов"
7  while (($c!=($M+1)))
8  do
9      echo $(for((i=1; i<=10; i++));
10         do
11             printf '%s' "${RANDOM:0:1}"
12         done
13         tr '[0-9]' '[a-z]'
14         echo $d
15         ((c+=1))
16         ((d+=1))
17     done
```

Рис. 9: код random

проверяю программу на работу


```
[root@10 ~]# ./12lab2.sh
```

```
10 случайных слов
```

```
ecccbchcbj
```

```
1
```

```
jcceiecbbj
```

```
2
```

```
cbbccbcbeb
```

```
3
```

```
hbgihcfcdd
```

```
4
```

```
ccbccccchc
```

```
5
```

```
ebgbcbcgbd
```

```
6
```

```
cbbccddcbc
```

```
7
```

```
bbibccbfcg
```

```
8
```

```
dbcchcbccb
```

```
9
```

```
cbdjfejhdb
```

```
10
```

```
[root@10 ~]#
```

Рис. 10: проверка кода с random

Выводы

В ходе выполнения лабораторной работы №12 я изучила основы программирования в оболочке ОС Linux, а так же научилась писать более сложные командные файлы с использованием логических управляющих конструкций и циклов