

Лабораторная работа №14

Операционные системы

Калашникова Ольга Сергеевна НПИбд-01-23

11 мая 2024

Российский университет дружбы народов, Москва, Россия

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

Создаю командный файл для первой программы, пишу ее, проверяю ее работу (рис.1).

```
[oskalashnikova@oskalashnikova ~]$ touch 121.sh
[oskalashnikova@oskalashnikova ~]$ chmod +x 121.sh
[oskalashnikova@oskalashnikova ~]$ bash 121.sh
File is blocked
File is unlocked
File is blocked
File is unlocked
File is blocked
File is unlocked
File is blocked
```

Рис. 1: Создание и исполнение файла

Выполнение лабораторной работы

Командный файл, реализующий упрощённый механизм семафоров. (рис.2).

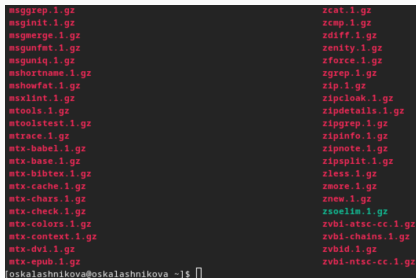
```
#!/bin/bash

lockfile="./lock.file"
exec {fn}>$lockfile

while test -f "$lockfile"
do
if flock -n ${fn}
then
    echo "File is_blocked"
    sleep 5
    echo "File is unlocked"
    flock -u ${fn}
else
    echo "File is blocked"
    sleep 5
fi
done
```

Выполнение лабораторной работы

Чтобы реализовать команду `man` с помощью командного файла, изучаю содержимое каталога `/usr/share/man/man1`. В нем находятся архивы текстовых файлов, содержащих справку по большинству установленных в системе программ и команд (рис.3).



```
msggrep.1.gz      zcat.1.gz
msginit.1.gz      zcmp.1.gz
msgmerge.1.gz     zdiff.1.gz
msgunfmt.1.gz     zenity.1.gz
msguniq.1.gz      zforce.1.gz
mshortname.1.gz   zgrep.1.gz
mshowfat.1.gz     zip.1.gz
msxlint.1.gz      zipcloak.1.gz
mtools.1.gz       zipdetails.1.gz
mtoolstest.1.gz   zipgrep.1.gz
mtrace.1.gz       zipinfo.1.gz
mtx-babel.1.gz    zipnote.1.gz
mtx-base.1.gz     zipsplit.1.gz
mtx-bibtex.1.gz   zless.1.gz
mtx-cache.1.gz    zmore.1.gz
mtx-chars.1.gz    znew.1.gz
mtx-check.1.gz    zsoelim.1.gz
mtx-colors.1.gz   zvbi-atsc-cc.1.gz
mtx-context.1.gz  zvbi-chains.1.gz
mtx-dvi.1.gz      zvbid.1.gz
mtx-epub.1.gz     zvbi-ntsc-cc.1.gz
oskalashnikova@oskalashnikova ~]$
```

Рис. 3: Изучение содержимого папки

Выполнение лабораторной работы

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 (рис.4).

```
#!/bin/bash

a=$1
if test -f "/usr/share/man/man1/$a.1.gz"
then less /usr/share/man/man1/$a.1.gz
else
echo "There is no such command"
fi
```

Проверяю работу командного файла (рис.5).

```
[oskalashnikova@oskalashnikova ~]$ touch 122.sh  
[oskalashnikova@oskalashnikova ~]$ chmod +x 122.sh  
[oskalashnikova@oskalashnikova ~]$ ./122.sh ls  
[oskalashnikova@oskalashnikova ~]$
```

Рис. 5: Исполнение программы

Выполнение лабораторной работы

Командный файл работает так же, как и команда `man`, открывает справку по указанной утилите (рис.6).

```
ESC[1m--authorESC[0m
with ESC[1m-lESC[22m, print the author of each file

ESC[1m-bESC[22m, ESC[1m--escapeESC[0m
print C-style escapes for nongraphic characters

ESC[1m--block-sizeESC[22m=ESC[4mSIZEESC[0m
with ESC[1m-lESC[22m, scale sizes by SIZE when pri
nting them;
e.g., '--block-size=M'; see SIZE format below

ESC[1m-BESC[22m, ESC[1m--ignore-backupsESC[0m
do not list implied entries ending with ~

ESC[1m-c ESC[22mwith ESC[1m-ltESC[22m: sort by, and show
, ctime (time of last
change of file status information); with ESC[1m-l
ESC[22m: show
/usr/share/man/man1/ls.1.gz
```

Рис. 6: Результат работы программы

Создаю файл для кода третьей программы, пишу программу и проверяю ее работу (рис.7).

```
[oskalashnikova@oskalashnikova ~]$ touch 123.sh
[oskalashnikova@oskalashnikova ~]$ chmod +x 123.sh
[oskalashnikova@oskalashnikova ~]$ bash 123.sh 1
p
[oskalashnikova@oskalashnikova ~]$ bash 123.sh 11
hsuaggkdgzz
[oskalashnikova@oskalashnikova ~]$ bash 123.sh 20
ibeqnoirbgzhjhjbndoj
[oskalashnikova@oskalashnikova ~]$
```

Рис. 7: Создание и исполнение файла

Выполнение лабораторной работы

Используя встроенную переменную \$RANDOM, пишу командный файл, генерирующий случайную последовательность букв латинского алфавита. Т.к. \$RANDOM выдаёт псевдослучайные числа в диапазоне от 0 до 32767, ввожу ограничения так, чтобы была генерация чисел от 1 до 26 (рис.8).

```
#!/bin/bash

a=$1

for ((i=0; i<$a; i++))
do
    ((char=$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 r;; 18) echo -n s;;
        19) echo -n t;; 20) echo -n q;; 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
```

Рис. 8: Код программы

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