

Лабораторная работа 14

Программирование в командном процессоре ОС UNIX. Расширенное программирование

Богданюк Анна Васильевна

Содержание

1	Цель работы	1
2	Задание.....	1
3	Выполнение лабораторной работы	1
4	Выводы.....	4

1 Цель работы

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

2 Задание

1. Написать командный файл, реализующий упрощённый механизм семафоров.
2. Реализовать команду `map` с помощью командного файла.
3. Используя встроенную переменную `$RANDOM`, напишите командный файл, генерирующий случайную последовательность букв латинского алфавита.

3 Выполнение лабораторной работы

Для начала пишу 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).(рис. 1).

```
avbogdanyuk@Bogdanyuk: ~/work/why
GNU nano 6.2 1.sh *
lockfile="./lockfile"
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
```

Текст файла

Затем делаю файл исполняемым, запускаю его, все работает корректно (рис. 2).

```
avbogdanyuk@Bogdanyuk:~/work/why$ touch 1.sh
avbogdanyuk@Bogdanyuk:~/work/why$ chmod +x 1.sh
avbogdanyuk@Bogdanyuk:~/work/why$ ls
1.sh amigo.cpp bla bla.txt dude text.txt
avbogdanyuk@Bogdanyuk:~/work/why$ nano 1.sh
avbogdanyuk@Bogdanyuk:~/work/why$ bash 1.sh
File is blocked
```

Запуск файла

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

```
avbogdanyuk@Bogdanyuk: ~/work/why
GNU nano 6.2 2.sh *
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
```

Текст файла

Затем делаю файл исполняемым, запускаю его, команды cd не нашлось. как и должно было произойти, а ls нашлась (рис. 4).

```
avbogdanyuk@Bogdanyuk: ~/work/why
avbogdanyuk@Bogdanyuk:~/work/why$ touch 2.sh
avbogdanyuk@Bogdanyuk:~/work/why$ chmod +x 2.sh
avbogdanyuk@Bogdanyuk:~/work/why$ nano 2.sh
avbogdanyuk@Bogdanyuk:~/work/why$ ./2.sh cd
There is no such command
avbogdanyuk@Bogdanyuk:~/work/why$ ./2.sh ls

[7]+ Stopped ./2.sh ls
avbogdanyuk@Bogdanyuk:~/work/why$
```

Запуска файл

Вот так выглядит результат запуска командного файла для команды ls (рис. 5).

```
avbogdanyuk@Bogdanyuk: ~/work/why
.\ DO NOT MODIFY THIS FILE! It was generated by Help2man 1.47.3.
.TH LS "1" "January 2024" "GNU coreutils 8.32" "User Commands"
.SH NAME
ls \- list directory contents
.SH SYNOPSIS
.B ls
[-fiv,OPTION][-l]... [-fi][-FILE]...
.SH DESCRIPTION
.\ Add any additional description here
.PP
List information about the FILES (the current directory by default).
Sort entries alphabetically if none of -fb, -cftuvSUX, -fr nor -fb, -l, -sort -fr is specified.
.PP
Mandatory arguments to long options are mandatory for short options too.
.TP
-fb, -a, -fr, -l, -all -fr
do not ignore entries starting with .
.TP
-fb, -A, -fr, -l, -almost -all -fr
do not list implied . and ..
.TP
-fb, -l, -author -fr
with -fb, -l -fr, print the author of each file
.TP
-fb, -b, -fr, -l, -escape -fr
print C-style escapes for nongraphic characters
```

Команда ls

Используя встроенную переменную \$RANDOM, написала командный файл, генерирующий случайную последовательность букв латинского алфавита. (рис. 6).

```
avbogdanyuk@Bogdanyuk: ~/work/why
GNU nano 6.2 3.sh
a=$!
for ((i=0;i<$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
```

Текст файла

Затем делаю файл исполняемым, запускаю его, все работает корректно (рис. 7).

```
avbogdanyuk@Bogdanyuk:~/work/why$ touch 3.sh
avbogdanyuk@Bogdanyuk:~/work/why$ chmod +x 3.sh
avbogdanyuk@Bogdanyuk:~/work/why$ nano 3.sh
avbogdanyuk@Bogdanyuk:~/work/why$ bash 3.sh 12
nnvdlqizzau
avbogdanyuk@Bogdanyuk:~/work/why$
```

Запуск файла

4 Выводы

В ходе выполнения лабораторной работы были изучены основы программирования в оболочке ОС UNIX. Научилась писать более сложные командные файлы с использованием логических управляющих конструкций и циклов.