

Лабораторная работа №14

Программирование в командном процессоре ОС UNIX. Расширенное программирование

Юсупова Ксения Равиловна

Российский университет дружбы народов, Москва, Россия

Информация

- Юсупова Ксения Равилевна
- Российский университет дружбы народов
- Номер студенческого билета- 1132247531
- [1132247531@pfur.ru]

Вводная часть

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.

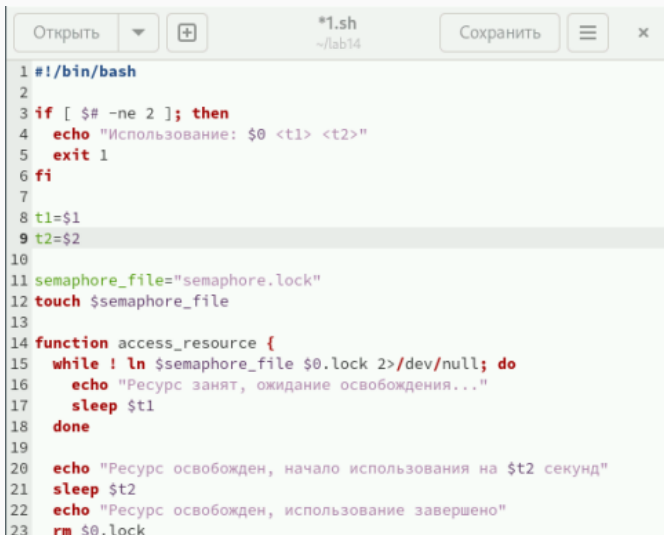
Командный файл должен в течение некоторого времени t_1 дожидаться освобождения ресурса, выдавая об этом сообщение, а дождавшись его освобождения, использовать его в течение некоторого времени $t_2 < t_1$, также выдавая информацию о том, что ресурс используется соответствующим командным файлом (процессом). 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 возможность взаимодействия трёх и более процессов.

2. Реализовать 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. Каждый архив можно открыть командой `less` сразу же просмотрев содержимое справки. 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`.
3. Используя встроенную переменную `$RANDOM`, напишите командный файл, генерирующий случайную последовательность букв латинского алфавита. Учтите, что `$RANDOM` выдаёт псевдослучайные числа в диапазоне от 0 до 32767.

Выполнение лабораторной работы

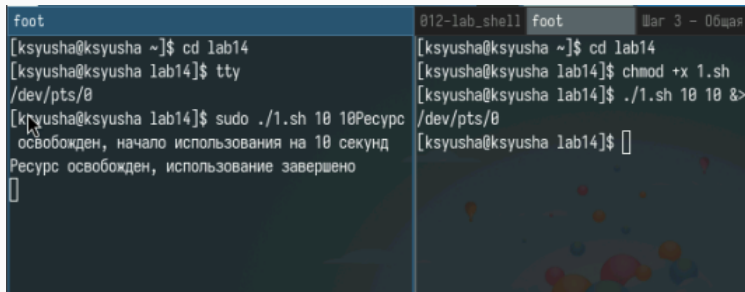
Выполнение лабораторной работы

Напишем код для первой программы



```
1 #!/bin/bash
2
3 if [ $# -ne 2 ]; then
4     echo "Использование: $0 <t1> <t2>"
5     exit 1
6 fi
7
8 t1=$1
9 t2=$2
10
11 semaphore_file="semaphore.lock"
12 touch $semaphore_file
13
14 function access_resource {
15     while ! ln $semaphore_file $0.lock 2>/dev/null; do
16         echo "Ресурс занят, ожидание освобождения..."
17         sleep $t1
18     done
19
20     echo "Ресурс освобожден, начало использования на $t2 секунд"
21     sleep $t2
22     echo "Ресурс освобожден, использование завершено"
23     rm $0.lock
```

Проверили код на работу



The image shows two terminal windows side-by-side. The left window has a title bar 'foot' and shows the following commands and output: `[ksyusha@ksyusha ~]$ cd lab14`, `[ksyusha@ksyusha lab14]$ tty` (output: `/dev/pts/0`), and `[ksyusha@ksyusha lab14]$ sudo ./1.sh 10 10` (output: `Ресурс освобожден, начало использования на 10 секунд` and `Ресурс освобожден, использование завершено`). The right window has a title bar '012-lab_shell foot' and shows: `[ksyusha@ksyusha ~]$ cd lab14`, `[ksyusha@ksyusha lab14]$ chmod +x 1.sh`, and `[ksyusha@ksyusha lab14]$./1.sh 10 10 &> /dev/pts/0` (output: `[ksyusha@ksyusha lab14]$`).

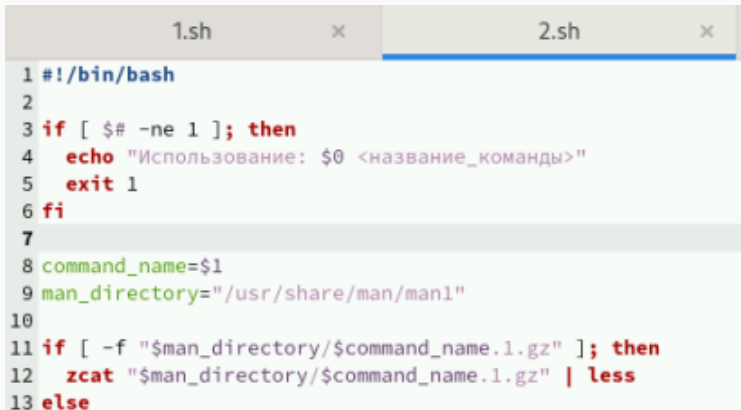
```
foot
[ksyusha@ksyusha ~]$ cd lab14
[ksyusha@ksyusha lab14]$ tty
/dev/pts/0
[ksyusha@ksyusha lab14]$ sudo ./1.sh 10 10
Ресурс освобожден, начало использования на 10 секунд
Ресурс освобожден, использование завершено
[ksyusha@ksyusha lab14]$

012-lab_shell foot
[ksyusha@ksyusha ~]$ cd lab14
[ksyusha@ksyusha lab14]$ chmod +x 1.sh
[ksyusha@ksyusha lab14]$ ./1.sh 10 10 &> /dev/pts/0
[ksyusha@ksyusha lab14]$
```

Рис. 2: проверили первый код

Выполнение лабораторной работы

Напишем код для второй программы (Реализовать 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.)



```
1 #!/bin/bash
2
3 if [ $# -ne 1 ]; then
4     echo "Использование: $0 <название_команды>"
5     exit 1
6 fi
7
8 command_name=$1
9 man_directory="/usr/share/man/man1"
10
11 if [ -f "$man_directory/$command_name.1.gz" ]; then
12     zcat "$man_directory/$command_name.1.gz" | less
13 else
```

Выполнение лабораторной работы

Проверили код на работу

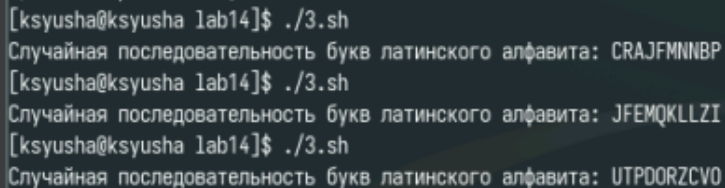
```
generate output designed for Emacs' dired mode
.TP
\fb\-f\fr
do not sort, enable \fb\-aU\fr, disable \fb\-ls\fr \fb\--color\fr
.TP
\fb\-F\fr, \fb\--classify\fr[=\fI\,WHEN\/\fr]
append indicator (one of */=>@|) to entries WHEN
.TP
\fb\--file\--type\fr
likewise, except do not append '*'
.TP
\fb\--format\fr=\fI\,WORD\/\fr
across \fb\-x\fr, commas \fb\-m\fr, horizontal \fb\-x\fr, long \fb\-l\fr,
single\--column \fb\-l\fr, verbose \fb\-l\fr, vertical \fb\-C\fr
.TP
\fb\--full\--time\fr
like \fb\-l\fr \fb\--time\--style\fr=\fI\,full\--iso\/\fr
.TP
\fb\-g\fr
like \fb\-l\fr, but do not list owner
.TP
\fb\--group\--directories\--first\fr
group directories before files;
can be augmented with a \fb\--sort\fr option, but any
use of \fb\--sort\fr=\fI\,none\/\fr (\fb\-U\fr) disables grouping
.TP
\fb\-G\fr, \fb\--no\--group\fr
in a long listing, don't print group names
```

Выполнение лабораторной работы

Написали код для третьей программы (Используя встроенную переменную \$RANDOM, напишите командный файл, генерирующий случайную последовательность букв латинского алфавита)

```
1.sh × 2.sh × 3.sh ×
1 #!/bin/bash
2
3 generate_random_letter() {
4   # Случайное число от 0 до 25
5   random_number=$((RANDOM % 26))
6
7   letter=$(printf \\$(printf '%03o' $((65 + random_number))))
8
9   echo -n "$letter"
10 }
11
12 random_sequence=""
13 for ((i=0; i<10; i++)); do
14   random_sequence="$random_sequence$(generate_random_letter)"
15 done
16
```

Проверили код на работу



```
[ksyusha@ksyusha lab14]$ ./3.sh
Случайная последовательность букв латинского алфавита: CRAJFMNNBP
[ksyusha@ksyusha lab14]$ ./3.sh
Случайная последовательность букв латинского алфавита: JFEMQKLLZI
[ksyusha@ksyusha lab14]$ ./3.sh
Случайная последовательность букв латинского алфавита: UTPDORZCVO
```

Рис. 6: Проверили код на работу

Выводы

В ходе лабораторной работы мы изучили основы программирования в оболочке ОС UNIX. Научились писать более сложные командные файлы с использованием логических управляющих конструкций и циклов.