

Отчёт по лабораторной работе №13

"Программирование в командном процессоре ОС UNIX. Расширенное программирование"

author: Малащенко Марина Владимировна

Цель работы

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

Ход работы

1. Напишем 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)

```
lab10_1.sh          [-----]  2 L:[  1+17
lockfile="./locking.file"
exec {fn}>$lockfile
if test -f "$lockfile"
then
while [ 1!=0 ]
do
if flock -n ${fn}
then
echo "File was locked"
sleep 2
echo "Unlocking..."
flock -u ${fn}
else
echo "File already locked"
sleep 2
fi
done
fi
```

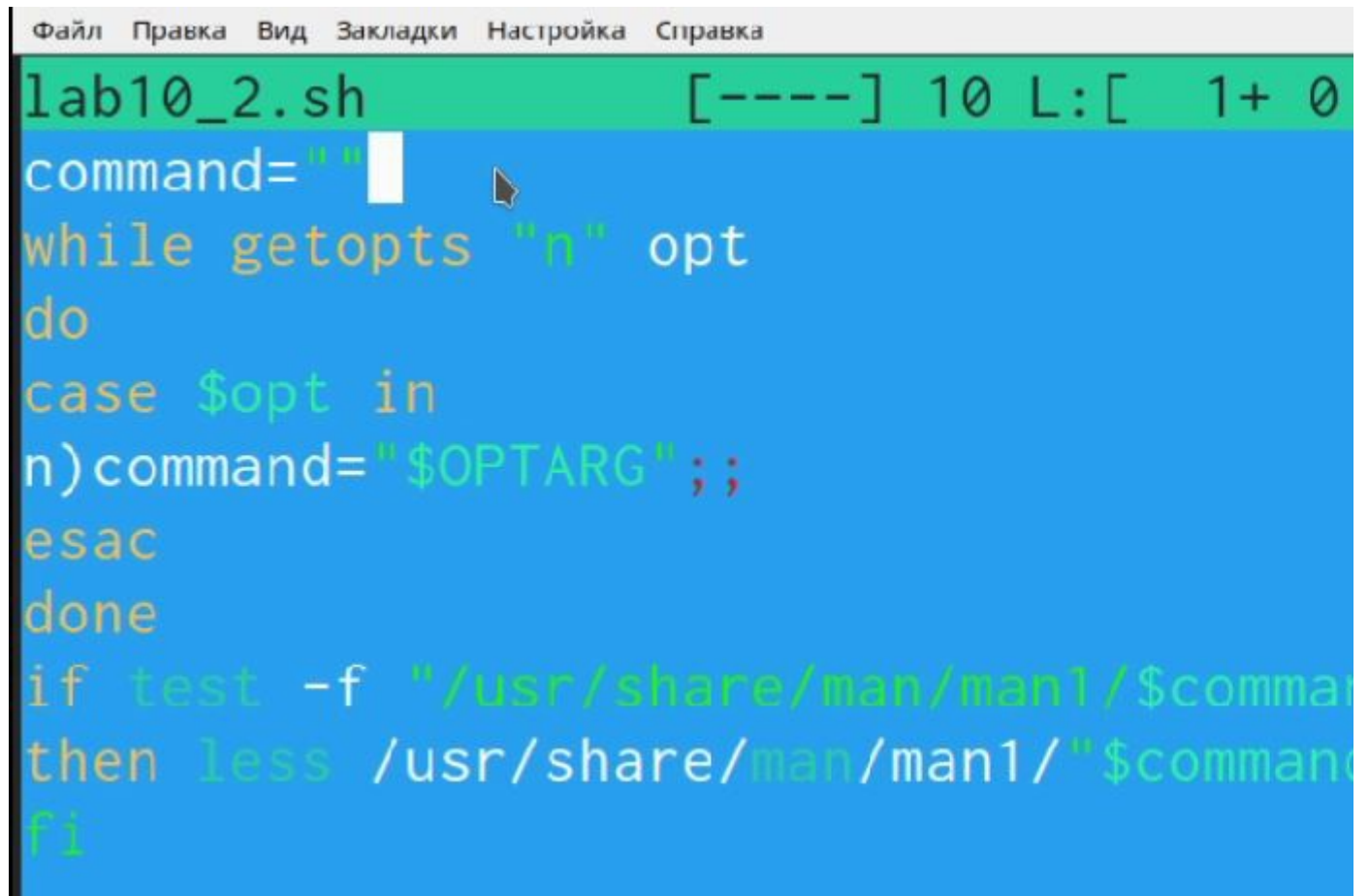
```
mvmalashenko@dk6n53 ~ $ mcedit 1.sh

mvmalashenko@dk6n53 ~ $ mcedit lab10_1

mvmalashenko@dk6n53 ~ $ mcedit lab10_1

mvmalashenko@dk6n53 ~ $ ./lab10_1.sh
bash: ./lab10_1.sh: Отказано в доступе
mvmalashenko@dk6n53 ~ $ chmod ugo+rwx
mvmalashenko@dk6n53 ~ $ ./lab10_1.sh
File was locked
Unlocking...
File already locked
File already locked
File already locked
File already locked
File already locked
File already locked
File already locked
File already locked
File already locked
File already locked
File already locked
File already locked
File already locked
```

/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.

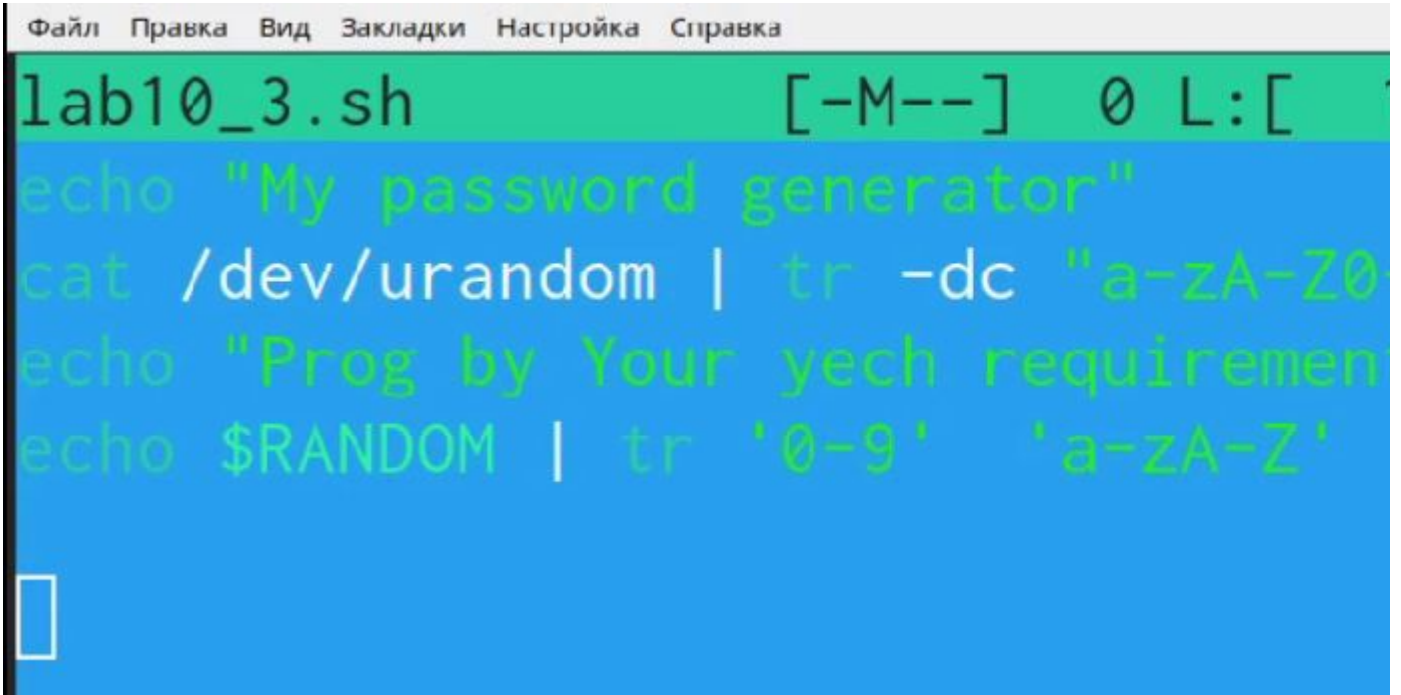


```
Файл  Правка  Вид  Закладки  Настройка  Справка
lab10_2.sh  [-----]  10  L:[  1+  0
command=""
while getopts "n" opt
do
case $opt in
n)command="$OPTARG";;
esac
done
if test -f "/usr/share/man/man1/$command"
then less /usr/share/man/man1/"$command"
fi
```



```
mvmalashenko@dk6n53 ~ $ mcedit lab10_2
mvmalashenko@dk6n53 ~ $ chmod ugo+rwx lab10_2
mvmalashenko@dk6n53 ~ $ ./lab10_2.sh -r
./lab10_2.sh: строка 9: неожиданный конец файла
./lab10_2.sh: строка 11: синтаксическая ошибка
mvmalashenko@dk6n53 ~ $ mcedit lab10_2
mvmalashenko@dk6n53 ~ $ mcedit lab10_2
mvmalashenko@dk6n53 ~ $ chmod ugo+rwx lab10_2
mvmalashenko@dk6n53 ~ $ ./lab10_2.sh -r
```

3. Используя встроенную переменную \$RANDOM, написали командный файл, генерирующий случайную последовательность букв латинского алфавита. Учтя, что \$RANDOM выдаёт псевдослучайные числа в диапазоне от 0 до 32767.



```
Файл  Правка  Вид  Закладки  Настройка  Справка
lab10_3.sh  [ -M-- ]  0  L:
echo "My password generator"
cat /dev/urandom | tr -dc "a-zA-Z0
echo "Prog by Your yech requiremen
echo $RANDOM | tr '0-9' 'a-zA-Z'
```

Для наглядности запустили скрипт несколько раз, чтобы увидеть, что генерируются разные случайные пароли.

```
mvmalashenko@dk6n53 ~ $ chmod ugo+
mvmalashenko@dk6n53 ~ $ ./lab10_3.
My password generator
MqsvtTk67p
Prog by Your yech requirement:
cjcgc
mvmalashenko@dk6n53 ~ $ ./lab10_3.
My password generator
mizqDdgCNp
Prog by Your yech requirement:
ejcd
mvmalashenko@dk6n53 ~ $ ./lab10_3.
My password generator
wKU2prlGgo
Prog by Your yech requirement:
cdedh
mvmalashenko@dk6n53 ~ $
```

Вывод

Анализ результатов

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

Контрольные вопросы

1: **Найдите синтаксическую ошибку в следующей строке: while [\$1 != "exit"]**

\$1. Так же между скобками должны быть пробелы. В противном случае скобки и рядом стоящие символы будут восприниматься как одно целое

2: **Как объединить (конкатенация) несколько строк в одну?**

```
cat file.txt | xargs | sed -e 's/./.\n/g'
```

3: **Найдите информацию об утилите seq. Какими иными способами можно реализовать её функционал при программировании на bash?**

seq - выдает последовательность чисел. Реализовать ее функционал можно командой `for n in {1..5} do <КОМАНДА> done`

4: **Какой результат даст вычисление выражения \$((10/3))?**

3

5: **Укажите кратко основные отличия командной оболочки zsh от bash.**

Zsh очень сильно упрощает работу. Но существуют различия. Например, в zsh после `for` обязательно вставлять пробел, нумерация массивов в zsh начинается с 1 (что не особо удобно на самом деле). Если вы собираетесь писать скрипт, который легко будет запускать множество разработчиков, то я рекомендую Bash. Если скрипты вам не нужны - Zsh (более простая работа с файлами, например)

6: **Проверьте, верен ли синтаксис данной конструкции `for ((a=1; a <= LIMIT; a++))`**

Верен

7: **Сравните язык bash с какими-либо языками программирования. Какие преимущества у bash по сравнению с ними? Какие недостатки?**

Bash позволяет очень легко работать с файловой системой без лишних конструкций (в отличие от обычного языка программирования). Но относительно обычных языков программирования bash очень сжат. Тот же Си имеет гораздо более широкие возможности для разработчика.

Библиография

- Bash Reference Manual — Перевод man-страницы от 2004 года.
- Advanced Bash-Scripting Guide — Расширенное руководство по написанию bash-скриптов. Дата обращения: 6 августа 2011.
- Частые ошибки программирования на Bash. Дата обращения: 22 ноября 2010.
- Введение в программирование на bash. Дата обращения: 22 ноября 2010.
- Описание команд bash (англ.). Дата обращения: 22 ноября 2010.
- Ян Шилдс (Ian Shields). Полезные советы Linux: Параметры bash и расширения параметров.

