

Отчет по лабораторной работе 12

Программирование в командном процессоре ОС UNIX. Командные файлы.

Власов Артем Сергеевич

Содержание

1	Цель работы	1
2	Задание.....	1
3	Выполнение лабораторной работы 12.	1
4	Контрольные вопросы	5
5	Выводы.....	5
	Список литературы.....	6

1 Цель работы

Изучить основы программирования в оболочке ОС UNIX. Научиться писать небольшие командные файлы.

2 Задание

Выполнить последовательность действий по заданному сценарию, написать 4 скрипта для разных целей.

3 Выполнение лабораторной работы 12.

Создание файла первого скрипта и изменение его прав доступа. (рис. fig. 1).

```
[vlasov@vbox ~]$ touch ex1.sh  
[vlasov@vbox ~]$ chmod 777 ex1.sh
```

Рис. 1: Создание файла первого скрипта и изменение его прав доступа

Код первого скрипта(бэкап в архиве). (рис. fig. 2).

```
tar -cvf ~/backup/ex1back.tar $0
```

Рис. 2: Первый скрипт

Проверка работы первого скрипта. (рис. fig. 3).

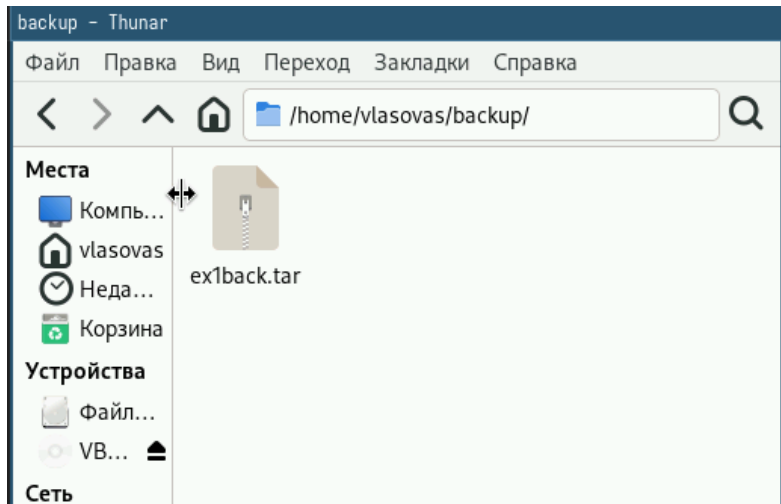


Рис. 3: Проверка

Создание файла второго скрипта и изменение его прав доступа. (рис. fig. 4).

```
vlasovas@vbox ~]$ touch ex2.sh  
[vlasovas@vbox ~]$ chmod 777 ex2.sh
```

Рис. 4: Создание файла второго скрипта и изменение его прав доступа

Код второго скрипта(вывод аргументов). (рис. fig. 5).

```
for i in "$@"  
do echo ${i}  
done
```

Рис. 5: Второй скрипт

Проверка работы второго скрипта. (рис. fig. 6).

```
[vlasovas@vbox ~]$ ./ex2.sh sdoifj soidjf astftsd sduhua  
sd asdp9asouihf asdygasydghjsd 'jkuasduihayuisd iuahsduy  
gyasdlo nhd dhsui'  
sdoifj  
soidjf  
astftsd  
sduhuasd  
asdp9asouihf  
asdygasydghjsd  
jkuasduihayuisd iuahsduygyasdlo nhd dhsui
```

Рис. 6: Проверка

Создание файла третьего скрипта и изменение его прав доступа. (рис. fig. 7).

```
[vlasovas@vbox ~]$ touch ex3.sh  
[vlasovas@vbox ~]$ chmod 777 ex3.sh
```

Рис. 7: Создание файла третьего скрипта и изменение его прав доступа

Код третьего скрипта(файлы каталога и их права доступа) (рис. fig. 8).

```
echo "$1/ " | tr -d "\n";  
stat --printf "%A" "$1/";  
echo  
  
for i in $1/*  
do echo "${i} " | tr -d "\n";  
stat --printf "%A" "${i}";  
echo  
  
done
```

Рис. 8: Третий скрипт

Проверка работа третьего скрипта(рис. fig. 9).

```
[vlasovas@vbox ~]$ ./ex3.sh ~  
/home/vlasovas/ drwx-----  
/home/vlasovas/abc1 -rw-rw-r--  
/home/vlasovas/asd -rw-r--r--  
/home/vlasovas/australia -rwxr--r--  
/home/vlasovas/backup drwxr-xr-x  
/home/vlasovas/bin drwxr-xr-x  
/home/vlasovas/conf.txt -rw-r--r--  
/home/vlasovas/Documents drwxr-xr-x  
/home/vlasovas/Downloads drwxr-xr-x  
/home/vlasovas/ex1.sh -rwxrwxrwx  
/home/vlasovas/ex2.sh -rwxrwxrwx  
/home/vlasovas/ex3.sh -rwxrwxrwx  
/home/vlasovas/feathers -rw-rw-r--  
/home/vlasovas/file.txt -rw-r--r--  
/home/vlasovas/git-extended drwxr-xr-x  
/home/vlasovas/lab11.sh -rw-r--r--  
/home/vlasovas/LICENSE -rw-r--r--  
/home/vlasovas/~logfile -rw-r--r--  
/home/vlasovas/main.cpp -rw-r--r--  
/home/vlasovas/may -rw-r--r--  
/home/vlasovas/monthly drwx--x--x  
/home/vlasovas/my_os -r-xr--r--  
/home/vlasovas/pass.txt -rw-r--r--  
/home/vlasovas/play drwxr-xr-x  
/home/vlasovas/qwe drwxr-xr-x  
/home/vlasovas/reports drwxr-xr-x  
/home/vlasovas/ski.places drwxr-xr-x  
/home/vlasovas/ski.places drwxr-xr-x
```

Рис. 9: Проверка

Создание файла четвертого скрипта и изменение его прав доступа.(рис. fig. 10).

```
[vlasovas@vbox ~]$ touch ex4.sh  
[vlasovas@vbox ~]$ chmod 777 ex4.sh
```

Рис. 10: Создание файла четвертого скрипта и изменение его прав доступа.

Код четвертого скрипта(количество файлов с заданным расширением). (рис. fig. 11).

```

let COUNTER = 0
for i in $2/*.$1
do let COUNTER++
done
echo $COUNTER

```

Рис. 11: Четвертый скрипт

Проверка работы четвертого скрипта. (рис. fig. 12).

```

[vlasovas@vbox ~]$ ./ex4.sh txt ~
4

```

Рис. 12: Проверка

4 Контрольные вопросы

Командная оболочка - это интерпретатор команд, обеспечивающий взаимодействие пользователя с операционной системой, примеры: Bash, Zsh, Ksh. POSIX - это стандарт, обеспечивающий совместимость между UNIX-подобными операционными системами. В Bash переменные объявляются как VAR="значение", массивы - ARR=("val1" "val2") или declare -A DICT=[{"key"]="value"}. Оператор let выполняет арифметические вычисления, а read используется для ввода данных пользователем. Bash поддерживает арифметические операции: +, -, *, /, %, **, ++, --. Конструкция (()) предназначена для выполнения арифметических операций и сравнений. Стандартные переменные включают \$HOME, \$PATH, \$USER, \$PWD, \$?, \$\$.

Метасимволы - это специальные символы (*, ?, >, |, & и др.), имеющие особое значение в shell. Экранировать метасимволы можно с помощью обратного слэша или кавычек. Скрипты создаются как файлы .sh, делаются исполняемыми через chmod +x и запускаются ./script.sh. Функции определяются как name() { команды } или function name { команды }. Тип файла проверяется операторами -f (обычный файл) и -d (каталог). Команда set управляет параметрами shell, typeset задает атрибуты переменных, unset удаляет переменные. Параметры передаются через \$1, \$2..., \$# (все аргументы), \$# (количество аргументов). Специальные переменные включают \$0 (имя скрипта), \$! (PID фонового процесса), \$_ (последний аргумент).

5 Выводы

Мы изучили основы программирования в оболочке ОС UNIX. Научились писать небольшие командные файлы.

Список литературы