

Отчет по лабораторной работе 14

Программирование в командном процессоре ОС UNIX. Расширенное программирование.

Власов Артем Сергеевич

Содержание

1	Цель работы	1
2	Задание.....	1
3	Выполнение лабораторной работы 14.	1
4	Контрольные вопросы	5
4.1	1. Синтаксическая ошибка	5
4.2	2. Конкатенация строк.....	5
4.3	3. Утилита seq.....	5
4.4	4. Вычисление выражения	5
4.5	5. Отличия zsh от bash.....	5
4.6	6. Проверка синтаксиса.....	5
4.7	7. Сравнение языков	6
5	Выводы.....	6
	Список литературы.....	6

1 Цель работы

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

2 Задание

Выполнить последовательность действий по заданному сценарию, написать 3 скрипта для разных целей.

3 Выполнение лабораторной работы 14.

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

```
vlasovas@vbox lab14]$ touch 1.sh
[vlasovas@vbox lab14]$ chmod 777 1.sh
[vlasovas@vbox lab14]$
```

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

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

```
1 RESOURCE_FILE="/tmp/resource.lock"
2 WAIT_TIME=5
3 USE_TIME=3
4 TERMINAL=$(tty)
5
6 echo "Процесс $$ запущен в $TERMINAL"
7
8 while true; do
9     echo "Процесс $$: Ожидание ресурса..." > $TERMINAL
10    while [ -f "$RESOURCE_FILE" ]; do
11        sleep $WAIT_TIME
12    done
13    touch "$RESOURCE_FILE"
14    echo "Процесс $$: Ресурс получен" > $TERMINAL
15
16    echo "Процесс $$: Использую ресурс ($USE_TIME сек)" >
$TERMINAL
17    sleep $USE_TIME
18
19    rm -f "$RESOURCE_FILE"
20    echo "Процесс $$: Ресурс освобожден" > $TERMINAL
21    sleep 1
22 done
```

Рис. 2: Первый скрипт

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

[vlasovas@vbox lab14]\$ tty	Процесс 23956: Ресурс получен
/dev/pts/0	Процесс 23956: Использую ресурс (3 сек)
[vlasovas@vbox lab14]\$	Процесс 23956: Ресурс освобожден
Процесс 23956 запущен в /dev/pts/1	Процесс 23956: Ожидание ресурса...
[vlasovas@vbox lab14]\$./1.sh 10	Процесс 23956: Ресурс получен
10	Процесс 23956: Использую ресурс (3 сек)
Процесс 24093 запущен в /dev/pts/0	Процесс 23956: Ресурс освобожден
Процесс 24093: Ожидание ресурса...	Процесс 23956: Ожидание ресурса...
.	Процесс 23956: Ресурс получен
Процесс 24093: Ресурс получен	Процесс 23956: Использую ресурс (3 сек)
Процесс 24093: Использую ресурс (3 сек)	Процесс 23956: Ресурс освобожден
Процесс 24093: Ресурс освобожден	Процесс 23956: Ожидание ресурса...
Процесс 24093: Ожидание ресурса...	Процесс 23956: Ресурс получен
.	Процесс 23956: Использую ресурс (3 сек)
Процесс 24093: Ресурс получен	Процесс 23956: Ресурс освобожден
Процесс 24093: Использую ресурс (3 сек)	Процесс 23956: Ожидание ресурса...
Процесс 24093: Ресурс освобожден	Процесс 23956: Ресурс получен
Процесс 24093: Ожидание ресурса...	Процесс 23956: Использую ресурс (3 сек)
.	Процесс 23956: Ресурс освобожден
Процесс 24093: Ресурс получен	Процесс 23956: Ожидание ресурса...
Процесс 24093: Использую ресурс (3 сек)	Процесс 23956: Ресурс получен
	Процесс 23956: Использую ресурс (3 сек)
	Процесс 23956: Ресурс освобожден
	Процесс 23956: Ожидание ресурса...
	Процесс 23956: Ресурс получен

Рис. 3: Проверка

Код второго скрипта (рис. fig. 4).

```

1 if [ $# -eq 0 ]; then
2     echo "Usage: $0 command_name"
3     exit 1
4 fi
5
6 man_file="/usr/share/man/man1/$1.1.gz"
7
8 if [ -f "$man_file" ]; then
9     less "$man_file"
10 else
11     echo "No manual entry for $1"
12 fi

```

Рис. 4: Второй скрипт

Запуск второго скрипта. (рис. fig. 5).

```
[vlasovas@vbox lab14]$ ./2.sh gedit
```

Рис. 5: Запуск

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

```

ESC[4mGEDITESC[24m(1)          General Commands Manual
ESC[4mGEDITESC[24m(1)

ESC[1mNAMEESC[0m
ESC[1mgedit ESC[22m- a general-purpose text editor

ESC[1mSYNOPSISESC[0m
ESC[1mgedit ESC[22m[ESC[4mOPTIONESC[24m...] [ESC[4mFILE
ESC[24m...] [+ESC[4mLINEESC[24m[:ESC[4mCOLUMNESC[24m]]
ESC[1mgedit ESC[22m[ESC[4mOPTIONESC[24m...] -

ESC[1mDESCRIPTIONESC[0m
ESC[1mgedit ESC[22mis an easy-to-use and general-purpose text editor. Its development started in 1998, at the beginnings of the GNOME project, with a good integration with that desktop environment.

You can use it to write simple notes and documents, or you can enable more advanced features that are useful for software development.

ESC[1mOPTIONSESC[0m
ESC[1m--encodingESC[0m
Set the character encoding to be used for opening the files listed on the command line.
/usr/share/man/man1/gedit.1.gz

```

Рис. 6: Проверка

Код третьего скрипта (рис. fig. 7).

```

length=${1:-10}
for ((i=0; i<length; i++)); do
    rand=$((RANDOM % 52))
    if [ $rand -lt 26 ]; then
        printf "\x$(printf %x $((65 + rand)))"
    else
        printf "\x$(printf %x $((97 + rand - 26)))"
    fi
done
echo

```

Рис. 7: Третий скрипт

Проверка работа третьего скрипта(рис. fig. 8).

```
[vlasovas@vbox lab14]$ ./3.sh
KpJDjAnvmc
[vlasovas@vbox lab14]$ ./3.sh
ZQbLakWmaF
[vlasovas@vbox lab14]$ ./3.sh
JslCvukMef
[vlasovas@vbox lab14]$ ./3.sh
DeBqZdeNHX
[vlasovas@vbox lab14]$ ./3.sh
XoiWGPcmnw
[vlasovas@vbox lab14]$
```

Рис. 8: Проверка

4 Контрольные вопросы

4.1 1. Синтаксическая ошибка

В строке `while [$1 != "exit"]` отсутствуют обязательные пробелы внутри квадратных скобок и кавычки для переменной. Правильная форма: `while ["$1" != "exit"]`.

4.2 2. Конкатенация строк

Объединение строк выполняется через простое соположение: `result="str1str2"`. Для надежности рекомендуется заключать переменные в фигурные скобки: `result="{str1}{str2}"`.

4.3 3. Утилита seq

Альтернативы генерации числовых последовательностей: - Цикл с фиксированными параметрами - Арифметический цикл `for` - Команда `printf` с диапазонами

4.4 4. Вычисление выражения

Выражение содержит некорректный синтаксис. Для целочисленного деления в `bash` используется конструкция `$((10/3))`, что дает результат 3.

4.5 5. Отличия zsh от bash

Ключевые различия включают расширенные возможности автодополнения, поддержку более сложных шаблонов поиска файлов и встроенную арифметику с плавающей точкой в `zsh`.

4.6 6. Проверка синтаксиса

Представленная конструкция содержит три ошибки: отсутствие пробела вокруг оператора сравнения, опечатку в имени переменной и лишнюю закрывающую скобку.

4.7 7. Сравнение языков

Bash оптимален для системного администрирования, но уступает универсальным языкам в вычислительной эффективности и поддерживаемых структурах данных. Основное преимущество - глубокая интеграция с shell-окружением.

5 Выводы

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

Список литературы