

Презентация к лабораторной работе №10

Старков Никита Алексеевич

¹RUDN University, Moscow, Russian Federation

Цель работы: изучить основы программирования в оболочке ОС UNIX/Linux. Научиться писать небольшие командные файлы

1) Изучаем архиваторы zip, bzip 2 и tar с помощью команды man.

```
ZIP(1L)                                ZIP(1L)

NAME
    zip - package and compress (archive) files

SYNOPSIS
    zip [-aA8cdDeffghjklmoqrRStuvVwXyz{0*}] [--longoption ...] [-b path]
        [-n suffixes] [-t date] [-tt date] [zipfile [file ...]] [-xi list]

    zipcloak (see separate man page)

    zipnote (see separate man page)

    zipsplit (see separate man page)

Note: Command line processing in zip has been changed to support long
options and handle all options and arguments more consistently. Some
old command lines that depend on command line inconsistencies may no
longer work.

DESCRIPTION
    zip is a compression and file packaging utility for Unix, VMS, MSDOS,
    OS/2, Windows 9x/NT/XP, Minix, Atari, Macintosh, Amiga, and Acorn RISC
    OS. It is analogous to a combination of the Unix commands tar(1) and
    compress(1) and is compatible with PKZIP (Phil Katz's ZIP for MSDOS
    systems).

    A companion program (unzip(1L)) unpacks zip archives. The zip and un-
    zip(1L) programs can work with archives produced by PKZIP (supporting
```

Figure 1: Архиватор zip

```
bzip2(1)                                General Commands Manual                                bzip2(1)

NAME
  bzip2, bunzip2 - a block-sorting file compressor, v1.0.8
  bzipcat - decompresses files to stdout
  bzip2recover - recovers data from damaged bzip2 files

SYNOPSIS
  bzip2 [ -cdFkqstvzVL123456789 ] [ filenames ... ]
  bunzip2 [ -fkvsVL ] [ filenames ... ]
  bzipcat [ -s ] [ filenames ... ]
  bzip2recover filename

DESCRIPTION
  bzip2 compresses files using the Burrows-Wheeler block sorting text compression algorithm, and Huffman coding. Compression is generally considerably better than that achieved by more conventional LZW/LZ78-based compressors, and approaches the performance of the PPM family of statistical compressors.

  The command-line options are deliberately very similar to those of GNU gzip, but they are not identical.

  bzip2 expects a list of file names to accompany the command-line flags. Each file is replaced by a compressed version of itself, with the name "original_name.bz2". Each compressed file has the same modification date, permissions, and, when possible, ownership as the corresponding original, so that these properties can be correctly restored at decompression time. File name handling is naive in the sense that there is no mechanism for preserving original file names, permissions, ownerships or dates in filesystems which lack these concepts, or have serious file name length restrictions, such as MS-
```

Figure 2: Архиватор bzip2

DESCRIPTION

GNU `tar` is an archiving program designed to store multiple files in a single file (an archive), and to manipulate such archives. The archive can be either a regular file or a device (e.g. a tape drive, hence the name of the program, which stands for `tape archiver`), which can be located either on the local or on a remote machine.

Option styles

Options to GNU `tar` can be given in three different styles. In **traditional style**, the first argument is a cluster of option letters and all subsequent arguments supply arguments to those options that require them. The arguments are read in the same order as the option letters. Any command line words that remain after all options has been processed are treated as non-optional arguments: file or archive member names.

For example, the `c` option requires creating the archive, the `v` option requests the verbose operation, and the `f` option takes an argument that sets the name of the archive to operate upon. The following command, written in the traditional style, instructs `tar` to store all files from the directory `/etc` into the archive file `etc.tar` verbosely listing the files being archived:

```
tar cfv etc.tar /etc
```

Figure 3: Архиватор tar

Создаем файл, в котором будем писать скрипт и открываем emacs

```
nastarkov@dk3n59 ~ $ touch backup.sh  
nastarkov@dk3n59 ~ $ emacs &
```

Figure 4: Создание файла backup.sh

Создание скрипта 1

Пишем скрипт, который при запуске будет делать резервную копию самого себя (то есть файла, в котором содержится его исходный код) в другую директорию backup в вашем домашнем каталоге. При этом файл должен архивироваться одним из архиваторов на выбор zip, bzip2 или tar.

```
#!/bin/bash
name='backup.sh'          #В переменную name сохраняем файл со скриптом
mkdir ~/backup             #Создаем каталог ~/backup
bzip2 -k ${name}           #Архивируем скрипт
mv ${name}.bz2 ~/backup    #Перемещаем архивированный скрипт в каталог ~/backup
echo 'Выполнено'
```

```
--- backup.sh All L7 (Shell-script[sh]) 4т мая 19 16:20 0.96
Warning (initialization): An error occurred while loading '~/.emacs':
```

```
error: Package 'fira-code-mode-' is unavailable
```

```
To ensure normal operation, you should investigate and remove the
cause of the error in your initialization file. Start Emacs with
the '--debug-init' option to view a complete error backtrace.
```

```
□
```

```
;%*- *Warnings* All L8 (Special) 4т мая 19 16:20 0.96
Wrote /afs/.dk.sci.pfu.edu.ru/home/n/a/nastarkov/backup.sh
```

Figure 5: Скрипт №01

Проверяем работу скрипта, предварительно добавив для него право на выполнение

A terminal window showing a command prompt. The prompt is 'nastarkov@dk3n59 ~ \$' and the command entered is 'chmod +x *.sh'. The command is highlighted in blue.

```
nastarkov@dk3n59 ~ $ chmod +x *.sh
```

Figure 6: Проверка работы скрипта



```
nasarkov@dk259 ~$ ./backup.sh
mkdir: невозможно создать каталог '/afs/dk.sci.pfu.edu.ru/home/n/a/nasarkov/backup': файл существует
Выполнено
```

Figure 7: Проверка работы скрипта

Создание скрипта 1

```
nastarkov@dk3n59 ~ $ ./backup.sh
md5r: невозможно создать каталог ~/afs/dk.sci.pfu.edu.ru/home/n/a/nastarkov/backup: файл существует
Выполнено
[4] Завершил эмса
nastarkov@dk3n59 ~ $ cd backup/
nastarkov@dk3n59 ~/backup $ ls
backup.sh bz2
nastarkov@dk3n59 ~/backup $ bunzip2 -c backup.sh.bz2
#! /bin/bash

name="backup.sh"      ## переменная name сохранен файл со скриптом
md5r -r ~/backup      ##Создан каталог ~/backup
bzzip2 -k $(name)      ##Архивируем скрипт
mv $(name).bz2 ~/backup ##Перемещаем архивированный скрипт в каталог ~/backup
echo "Выполнено"
```

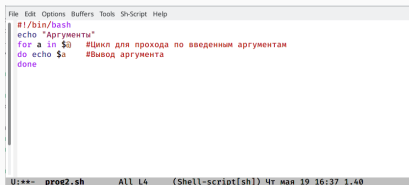
Figure 8: Проверка работы скрипта

2) Создаем файл, в котором будем писать второй скрипт и так же открываем его в emacs.

```
nastarkov@bk.ru: ~/backup $ cd -  
nastarkov@bk.ru: $ touch prog2.sh  
nastarkov@bk.ru: $ emacs
```

Figure 9: Создание файла prog2.sh

Написал пример командного файла, обрабатывающего любое произвольное число аргументов командной строки, в том числе превышающее десять. Например, скрипт может последовательно распечатывать значения всех переданных аргументов



```
File Edit Options Buffers Tools Sh-Script Help
#!/bin/bash
echo "Аргументы"
for a in $@ #Цикл для прохода по введенным аргументам
do echo $a #Вывод аргумента
done

U:*** prog2.sh All L4 (Shell-script[sh]) 4т мая 19 16:37 1.40
```

Figure 10: Скрипт №2

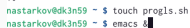
Проверил работу данного скрипта, предварительно добавив для него право на выполнение

```
nastarkov@dk3n33 - $ chmod +x *.sh
nastarkov@dk3n33 - $ ls
abc1      conf.txt  f3.txt   lab07.sh  play      text.txt  Музыка
astralia  course-directory-student-template  f4.txt   lab07.sh-  prog2.sh  work      Общедоступные
backup.sh f1.txt   f4.txt   lab.cpp   prog2.sh- 'Рабочий стол'
backup.sh- f3.txt   feathera monthly  public_html  Документы
bin       f2.txt   file.txt my.sh     reports      Заргузан
blog      f2.txt   OHexstep 'NOVEL KATALOG'  ski.places  Изображения

nastarkov@dk3n33 - $ ./prog2.sh 0 1 2 3 4
Apргymenru
0
1
2
3
4
nastarkov@dk3n33 - $ ./prog2.sh 0 1 2 3 4 5 6 7 8 9 10
Apргymenru
0
1
2
3
4
5
6
7
8
9
10
```

Figure 11: Проверка работы скрипта

3) Создаем файл для третьего скрипта и открываем в редакторе emacs



```
nastarkov@dk3n59 ~ $ touch progl.s.sh
nastarkov@dk3n59 ~ $ emacs &
```

Figure 12: Создание файла progl.s.sh

Создание скрипта 3

Пишем командный файл аналог команды ls (без использования самой этой команды и команды dir). Требуется, чтобы он выдавал информацию о нужном каталоге и выводил информацию о возможностях доступа к файлам этого каталога.

```
#!/bin/bash
a="$1"
for i in ${a}/*
do
    echo "$i"

    if test -f $i
    then echo "Обычный файл"
    fi

    if test -s $i
    then echo "Каталог"
    fi

    if test -r $i
    then echo "Чтение разрешено"
    fi

    if test -w $i
    then echo "Запись разрешена"
    fi

    if test -x $i
    then echo "Выполнение разрешено"
    fi
done
```

Figure 13: Скрипт №3


Далее проверяем работу скрипта, предварительно добавив для него право на выполнение

```
nastarkov@bk3519 ~$ chmod +x *.sh
nastarkov@bk3519 ~$ ls
abc1      conf.txt  f3.txt    lab07.sh  play      reports   Зеркала
australia course-directory-student-template f3.txt    lab07.sh~ prog2.sh  ski_places Иллюстрации
backup    equipment f4.txt    lab.cpp   prog2.sh~ text.txt  Музыка
backup.sh f1.txt    f4.txt~   may       progls.sh top       Общеобразователь
backup.sh~ f1.txt~   feathers  monthly   progls.sh~ work     "Рабочие стол"
bin       f2.txt    file.txt  my_os     public    Видео     Школы
blog      f2.txt~   OWStap    "MOVIE KATALOG" public_html Документы

nastarkov@bk3519 ~$ ./proglis.sh -
/aFu/.dk.sci.pfu.edu.ru/home/n/a/nastarkov/abc1
Обычный файл
Имя: разрешено
Затис: разрешено
/aFu/.dk.sci.pfu.edu.ru/home/n/a/nastarkov/australia
Каталог
Имя: разрешено
Затис: разрешено
Выполнение: разрешено
/aFu/.dk.sci.pfu.edu.ru/home/n/a/nastarkov/backup
Каталог
Имя: разрешено
Затис: разрешено
Выполнение: разрешено
/aFu/.dk.sci.pfu.edu.ru/home/n/a/nastarkov/backup.sh
Обычный файл
Каталог
Имя: разрешено
Затис: разрешено
Выполнение: разрешено
/aFu/.dk.sci.pfu.edu.ru/home/n/a/nastarkov/backup.sh~
Обычный файл
Каталог
Имя: разрешено
Затис: разрешено
Выполнение: разрешено
/aFu/.dk.sci.pfu.edu.ru/home/n/a/nastarkov/bin
Каталог
Имя: разрешено
Затис: разрешено
```

Figure 14: Перемещение курсора в конец буфера обмена.

4) Создаем файл для четвертого скрипта и открываем в редакторе emacs

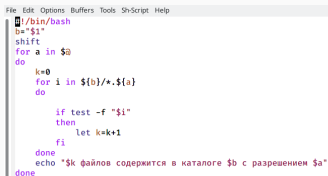


```
nestarkov@k3n59 ~ % touch format.sh
nestarkov@k3n59 ~ % emacs &
```

Figure 15: Создание файла format.sh

Создание скрипта 4

Пишем командный файл, который получает в качестве аргумента командной строки формат файла (.txt,.doc,.jpg,.pdfи т.д.) и вычисляет количество таких файлов в указанной директории. Путь к директории также передаётся в виде аргумента командной строки



```
File Edit Options Buffers Tools Sh-Script Help
#!/bin/bash
b="$1"
shift
for a in $@
do
    k=0
    for i in ${b}/+.*${a}
    do
        if test -f "$i"
        then
            let k=k+1
        fi
    done
    echo "$k файлов содержится в каталоге $b с разрешением $a"
done
```

Figure 16: Скрипт №4

Далее проверяем работу скрипта, предварительно добавив для него право на выполнение

```
nastarkov@ok3555 ~$ touch file.pdf file.doc file2.doc
nastarkov@ok3555 ~$ ls
australia  course-directory-student-template  f3.txt  file.txt  my_05  public  Изображения
backup    equipment                            f4.txt  format.sh  'NOVY KATALOG'  public_html  Музыка
backup.sh  f1.txt                               f4.txt~  format.sh~  play          text.txt  Общедоступные
backup.sh~ f3.txt~                              f4thors  OWBtop    prog2.sh      work      "Рабочий стол"
bin       f2.txt                               file2.doc  lab07.sh  prog2.sh~    video     Видео
blog      f2.txt~                              file.doc  lab07.sh~  prog1a.sh    документы
conf.txt  f2.txt                               file.pdf  nag       prog1a.sh~   Зеркала

nastarkov@ok3555 ~$ ./format.sh - pdf sh txt doc
1 файл(ов) содержится в каталоге /afs/.dk.sci.pfu.edu.ru/home/n/a/nastarkov с разрешением pdf
5 файл(ов) содержится в каталоге /afs/.dk.sci.pfu.edu.ru/home/n/a/nastarkov с разрешением sh
7 файл(ов) содержится в каталоге /afs/.dk.sci.pfu.edu.ru/home/n/a/nastarkov с разрешением txt
2 файл(ов) содержится в каталоге /afs/.dk.sci.pfu.edu.ru/home/n/a/nastarkov с разрешением doc
```

Figure 17: Проверка работы скрипта

Вывод: в ходе выполнения лабораторной работы я изучил основы программирования в оболочке ОС UNIX/Linux, научился писать небольшие командные файлы