

Лабораторная работа №12

Операционные системы

Панявкина И.В.

28 апреля 2025

Российский университет дружбы народов, Москва, Россия

Цель данной лабораторной работы - изучить основы программирования в оболочке ОС UNIX/Linux, научиться писать небольшие командные файлы.

1. Написать 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. Способ использования команд архивации необходимо узнать, изучив справку.

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

3. Написать 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.

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

Командный процессор (командная оболочка, интерпретатор команд shell) — это программа, позволяющая пользователю взаимодействовать с операционной системой компьютера. В операционных системах типа UNIX/Linux наиболее часто используются следующие реализации командных оболочек:

1. оболочка Борна (Bourne shell или sh) — стандартная командная оболочка UNIX/Linux, содержащая базовый, но при этом полный набор функций;
2. C-оболочка (или csh) — надстройка на оболочкой Борна, использующая C-подобный синтаксис команд с возможностью сохранения истории выполнения команд;
3. оболочка Корна (или ksh) — напоминает оболочку C, но операторы управления программой совместимы с операторами оболочки Борна;
4. BASH — сокращение от Bourne Again Shell (опять оболочка Борна), в основе своей совмещает свойства оболочек C и Корна (разработка компании Free Software Foundation).

POSIX (Portable Operating System Interface for Computer Environments) — набор стандартов описания интерфейсов взаимодействия операционной системы и прикладных программ. Стандарты POSIX разработаны комитетом IEEE (Institute of Electrical and Electronics Engineers) для обеспечения совместимости различных UNIX/Linux-подобных операционных систем и переносимости прикладных программ на уровне исходного кода. POSIX-совместимые оболочки разработаны на базе оболочки Корна. Рассмотрим основные элементы программирования в оболочке `bash`. В других оболочках большинство команд будет совпадать с описанными ниже.

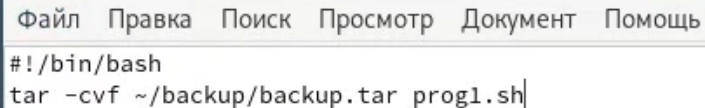
Создаю файл `prog1.sh` в котором буду писать программу с расширением `sh` (shell) с помощью утилиты `touch`, далее делаю его исполняемым с помощью `chmod +x`, открываю файл в текстовом редакторе, пишу в нем код, и после того как я написала программу в файле, я могу его запустить `bash` (рис. 1).

Выполнение лабораторной работы

```
[irinapanyavkina@irinapanyavkina ~]$ touch prog1.sh
[irinapanyavkina@irinapanyavkina ~]$ 
[irinapanyavkina@irinapanyavkina ~]$ 
[irinapanyavkina@irinapanyavkina ~]$ 
[irinapanyavkina@irinapanyavkina ~]$ 
[irinapanyavkina@irinapanyavkina ~]$ 
[irinapanyavkina@irinapanyavkina ~]$ 
[irinapanyavkina@irinapanyavkina ~]$ 
[irinapanyavkina@irinapanyavkina ~]$ 
[irinapanyavkina@irinapanyavkina ~]$ 
[irinapanyavkina@irinapanyavkina ~]$ 
[irinapanyavkina@irinapanyavkina ~]$ 
[irinapanyavkina@irinapanyavkina ~]$ 
[irinapanyavkina@irinapanyavkina ~]$ 
[irinapanyavkina@irinapanyavkina ~]$ 
[irinapanyavkina@irinapanyavkina ~]$ 
[irinapanyavkina@irinapanyavkina ~]$ 
[irinapanyavkina@irinapanyavkina ~]$ 
[irinapanyavkina@irinapanyavkina ~]$ 
[irinapanyavkina@irinapanyavkina ~]$ 
[irinapanyavkina@irinapanyavkina ~]$ 
[irinapanyavkina@irinapanyavkina ~]$ 
[irinapanyavkina@irinapanyavkina ~]$ 
[irinapanyavkina@irinapanyavkina ~]$ 
[irinapanyavkina@irinapanyavkina ~]$ 
[irinapanyavkina@irinapanyavkina ~]$ 
[irinapanyavkina@irinapanyavkina ~]$ 
[irinapanyavkina@irinapanyavkina ~]$ 
[irinapanyavkina@irinapanyavkina ~]$ 
[irinapanyavkina@irinapanyavkina ~]$ 
[irinapanyavkina@irinapanyavkina ~]$ 
[irinapanyavkina@irinapanyavkina ~]$ 
[irinapanyavkina@irinapanyavkina ~]$ 
[irinapanyavkina@irinapanyavkina ~]$ 
[irinapanyavkina@irinapanyavkina ~]$ 
tar: /home/irinapanyavkina/backup/backup.tar: Функция open завершилась с ошибкой: Нет такого файла или каталога
tar: Error is not recoverable: exiting now
[irinapanyavkina@irinapanyavkina ~]$ mkdir backup
[irinapanyavkina@irinapanyavkina ~]$ bash prog1.sh
prog1.sh
```

Рис. 1: Выполнение программы 1

скрипт, который при запуске будет делать резервную копию самого себя (то есть файла, в котором содержится его исходный код) в другую директорию backup в моем домашнем каталоге. При этом файл должен архивироваться одним из архиваторов на выбор zip, bzip2 или tar (рис. 2).



The image shows a terminal window with a menu bar at the top containing the items: 'Файл', 'Правка', 'Поиск', 'Просмотр', 'Документ', and 'Помощь'. Below the menu bar, the first line of the script is the shebang '#!/bin/bash'. The second line is the command 'tar -cvf ~/backup/backup.tar prog1.sh', followed by a vertical cursor indicating the end of the line.

```
Файл  Правка  Поиск  Просмотр  Документ  Помощь
#!/bin/bash
tar -cvf ~/backup/backup.tar prog1.sh|
```

Рис. 2: Написанная программа 1

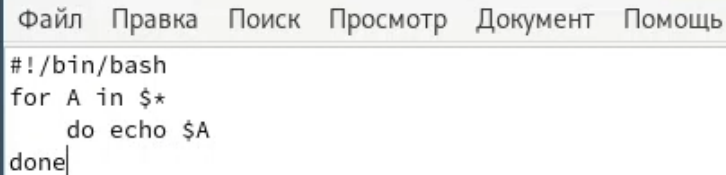
Создаю файл `prog2.sh` в котором буду писать программу с расширением `sh` (shell) с помощью утилиты `touch`, далее делаю его исполняемым с помощью `chmod +x`, открываю файл в текстовом редакторе, пишу в нем код, и после того как я написала программу в файле, я могу его запустить `bash` (рис. 3).

Выполнение лабораторной работы

```
(irinapanyavkina@irinapanyavkina ~)$ touch prog2.sh
(irinapanyavkina@irinapanyavkina ~)$ ls
abc1      bin          dotfl        feathers    git-extended lab07.sh    monthly      pandoc-3.1.11.1-linux-amd64.tar.gz  play
australia 'd (1-я копия).md' dotfiles     file.txt   gitflow     lab07.sh~  my_os       pandoc-crossref.1                  prog1.sh
backup    d.md        dotfiles-template fun         iloveos     may         pandoc-3.1.11.1                  pandoc-crossref-linux.tar.xz      prog2.sh
(irinapanyavkina@irinapanyavkina ~)$ chmod +x prog2.sh
(irinapanyavkina@irinapanyavkina ~)$ bash prog2.sh 3o eo ht 5 7 sk dk ok 1 i s e heii ooo 22 3 4 oil
3o
eo
ht
5
7
sk
dk
ok
1
i
s
e
heii
ooo
22
3
4
oil
(irinapanyavkina@irinapanyavkina ~)$
```

Рис. 3: Выполнение программы 2

Пример командного файла, обрабатывающего любое произвольное число аргументов командной строки, в том числе превышающее десять. Например, скрипт может последовательно распечатывать значения всех переданных аргументов (рис. 4).



The image shows a screenshot of a text editor window. The title bar at the top contains the text: "Файл Правка Поиск Просмотр Документ Помощь". The editor area contains the following text:

```
#!/bin/bash
for A in $*
do echo $A
done|
```

Рис. 4: Написанная программа 2

Создаю файл `prog3.sh` в котором буду писать программу с расширением `sh` (shell) с помощью утилиты `touch`, далее делаю его исполняемым с помощью `chmod +x`, открываю файл в текстовом редакторе, пишу в нем код, и после того как я написала программу в файле, я могу его запустить `bash` (рис. 5).

Выполнение лабораторной работы

```
[irinapanyavkina@irinapanyavkina ~]$ touch prog3.sh
[irinapanyavkina@irinapanyavkina ~]$ chmod +x prog3.sh
[irinapanyavkina@irinapanyavkina ~]$ bash prog3.sh iloveos/
abc1: is a file and writeable
readable
australia^ is a directory
backup^ is a directory
bin^ is a directory
d (1-я копия).md: is a file and prog3.sh: строка 9: test: слишком много аргументов
d.md: is a file and writeable
readable
dotf1^ is a directory
dotfiles^ is a directory
dotfiles-template^ is a directory
feathers: is a file and writeable
readable
file.txt: is a file and writeable
readable
fun^ is a directory
git-extended^ is a directory
gitflow^ is a directory
iloveos^ is a directory
lab07.sh: is a file and writeable
readable
lab07.sh~: is a file and writeable
readable
may: is a file and writeable
readable
monthly^ is a directory
my_os: is a file and pandoc-3.1.11.1^ is a directory
pandoc-3.1.11.1-linux-amd64.tar.gz: is a file and writeable
readable
```

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. (рис. 6).

Файл Правка Поиск Просмотр Документ Помощь

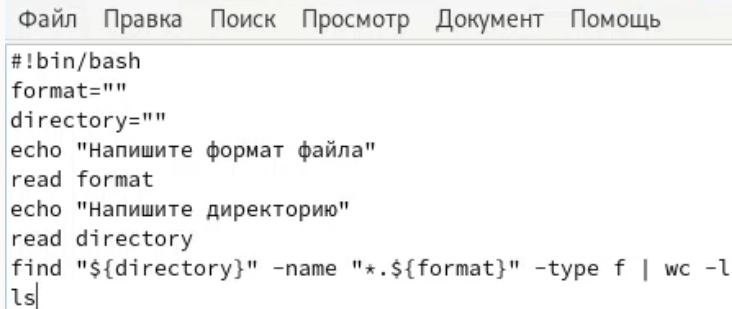
```
#!/bin/bash
for A in *
do
    if test -d "$A"
    then
        echo "$A^ is a directory"
    else
        echo -n "$A: is a file and "
        if test -w $A
        then
            echo writeable
            if test -r $A
            then
                echo "readable"
            else
                echo "neither readable or writeable"
            fi
        fi
    fi
done
```

Создаю файл `prog4.sh` в котором буду написать программу с расширением `sh` (shell) с помощью утилиты `touch`, далее делаю его исполняемым с помощью `chmod +x`, открываю файл в любом текстовом редакторе, пишу в нем код (рис. 7).

```
[irinapanyavkina@irinapanyavkina ~]$ touch prog4.sh  
[irinapanyavkina@irinapanyavkina ~]$ chmod +x prog4.sh
```

Рис. 7: Создание файла с программой 4

Командный файл, который получает в качестве аргумента командной строки формат файла (.txt, .doc, .jpg, .pdf и т.д.) и вычисляет количество таких файлов в указанной директории. Путь к директории также передаётся в виде аргумента командной строки (рис. 8).



The image shows a terminal window with a menu bar at the top containing the items: 'Файл', 'Правка', 'Поиск', 'Просмотр', 'Документ', and 'Помощь'. Below the menu bar, the following shell script is displayed:

```
#!/bin/bash
format=""
directory=""
echo "Напишите формат файла"
read format
echo "Напишите директорию"
read directory
find "${directory}" -name "*.${format}" -type f | wc -l
ls|
```

Рис. 8: Написанная программа 4

И после того как я написала программу в файле, я могу его запустить bash (рис. 9).

```
[irinapanyavkina@irinapanyavkina ~]$ bash prog4.sh
Напишите формат файла
txt
Напишите директорию
/home/irinapanyavkina
20
abcl      bin          dotfl      feathers  git-extended  lab07.sh  monthly    pandoc-3.1.11.1-linux-amd64.tar.gz  play    prog3.sh  re
australia 'd (1-я копия).md' dotfiles   file.txt   gitflow    lab07.sh~  my_os      pandoc-crossref.1  progl.sh prog4.sh  sk
backup    d.md          dotfiles-template fun        iloveos     may        pandoc-3.1.11.1  pandoc-crossref-linux.tar.xz  prog2.sh README.md sn
[irinapanyavkina@irinapanyavkina ~]$
```

Рис. 9: Выполнение программы 4

Выводы

При выполнении данной лабораторной работы я изучила основы программирования в оболочке ОС UNIX/Linux, научилась писать небольшие командные файлы.

Список литературы

1. Лабораторная работа №12 [Электронный ресурс] URL:
<https://esystem.rudn.ru/mod/resource/view.php?id=1224391>