

Лабораторная работа №12

талебу тенке франк

¹RUDN University, Moscow, Russian Federation

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.

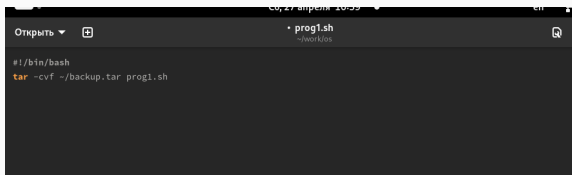
3. Написать командный файл — аналог команды `ls` (без использования самой этой команды и команды `dir`). Требуется, чтобы он выдавал информацию о нужном каталоге и выводил информацию о возможностях доступа к файлам этого каталога.
4. Написать командный файл, который получает в качестве аргумента командной 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:

- оболочка Борна (Bourne shell или sh) — стандартная командная оболочка UNIX/Linux, содержащая базовый, но при этом полный набор функций;
- C-оболочка (или csh) — надстройка на оболочкой Борна, использующая C-подобный синтаксис команд с возможностью сохранения истории выполнения команд;
- оболочка Корна (или ksh) — напоминает оболочку C, но операторы управления программой совместимы с операторами оболочки Борна;
- 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.

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. рис. ((fig:001?; fig:002?; fig:003?))



A screenshot of a terminal window. The title bar at the top shows "Открыть" (Open) on the left, "prog1.sh" in the center, and a close button on the right. Below the title bar, the terminal content shows the prompt "#!/bin/bash" followed by the command "tar -cvf -/backup.tar prog1.sh". The command is partially highlighted in orange.

```
#!/bin/bash
tar -cvf -/backup.tar prog1.sh
```

Рис. 1: Текст первой программы

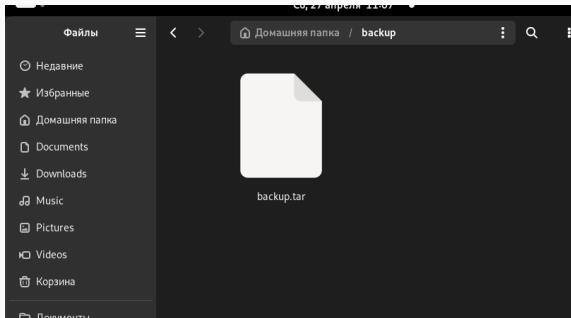
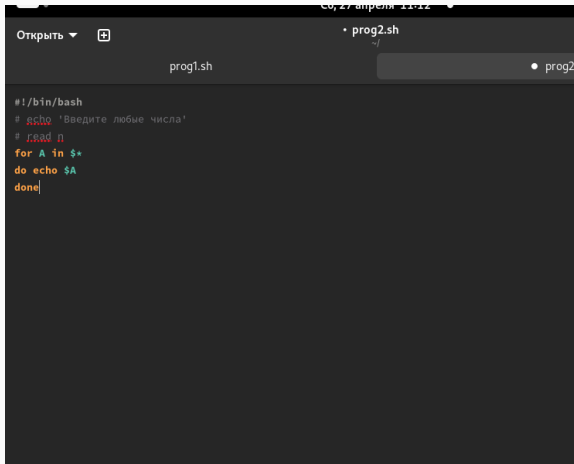


Рис. 2: Проверка работы программы

Создание файла для второй программы, проверка содержимого
домашнего каталога

Рис. 3: Создание файла для второй программы, проверка содержимого
домашнего каталога

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. ((fig:004?; fig:005?))



```
#!/bin/bash
# echo 'Введите любые числа'
# read n
for A in $*
do echo $A
done
```

Рис. 4: Текст второй программы

Выполнение лабораторной работы

```
done
thouston@username:~$ bash prog2.sh 1 2 3 4 3 2 3 4 5 7 1
1
2
3
4
3
2
3
4
5
7
1
thouston@username:~$ 1 2 3 4 52 4 354 32 56 23 78
bash: 1: команда не найдена...
thouston@username:~$ bash prog2.sh 35 4872 452 372 482
35
4872
452
372
482
thouston@username:~$
```

Рис. 5: Проверка работы второй программы

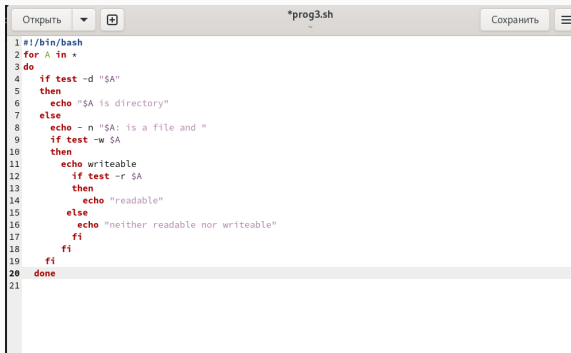
3. Написать командный файл — аналог команды `ls` (без использования самой этой команды и команды `dir`). Требуется, чтобы он выдавал информацию о нужном каталоге и выводил информацию о возможностях доступа к файлам этого каталога.
((fig:006?; fig:007?; fig:008?))

Выполнение лабораторной работы

```
thouston@username:~$ 1 2 3 4 52 4 354 32 56 23 78
bash: 1: команда не найдена...
thouston@username:~$ bash prog2.sh 35 4872 452 372 482
35
4872
452
372
482
thouston@username:~$ touch prog3.sh
thouston@username:~$ ls
abc1      git-extended  ls           prog3.sh      Загрузки
backup    git_repo      may          project_0C    Изображения
backup.tar '#lab07.sh#'  misk         reports       Музыка
bin        lab07.sh      monthly     ski.places    Общедоступные
conf.txt   lab07.sh~    nemos        text.txt      'Рабочий стол'
Desktop    letters       newdir       work          Шаблоны
Downloads  LICENSE       Pictures     Видео
'file (копия).txt' logfilr       prog2.sh     Документы
thouston@username:~$
```

Рис. 6: Создание файла для третьей программы

Выполнение лабораторной работы



```
1 #!/bin/bash
2 for A in *
3 do
4     if test -d "$A"
5     then
6         echo "$A is directory"
7     else
8         echo -n "$A: is a file and "
9         if test -w $A
10        then
11            echo writeable
12            if test -r $A
13            then
14                echo "readable"
15            else
16                echo "neither readable nor writeable"
17            fi
18        fi
19    fi
20 done
21
```

Рис. 7: Текст третьей программы

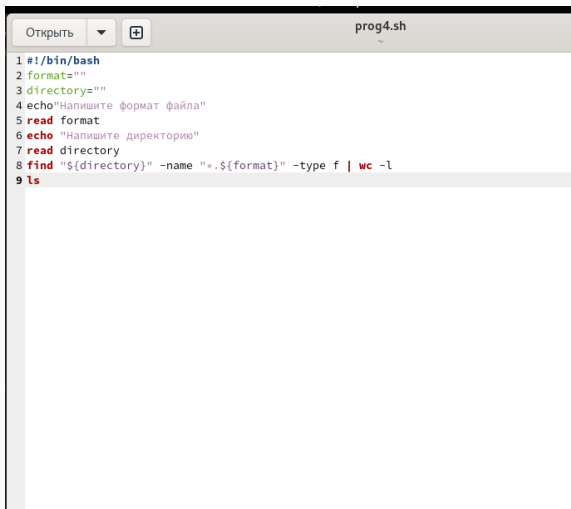
Выполнение лабораторной работы

```
thouston@username:~$ bash prog3.sh
abc1: is a file andwriteable
readable
backup: is a directory
backup.tar: is a file andwriteable
readable
bin: is a directory
conf.txt: is a file andwriteable
readable
Desktop: is a directory
Downloads: is a directory
prog3.sh: строка 4: test: file: ожидается бинарный оператор
file (копия).txt: is a file andprog3.sh: строка 9: test: file: ожидается бинарный опе
git-extended: is a directory
git_repo: is a directory
#lab07.sh#: is a file andwriteable
readable
lab07.sh: is a file andwriteable
readable
lab07.sh~: is a file andwriteable
readable
letters: is a directory
LICENSE: is a file andwriteable
readable
logfile: is a file andwriteable
readable
ls: is a directory
may: is a file andwriteable
readable
misk: is a directory
monthly: is a directory
nemos: is a directory
newdir: is a directory
Pictures: is a directory
```

Рис. 8: Проверка работы третьей программы

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. ((fig:009?; fig:010?))

Выполнение лабораторной работы



The image shows a terminal window with a title bar containing the text "Открыть" (Open), a dropdown arrow, and a plus icon. The window title is "prog4.sh". The terminal content is as follows:

```
1 #!/bin/bash
2 format=""
3 directory=""
4 echo "Напишите формат файла"
5 read format
6 echo "Напишите директорию"
7 read directory
8 find "${directory}" -name "*.${format}" -type f | wc -l
9 ls
```

Рис. 9: Текст четвертой программы

Выполнение лабораторной работы

```
thouston@username:/home/thouston
thouston@username:~$ gedit prog4.sh
thouston@username:~$ bash prog4.sh
Напишите формат файла
txt
Напишите директорию
/home/thouston
22
abc1      'file (копия).txt'  LICENSE  newdir      ski.places  Музыка
backup    git-extended        logfilr   Pictures    text.txt    Общедоступные
backup.tar git_repo            ls        prog2.sh    work        'Рабочий стол'
bin        '#lab07.sh#'        may       prog3.sh    Видео       Шаблоны
conf.txt   lab07.sh            misk      prog4.sh    Документы
Desktop    lab07.sh~           monthly   project_OC  Загрузки
Downloads  letters             nemos     reports     Изображения
thouston@username:~$ bash prog4.sh
Напишите формат файла
/home/thouston
Напишите директорию
js
find: warning: '-name' matches against basenames only, but the given pattern contains a directory
separator ('/'), thus the expression will evaluate to false all the time. Did you mean '-wholen
ame'?
find: 'js': Нет такого файла или каталога
0
abc1      'file (копия).txt'  LICENSE  newdir      ski.places  Музыка
backup    git-extended        logfilr   Pictures    text.txt    Общедоступные
backup.tar git_repo            ls        prog2.sh    work        'Рабочий стол'
bin        '#lab07.sh#'        may       prog3.sh    Видео       Шаблоны
conf.txt   lab07.sh            misk      prog4.sh    Документы
Desktop    lab07.sh~           monthly   project_OC  Загрузки
Downloads  letters             nemos     reports     Изображения
thouston@username:~$
```

Рис. 10: Проверка работы четвертой программы

В процессе выполнения лабораторной работы я изучила основы программирования в оболочке ОС UNIX/Linux. Научилась писать небольшие командные файлы.

1. Лабораторная работа № 10. Программирование в командном процессоре ОС UNIX. Командные файлы [Электронный ресурс]. URL: <https://esystem.rudn.ru/>.

Спасибо за внимание!