

Отчет по лабораторной работе № 10

Российский университет дружбы народов

Абдулгалимов Мурад

Содержание

Цель работы	5
Задание	6
Выполнение лабораторной работы	7
Написал 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. (рис. [-@fig:001])	7
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. (рис. [-@fig:002])	8
Написал командный файл — аналог команды ls (без использования самой этой команды и команды dir). Требовалось, чтобы он выдавал информацию о нужном каталоге и выводил информацию о возможностях доступа к файлам этого каталога. (рис. [-@fig:003])	9
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. (рис. [-@fig:004])	10
Выводы	11
Контрольные вопросы	12

Список иллюстраций

0.1 Скрипт 1	7
0.2 Скрипт 2	8
0.3 Скрипт 3	9
0.4 Скрипт 4	10

Список таблиц

Цель работы

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

Задание

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.

Выполнение лабораторной работы

Написал 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. (рис. [-@fig:001])

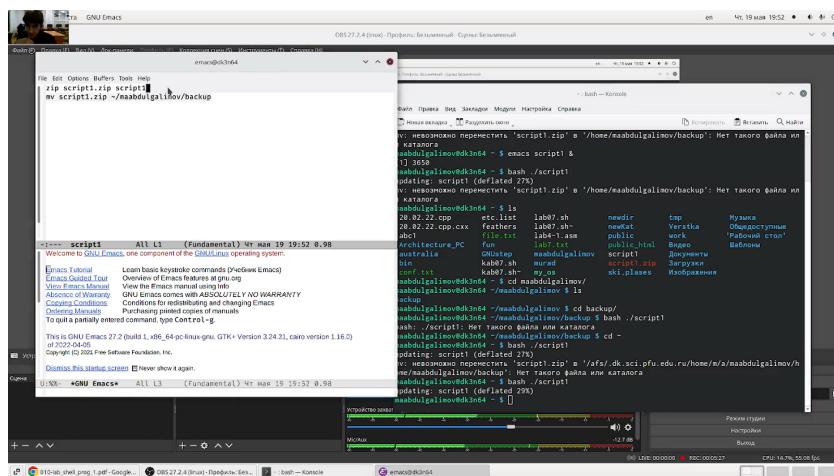


Рис. 0.1: Скрипт 1

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. (рис. [-@fig:002])

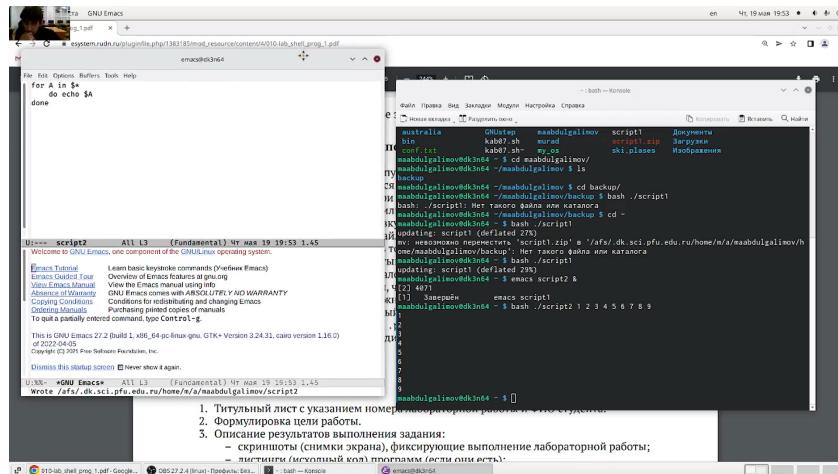


Рис. 0.2: Скрипт 2

Написал командный файл — аналог команды ls (без использования самой этой команды и команды dir). Требовалось, чтобы он выдавал информацию о нужном каталоге и выводил информацию о возможностях доступа к файлам этого каталога. (рис. [-@fig:003])

```

#!/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 -r $A
        then echo "readable"
        elif test -w $A
        then echo "writeable"
        else echo neither readable nor writeable
        fi
    fi
done

```

U:--- script3.sh All L12 (Shell-script[sh]) Чт май 19 20:05 1.81
Welcome to GNU Emacs, one component of the [GNU/Linux](#) operating system.
Emacs Tutorial Learn basic keystroke commands (Preface Emacs)
Emacs User's Manual View the Emacs manual using Info
Vito's Emacs Manual View the Emacs manual using Info
Absence of Warranty GNU Emacs comes with ABSOLUTELY NO WARRANTY
Copyright © 2002 Free Software Foundation, Inc.
Purchasing printed copies of manuals Ordering Manuals
To quit a partially entered command type Control-C
This is GNU Emacs 22.2 (build 1, #6, 64-bit Linux-gnu, GTK+ Version 3.24.31, cairo version 1.16.0)
of 2022-04-11. Copyright (C) 2022 Free Software Foundation, Inc.
Документация: /usr/share/info/emacs.info
Помощь: /usr/share/info/emacs.info
Рабочий стол: /usr/share/info/emacs.info
Скрипты: /usr/share/info/emacs.info
Музыка: /usr/share/info/emacs.info
Общие темы: /usr/share/info/emacs.info
Логотипы: /usr/share/info/emacs.info
Рабочий стол: /usr/share/info/emacs.info
Скрипты: /usr/share/info/emacs.info
neither readable nor writeable
nobody: is a directory
nobody: is a file and readable
nobody: is a file and writable
nobody: is a file and neither readable nor writable
nobody: is a file and writeable
script1.sh: is a file and readable
script1.sh: is a file and writeable
script2.sh: is a file and readable
script2.sh: is a file and writeable
script3.sh: is a file and readable
script3.sh: is a file and writeable
skiplines: is a directory
тест: is a file
Verstaka: is a directory
work: is a directory
Базы данных: is a directory
Документы: is a directory
Заргузки: is a directory
Команды: is a directory
Музыка: is a directory
Общие темы: is a directory
Логотипы: is a directory
Рабочий стол: is a file and ./script3.sh строка 5: test Рабочий: ожидается бинарный оператор
/script3.sh: строка 7: test Рабочий: ожидается бинарный оператор
neither readable nor writeable
nobody: is a directory
nobody: is a file and readable
nobody: is a file and writable
nobody: is a file and neither readable nor writable
nobody: is a file and writeable

Рис. 0.3: Скрипт 3

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.
(рис. [-@fig:004])

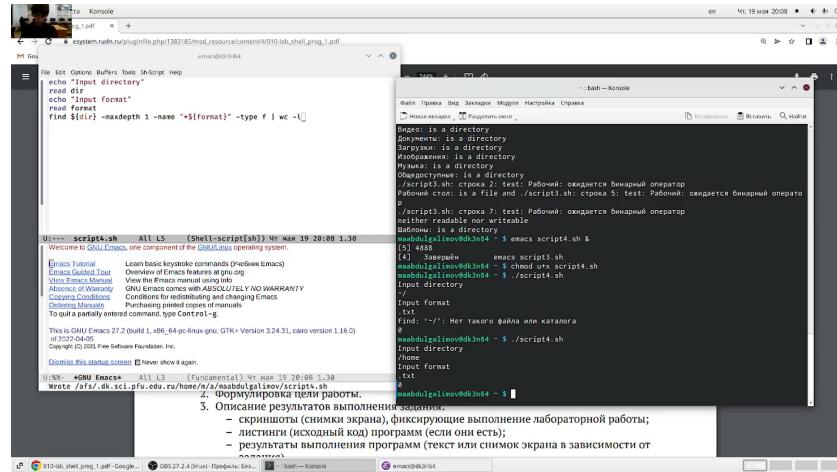


Рис. 0.4: Скрипт 4

Выводы

Научился писать небольшие командные файлы.

Контрольные вопросы

1. 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).
2. POSIX (Portable Operating System Interface for Computer Environments) — набор стандартов описания интерфейсов взаимодействия операционной системы и прикладных программ.
3. `mark=/usr/andy/bin` Данная команда присваивает значение строки символов `/usr/andy/bin` переменной `mark` типа строка символов. Для создания массива используется команда `set` с флагом `-A`. За флагом следует имя переменной, а затем список значений, разделённых пробелами. Например, `set -A states Delaware Michigan "New Jersey"`
4. Команда `let` является показателем того, что последующие аргументы пред-

ставляют собой выражение, подлежащее вычислению. Команда read позволяет читать значения переменных со стандартного ввода

5. Простейшими математическими выражениями являются сложение (+), вычитание (-), умножение (*), целочисленное деление (/) и целочисленный остаток от деления (%).
6. Для облегчения программирования можно записывать условия оболочки bash в двойные скобки — (()).
7. Переменные PS1 и PS2 предназначены для отображения промптера командного процессора. PS1 — это промптер командного процессора, по умолчанию его значение равно символу \$ или #. Если какая-то интерактивная программа, запущенная командным процессором, требует ввода, то используется промптер PS2. Он по умолчанию имеет значение символа >. Другие стандартные переменные: HOME — имя домашнего каталога пользователя. Если команда cd вводится без аргументов, то происходит переход в каталог, указанный в этой переменной. IFS — последовательность символов, являющихся разделителями в командной строке, например, пробел, табуляция и перевод строки (new line). MAIL — командный процессор каждый раз перед выводом на экран промптера проверяет содержимое файла, имя которого указано в этой переменной, и если содержимое этого файла изменилось с момента последнего ввода из него, то перед тем как вывести на терминал промптер, командный процессор выводит на терминал сообщение You have mail (у Вас есть почта). TERM — тип используемого терминала. LOGNAME — содержит регистрационное имя пользователя, которое устанавливается автоматически при входе в систему.
8. Такие символы, как ' < > * ? | " &, являются метасимволами и имеют для командного процессора специальный смысл
9. Снятие специального смысла с метасимвола называется экранированием метасимвола. Экранирование может быть осуществлено с помощью предшествую-

щего метасимволу символа , который, в свою очередь, является метасимволом.

10. Командный файл можно создать с помощью какого-либо редактора, затем сделать его исполняемым и запустить его из терминала, введя “./название файла”.
11. помощью ключевого слова function.
12. Вводим команду ls -lrt и если первым в правах доступа стоит d то это каталог.
Иначе это файл.
13. Для создания массива используется команда set с флагом -A. Если использовать typeset -i для объявления и присвоения переменной, то при последующем её применении она станет целой. Изъять переменную из программы можно с помощью команды unset.
14. При вызове командного файла на выполнение параметры ему могут быть переданы точно таким же образом, как и выполняемой программе. С точки зрения командного файла эти параметры являются позиционными. Символ \$ является метасимволом командного процессора. Он используется, в частности, для ссылки на параметры, точнее, для получения их значений в командном файле. В командный файл можно передать до девяти параметров.
15. \$* — отображается вся командная строка или параметры оболочки; \$? — код завершения последней выполненной команды; \$\$ — уникальный идентификатор процесса, в рамках которого выполняется командный процессор; \$! — номер процесса, в рамках которого выполняется последняя вызванная на выполнение в командном режиме команда; \$- — значение флагов командного процессора; \${#} — возвращает целое число — количество слов, которые были результатом \$; \${#name} — возвращает целое значение длины строки в переменной name; \${name[n]} — обращение к n-му элементу массива; \${name[*]} — перечисляет все элементы массива, разделённые пробелом; \${name[@]} — то же самое, но позволяет учитывать символы пробелы в самих переменных; \${name:-value}

— если значение переменной name не определено, то оно будет заменено на указанное value; \${name:value} — проверяется факт существования переменной; \${name=value} — если name не определено, то ему присваивается значение value; \${name?value} — останавливает выполнение, если имя переменной не определено, и выводит value как сообщение об ошибке; \${name+value} — это выражение работает противоположно \${name-value}. Если переменная определена, то подставляется value; \${name#pattern} — представляет значение переменной name с удалённым самым коротким левым образцом (pattern); \${#name[*]} и \${#name[@]} — эти выражения возвращают количество элементов в массиве name.