

Лабораторная работа-10

**Программирование в командном процессоре ОС UNIX. Командные
файлы**

Овчинников Данил НБИбд-03-22

Содержание

1	Цель работы	5
2	Выполнение лабораторной работы	6
3	Выводы	12
4	Ответы на контрольные вопросы:	13
	Список литературы	16

Список иллюстраций

2.1	Команды1	6
2.2	Код1	7
2.3	Команды2	7
2.4	Код2	8
2.5	Команды3	9
2.6	Код3	10
2.7	Команды4	10
2.8	Код4	11

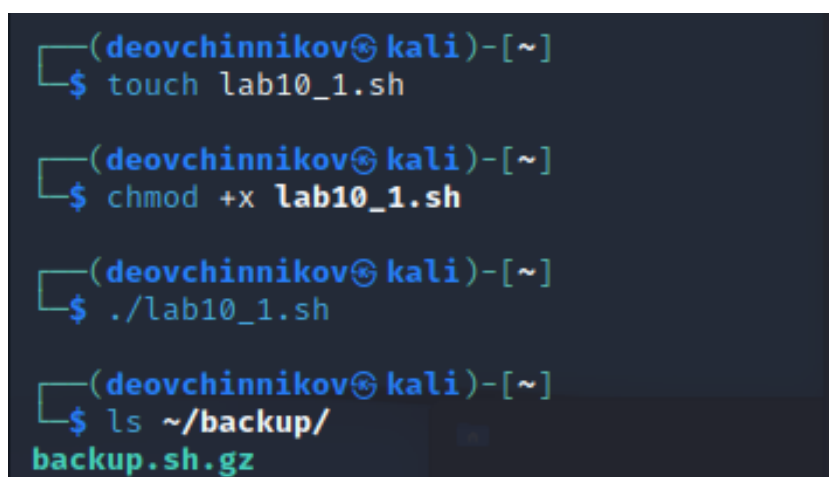
Список таблиц

1 Цель работы

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

2 Выполнение лабораторной работы

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. 2.1 2.2)



```
(deovchinnikov@kali)-[~]  
$ touch lab10_1.sh  
  
(deovchinnikov@kali)-[~]  
$ chmod +x lab10_1.sh  
  
(deovchinnikov@kali)-[~]  
$ ./lab10_1.sh  
  
(deovchinnikov@kali)-[~]  
$ ls ~/backup/  
backup.sh.gz
```

Рис. 2.1: Команды1

```

$ lab10_1.sh • presentation.md •
home > deovchinnikov > $ lab10_1.sh
1  #!/bin/bash
2  mkdir ~/backup
3  cp lab10_1.sh ~/backup/backup.sh
4  gzip ~/backup/backup.sh

```

Рис. 2.2: Код1

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(рис. 2.3 2.4).

```

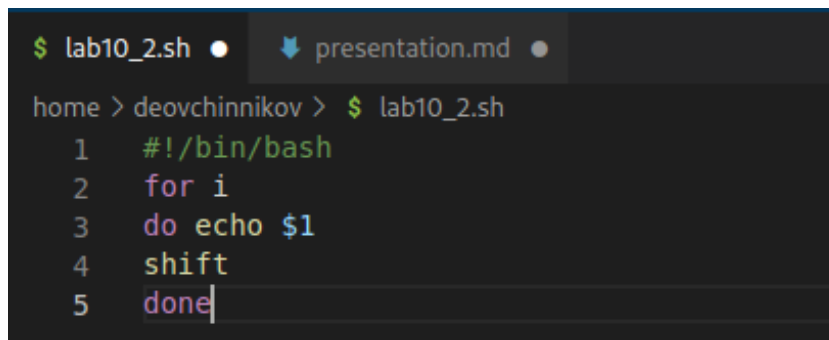
(deovchinnikov@kali)-[~]
$ touch lab10_2.sh

(deovchinnikov@kali)-[~]
$ chmod +x lab10_2.sh

(deovchinnikov@kali)-[~]
$ ./lab10_2.sh S D K 5 3
S
D
K
5
3

```

Рис. 2.3: Команды2



```
$ lab10_2.sh presentation.md
home > deovchinnikov > $ lab10_2.sh
1  #!/bin/bash
2  for i
3  do echo $1
4  shift
5  done
```

Рис. 2.4: Код2

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. (рис. 2.5 2.6).


```

(deovchinnikov@kali)-[~]
$ touch lab10_3.sh

(deovchinnikov@kali)-[~]
$ chmod +x lab10_3.sh

(deovchinnikov@kali)-[~]
$ ./lab10_3.sh
READ
*
./vboxclient-clipboard.pid
./local
./xsession-errors.old
./Public
./backup
./face.icon
./texlive2022
./rpmdb
./Videos
./gnupg
./work
./zsh_history
./bashrc.original
./config
./lab10_3.sh
./viminfo
./Music
./face
./sudo_as_admin_successful
./wyare.txt.swp
./mozilla
./Pictures
./vboxclient-display-svgx-x11.pid
./Documents
./selected_editor
./bash_logout
./pki
./hello.sh.swo
./zshrc
./lab10_1.sh

```

Рис. 2.5: Команды3

```

$ lab10_3.sh presentation.md
home > deovchinnikov > $ lab10_3.sh
1  #!/bin/bash
2  echo "READ"
3  find $1 -maxdepth 1 -perm /u=r
4  echo "WRITE"
5  find $1 -maxdepth 1 -perm /u=w
6  echo "EXECUTE"
7  find $1 -maxdepth 1 -perm /u=x

```

Рис. 2.6: Код3

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. 2.7 2.8)

```

(deovchinnikov@kali)-[~]
$ touch lab10_4.sh
(deovchinnikov@kali)-[~]
$ chmod +x lab10_4.sh
(deovchinnikov@kali)-[~]
$ ./lab10_4.sh
write format
dll
write directory
os
find: 'os': No such file or directory
0
backup Desktop Documents Downloads equipment lab10_1.sh lab10_2.sh lab10_3.sh lab10_4.sh

```

Рис. 2.7: Команды4

```
$ lab10_3.sh $ lab10_4.sh presentation.md
home > deovchinnikov > $ lab10_4.sh
1  #!/bin/bash
2  direct=''
3  form=''
4  echo 'write format'
5  read form
6  echo 'write directory'
7  read direct
8  find "$direct" -name "$*.form" -type f | wc -l
9  ls
```

Рис. 2.8: Код4

3 Выводы

Я изучил основы программирования в оболочке ОС UNIX/Linux. Научился писать небольшие командные файлы.

4 Ответы на контрольные вопросы:

1. Объясните понятие командной оболочки. Приведите примеры командных оболочек. Чем они отличаются?
 - a) sh — стандартная командная оболочка UNIX/Linux, содержащая базовый, полный набор функций
 - b) csh — использующая C-подобный синтаксис команд с возможностью сохранения истории выполнения команд
 - c) ksh — напоминает оболочку C, но операторы управления программой совместимы с операторами оболочки Борна
 - d) bash — сокращение от Bourne Again Shell (опять оболочка Борна), в основе своей совмещает свойства оболочек C и Корна
2. Что такое POSIX? POSIX (Portable Operating System Interface for Computer Environments) — набор стандартов описания интерфейсов взаимодействия операционной системы и прикладных программ.
3. Как определяются переменные и массивы в языке программирования bash? Переменные вызываются \$var, где var=чему-то, указанному пользователем, неважно что бы то не было, название файла, каталога или еще чего. Для массивов используется команда set -A
4. Каково назначение операторов let и read? let — вычисляет далее заданное математическое значение read — позволяет читать значения переменных со стандартного ввода 10
5. Какие арифметические операции можно применять в языке программирования

- bash? Прибавление, умножение, вычисление, деление), сравнение значений, экспонирование и др.
6. Что означает операция (())? Это обозначение используется для облегчения программирования для условий bash
 7. Какие стандартные имена переменных Вам известны? Нам известны HOME, PATH, BASH, ENV, PWD, UID, OLDPWD, PPID, GROUPS, OSTYPE, PS1 - PS4, LANG, HOSTFILE, MAIL, TERM, LOGNAME, USERNAME, IFS и др.
 8. Что такое метасимволы? Метасимволы это специальные знаки, которые могут использоваться для сокращения пути, поиска объекта по расширению, перед переменными, например «\$» или «*» .
 9. Как экранировать метасимволы? Добавить перед метасимволом метасимвол «\»
 10. Как создавать и запускать командные файлы? При помощи команды chmod. Надо дать права на запуск chmod +x название файла, затем запустить bash ./название файла Например у нас файл lab Пишем: chmod +x lab ./lab
 11. Как определяются функции в языке программирования bash? Объединяя несколько команд с помощью function
 12. Каким образом можно выяснить, является файл каталогом или обычным файлом? Можно задать команду на проверку директория ли это test -d директория
 13. Каково назначение команд set, typeset и unset? Set — используется для создания массивов Unset — используется для изъятия переменной Typeset — используется для присваивания каких-либо функций
 14. Как передаются параметры в командные файлы? Добавлением аргументов после команды запуска bash скрипта
 15. Назовите специальные переменные языка bash и их назначение. – \$* — отображается вся командная строка или параметры оболочки; – \$? — код завершения последней выполненной команды; – \$\$ — уникальный идентификатор процесса, в рамках которого выполняется командн– \$! —

номер процесса, в рамках которого выполняется последняя вызванная на вып- \$- — 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 пере- $\{name:-value\}$ — если значение переменной name не определено, то оно будет заменено на- $\{name:value\}$ — проверяется факт существования переменной; — $\{name=value\}$ — если name не определено, то ему присваивается значение value; — $\{name?value\}$ — останавливает выполнение, если имя переменной не определено, — $\{name+value\}$ — это выражение работает противоположно $\{namevalue\}$. Если переменная определена, то подставляется value;

Список литературы