

**МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ
ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ
ВЫСШЕГО ОБРАЗОВАНИЯ «РОССИЙСКИЙ УНИВЕРСИТЕТ ДРУЖБЫ НАРОДОВ»**

Факультет физико-математических и естественных наук

Кафедра информационных технологий

ОТЧЕТ

по лабораторной работе 10

ТЕМА «Программирование в командном процессоре ОС UNIX. Командные файлы» по дисциплине «Операционные системы»

Выполнил/ла:

Студент/ка группы НПИбд-02-21

Студенческий билет No 1032205421

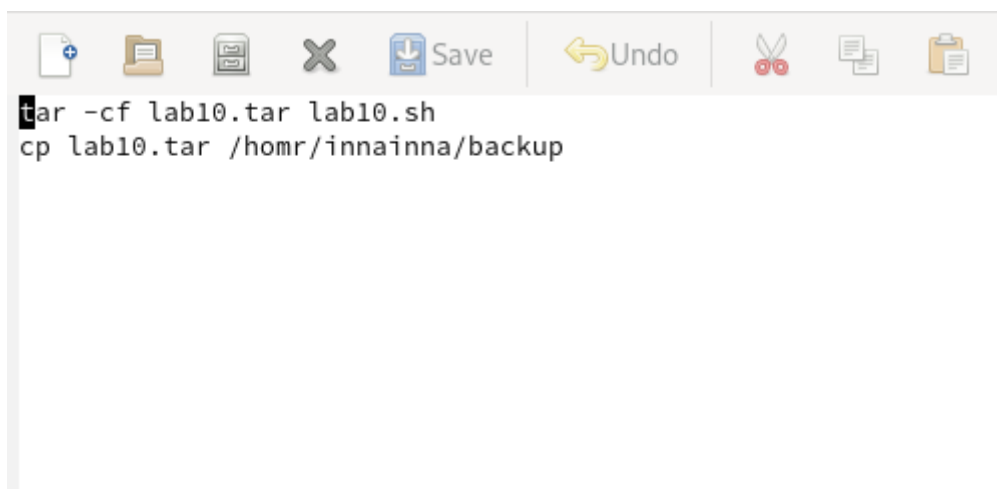
Студент/ка: Стелина Петрити

Цель работы

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

Последовательность выполнения работы

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.



```
tar -cf lab10.tar lab10.sh
cp lab10.tar /home/innainna/backup
```

1.1. исходный код

```

[inna@fedora ~]$ emacs lab10.sh
[inna@fedora ~]$ emacs lab10.sh
[inna@fedora ~]$ chmod +x lab10.sh
[inna@fedora ~]$ ./lab10.sh
cp: cannot create regular file '/homr/innainna/backup': No such file
y
[inna@fedora ~]$ emacs lab10.sh
[inna@fedora ~]$ ./lab10.sh
[inna@fedora ~]$ ls
123.sh          lab10.sh~
123.tar         lab10.tar
456.sh         lab4.md
abc1.sh        lab4.pdf
abc1.tar       letters
archive        may
[inna@fedora ~]$ ls backup
123.tar lab10.tar
[inna@fedora ~]$ cd backup
[inna@fedora backup]$ tar -xvf lab10.tar
lab10.sh
[inna@fedora backup]$ la
bash: la: command not found...
l[inna@fedora backup]$ la
bash: la: command not found...
[inna@fedora backup]$ ls
123.tar lab10.sh lab10.tar

```

1.2. использование команд архивации, 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.

```

#!/bin/bash
for a in "$@"; do
    echo "$a"
done

```

2.1. скрипт/исходный код

```

[inna@fedora backup]$ emacs point2.sh
[inna@fedora backup]$ ./point2.sh 10 20 30
bash: ./point2.sh: Permission denied
[inna@fedora backup]$ chmod +x point2.sh
[inna@fedora backup]$ ./point2.sh 10 20 30
10
20
30
[inna@fedora backup]$

```

2.2.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.

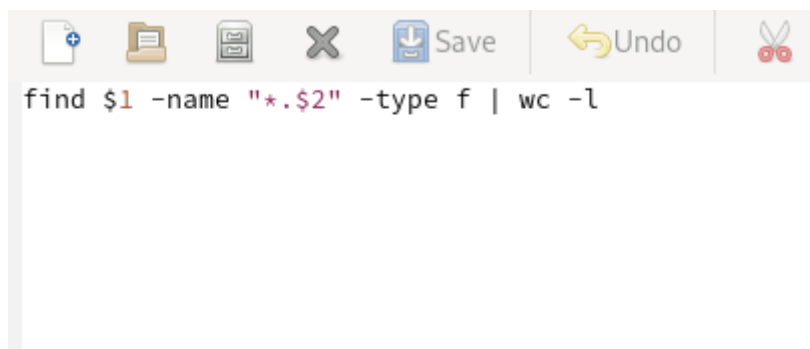
```
#!/bin/bash
echo "Enter the way:"
read ctlg
change=$ctlg
cd $change
echo *
```

3.1.скрипт/исходный код

```
[inna@fedora backup]$ emacs point3.sh
[inna@fedora backup]$ chmod +x point3.sh
[inna@fedora backup]$ ./point3.sh
Enter the way:
/home/inna/backup
123.tar lab10.sh lab10.tar point2.sh point3.sh
[inna@fedora backup]$
```

3.2.проверка

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.

A screenshot of a file manager window. The title bar contains icons for a new file, a folder, a disk, a close button, a save button labeled 'Save', an undo button labeled 'Undo', and a scissors icon. The main area is a terminal window with the command `find $1 -name "*. $2" -type f | wc -l` entered in red text.

```
find $1 -name "*. $2" -type f | wc -l
```

4.1.скрипт/исходный код

```
[inna@fedora backup]$ emacs point4.sh
[inna@fedora backup]$ chmod +x point4.sh
[inna@fedora backup]$ ./point4.sh /home/inna/backup sh
find: missing argument to `-type'
0
[inna@fedora backup]$ emacs point4.sh
[inna@fedora backup]$ ./point4.sh /home/inna/backup sh
4
[inna@fedora backup]$ ./point4.sh /home/inna/backup pdf
0
[inna@fedora backup]$ ./point4.sh /home/inna/backup txt
0
[inna@fedora backup]$ ./point4.sh /home/inna/backup png
0
[inna@fedora backup]$
```

4.2. проверка

Выводы

В этой лаборатории мы изучали основы программирования в оболочке ОС UNIX/Linux. и научился писать пакетные файлы в терминальном unix /linux

Контрольные вопросы

1.Объясните понятие командной оболочки. Приведите примеры командных оболочек. Чем они отличаются?

Оболочка Борна - содержащая базовый, но при этом полный набор функций. С-оболочка - использующая С-подобный синтаксис команд с возможностью сохранения истории выполнения команд. Оболочка Корна - операторы управления программой совместимы с операторами оболочки Борна. BASH - в основе своей совмещает свойства оболочек С и Корна. Оболочка операционной системы — интерпретатор команд операционной системы (ОС),обеспечивающий интерфейс для взаимодействия пользователя с функциями системы.

2.Что такое POSIX?

POSIX - это стандарт, описывающий интерфейс между операционной системой и прикладной программой.

3.Как определяются переменные и массивы в языке программирования bash?

Переменная/=значение.set -А переменная, список значений

4.Каково назначение операторов let и read? let - берет два операнда и присваивает их переменной. read - чтение значения переменных со стандартного ввода.

5.Какие арифметические операции можно применять в языке программирования bash?

Опер. логики, умножение, деление, сложение, вычитание.

6.Что означает операция (())? Условия оболочки bash.

7.Какие стандартные имена переменных Вам известны? MAIL, PATH, IFS, TERM

8. Что такое метасимволы?

' * \ ? | " < > &, являются метасимволами и имеют для командного процессора отличный от обычных символом смысл

9. Как экранировать метасимволы?

Для экранирования нужно заключить её в одинарные кавычки. Строка, заключённая в двойные кавычки, экранирует все метасимволы, кроме ' , , "\$.

10. Как создавать и запускать командные файлы?

```
bash <командный_файл> [аргументы] chmod +x <командный_файл> ./командный_файл
```

11. Как определяются функции в языке программирования bash?

```
function <fun_name>{тело функции}
```

12. Каким образом можно выяснить, является файл каталогом или обычным файлом?

– test -d file — истина, если file является каталогом, ложь - является файлом.

13. Каково назначение команд set, typeset и unset?

Для создания массива используется команда set с флагом -A typeset является встроенной инструкцией и предназначена для наложения ограничений на переменные. Unset можно изъять переменную из программы.

14. Как передаются параметры в командные файлы?

Символ \$ является метасимволом командного процессора, используется, в частности, для ссылки на параметры. При использовании где-либо в командном файле комбинации символов \$i, где 0 < i < 10, вместо неё будет осуществлена подстановка значения параметра с порядковым номером i. \$0 приводит к подстановке вместо неё имени данного командного файла.

15. Назовите специальные переменные языка bash и их назначение.

\$! 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;

\$* отображается вся командная строка или параметры оболочки;

\$? код завершения последней выполненной команды;

\$\$ уникальный идентификатор процесса, в рамках которого выполняется командный процессор;