Лабораторная работа №11 по предмету Операционные системы

НПМбв-02-19

Нечаева Виктория Алексеевна

Содержание

# Цель работы

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

# Задание

Здесь приводится описание задания в соответствии с рекомендациями методического пособия и выданным вариантом.

# Выполнение лабораторной работы

## Задача 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. 1-2)

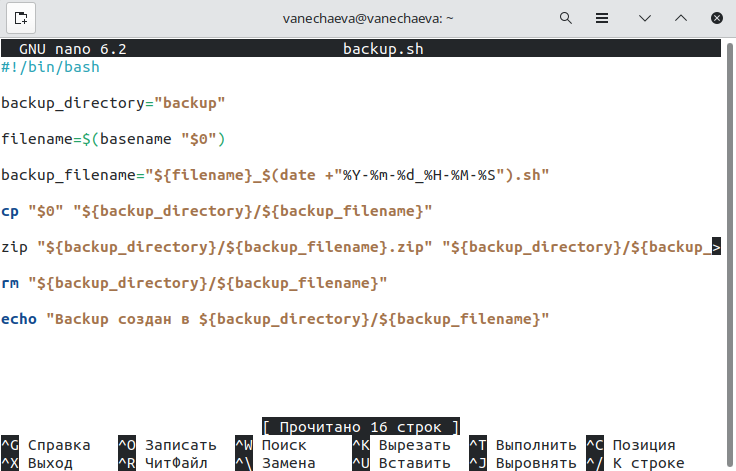


Рисунок 1

В backup\_directory задается название каталога куда будет отправляться бэкап скрипта.  
В filename задается переменная с указателем , ссылаясь на название текущего скрипта  
backup\_filename задает название нового бэкапа, в новом названии генерируется дата и время создания  
cp в данном случае копирует текущий скрипт в backup  
zip сжимает созданный в папке backup бэкапный файл, так как сжать можно только каталог, файл автоматически кладется в папку backup  
rm удаляет backup файл, так как по заданию нужен именно архив  
echo выводит в консоль инфо о созданном бэкапе

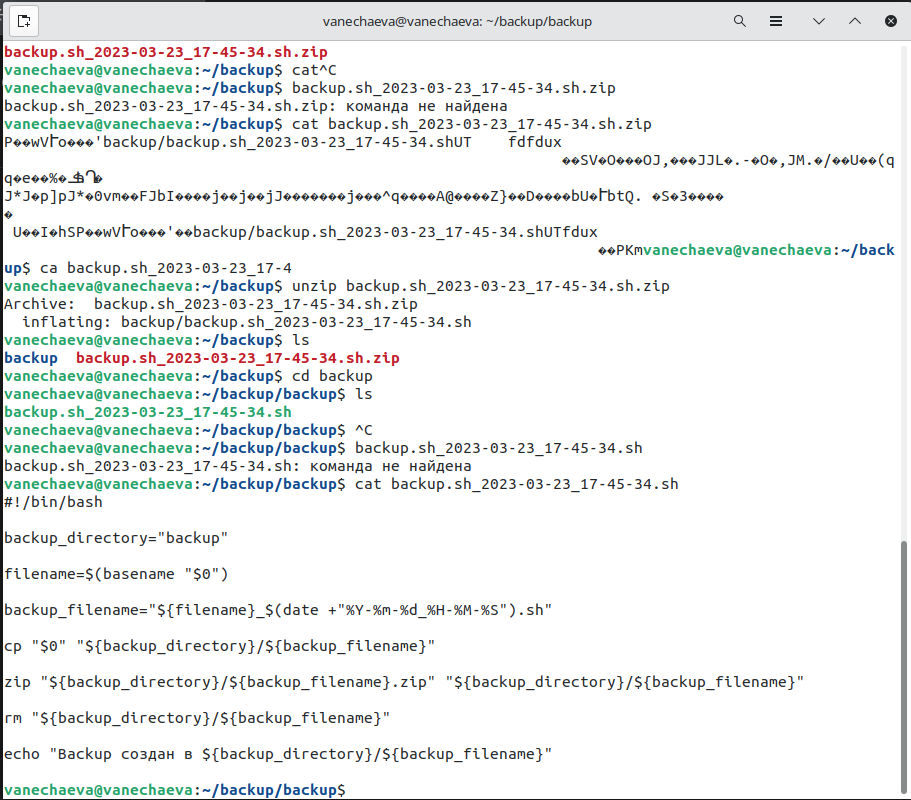


Рисунок 2

## Задача 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.

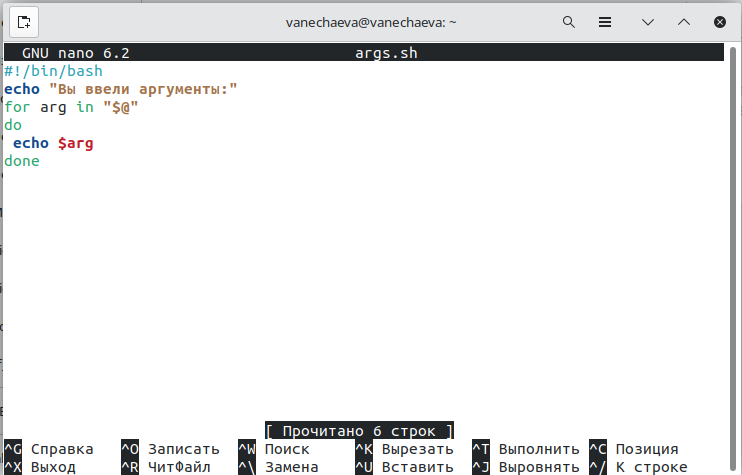


Рисунок 3

В коде скрипта только цикл по всем аргументам, указанным при вызове скрипта в командной строке. Выводит более 10 аргументов.

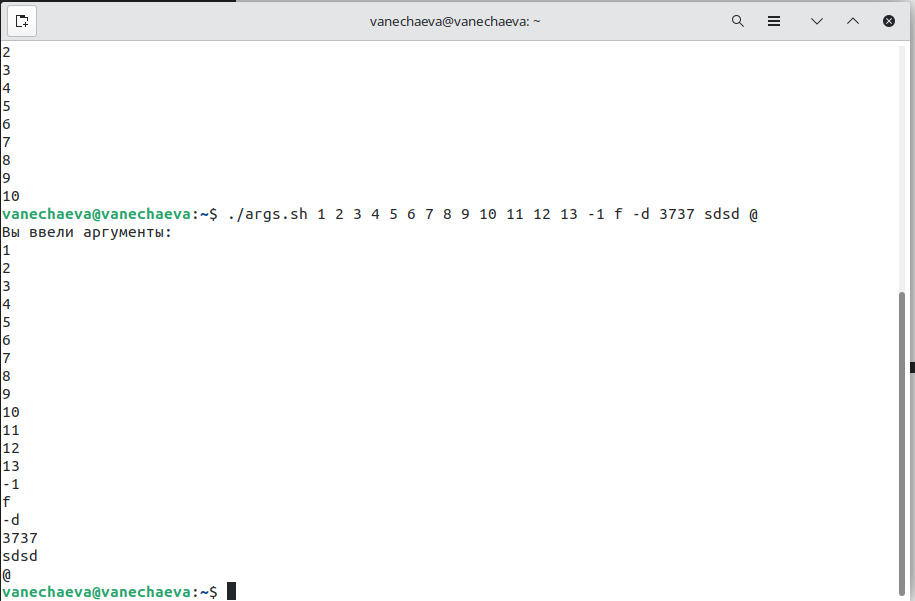


Рисунок 4

## Задача 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.

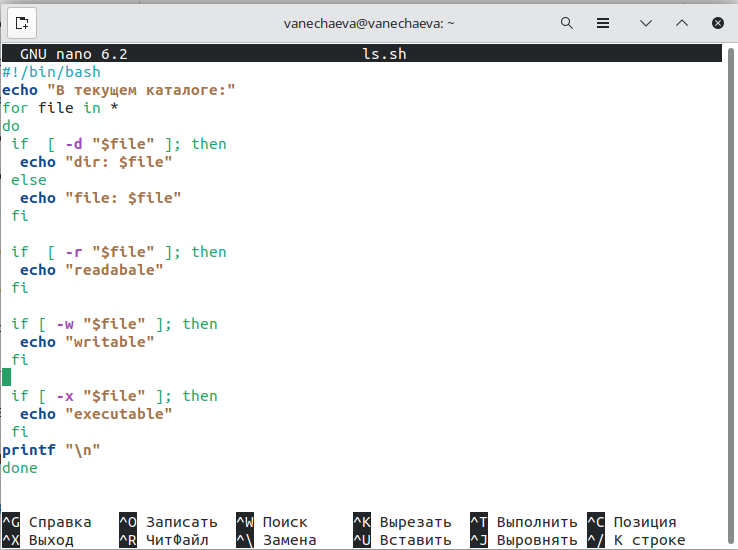


Рисунок 5

Цикл проходит по всем файлам директории и проверяет на несколько критериев:  
Если тип это файл (-f) или каталог (-d), оно выводится в консоли. Таким образом выводятся все файлы и каталоги. Остается вывести инфо о доступах.  
В трех следующих проверках условий проверяются права доступа -r (readable), -w (writable), -x (executable), информация об этом также выводится в консоли.

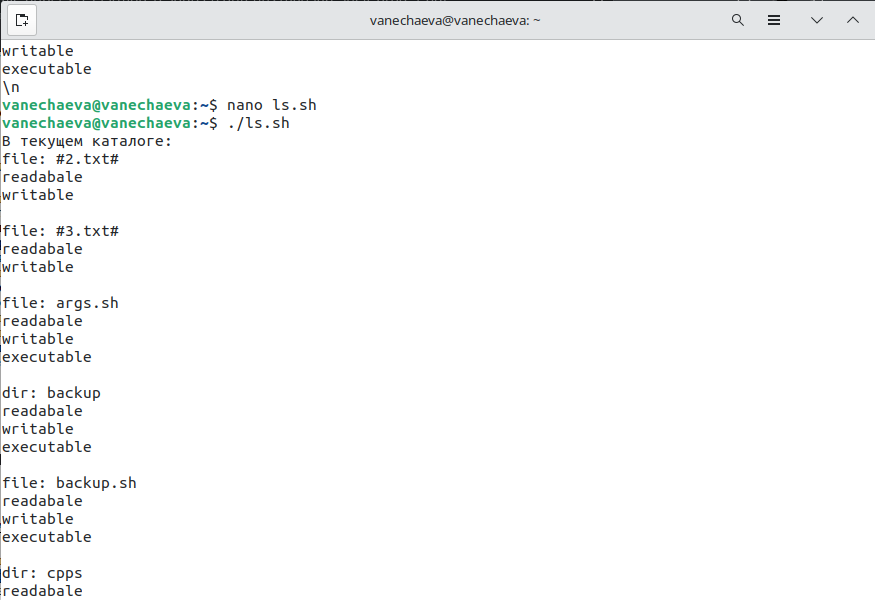


Рисунок 6

## Задача 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.

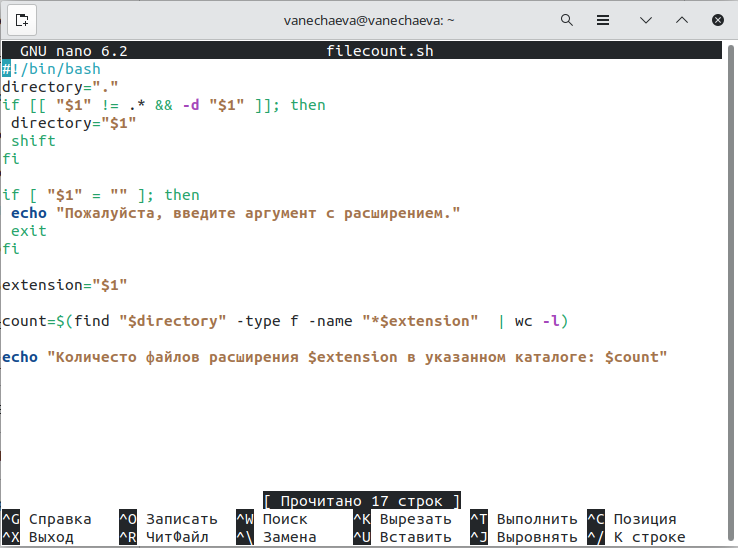


Рисунок 7

Директория по умолчанию устанавливается текущая - “.”  
В первом условии if проверяется, что первый аргумент это НЕ расширение и что первый аргумент это каталог. Если условия подходят, то переменной directory вместо значения по умолчанию присваивается каталог из первого аргумента. shift сдвигает аргумент влево, чтобы скрипт мог прочитать расширение, указанное пользователем.

Во второй проверке if проверяется условие для случая, если мы решили не указывать директорию в аргументе и указали только нужное расширение. Проверяется, что первый аргумент пустой, и в таком случае выводится сообщение о необходимости указать расширение и скрипт завершает выполнение с помощью exit.

Далее extension присвается значение первого введенного нами аргумента. count ищет с помощью find объекты типа f (файл) с названием по маске extension введенного нами, конвеер передает результат поиска функции wc которая с опцией -l считает количество строк и в последней строке через echo мы его выводим в консоли.

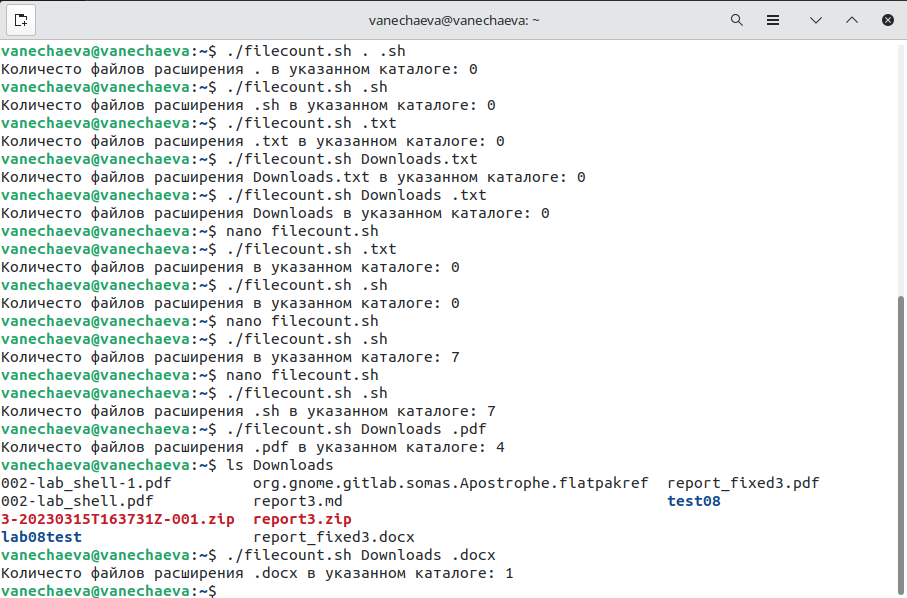


Рисунок 8

# Выводы

В ходе данной лабораторной работы мною были изучены основы программирования в оболочке OC UNIX/Linux.

# Контрольные вопросы

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

Командная оболочка — программа, через которую пользователь или администратор управляет операционной системой и установленными программами, используя командную строку. У этого понятия множество синонимов — shell, командный интерпретатор и командный процессор. Пример bash, zsh.

1. Что такое POSIX?

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

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

Переменная инициализруется с помощью значка доллара.

Инициализация, или объявление массивов происходит либо явно, с использованием команды declare, либо при указании массива прямо в коде. Обратите внимание на то, что если не использовать команду declare – то массив по-умолчанию будет считаться индексированным.

Примеры объявления массивов:  
$ declare -a array  
$ array=( [1]=one [2]=two )  
$ echo ${array[@]}  
one two

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

let a=5+4  
echo $a # 9  
let “a = 5 + 4”  
echo $a # 9  
let a++  
echo $a # 10  
let “a = 4 \* 5”  
echo $a # 20  
let “a = $1 + 30”  
echo $a

1. Что означает операция (( ))?

В двойных скобках выполняется целочисленная арифметика.

1. Какие стандартные имена переменных Вам известны?

arg

1. Что такое метасимволы?

Метасимволом называется особая комбинация символов, которые в реальной строке могут совпадать с разными символами. Кроме того, некоторые метасимволы совпадают с границами между символами (например с началом строки).

1. Как экранировать метасимволы?

С помощью бэкслеша

1. Как создавать и запускать командные файлы?

Создать файл filename.sh, дать права на запуск chmod +x filename.sh, запустить скрипт ./filename.sh

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

в консоли через ls  
в шелл скрипте [ -d “/path/dir/” ] && echo “Directory /path/dir/ exists”

1. Как передаются параметры в командные файлы?

При запуске скрипта указываются через пробел

1. Назовите специальные переменные языка bash и их назначение.

$\* – все аргументы; · $@ – все аргументы; · $# – количество аргументов; · $0 – имя скрипта;