Лабораторная работа № 12. Программирование в командном процессоре ОС UNIX. Командные файлы

Отчёт

Сергеев Даниил Олегович

Содержание

# 1 Цель работы

Изучить основы программирования в оболочке ОС UNIX/Linux. Научиться писать небольшие командные файлы. [1]

# 2 Задание

Написать скрипты для задач лабораторной работы

# 3 Ход выполнения лабораторной работы

## 3.1 Выполнение упражнений

Перейдем в каталог lab07 и создадим файл first.sh, открыв emacs в фоновом режиме.

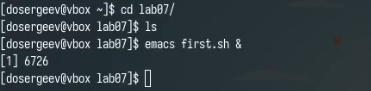


Рис. 1: Создание первого скрипта.

Напишем код для первого скрипта. Он будет создавать резервную копию самого себя в каталоге ~/backup. Созданный файл будеть архивироваться с помощью tar. (рис. 2)



Рис. 2: Код первого скрипта

Проверим работу командного файл.

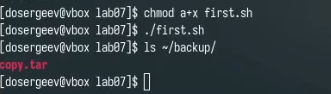


Рис. 3: Проверка первого скрипта

Теперь напишем командный файл для распечатывания значений всех переданных ему аргументов. (рис. 4)

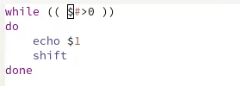


Рис. 4: Код второго скрипта

Проверим его работу, обязательно на 11 и более аргументах.



Рис. 5: Проверка второго скрипта

Следующий скрипт станет аналогом ls - он будет выдавать информацию о нужном каталоге и выводить информацию о возможностях доступа к его файлам. (рис. 6)

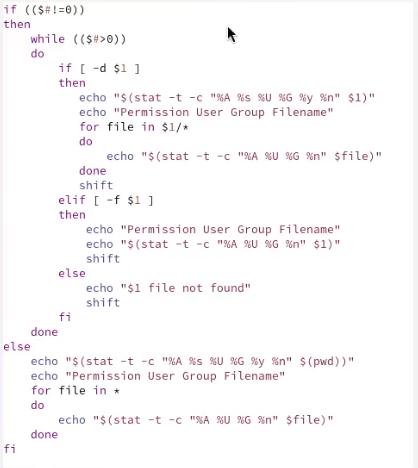


Рис. 6: Код третьего скрипта

Проверим его работу на домашнем каталоге. Укажем путь к нему в качестве аргумента.

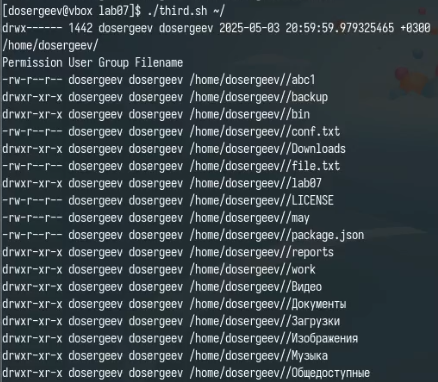


Рис. 7: Проверка третьего скрипта

Последний скрипт будет выводить все файлы указанного формата (.txt, .doc, .jpg, .pdf) в заданной директории. Для этого попробуем использовать команду getopts. Будем принимать директорию и формат файла в качестве аргумента ключей -d и -f. (рис. 8)

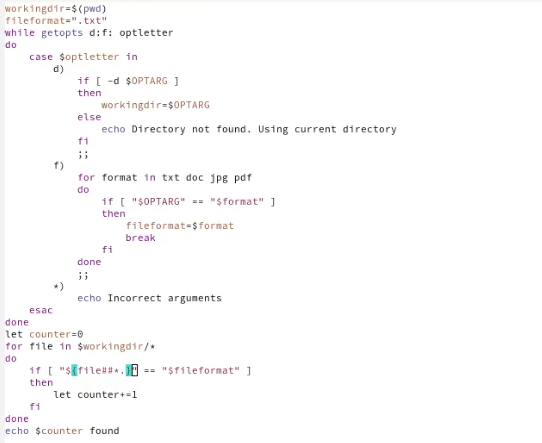


Рис. 8: Код четвертого скрипта

Проверим его работу на домашнем каталоге. Укажем путь ~/ и формат .txt

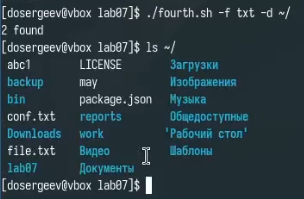


Рис. 9: Проверка четвертого скрипта

## 3.2 Ответы на контрольные вопросы

1. Командная оболочка - это программа, позволяющая пользователю взаимодействовать с операционной системой. Примеры:
   * Bourne shell (sh)
   * C shell (csh)
   * Korn shell (ksh)
   * BASH (Bourne Again Shell) Основные отличия: синтаксис, функциональность, совместимость с POSIX.
2. POSIX (Portable Operating System Interface) - набор стандартов, описывающих интерфейсы взаимодействия ОС и прикладных программ для обеспечения совместимости UNIX-подобных систем.
3. В bash:
   * Переменные: name=value
   * Массивы: set -A array\_name value1 value2 "value 3" или array\_name[index]=value
4. Операторы:
   * let - вычисление арифметических выражений
   * read - чтение значений переменных со стандартного ввода
5. Можно применять данные арифметические операции в bash: + (сложение), - (вычитание), \* (умножение), / (деление), % (остаток от деления), битовые операции и операции сравнения.
6. Операция (( )) используется для вычисления арифметических выражений и возвращает статус 0 (истина), если результат не нулевой.
7. Стандартные переменные:
   * PATH - пути поиска команд
   * PS1, PS2 - промптеры
   * HOME - домашний каталог
   * IFS - разделители полей
   * MAIL - файл почты
   * TERM - тип терминала
   * LOGNAME - имя пользователя
8. Метасимволы - символы с особым значением в командной оболочке: ' < > \* ? | \ " & и другие.
9. Экранирование метасимволов:
   * Перед символом: \\*
   * Группа в одинарных кавычках: '\*\|\*'
   * В двойных кавычках (кроме $, ', \, ")
   * Создать текстовый файл с командами
   * Дать права на выполнение: chmod +x filename
   * Запустить: ./filename или bash filename
10. Функции определяются так: bash function name { команды } Удаление: unset -f name
11. Проверить тип файла можно командой test:
    * test -f file - обычный файл
    * test -d file - каталог Или: [ -f file ], [ -d file ]
12. Назначение команд:
    * set - установка параметров оболочки/вывод переменных
    * typeset (или declare) - задание типа переменной
    * unset - удаление переменной/функции
13. Параметры в командные файлы передаются как аргументы при вызове и доступны через $1, $2, …, $9, $0 - имя скрипта.
14. Специальные переменные bash:
    * $\*, $@ - все параметры
    * $# - количество параметров
    * $? - код завершения последней команды
    * $$ - PID текущего процесса
    * $! - PID последнего фонового процесса
    * $- - флаги оболочки
    * ${#\*} - количество слов в $\*
    * ${#name} - длина строки переменной

# 4 Вывод

В результате выполнения лабораторной работы я изучил основы программирования в оболочке ОС UNIX/Linux и научился писать небольшие командные файлы.

# Список литературы

1. Kulyabov. Лабораторная работа № 11. Текстовой редактор emacs. https://esystem.rudn.ru/pluginfile.php/2586589/mod\_resource/content/4/010-lab\_shell\_prog\_1.pdf; RUDN.